Fashion MNIST Dataset:

The Fashion MNIST dataset is a popular benchmark dataset often used as an alternative to the original MNIST dataset for testing machine learning and computer vision models. It consists of grayscale images of various fashion items, and the task is to classify these items into one of ten categories. Each image is a 28x28 pixel grayscale image.

Here are the ten categories in the Fashion MNIST dataset:

1. T-shirt/top
2. Trouser
3. Pullover
4. Dress
5. Coat
6. Sandal
7. Shirt
8. Sneaker
9. Bag
10. Ankle boot

The dataset is widely used for tasks related to image classification and deep learning model evaluation. It's a bit more challenging than the original MNIST dataset due to the greater variety in the types of items and their more complex patterns.

In [1]:
```
# Importing basic packages

import os, sys
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import statsmodels.api as sm
import scipy.stats as stats
from datetime import datetime
import warnings
warnings.filterwarnings('ignore')
%matplotlib inline
sns.set()
pd.set_option('display.max_rows',None)
pd.set_option('display.max_columns',None)
sns.set_style('whitegrid')
```

In [2]:
```
# Importing tensorflow,keras

import tensorflow as tf
from tensorflow import keras
from keras.layers import *
from keras.datasets import fashion_mnist
```

In [3]:
```
# Loading the fashion mnist dataset:

(x_train, y_train), (x_test, y_test) = fashion_mnist.load_data()
```

In [4]:
```
# Visualising the objects:

class_names = ["T-shirt/top", "Trouser", "Pullover", "Dress", "Coat", "Sandal", "Shirt", "Sneaker", "Bag", "Ankle boot"

plt.figure(figsize=(12,12))
for i in range(25):
  plt.subplot(5,5,i+1)
  plt.xticks([])
  plt.yticks([])
  plt.imshow(x_train[i], cmap=plt.cm.binary)
  plt.xlabel(class_names[y_train[i]])

plt.show()
```

In [5]:
```
# Now reshaping the training and testing data to have a single channel (grayscale images) and the dimensions expected b

x_train = x_train.reshape(x_train.shape[0],28,28,1)
x_test = x_test.reshape(x_test.shape[0],28,28,1)
```

In [6]:
```
# Now normalising the pixel values in order help the model converge faster:

x_train = x_train/255.
x_test = x_test/255.
```

In [7]:
```
# Now converting the categorical labels into one hot encoding:

y_train = keras.utils.to_categorical(y_train, 10)
y_test = keras.utils.to_categorical(y_test, 10)
```

Building a CNN Model

In [8]:
```
# Importing necessary libraries

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

In [9]:
```
# Building  a simple CNN architecture

# Building a sequential API
model = Sequential()

# part 1
model.add(Conv2D(32, kernel_size=(3,3), activation='relu', input_shape = (28,28,1)))
model.add(MaxPooling2D(pool_size=(2,2), strides=2))

# part 2
model.add(Conv2D(64, kernel_size=(3,3), activation='relu'))
model.add(MaxPooling2D(pool_size=(2,2), strides=2))

# part 3
model.add(Conv2D(64, kernel_size=(3,3), activation='relu'))
model.add(MaxPooling2D(pool_size=(2,2), strides=2))

# Flatten
model.add(Flatten())

# Building the DNN model
model.add(Dense(units=64, activation = 'relu'))
model.add(Dropout(0.5))
model.add(BatchNormalization())
model.add(Dense(units=32, activation = 'relu'))
model.add(Dropout(0.4))
model.add(BatchNormalization())
model.add(Dense(units=10, activation = 'softmax'))

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

In [10]:
```
# Training the model

history = model.fit(x_train, y_train, validation_data = (x_test, y_test), batch_size = 64, epochs = 100)
```

```
Epoch 1/100
938/938 [==============================] - 23s 9ms/step - loss: 1.0339 - accuracy: 0.6481 - val_loss: 0.5399 - val_acc
uracy: 0.7987
Epoch 2/100
938/938 [==============================] - 7s 7ms/step - loss: 0.6560 - accuracy: 0.7750 - val_loss: 0.4670 - val_accu
racy: 0.8286
Epoch 3/100
938/938 [==============================] - 7s 7ms/step - loss: 0.5759 - accuracy: 0.8052 - val_loss: 0.4328 - val_accu
racy: 0.8494
Epoch 4/100
938/938 [==============================] - 8s 8ms/step - loss: 0.5300 - accuracy: 0.8236 - val_loss: 0.3953 - val_accu
racy: 0.8588
Epoch 5/100
938/938 [==============================] - 6s 7ms/step - loss: 0.4953 - accuracy: 0.8360 - val_loss: 0.3850 - val_accu
racy: 0.8655
Epoch 6/100
938/938 [==============================] - 7s 8ms/step - loss: 0.4669 - accuracy: 0.8450 - val_loss: 0.3756 - val_accu
racy: 0.8706
Epoch 7/100
938/938 [==============================] - 7s 7ms/step - loss: 0.4449 - accuracy: 0.8536 - val_loss: 0.3631 - val_accu
racy: 0.8723
Epoch 8/100
938/938 [==============================] - 7s 7ms/step - loss: 0.4307 - accuracy: 0.8587 - val_loss: 0.3480 - val_accu
racy: 0.8786
Epoch 9/100
938/938 [==============================] - 7s 7ms/step - loss: 0.4158 - accuracy: 0.8648 - val_loss: 0.3482 - val_accu
racy: 0.8780
Epoch 10/100
938/938 [==============================] - 6s 7ms/step - loss: 0.4033 - accuracy: 0.8690 - val_loss: 0.3353 - val_accu
racy: 0.8826
Epoch 11/100
938/938 [==============================] - 7s 7ms/step - loss: 0.3918 - accuracy: 0.8719 - val_loss: 0.3565 - val_accu
racy: 0.8688
Epoch 12/100
938/938 [==============================] - 6s 7ms/step - loss: 0.3786 - accuracy: 0.8777 - val_loss: 0.4130 - val_accu
racy: 0.8489
Epoch 13/100
938/938 [==============================] - 7s 8ms/step - loss: 0.3722 - accuracy: 0.8798 - val_loss: 0.3448 - val_accu
racy: 0.8782
Epoch 14/100
938/938 [==============================] - 6s 7ms/step - loss: 0.3601 - accuracy: 0.8837 - val_loss: 0.3573 - val_accu
racy: 0.8732
Epoch 15/100
938/938 [==============================] - 8s 9ms/step - loss: 0.3591 - accuracy: 0.8856 - val_loss: 0.3354 - val_accu
racy: 0.8860
Epoch 16/100
938/938 [==============================] - 7s 7ms/step - loss: 0.3451 - accuracy: 0.8895 - val_loss: 0.3486 - val_accu
racy: 0.8832
Epoch 17/100
938/938 [==============================] - 7s 7ms/step - loss: 0.3463 - accuracy: 0.8890 - val_loss: 0.3352 - val_accu
racy: 0.8901
Epoch 18/100
938/938 [==============================] - 7s 8ms/step - loss: 0.3303 - accuracy: 0.8944 - val_loss: 0.3448 - val_accu
racy: 0.8911
Epoch 19/100
938/938 [==============================] - 6s 7ms/step - loss: 0.3252 - accuracy: 0.8971 - val_loss: 0.3821 - val_accu
racy: 0.8739
Epoch 20/100
938/938 [==============================] - 7s 8ms/step - loss: 0.3229 - accuracy: 0.8971 - val_loss: 0.3405 - val_accu
racy: 0.8906
Epoch 21/100
938/938 [==============================] - 6s 7ms/step - loss: 0.3110 - accuracy: 0.9004 - val_loss: 0.3732 - val_accu
racy: 0.8851
Epoch 22/100
938/938 [==============================] - 7s 7ms/step - loss: 0.3037 - accuracy: 0.9030 - val_loss: 0.3863 - val_accu
racy: 0.8848
Epoch 23/100
938/938 [==============================] - 6s 7ms/step - loss: 0.3050 - accuracy: 0.9032 - val_loss: 0.3456 - val_accu
racy: 0.8916
Epoch 24/100
938/938 [==============================] - 7s 8ms/step - loss: 0.2969 - accuracy: 0.9064 - val_loss: 0.3832 - val_accu
racy: 0.8836
Epoch 25/100
938/938 [==============================] - 7s 7ms/step - loss: 0.2968 - accuracy: 0.9062 - val_loss: 0.3554 - val_accu
racy: 0.8900
Epoch 26/100
938/938 [==============================] - 6s 7ms/step - loss: 0.2860 - accuracy: 0.9085 - val_loss: 0.3461 - val_accu
racy: 0.8904
Epoch 27/100
938/938 [==============================] - 7s 7ms/step - loss: 0.2836 - accuracy: 0.9111 - val_loss: 0.3585 - val_accu
racy: 0.8946
Epoch 28/100
938/938 [==============================] - 6s 7ms/step - loss: 0.2798 - accuracy: 0.9125 - val_loss: 0.3586 - val_accu
racy: 0.8918
Epoch 29/100
938/938 [==============================] - 7s 8ms/step - loss: 0.2694 - accuracy: 0.9140 - val_loss: 0.3909 - val_accu
racy: 0.8860
Epoch 30/100
938/938 [==============================] - 6s 7ms/step - loss: 0.2671 - accuracy: 0.9149 - val_loss: 0.3644 - val_accu
racy: 0.8911
Epoch 31/100
938/938 [==============================] - 7s 8ms/step - loss: 0.2643 - accuracy: 0.9179 - val_loss: 0.3797 - val_accu
racy: 0.8847
Epoch 32/100
938/938 [==============================] - 7s 7ms/step - loss: 0.2568 - accuracy: 0.9183 - val_loss: 0.3639 - val_accu
```

```
racy: 0.8925
Epoch 33/100
938/938 [==============================] - 7s 8ms/step - loss: 0.2563 - accuracy: 0.9201 - val_loss: 0.3704 - val_accu
racy: 0.8931
Epoch 34/100
938/938 [==============================] - 7s 8ms/step - loss: 0.2534 - accuracy: 0.9208 - val_loss: 0.3946 - val_accu
racy: 0.8914
Epoch 35/100
938/938 [==============================] - 7s 7ms/step - loss: 0.2481 - accuracy: 0.9233 - val_loss: 0.3809 - val_accu
racy: 0.8834
Epoch 36/100
938/938 [==============================] - 7s 8ms/step - loss: 0.2426 - accuracy: 0.9240 - val_loss: 0.3933 - val_accu
racy: 0.8937
Epoch 37/100
938/938 [==============================] - 6s 7ms/step - loss: 0.2472 - accuracy: 0.9221 - val_loss: 0.3879 - val_accu
racy: 0.8868
Epoch 38/100
938/938 [==============================] - 7s 7ms/step - loss: 0.2344 - accuracy: 0.9242 - val_loss: 0.3939 - val_accu
racy: 0.8885
Epoch 39/100
938/938 [==============================] - 6s 7ms/step - loss: 0.2407 - accuracy: 0.9249 - val_loss: 0.3977 - val_accu
racy: 0.8920
Epoch 40/100
938/938 [==============================] - 7s 7ms/step - loss: 0.2341 - accuracy: 0.9283 - val_loss: 0.3960 - val_accu
racy: 0.8894
Epoch 41/100
938/938 [==============================] - 7s 7ms/step - loss: 0.2292 - accuracy: 0.9298 - val_loss: 0.3884 - val_accu
racy: 0.8911
Epoch 42/100
938/938 [==============================] - 7s 7ms/step - loss: 0.2168 - accuracy: 0.9322 - val_loss: 0.4036 - val_accu
racy: 0.8885
Epoch 43/100
938/938 [==============================] - 7s 7ms/step - loss: 0.2195 - accuracy: 0.9336 - val_loss: 0.4172 - val_accu
racy: 0.8887
Epoch 44/100
938/938 [==============================] - 7s 7ms/step - loss: 0.2206 - accuracy: 0.9330 - val_loss: 0.3812 - val_accu
racy: 0.8919
Epoch 45/100
938/938 [==============================] - 7s 8ms/step - loss: 0.2206 - accuracy: 0.9325 - val_loss: 0.4368 - val_accu
racy: 0.8775
Epoch 46/100
938/938 [==============================] - 7s 7ms/step - loss: 0.2170 - accuracy: 0.9324 - val_loss: 0.3943 - val_accu
racy: 0.8950
Epoch 47/100
938/938 [==============================] - 7s 8ms/step - loss: 0.2088 - accuracy: 0.9359 - val_loss: 0.4096 - val_accu
racy: 0.8928
Epoch 48/100
938/938 [==============================] - 6s 7ms/step - loss: 0.2129 - accuracy: 0.9342 - val_loss: 0.4009 - val_accu
racy: 0.8924
Epoch 49/100
938/938 [==============================] - 7s 8ms/step - loss: 0.2059 - accuracy: 0.9368 - val_loss: 0.4313 - val_accu
racy: 0.8903
Epoch 50/100
938/938 [==============================] - 7s 7ms/step - loss: 0.2028 - accuracy: 0.9370 - val_loss: 0.4540 - val_accu
racy: 0.8892
Epoch 51/100
938/938 [==============================] - 7s 7ms/step - loss: 0.2053 - accuracy: 0.9361 - val_loss: 0.4121 - val_accu
racy: 0.8939
Epoch 52/100
938/938 [==============================] - 7s 8ms/step - loss: 0.1981 - accuracy: 0.9387 - val_loss: 0.4368 - val_accu
racy: 0.8901
Epoch 53/100
938/938 [==============================] - 6s 7ms/step - loss: 0.1970 - accuracy: 0.9396 - val_loss: 0.4591 - val_accu
racy: 0.8843
Epoch 54/100
938/938 [==============================] - 7s 8ms/step - loss: 0.1930 - accuracy: 0.9413 - val_loss: 0.4432 - val_accu
racy: 0.8902
Epoch 55/100
938/938 [==============================] - 6s 7ms/step - loss: 0.1881 - accuracy: 0.9412 - val_loss: 0.4288 - val_accu
racy: 0.8900
Epoch 56/100
938/938 [==============================] - 7s 8ms/step - loss: 0.1858 - accuracy: 0.9433 - val_loss: 0.4316 - val_accu
racy: 0.8903
Epoch 57/100
938/938 [==============================] - 7s 7ms/step - loss: 0.1822 - accuracy: 0.9434 - val_loss: 0.4428 - val_accu
racy: 0.8907
Epoch 58/100
938/938 [==============================] - 7s 8ms/step - loss: 0.1867 - accuracy: 0.9424 - val_loss: 0.4830 - val_accu
racy: 0.8872
Epoch 59/100
938/938 [==============================] - 7s 7ms/step - loss: 0.1827 - accuracy: 0.9444 - val_loss: 0.4607 - val_accu
racy: 0.8925
Epoch 60/100
938/938 [==============================] - 7s 7ms/step - loss: 0.1757 - accuracy: 0.9458 - val_loss: 0.4538 - val_accu
racy: 0.8900
Epoch 61/100
938/938 [==============================] - 8s 8ms/step - loss: 0.1837 - accuracy: 0.9442 - val_loss: 0.4471 - val_accu
racy: 0.8896
Epoch 62/100
938/938 [==============================] - 7s 7ms/step - loss: 0.1747 - accuracy: 0.9461 - val_loss: 0.4655 - val_accu
racy: 0.8903
Epoch 63/100
938/938 [==============================] - 7s 8ms/step - loss: 0.1709 - accuracy: 0.9473 - val_loss: 0.4878 - val_accu
racy: 0.8900
Epoch 64/100
938/938 [==============================] - 7s 7ms/step - loss: 0.1697 - accuracy: 0.9484 - val_loss: 0.4887 - val_accu
```

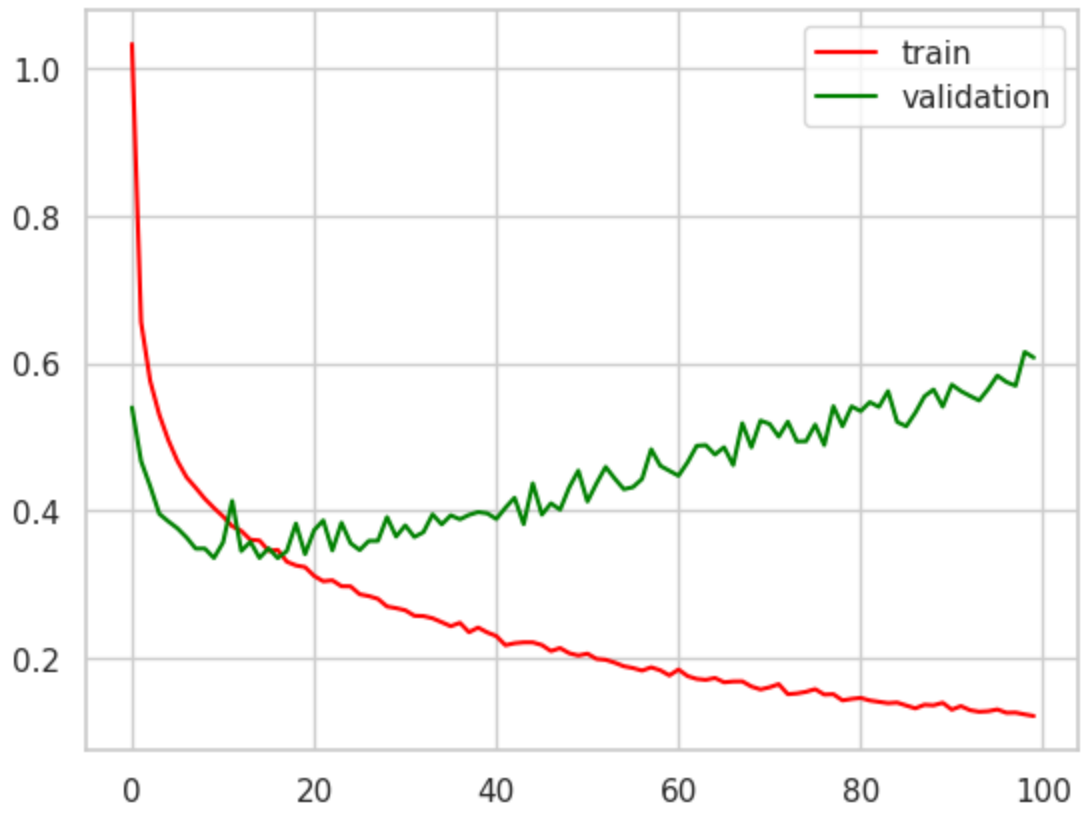
```
racy: 0.8891
Epoch 65/100
938/938 [==============================] - 7s 7ms/step - loss: 0.1722 - accuracy: 0.9478 - val_loss: 0.4759 - val_accu
racy: 0.8912
Epoch 66/100
938/938 [==============================] - 7s 7ms/step - loss: 0.1664 - accuracy: 0.9485 - val_loss: 0.4859 - val_accu
racy: 0.8899
Epoch 67/100
938/938 [==============================] - 7s 7ms/step - loss: 0.1675 - accuracy: 0.9478 - val_loss: 0.4615 - val_accu
racy: 0.8917
Epoch 68/100
938/938 [==============================] - 7s 8ms/step - loss: 0.1676 - accuracy: 0.9493 - val_loss: 0.5186 - val_accu
racy: 0.8857
Epoch 69/100
938/938 [==============================] - 6s 7ms/step - loss: 0.1606 - accuracy: 0.9523 - val_loss: 0.4858 - val_accu
racy: 0.8849
Epoch 70/100
938/938 [==============================] - 8s 9ms/step - loss: 0.1566 - accuracy: 0.9531 - val_loss: 0.5220 - val_accu
racy: 0.8895
Epoch 71/100
938/938 [==============================] - 7s 7ms/step - loss: 0.1596 - accuracy: 0.9521 - val_loss: 0.5175 - val_accu
racy: 0.8881
Epoch 72/100
938/938 [==============================] - 7s 8ms/step - loss: 0.1638 - accuracy: 0.9521 - val_loss: 0.5002 - val_accu
racy: 0.8844
Epoch 73/100
938/938 [==============================] - 8s 8ms/step - loss: 0.1502 - accuracy: 0.9548 - val_loss: 0.5206 - val_accu
racy: 0.8890
Epoch 74/100
938/938 [==============================] - 7s 8ms/step - loss: 0.1512 - accuracy: 0.9554 - val_loss: 0.4938 - val_accu
racy: 0.8909
Epoch 75/100
938/938 [==============================] - 7s 8ms/step - loss: 0.1534 - accuracy: 0.9536 - val_loss: 0.4940 - val_accu
racy: 0.8903
Epoch 76/100
938/938 [==============================] - 7s 7ms/step - loss: 0.1570 - accuracy: 0.9536 - val_loss: 0.5166 - val_accu
racy: 0.8892
Epoch 77/100
938/938 [==============================] - 7s 8ms/step - loss: 0.1497 - accuracy: 0.9530 - val_loss: 0.4888 - val_accu
racy: 0.8923
Epoch 78/100
938/938 [==============================] - 6s 7ms/step - loss: 0.1502 - accuracy: 0.9541 - val_loss: 0.5417 - val_accu
racy: 0.8901
Epoch 79/100
938/938 [==============================] - 7s 7ms/step - loss: 0.1417 - accuracy: 0.9571 - val_loss: 0.5143 - val_accu
racy: 0.8926
Epoch 80/100
938/938 [==============================] - 7s 7ms/step - loss: 0.1438 - accuracy: 0.9567 - val_loss: 0.5414 - val_accu
racy: 0.8907
Epoch 81/100
938/938 [==============================] - 6s 7ms/step - loss: 0.1451 - accuracy: 0.9560 - val_loss: 0.5349 - val_accu
racy: 0.8920
Epoch 82/100
938/938 [==============================] - 7s 7ms/step - loss: 0.1416 - accuracy: 0.9574 - val_loss: 0.5471 - val_accu
racy: 0.8888
Epoch 83/100
938/938 [==============================] - 7s 7ms/step - loss: 0.1398 - accuracy: 0.9580 - val_loss: 0.5407 - val_accu
racy: 0.8878
Epoch 84/100
938/938 [==============================] - 7s 8ms/step - loss: 0.1380 - accuracy: 0.9590 - val_loss: 0.5621 - val_accu
racy: 0.8869
Epoch 85/100
938/938 [==============================] - 7s 7ms/step - loss: 0.1388 - accuracy: 0.9586 - val_loss: 0.5202 - val_accu
racy: 0.8889
Epoch 86/100
938/938 [==============================] - 7s 8ms/step - loss: 0.1348 - accuracy: 0.9588 - val_loss: 0.5143 - val_accu
racy: 0.8884
Epoch 87/100
938/938 [==============================] - 7s 8ms/step - loss: 0.1309 - accuracy: 0.9608 - val_loss: 0.5328 - val_accu
racy: 0.8888
Epoch 88/100
938/938 [==============================] - 7s 7ms/step - loss: 0.1358 - accuracy: 0.9594 - val_loss: 0.5551 - val_accu
racy: 0.8913
Epoch 89/100
938/938 [==============================] - 7s 8ms/step - loss: 0.1351 - accuracy: 0.9587 - val_loss: 0.5643 - val_accu
racy: 0.8899
Epoch 90/100
938/938 [==============================] - 6s 7ms/step - loss: 0.1385 - accuracy: 0.9580 - val_loss: 0.5411 - val_accu
racy: 0.8895
Epoch 91/100
938/938 [==============================] - 7s 8ms/step - loss: 0.1292 - accuracy: 0.9612 - val_loss: 0.5709 - val_accu
racy: 0.8888
Epoch 92/100
938/938 [==============================] - 6s 7ms/step - loss: 0.1340 - accuracy: 0.9596 - val_loss: 0.5622 - val_accu
racy: 0.8879
Epoch 93/100
938/938 [==============================] - 7s 7ms/step - loss: 0.1285 - accuracy: 0.9616 - val_loss: 0.5556 - val_accu
racy: 0.8899
Epoch 94/100
938/938 [==============================] - 6s 7ms/step - loss: 0.1263 - accuracy: 0.9618 - val_loss: 0.5495 - val_accu
racy: 0.8874
Epoch 95/100
938/938 [==============================] - 7s 8ms/step - loss: 0.1270 - accuracy: 0.9625 - val_loss: 0.5651 - val_accu
racy: 0.8868
Epoch 96/100
938/938 [==============================] - 7s 7ms/step - loss: 0.1295 - accuracy: 0.9613 - val_loss: 0.5833 - val_accu
```

```
racy: 0.8885
Epoch 97/100
938/938 [==============================] - 7s 7ms/step - loss: 0.1249 - accuracy: 0.9622 - val_loss: 0.5746 - val_accu
racy: 0.8874
Epoch 98/100
938/938 [==============================] - 7s 7ms/step - loss: 0.1252 - accuracy: 0.9631 - val_loss: 0.5694 - val_accu
racy: 0.8906
Epoch 99/100
938/938 [==============================] - 6s 7ms/step - loss: 0.1227 - accuracy: 0.9624 - val_loss: 0.6153 - val_accu
racy: 0.8871
Epoch 100/100
938/938 [==============================] - 7s 8ms/step - loss: 0.1205 - accuracy: 0.9635 - val_loss: 0.6077 - val_accu
racy: 0.8856
```

In [11]:
```
# This code trains your CNN model on the Fashion MNIST dataset for 10 epochs with a batch size of 64.
```

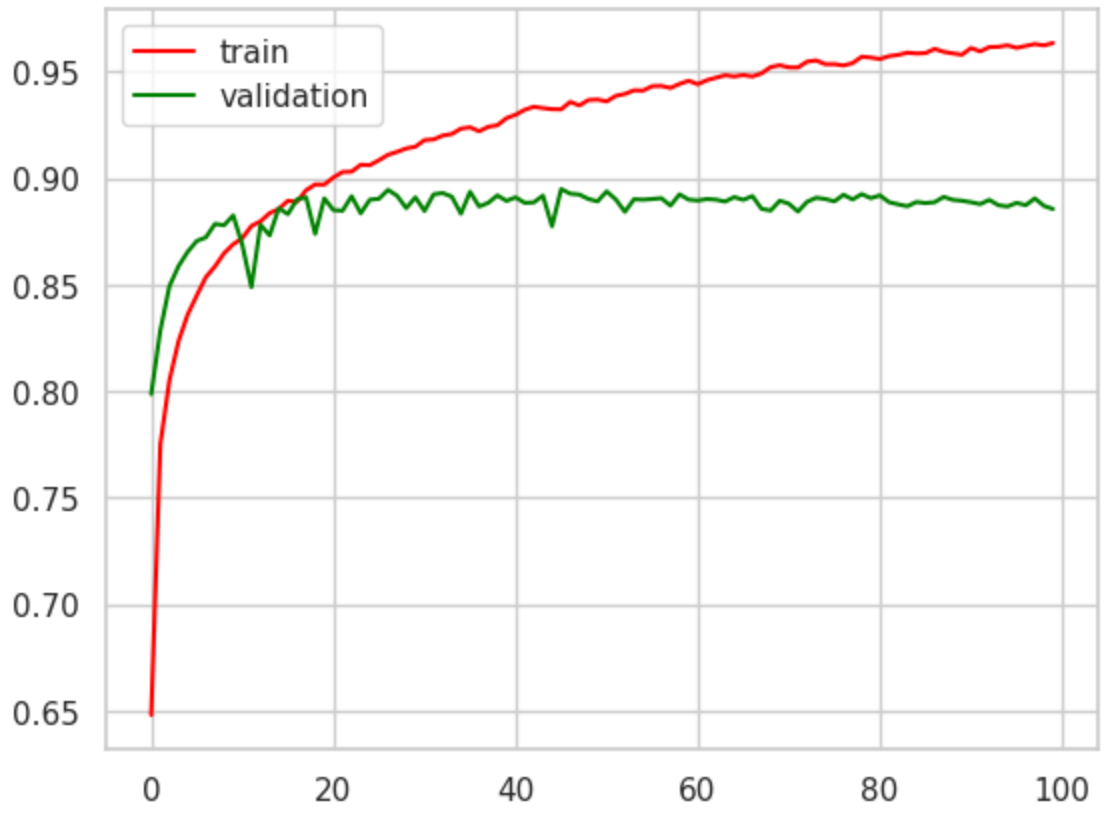
In [12]:
```
# Plotting the graph to visualize the loss over the epochs

plt.plot(history.history['loss'], color = 'red', label='train')
plt.plot(history.history['val_loss'], color = 'green', label='validation')
plt.legend()
plt.show()
```



In [13]:
```
# Plotting the graph to visualize the accuracy over the epochs

plt.plot(history.history['accuracy'], color = 'red', label='train')
plt.plot(history.history['val_accuracy'], color = 'green', label='validation')
plt.legend()
plt.show()
```



Conclusion

In this project, a Convolutional Neural Network (CNN) model was successfully constructed and trained to classify fashion items using the Fashion MNIST dataset. Fashion MNIST is a well-established benchmark for image classification, and the model was designed to identify and categorize clothing and accessory items across ten distinct classes.

The CNN model, developed with an efficient yet robust architecture, achieved a training accuracy of 96.35% and a validation accuracy of 88.56%. These results highlight the model's capacity to learn from the training data and generalize effectively to previously unseen fashion items.

The significance of these achievements extends beyond the numerical outcomes themselves. Image classification models have a wide array of

Introduction to the CIFAR-10 Dataset:

The CIFAR-10 dataset is a widely-used benchmark in the field of computer vision and deep learning. It consists of 60,000 32x32 color images in 10 different classes, with each class containing 6,000 images. The dataset is divided into a training set of 50,000 images and a testing set of 10,000 images. Each image belongs to one of the following ten classes:

1. Airplane
2. Automobile
3. Bird
4. Cat
5. Deer
6. Dog
7. Frog
8. Horse
9. Ship
10. Truck

The CIFAR-10 dataset is designed for image classification tasks, making it a valuable resource for developing and evaluating deep learning models. It poses unique challenges due to its small image size and the diversity of object classes it covers.

With this dataset, you can explore a wide range of computer vision tasks, from basic image classification to more complex challenges in object recognition and pattern analysis. It serves as an excellent starting point for experimenting with convolutional neural networks (CNNs) and other image processing techniques.

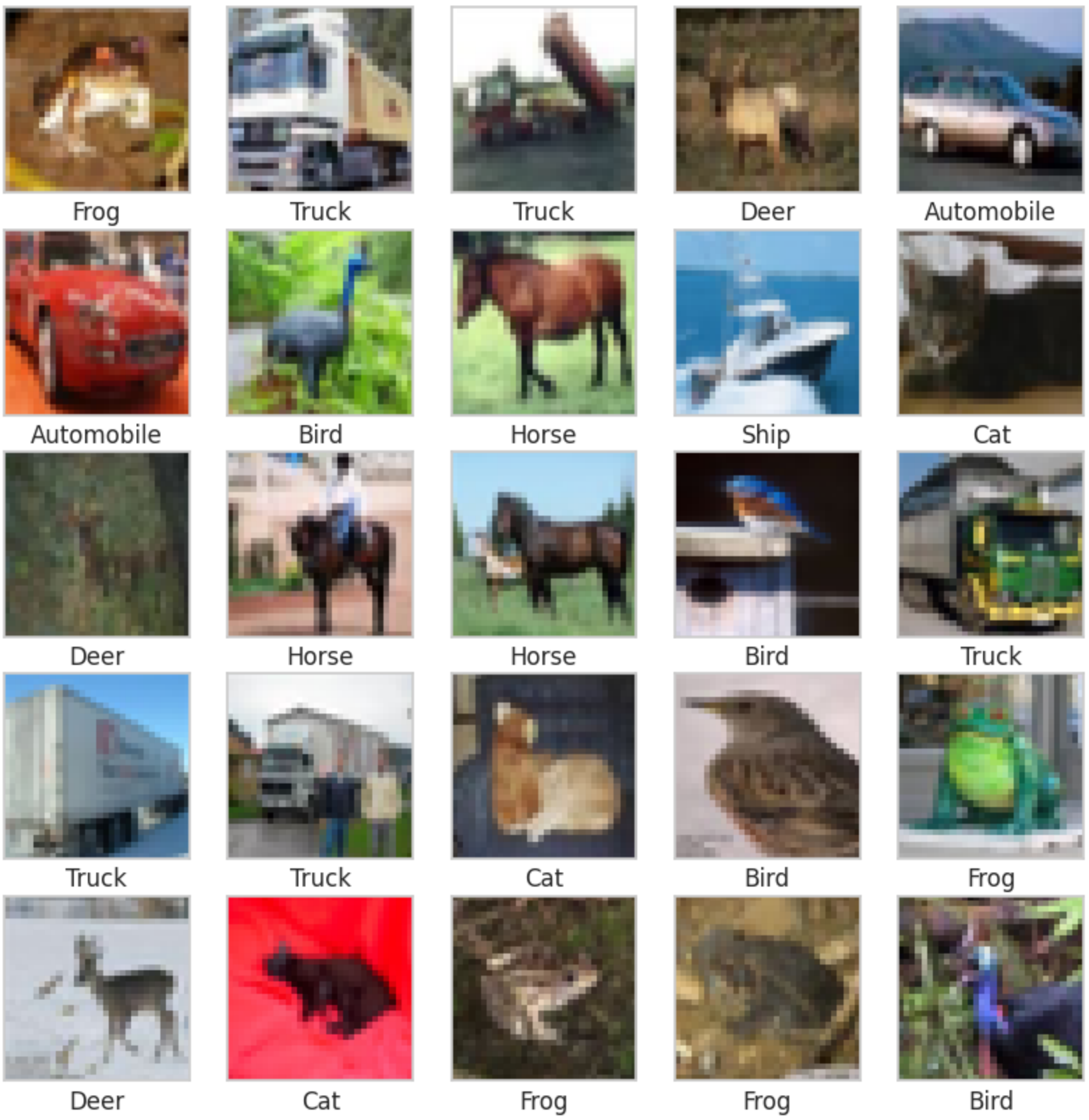
In [14]:
```
# Loading the CIFAR 10 dataset:

(x_train, y_train), (x_test, y_test) = keras.datasets.cifar10.load_data()
```

In [15]:
```
# Visualising the objects:

class_names = ["Airplane", "Automobile", "Bird", "Cat", "Deer", "Dog", "Frog", "Horse", "Ship", "Truck"]

plt.figure(figsize=(10,10))
for i in range(25):
  plt.subplot(5,5,i+1)
  plt.xticks([])
  plt.yticks([])
  plt.grid(False)
  plt.imshow(x_train[i])
  plt.xlabel(class_names[int(y_train[i])])
plt.show()
```



In [16]:
```
# Normalizing the pixels:

x_train = x_train/255.
x_test = x_test/255.
```

Normalizing the pixel values to a common range (usually between 0 and 1) is essential for efficient model training. It ensures that all features are on a similar scale, which helps the model converge faster and prevents any one feature from dominating the learning process.

In [17]:
```
# One hot encoding:

y_train = keras.utils.to_categorical(y_train, 10)
y_test = keras.utils.to_categorical(y_test, 10)
```

Depending on the model architecture, one-hot encoding of class labels might be necessary. It transforms categorical labels into a binary matrix format, making them suitable for classification tasks.

In [18]:
```
# Building a CNN architecture now:

from keras.models import Sequential
from keras.layers import Conv2D, MaxPooling2D, Flatten, Dense, Dropout

model = Sequential()

# Convolutional Layer 1
model.add(Conv2D(32, (3, 3), activation='relu', input_shape=(32, 32, 3)))
model.add(MaxPooling2D((2, 2)))

# Convolutional Layer 2
model.add(Conv2D(64, (3, 3), activation='relu'))
model.add(MaxPooling2D((2, 2)))

# Convolutional Layer 3
model.add(Conv2D(128, (3, 3), activation='relu'))
model.add(MaxPooling2D((2, 2)))

# Flatten the feature maps
model.add(Flatten())

# Fully Connected Layer 1
model.add(Dense(512, activation='relu', kernel_regularizer = keras.regularizers.l2(0.03)))
model.add(Dropout(0.3))  # Dropout for regularization
model.add(BatchNormalization())

# Fully Connected Layer 2
model.add(Dense(128, activation='relu', kernel_regularizer = keras.regularizers.l2(0.03)))
model.add(Dropout(0.3))
model.add(BatchNormalization())

# Fully Connected Layer 3
model.add(Dense(64, activation='relu', kernel_regularizer = keras.regularizers.l2(0.03)))
model.add(Dropout(0.3))
model.add(BatchNormalization())

# Fully Connected Layer 4
model.add(Dense(32, activation='relu', kernel_regularizer = keras.regularizers.l2(0.03)))
model.add(Dropout(0.3))
model.add(BatchNormalization())

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

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

Early stopping is a technique used to prevent overfitting in neural networks by monitoring a chosen validation metric and stopping the training process when this metric stops improving. In your case, you're concerned with the validation accuracy not improving much after a certain point. To implement early stopping, you can use the EarlyStopping callback provided by Keras. You can monitor the validation accuracy and specify a patience value, which determines how many epochs to wait for improvement before stopping the training.

In [19]:
```
from keras.callbacks import EarlyStopping

# Define early stopping
early_stopping = EarlyStopping(monitor='val_accuracy',  # Monitor validation accuracy
                               patience=3,            # Number of epochs with no improvement to wait before stopping
                               verbose=1,            # Display a message when stopping
                               restore_best_weights=True, start_from_epoch=70)  # Restore the model weights from the ep

# The training will automatically stop when validation accuracy stops improving.
history = model.fit(x_train, y_train, validation_data = (x_test, y_test), batch_size = 64, epochs = 500, callbacks=[ear
```

```
Epoch 1/500
782/782 [==============================] - 13s 12ms/step - loss: 4.6037 - accuracy: 0.2781 - val_loss: 1.9948 - val_ac
curacy: 0.3192
Epoch 2/500
782/782 [==============================] - 8s 10ms/step - loss: 1.7241 - accuracy: 0.4220 - val_loss: 1.6874 - val_acc
uracy: 0.4316
Epoch 3/500
782/782 [==============================] - 8s 10ms/step - loss: 1.6131 - accuracy: 0.4695 - val_loss: 1.5244 - val_acc
uracy: 0.4734
Epoch 4/500
782/782 [==============================] - 9s 11ms/step - loss: 1.5362 - accuracy: 0.5042 - val_loss: 1.5556 - val_acc
uracy: 0.4924
Epoch 5/500
782/782 [==============================] - 9s 12ms/step - loss: 1.4925 - accuracy: 0.5367 - val_loss: 1.4433 - val_acc
uracy: 0.5623
Epoch 6/500
782/782 [==============================] - 8s 10ms/step - loss: 1.4426 - accuracy: 0.5718 - val_loss: 1.3114 - val_acc
uracy: 0.6182
Epoch 7/500
782/782 [==============================] - 9s 11ms/step - loss: 1.3987 - accuracy: 0.5915 - val_loss: 1.3424 - val_acc
uracy: 0.5971
Epoch 8/500
782/782 [==============================] - 8s 11ms/step - loss: 1.3683 - accuracy: 0.6054 - val_loss: 1.3542 - val_acc
uracy: 0.5894
Epoch 9/500
782/782 [==============================] - 7s 9ms/step - loss: 1.3374 - accuracy: 0.6191 - val_loss: 1.3009 - val_accu
racy: 0.6076
Epoch 10/500
782/782 [==============================] - 8s 10ms/step - loss: 1.3200 - accuracy: 0.6261 - val_loss: 1.2712 - val_acc
uracy: 0.6224
Epoch 11/500
782/782 [==============================] - 8s 11ms/step - loss: 1.2913 - accuracy: 0.6369 - val_loss: 1.2993 - val_acc
uracy: 0.6275
Epoch 12/500
782/782 [==============================] - 8s 10ms/step - loss: 1.2676 - accuracy: 0.6487 - val_loss: 1.3758 - val_acc
uracy: 0.5992
Epoch 13/500
782/782 [==============================] - 8s 10ms/step - loss: 1.2510 - accuracy: 0.6528 - val_loss: 1.2149 - val_acc
uracy: 0.6501
Epoch 14/500
782/782 [==============================] - 9s 11ms/step - loss: 1.2333 - accuracy: 0.6621 - val_loss: 1.1449 - val_acc
uracy: 0.6692
Epoch 15/500
782/782 [==============================] - 8s 10ms/step - loss: 1.2209 - accuracy: 0.6670 - val_loss: 1.1749 - val_acc
uracy: 0.6661
Epoch 16/500
782/782 [==============================] - 8s 11ms/step - loss: 1.2060 - accuracy: 0.6704 - val_loss: 1.1865 - val_acc
uracy: 0.6595
Epoch 17/500
782/782 [==============================] - 8s 11ms/step - loss: 1.1873 - accuracy: 0.6773 - val_loss: 1.1722 - val_acc
uracy: 0.6730
Epoch 18/500
782/782 [==============================] - 8s 10ms/step - loss: 1.1757 - accuracy: 0.6845 - val_loss: 1.1929 - val_acc
uracy: 0.6665
Epoch 19/500
782/782 [==============================] - 8s 11ms/step - loss: 1.1649 - accuracy: 0.6856 - val_loss: 1.1740 - val_acc
uracy: 0.6765
Epoch 20/500
782/782 [==============================] - 9s 11ms/step - loss: 1.1629 - accuracy: 0.6865 - val_loss: 1.1682 - val_acc
uracy: 0.6707
Epoch 21/500
782/782 [==============================] - 8s 10ms/step - loss: 1.1439 - accuracy: 0.6971 - val_loss: 1.2053 - val_acc
uracy: 0.6578
Epoch 22/500
782/782 [==============================] - 8s 10ms/step - loss: 1.1367 - accuracy: 0.6982 - val_loss: 1.1395 - val_acc
uracy: 0.6800
Epoch 23/500
782/782 [==============================] - 8s 10ms/step - loss: 1.1215 - accuracy: 0.7031 - val_loss: 1.2470 - val_acc
uracy: 0.6565
Epoch 24/500
782/782 [==============================] - 8s 11ms/step - loss: 1.1221 - accuracy: 0.7058 - val_loss: 1.1965 - val_acc
uracy: 0.6648
Epoch 25/500
782/782 [==============================] - 8s 10ms/step - loss: 1.1063 - accuracy: 0.7106 - val_loss: 1.1522 - val_acc
uracy: 0.6793
Epoch 26/500
782/782 [==============================] - 8s 10ms/step - loss: 1.0990 - accuracy: 0.7147 - val_loss: 1.1634 - val_acc
uracy: 0.6899
Epoch 27/500
782/782 [==============================] - 8s 10ms/step - loss: 1.0895 - accuracy: 0.7157 - val_loss: 1.2053 - val_acc
uracy: 0.6647
Epoch 28/500
782/782 [==============================] - 8s 11ms/step - loss: 1.0948 - accuracy: 0.7165 - val_loss: 1.2233 - val_acc
uracy: 0.6661
Epoch 29/500
782/782 [==============================] - 8s 10ms/step - loss: 1.0739 - accuracy: 0.7226 - val_loss: 1.2736 - val_acc
uracy: 0.6570
Epoch 30/500
782/782 [==============================] - 8s 10ms/step - loss: 1.0649 - accuracy: 0.7232 - val_loss: 1.1767 - val_acc
uracy: 0.6793
Epoch 31/500
782/782 [==============================] - 8s 11ms/step - loss: 1.0592 - accuracy: 0.7285 - val_loss: 1.1744 - val_acc
uracy: 0.6860
Epoch 32/500
782/782 [==============================] - 8s 10ms/step - loss: 1.0430 - accuracy: 0.7321 - val_loss: 1.1610 - val_acc
```

```
uracy: 0.6892
Epoch 33/500
782/782 [==============================] - 8s 10ms/step - loss: 1.0533 - accuracy: 0.7325 - val_loss: 1.2924 - val_acc
uracy: 0.6657
Epoch 34/500
782/782 [==============================] - 8s 10ms/step - loss: 1.0498 - accuracy: 0.7313 - val_loss: 1.1181 - val_acc
uracy: 0.7042
Epoch 35/500
782/782 [==============================] - 8s 10ms/step - loss: 1.0360 - accuracy: 0.7386 - val_loss: 1.1312 - val_acc
uracy: 0.7047
Epoch 36/500
782/782 [==============================] - 8s 10ms/step - loss: 1.0341 - accuracy: 0.7377 - val_loss: 1.1185 - val_acc
uracy: 0.6954
Epoch 37/500
782/782 [==============================] - 9s 11ms/step - loss: 1.0233 - accuracy: 0.7419 - val_loss: 1.1824 - val_acc
uracy: 0.6840
Epoch 38/500
782/782 [==============================] - 9s 12ms/step - loss: 1.0187 - accuracy: 0.7419 - val_loss: 1.1614 - val_acc
uracy: 0.6934
Epoch 39/500
782/782 [==============================] - 8s 10ms/step - loss: 1.0110 - accuracy: 0.7462 - val_loss: 1.1629 - val_acc
uracy: 0.6975
Epoch 40/500
782/782 [==============================] - 9s 11ms/step - loss: 1.0076 - accuracy: 0.7497 - val_loss: 1.1293 - val_acc
uracy: 0.7012
Epoch 41/500
782/782 [==============================] - 9s 11ms/step - loss: 0.9942 - accuracy: 0.7534 - val_loss: 1.1522 - val_acc
uracy: 0.7036
Epoch 42/500
782/782 [==============================] - 8s 10ms/step - loss: 0.9955 - accuracy: 0.7537 - val_loss: 1.1767 - val_acc
uracy: 0.6878
Epoch 43/500
782/782 [==============================] - 8s 11ms/step - loss: 1.0021 - accuracy: 0.7527 - val_loss: 1.1940 - val_acc
uracy: 0.6821
Epoch 44/500
782/782 [==============================] - 8s 10ms/step - loss: 0.9915 - accuracy: 0.7541 - val_loss: 1.1605 - val_acc
uracy: 0.6779
Epoch 45/500
782/782 [==============================] - 8s 10ms/step - loss: 0.9839 - accuracy: 0.7580 - val_loss: 1.3073 - val_acc
uracy: 0.6699
Epoch 46/500
782/782 [==============================] - 8s 11ms/step - loss: 0.9855 - accuracy: 0.7598 - val_loss: 1.1097 - val_acc
uracy: 0.7129
Epoch 47/500
782/782 [==============================] - 8s 10ms/step - loss: 0.9835 - accuracy: 0.7601 - val_loss: 1.1609 - val_acc
uracy: 0.6970
Epoch 48/500
782/782 [==============================] - 8s 10ms/step - loss: 0.9643 - accuracy: 0.7638 - val_loss: 1.1363 - val_acc
uracy: 0.7127
Epoch 49/500
782/782 [==============================] - 9s 11ms/step - loss: 0.9625 - accuracy: 0.7666 - val_loss: 1.1439 - val_acc
uracy: 0.7112
Epoch 50/500
782/782 [==============================] - 8s 11ms/step - loss: 0.9587 - accuracy: 0.7684 - val_loss: 1.1643 - val_acc
uracy: 0.6886
Epoch 51/500
782/782 [==============================] - 8s 10ms/step - loss: 0.9484 - accuracy: 0.7698 - val_loss: 1.1369 - val_acc
uracy: 0.7116
Epoch 52/500
782/782 [==============================] - 8s 11ms/step - loss: 0.9636 - accuracy: 0.7680 - val_loss: 1.1431 - val_acc
uracy: 0.7081
Epoch 53/500
782/782 [==============================] - 8s 11ms/step - loss: 0.9515 - accuracy: 0.7721 - val_loss: 1.1509 - val_acc
uracy: 0.7030
Epoch 54/500
782/782 [==============================] - 7s 9ms/step - loss: 0.9387 - accuracy: 0.7732 - val_loss: 1.1598 - val_accu
racy: 0.7050
Epoch 55/500
782/782 [==============================] - 8s 10ms/step - loss: 0.9346 - accuracy: 0.7741 - val_loss: 1.1528 - val_acc
uracy: 0.7038
Epoch 56/500
782/782 [==============================] - 9s 11ms/step - loss: 0.9377 - accuracy: 0.7724 - val_loss: 1.1669 - val_acc
uracy: 0.7079
Epoch 57/500
782/782 [==============================] - 7s 10ms/step - loss: 0.9454 - accuracy: 0.7731 - val_loss: 1.1342 - val_acc
uracy: 0.7048
Epoch 58/500
782/782 [==============================] - 8s 10ms/step - loss: 0.9277 - accuracy: 0.7790 - val_loss: 1.1950 - val_acc
uracy: 0.6946
Epoch 59/500
782/782 [==============================] - 8s 10ms/step - loss: 0.9287 - accuracy: 0.7789 - val_loss: 1.2335 - val_acc
uracy: 0.6827
Epoch 60/500
782/782 [==============================] - 8s 10ms/step - loss: 0.9213 - accuracy: 0.7825 - val_loss: 1.1528 - val_acc
uracy: 0.6999
Epoch 61/500
782/782 [==============================] - 9s 11ms/step - loss: 0.9330 - accuracy: 0.7804 - val_loss: 1.2009 - val_acc
uracy: 0.7049
Epoch 62/500
782/782 [==============================] - 8s 11ms/step - loss: 0.9151 - accuracy: 0.7837 - val_loss: 1.1926 - val_acc
uracy: 0.7097
Epoch 63/500
782/782 [==============================] - 8s 10ms/step - loss: 0.9076 - accuracy: 0.7855 - val_loss: 1.1531 - val_acc
uracy: 0.7052
Epoch 64/500
782/782 [==============================] - 8s 10ms/step - loss: 0.9251 - accuracy: 0.7827 - val_loss: 1.2063 - val_acc
```

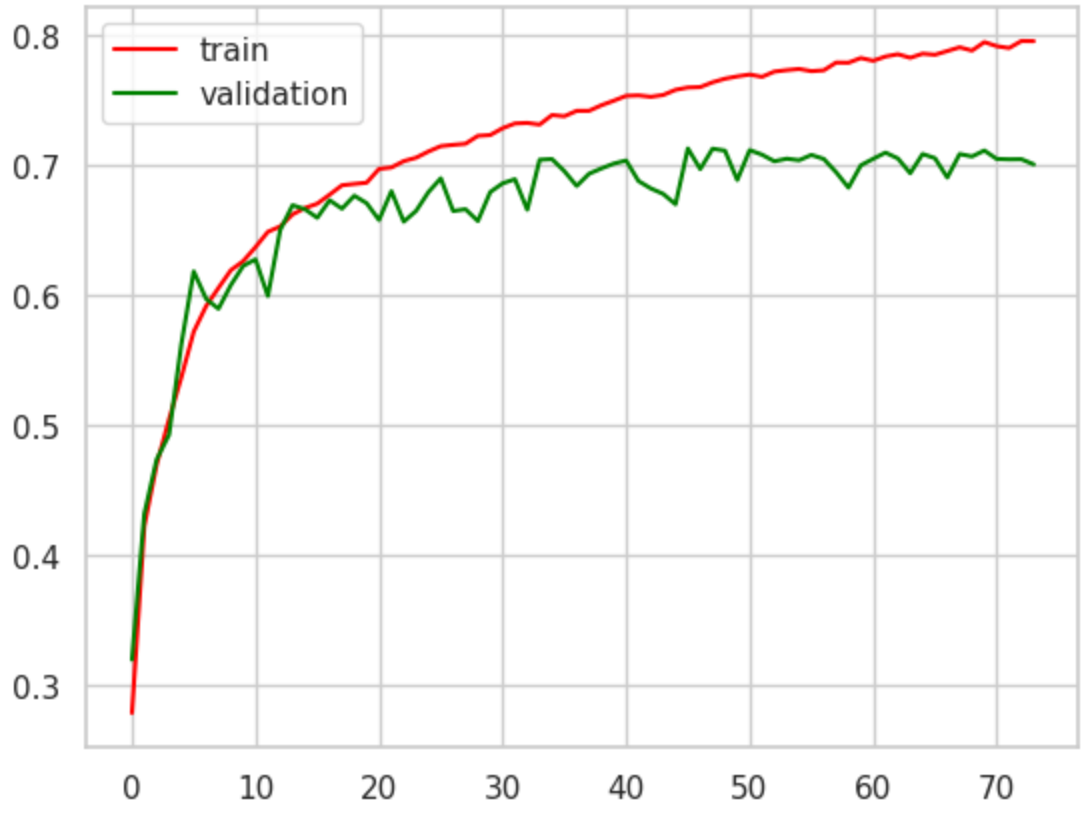
```
uracy: 0.6938
Epoch 65/500
782/782 [==============================] - 8s 11ms/step - loss: 0.9098 - accuracy: 0.7859 - val_loss: 1.1909 - val_acc
uracy: 0.7086
Epoch 66/500
782/782 [==============================] - 7s 9ms/step - loss: 0.9165 - accuracy: 0.7851 - val_loss: 1.1790 - val_accu
racy: 0.7054
Epoch 67/500
782/782 [==============================] - 8s 11ms/step - loss: 0.9044 - accuracy: 0.7879 - val_loss: 1.2177 - val_acc
uracy: 0.6905
Epoch 68/500
782/782 [==============================] - 8s 11ms/step - loss: 0.9068 - accuracy: 0.7908 - val_loss: 1.1732 - val_acc
uracy: 0.7086
Epoch 69/500
782/782 [==============================] - 8s 10ms/step - loss: 0.8958 - accuracy: 0.7883 - val_loss: 1.1774 - val_acc
uracy: 0.7067
Epoch 70/500
782/782 [==============================] - 8s 11ms/step - loss: 0.8899 - accuracy: 0.7948 - val_loss: 1.1496 - val_acc
uracy: 0.7113
Epoch 71/500
782/782 [==============================] - 9s 11ms/step - loss: 0.9017 - accuracy: 0.7917 - val_loss: 1.1784 - val_acc
uracy: 0.7048
Epoch 72/500
782/782 [==============================] - 8s 10ms/step - loss: 0.8937 - accuracy: 0.7902 - val_loss: 1.2060 - val_acc
uracy: 0.7045
Epoch 73/500
782/782 [==============================] - 8s 11ms/step - loss: 0.8900 - accuracy: 0.7956 - val_loss: 1.1951 - val_acc
uracy: 0.7046
Epoch 74/500
782/782 [==============================] - ETA: 0s - loss: 0.8898 - accuracy: 0.7955Restoring model weights from the e
nd of the best epoch: 71.
782/782 [==============================] - 9s 11ms/step - loss: 0.8898 - accuracy: 0.7955 - val_loss: 1.1993 - val_acc
uracy: 0.7006
Epoch 74: early stopping
```

In [20]:
```
# Plotting the graph to visualize the loss over the epochs

plt.plot(history.history['loss'], color = 'red', label='train')
plt.plot(history.history['val_loss'], color = 'green', label='validation')
plt.legend()
plt.show()
```

In [21]:
```
# Plotting the graph to visualize the accuracy over the epochs

plt.plot(history.history['accuracy'], color = 'red', label='train')
plt.plot(history.history['val_accuracy'], color = 'green', label='validation')
plt.legend()
plt.show()
```



In [22]:
```
# Using data augmentation techniques to reach satisfactory model accuracy:

from keras.preprocessing.image import ImageDataGenerator

datagen = ImageDataGenerator(rotation_range=15, width_shift_range=0.1, height_shift_range=0.1, horizontal_flip=True, zo

# Fitting the data generator on the training data
datagen.fit(x_train)

# creating a new model instance
model1 = Sequential()

# Convolutional Layer 1
model1.add(Conv2D(32, (3, 3), activation='relu', input_shape=(32, 32, 3)))
model1.add(MaxPooling2D((2, 2)))

# Convolutional Layer 2
model1.add(Conv2D(64, (3, 3), activation='relu'))
model1.add(MaxPooling2D((2, 2)))

# Convolutional Layer 3
model1.add(Conv2D(128, (3, 3), activation='relu'))
model1.add(MaxPooling2D((2, 2)))

# Flatten the feature maps
model1.add(Flatten())

# Fully Connected Layer 1
model1.add(Dense(512, activation='relu', kernel_regularizer = keras.regularizers.l2(0.03)))
model1.add(Dropout(0.3))  # Dropout for regularization
model1.add(BatchNormalization())

# Fully Connected Layer 2
model1.add(Dense(128, activation='relu', kernel_regularizer = keras.regularizers.l2(0.03)))
model1.add(Dropout(0.3))
model1.add(BatchNormalization())

# Fully Connected Layer 3
model1.add(Dense(64, activation='relu', kernel_regularizer = keras.regularizers.l2(0.03)))
model1.add(Dropout(0.3))
model1.add(BatchNormalization())

# Fully Connected Layer 4
model1.add(Dense(32, activation='relu', kernel_regularizer = keras.regularizers.l2(0.03)))
model1.add(Dropout(0.3))
model1.add(BatchNormalization())

# Output Layer
model1.add(Dense(10, activation='softmax'))

# Compile the model
model1.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])
```

In [23]:
```
# Define early stopping
early_stopping1 = EarlyStopping(monitor='val_accuracy',
                                patience=3,
                                verbose=1,
                                restore_best_weights=True, start_from_epoch=70)

# Training the model with data augmentation
history1 = model1.fit(datagen.flow(x_train, y_train, batch_size=64), epochs=500, validation_data=(x_test, y_test), call
```

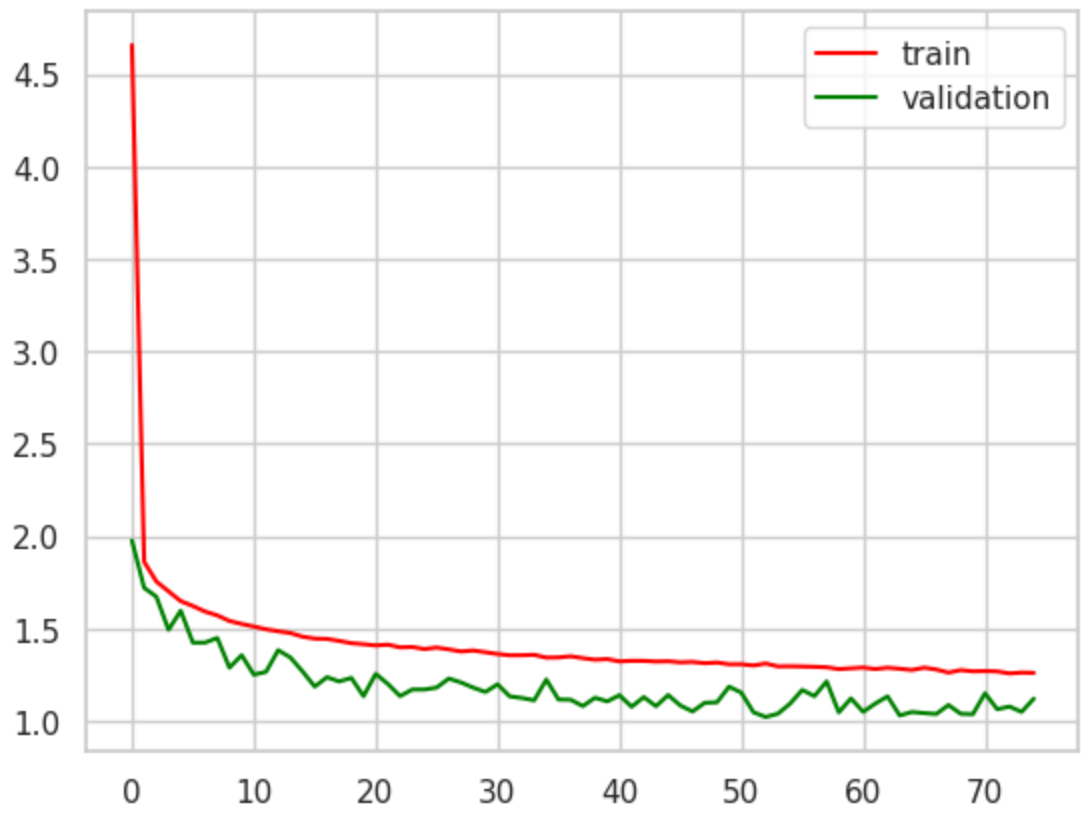
```
Epoch 1/500
782/782 [==============================] - 37s 42ms/step - loss: 4.6581 - accuracy: 0.2464 - val_loss: 1.9755 - val_ac
curacy: 0.3340
Epoch 2/500
782/782 [==============================] - 33s 42ms/step - loss: 1.8606 - accuracy: 0.3647 - val_loss: 1.7198 - val_ac
curacy: 0.4128
Epoch 3/500
782/782 [==============================] - 33s 42ms/step - loss: 1.7544 - accuracy: 0.4066 - val_loss: 1.6720 - val_ac
curacy: 0.4157
Epoch 4/500
782/782 [==============================] - 31s 40ms/step - loss: 1.7010 - accuracy: 0.4308 - val_loss: 1.4929 - val_ac
curacy: 0.5058
Epoch 5/500
782/782 [==============================] - 33s 42ms/step - loss: 1.6481 - accuracy: 0.4547 - val_loss: 1.5960 - val_ac
curacy: 0.4734
Epoch 6/500
782/782 [==============================] - 34s 43ms/step - loss: 1.6216 - accuracy: 0.4673 - val_loss: 1.4231 - val_ac
curacy: 0.5305
Epoch 7/500
782/782 [==============================] - 31s 40ms/step - loss: 1.5914 - accuracy: 0.4914 - val_loss: 1.4235 - val_ac
curacy: 0.5531
Epoch 8/500
782/782 [==============================] - 33s 42ms/step - loss: 1.5707 - accuracy: 0.5080 - val_loss: 1.4485 - val_ac
curacy: 0.5474
Epoch 9/500
782/782 [==============================] - 33s 42ms/step - loss: 1.5406 - accuracy: 0.5240 - val_loss: 1.2876 - val_ac
curacy: 0.6056
Epoch 10/500
782/782 [==============================] - 32s 42ms/step - loss: 1.5244 - accuracy: 0.5304 - val_loss: 1.3554 - val_ac
curacy: 0.5732
Epoch 11/500
782/782 [==============================] - 31s 40ms/step - loss: 1.5105 - accuracy: 0.5428 - val_loss: 1.2498 - val_ac
curacy: 0.6292
Epoch 12/500
782/782 [==============================] - 33s 43ms/step - loss: 1.4955 - accuracy: 0.5491 - val_loss: 1.2653 - val_ac
curacy: 0.6203
Epoch 13/500
782/782 [==============================] - 32s 41ms/step - loss: 1.4848 - accuracy: 0.5530 - val_loss: 1.3824 - val_ac
curacy: 0.5755
Epoch 14/500
782/782 [==============================] - 32s 41ms/step - loss: 1.4752 - accuracy: 0.5588 - val_loss: 1.3428 - val_ac
curacy: 0.6092
Epoch 15/500
782/782 [==============================] - 32s 41ms/step - loss: 1.4555 - accuracy: 0.5668 - val_loss: 1.2670 - val_ac
curacy: 0.6060
Epoch 16/500
782/782 [==============================] - 33s 42ms/step - loss: 1.4452 - accuracy: 0.5725 - val_loss: 1.1852 - val_ac
curacy: 0.6538
Epoch 17/500
782/782 [==============================] - 31s 40ms/step - loss: 1.4434 - accuracy: 0.5745 - val_loss: 1.2363 - val_ac
curacy: 0.6299
Epoch 18/500
782/782 [==============================] - 33s 42ms/step - loss: 1.4327 - accuracy: 0.5754 - val_loss: 1.2133 - val_ac
curacy: 0.6426
Epoch 19/500
782/782 [==============================] - 32s 41ms/step - loss: 1.4203 - accuracy: 0.5837 - val_loss: 1.2324 - val_ac
curacy: 0.6301
Epoch 20/500
782/782 [==============================] - 32s 41ms/step - loss: 1.4149 - accuracy: 0.5871 - val_loss: 1.1337 - val_ac
curacy: 0.6687
Epoch 21/500
782/782 [==============================] - 40s 51ms/step - loss: 1.4080 - accuracy: 0.5877 - val_loss: 1.2543 - val_ac
curacy: 0.6309
Epoch 22/500
782/782 [==============================] - 32s 41ms/step - loss: 1.4123 - accuracy: 0.5868 - val_loss: 1.1996 - val_ac
curacy: 0.6466
Epoch 23/500
782/782 [==============================] - 34s 43ms/step - loss: 1.3977 - accuracy: 0.5914 - val_loss: 1.1338 - val_ac
curacy: 0.6656
Epoch 24/500
782/782 [==============================] - 31s 40ms/step - loss: 1.3997 - accuracy: 0.5932 - val_loss: 1.1696 - val_ac
curacy: 0.6543
Epoch 25/500
782/782 [==============================] - 32s 41ms/step - loss: 1.3885 - accuracy: 0.5979 - val_loss: 1.1696 - val_ac
curacy: 0.6472
Epoch 26/500
782/782 [==============================] - 32s 40ms/step - loss: 1.3966 - accuracy: 0.5952 - val_loss: 1.1800 - val_ac
curacy: 0.6457
Epoch 27/500
782/782 [==============================] - 32s 41ms/step - loss: 1.3867 - accuracy: 0.5977 - val_loss: 1.2288 - val_ac
curacy: 0.6403
Epoch 28/500
782/782 [==============================] - 32s 40ms/step - loss: 1.3754 - accuracy: 0.6037 - val_loss: 1.2077 - val_ac
curacy: 0.6493
Epoch 29/500
782/782 [==============================] - 33s 42ms/step - loss: 1.3807 - accuracy: 0.6016 - val_loss: 1.1797 - val_ac
curacy: 0.6585
Epoch 30/500
782/782 [==============================] - 32s 40ms/step - loss: 1.3714 - accuracy: 0.6065 - val_loss: 1.1560 - val_ac
curacy: 0.6674
Epoch 31/500
782/782 [==============================] - 33s 42ms/step - loss: 1.3617 - accuracy: 0.6065 - val_loss: 1.1980 - val_ac
curacy: 0.6519
Epoch 32/500
782/782 [==============================] - 33s 42ms/step - loss: 1.3548 - accuracy: 0.6117 - val_loss: 1.1323 - val_ac
```

```
curacy: 0.6736
Epoch 33/500
782/782 [==============================] - 32s 40ms/step - loss: 1.3550 - accuracy: 0.6128 - val_loss: 1.1237 - val_ac
curacy: 0.6671
Epoch 34/500
782/782 [==============================] - 33s 42ms/step - loss: 1.3572 - accuracy: 0.6128 - val_loss: 1.1109 - val_ac
curacy: 0.6826
Epoch 35/500
782/782 [==============================] - 32s 40ms/step - loss: 1.3422 - accuracy: 0.6166 - val_loss: 1.2232 - val_ac
curacy: 0.6504
Epoch 36/500
782/782 [==============================] - 33s 42ms/step - loss: 1.3433 - accuracy: 0.6135 - val_loss: 1.1154 - val_ac
curacy: 0.6712
Epoch 37/500
782/782 [==============================] - 32s 41ms/step - loss: 1.3496 - accuracy: 0.6140 - val_loss: 1.1143 - val_ac
curacy: 0.6765
Epoch 38/500
782/782 [==============================] - 33s 43ms/step - loss: 1.3393 - accuracy: 0.6179 - val_loss: 1.0799 - val_ac
curacy: 0.6904
Epoch 39/500
782/782 [==============================] - 32s 41ms/step - loss: 1.3309 - accuracy: 0.6239 - val_loss: 1.1250 - val_ac
curacy: 0.6787
Epoch 40/500
782/782 [==============================] - 33s 42ms/step - loss: 1.3344 - accuracy: 0.6197 - val_loss: 1.1052 - val_ac
curacy: 0.6749
Epoch 41/500
782/782 [==============================] - 33s 42ms/step - loss: 1.3217 - accuracy: 0.6258 - val_loss: 1.1388 - val_ac
curacy: 0.6634
Epoch 42/500
782/782 [==============================] - 32s 40ms/step - loss: 1.3250 - accuracy: 0.6247 - val_loss: 1.0761 - val_ac
curacy: 0.7034
Epoch 43/500
782/782 [==============================] - 33s 43ms/step - loss: 1.3246 - accuracy: 0.6223 - val_loss: 1.1287 - val_ac
curacy: 0.6705
Epoch 44/500
782/782 [==============================] - 32s 41ms/step - loss: 1.3210 - accuracy: 0.6272 - val_loss: 1.0794 - val_ac
curacy: 0.6923
Epoch 45/500
782/782 [==============================] - 31s 40ms/step - loss: 1.3224 - accuracy: 0.6236 - val_loss: 1.1396 - val_ac
curacy: 0.6688
Epoch 46/500
782/782 [==============================] - 33s 42ms/step - loss: 1.3166 - accuracy: 0.6285 - val_loss: 1.0828 - val_ac
curacy: 0.6887
Epoch 47/500
782/782 [==============================] - 32s 41ms/step - loss: 1.3189 - accuracy: 0.6283 - val_loss: 1.0507 - val_ac
curacy: 0.6940
Epoch 48/500
782/782 [==============================] - 33s 42ms/step - loss: 1.3118 - accuracy: 0.6294 - val_loss: 1.0977 - val_ac
curacy: 0.6894
Epoch 49/500
782/782 [==============================] - 32s 40ms/step - loss: 1.3156 - accuracy: 0.6292 - val_loss: 1.1001 - val_ac
curacy: 0.6765
Epoch 50/500
782/782 [==============================] - 32s 41ms/step - loss: 1.3061 - accuracy: 0.6304 - val_loss: 1.1849 - val_ac
curacy: 0.6591
Epoch 51/500
782/782 [==============================] - 31s 40ms/step - loss: 1.3065 - accuracy: 0.6313 - val_loss: 1.1524 - val_ac
curacy: 0.6658
Epoch 52/500
782/782 [==============================] - 33s 42ms/step - loss: 1.2994 - accuracy: 0.6346 - val_loss: 1.0464 - val_ac
curacy: 0.6977
Epoch 53/500
782/782 [==============================] - 31s 40ms/step - loss: 1.3105 - accuracy: 0.6331 - val_loss: 1.0207 - val_ac
curacy: 0.7044
Epoch 54/500
782/782 [==============================] - 32s 41ms/step - loss: 1.2947 - accuracy: 0.6378 - val_loss: 1.0386 - val_ac
curacy: 0.7044
Epoch 55/500
782/782 [==============================] - 32s 41ms/step - loss: 1.2955 - accuracy: 0.6356 - val_loss: 1.0926 - val_ac
curacy: 0.6886
Epoch 56/500
782/782 [==============================] - 32s 41ms/step - loss: 1.2941 - accuracy: 0.6350 - val_loss: 1.1672 - val_ac
curacy: 0.6674
Epoch 57/500
782/782 [==============================] - 33s 43ms/step - loss: 1.2925 - accuracy: 0.6378 - val_loss: 1.1336 - val_ac
curacy: 0.6677
Epoch 58/500
782/782 [==============================] - 32s 40ms/step - loss: 1.2899 - accuracy: 0.6404 - val_loss: 1.2133 - val_ac
curacy: 0.6590
Epoch 59/500
782/782 [==============================] - 33s 42ms/step - loss: 1.2801 - accuracy: 0.6419 - val_loss: 1.0460 - val_ac
curacy: 0.7017
Epoch 60/500
782/782 [==============================] - 32s 41ms/step - loss: 1.2841 - accuracy: 0.6413 - val_loss: 1.1213 - val_ac
curacy: 0.6796
Epoch 61/500
782/782 [==============================] - 31s 40ms/step - loss: 1.2884 - accuracy: 0.6402 - val_loss: 1.0486 - val_ac
curacy: 0.6985
Epoch 62/500
782/782 [==============================] - 33s 42ms/step - loss: 1.2803 - accuracy: 0.6430 - val_loss: 1.0936 - val_ac
curacy: 0.6956
Epoch 63/500
782/782 [==============================] - 31s 40ms/step - loss: 1.2874 - accuracy: 0.6399 - val_loss: 1.1324 - val_ac
curacy: 0.6872
Epoch 64/500
782/782 [==============================] - 33s 42ms/step - loss: 1.2820 - accuracy: 0.6409 - val_loss: 1.0291 - val_ac
```

```
curacy: 0.7077
Epoch 65/500
782/782 [==============================] - 33s 42ms/step - loss: 1.2751 - accuracy: 0.6438 - val_loss: 1.0479 - val_ac
curacy: 0.7052
Epoch 66/500
782/782 [==============================] - 32s 41ms/step - loss: 1.2877 - accuracy: 0.6422 - val_loss: 1.0418 - val_ac
curacy: 0.7038
Epoch 67/500
782/782 [==============================] - 33s 42ms/step - loss: 1.2779 - accuracy: 0.6461 - val_loss: 1.0370 - val_ac
curacy: 0.7088
Epoch 68/500
782/782 [==============================] - 31s 40ms/step - loss: 1.2611 - accuracy: 0.6493 - val_loss: 1.0856 - val_ac
curacy: 0.6923
Epoch 69/500
782/782 [==============================] - 33s 43ms/step - loss: 1.2746 - accuracy: 0.6452 - val_loss: 1.0383 - val_ac
curacy: 0.7045
Epoch 70/500
782/782 [==============================] - 32s 41ms/step - loss: 1.2680 - accuracy: 0.6504 - val_loss: 1.0359 - val_ac
curacy: 0.7103
Epoch 71/500
782/782 [==============================] - 33s 42ms/step - loss: 1.2694 - accuracy: 0.6488 - val_loss: 1.1491 - val_ac
curacy: 0.6786
Epoch 72/500
782/782 [==============================] - 32s 41ms/step - loss: 1.2675 - accuracy: 0.6489 - val_loss: 1.0627 - val_ac
curacy: 0.6991
Epoch 73/500
782/782 [==============================] - 32s 41ms/step - loss: 1.2566 - accuracy: 0.6517 - val_loss: 1.0773 - val_ac
curacy: 0.6931
Epoch 74/500
782/782 [==============================] - 32s 41ms/step - loss: 1.2613 - accuracy: 0.6504 - val_loss: 1.0484 - val_ac
curacy: 0.6986
Epoch 75/500
782/782 [==============================] - ETA: 0s - loss: 1.2595 - accuracy: 0.6511Restoring model weights from the e
nd of the best epoch: 72.
782/782 [==============================] - 34s 44ms/step - loss: 1.2595 - accuracy: 0.6511 - val_loss: 1.1200 - val_ac
curacy: 0.6921
Epoch 75: early stopping
```

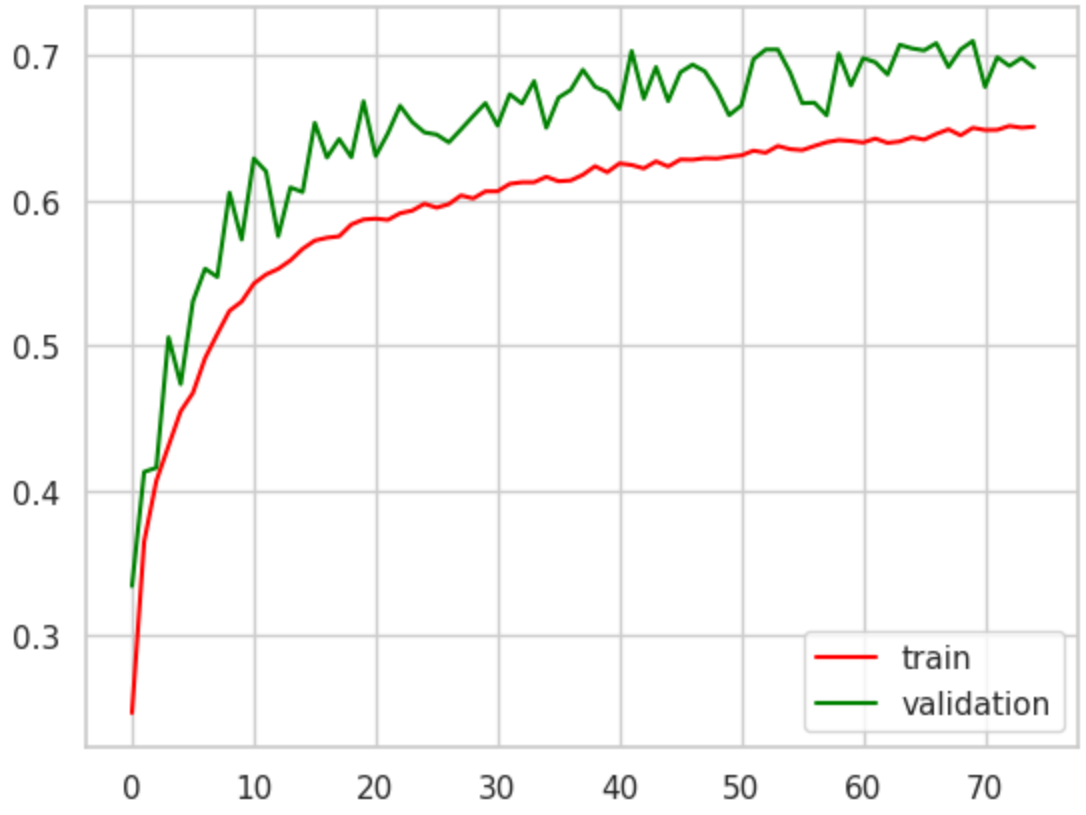
In [24]:
```
# Plotting the graph to visualize the loss over the epochs

plt.plot(history1.history['loss'], color = 'red', label='train')
plt.plot(history1.history['val_loss'], color = 'green', label='validation')
plt.legend()
plt.show()
```



In [25]:
```
# Plotting the graph to visualize the accuracy over the epochs

plt.plot(history1.history['accuracy'], color = 'red', label='train')
plt.plot(history1.history['val_accuracy'], color = 'green', label='validation')
plt.legend()
plt.show()
```



Conclusion

The CNN model is built on the CIFAR 10 dataset and the model that can be relied on is the CNN model without data augmentation with a train accuracy of 79.55% and validation accuracy of 70.06%.

The skills and insights gained during this project extend to various domains, from autonomous vehicles to medical imaging and beyond. The ability to train CNNs for image classification has far-reaching implications.