In [1]:

```
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
import keras
import tensorflow as tf

from tensorflow import keras
from keras.models import Sequential
from tensorflow.keras.layers import Input, Conv2D, Dense, Flatten, Dropout
from tensorflow.keras.layers import GlobalMaxPooling2D, MaxPooling2D
from tensorflow.keras.layers import BatchNormalization
from tensorflow.keras.models import Model
from tensorflow.keras import regularizers, optimizers
from tensorflow.keras.utils import to_categorical
from sklearn.metrics import accuracy_score

import warnings
warnings.filterwarnings('ignore')

print("Tensorflow version:",tf.__version__)
print("Keras version:",keras.__version__)
```

```
2022-12-02 15:51:46.728393: I tensorflow/core/platform/cpu_feature_guard.cc:193] This TensorFlow binary is optimized
with oneAPI Deep Neural Network Library (oneDNN) to use the following CPU instructions in performance-critical operat
ions:  AVX2 FMA
To enable them in other operations, rebuild TensorFlow with the appropriate compiler flags.
Tensorflow version: 2.11.0
Keras version: 2.11.0
```

In [2]:

```
from tensorflow.keras.datasets import cifar10
(X_train, Y_train), (X_test, Y_test) = cifar10.load_data()
```

In [3]:

```
# Normalizing
X_train = X_train/255
X_test = X_test/255
# One-Hot-Encoding
Y_train_en = to_categorical(Y_train,10)
Y_test_en = to_categorical(Y_test,10)
```

In [5]:

```
model = Sequential()
model.add(Conv2D(32,(4,4),input_shape = (32,32,3),activation='relu'))
model.add(MaxPooling2D(pool_size = (2,2)))
model.add(Conv2D(32,(4,4),input_shape = (32,32,3),activation='relu'))
model.add(MaxPooling2D(pool_size = (2,2)))
model.add(Flatten())
model.add(Dense(128, activation ='relu'))
model.add(Dense(10, activation ='softmax'))
model.compile(loss ='categorical_crossentropy', optimizer ='adam', metrics =['accuracy'])
```

```
2022-12-02 15:52:37.810351: I tensorflow/core/platform/cpu_feature_guard.cc:193] This TensorFlow binary is optimized
with oneAPI Deep Neural Network Library (oneDNN) to use the following CPU instructions in performance-critical operat
ions:  AVX2 FMA
To enable them in other operations, rebuild TensorFlow with the appropriate compiler flags.
```

In [6]:

```
model.summary()
history = model.fit(X_train, Y_train_en, epochs = 20, verbose=1,validation_data=(X_test,Y_test_en))
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #   
=================================================================
 conv2d (Conv2D)             (None, 29, 29, 32)        1568      
                                                                 
 max_pooling2d (MaxPooling2D  (None, 14, 14, 32)       0         
 )                                                               
                                                                 
 conv2d_1 (Conv2D)           (None, 11, 11, 32)        16416     
                                                                 
 max_pooling2d_1 (MaxPooling  (None, 5, 5, 32)         0         
 2D)                                                             
                                                                 
 flatten (Flatten)           (None, 800)               0         
                                                                 
 dense (Dense)               (None, 128)               102528    
                                                                 
 dense_1 (Dense)             (None, 10)                1290      
                                                                 
=================================================================
Total params: 121,802
Trainable params: 121,802
Non-trainable params: 0
_________________________________________________________________
Epoch 1/20
1563/1563 [==============================] - 28s 17ms/step - loss: 1.5370 - accuracy: 0.4424 - val_loss: 1.4239 - val
_accuracy: 0.4938
Epoch 2/20
1563/1563 [==============================] - 27s 17ms/step - loss: 1.2234 - accuracy: 0.5696 - val_loss: 1.1969 - val
_accuracy: 0.5801
Epoch 3/20
1563/1563 [==============================] - 29s 19ms/step - loss: 1.0867 - accuracy: 0.6179 - val_loss: 1.0858 - val
_accuracy: 0.6194
Epoch 4/20
1563/1563 [==============================] - 29s 19ms/step - loss: 0.9957 - accuracy: 0.6513 - val_loss: 1.0308 - val
_accuracy: 0.6424
Epoch 5/20
1563/1563 [==============================] - 25s 16ms/step - loss: 0.9211 - accuracy: 0.6781 - val_loss: 1.0586 - val
_accuracy: 0.6285
Epoch 6/20
1563/1563 [==============================] - 26s 17ms/step - loss: 0.8511 - accuracy: 0.7032 - val_loss: 0.9536 - val
_accuracy: 0.6705
Epoch 7/20
1563/1563 [==============================] - 26s 17ms/step - loss: 0.7961 - accuracy: 0.7229 - val_loss: 0.9651 - val
_accuracy: 0.6704
Epoch 8/20
1563/1563 [==============================] - 25s 16ms/step - loss: 0.7549 - accuracy: 0.7362 - val_loss: 0.9691 - val
_accuracy: 0.6709
Epoch 9/20
1563/1563 [==============================] - 25s 16ms/step - loss: 0.7116 - accuracy: 0.7510 - val_loss: 0.9797 - val
_accuracy: 0.6739
Epoch 10/20
1563/1563 [==============================] - 25s 16ms/step - loss: 0.6704 - accuracy: 0.7637 - val_loss: 0.9674 - val
_accuracy: 0.6806
Epoch 11/20
1563/1563 [==============================] - 25s 16ms/step - loss: 0.6310 - accuracy: 0.7785 - val_loss: 0.9342 - val
_accuracy: 0.6905
Epoch 12/20
1563/1563 [==============================] - 25s 16ms/step - loss: 0.5947 - accuracy: 0.7898 - val_loss: 1.0009 - val
_accuracy: 0.6798
Epoch 13/20
1563/1563 [==============================] - 25s 16ms/step - loss: 0.5656 - accuracy: 0.8018 - val_loss: 0.9938 - val
_accuracy: 0.6821
Epoch 14/20
1563/1563 [==============================] - 25s 16ms/step - loss: 0.5296 - accuracy: 0.8136 - val_loss: 1.0576 - val
_accuracy: 0.6768
Epoch 15/20
1563/1563 [==============================] - 25s 16ms/step - loss: 0.5060 - accuracy: 0.8213 - val_loss: 1.0897 - val
_accuracy: 0.6712
Epoch 16/20
1563/1563 [==============================] - 25s 16ms/step - loss: 0.4770 - accuracy: 0.8330 - val_loss: 1.1118 - val
_accuracy: 0.6764
Epoch 17/20
1563/1563 [==============================] - 26s 17ms/step - loss: 0.4498 - accuracy: 0.8413 - val_loss: 1.1038 - val
_accuracy: 0.6814
Epoch 18/20
1563/1563 [==============================] - 26s 16ms/step - loss: 0.4323 - accuracy: 0.8472 - val_loss: 1.1496 - val
_accuracy: 0.6805
Epoch 19/20
1563/1563 [==============================] - 26s 17ms/step - loss: 0.4102 - accuracy: 0.8513 - val_loss: 1.2052 - val
_accuracy: 0.6785
Epoch 20/20
1563/1563 [==============================] - 25s 16ms/step - loss: 0.3830 - accuracy: 0.8623 - val_loss: 1.2357 - val
_accuracy: 0.6731
```

In [8]:

```
# Model_1 with Dropouts
model_1 = Sequential()
model_1.add(Conv2D(64,(4,4),input_shape=(32,32,3),activation='relu'))
```

```
model_1.add(MaxPooling2D(pool_size=(2,2)))
model_1.add(Dropout(0.5))
model_1.add(Conv2D(64,(4,4),input_shape=(32,32,3),activation='relu'))
model_1.add(MaxPooling2D(pool_size=(2,2)))
model_1.add(Dropout(0.25))
model_1.add(Flatten())
model_1.add(Dense(256,activation='relu'))
model_1.add(Dense(10,activation='softmax'))
model_1.compile(loss='categorical_crossentropy',optimizer='adam',metrics=['accuracy'])
```

In [9]:

```
model_1.summary()
history = model_1.fit(X_train, Y_train_en, epochs = 20, verbose=1,validation_data=(X_test,Y_test_en))
```

```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #   
=================================================================
 conv2d_2 (Conv2D)           (None, 29, 29, 64)        3136      
                                                                 
 max_pooling2d_2 (MaxPooling  (None, 14, 14, 64)       0         
 2D)                                                             
                                                                 
 dropout (Dropout)           (None, 14, 14, 64)        0         
                                                                 
 conv2d_3 (Conv2D)           (None, 11, 11, 64)        65600     
                                                                 
 max_pooling2d_3 (MaxPooling  (None, 5, 5, 64)         0         
 2D)                                                             
                                                                 
 dropout_1 (Dropout)         (None, 5, 5, 64)          0         
                                                                 
 flatten_1 (Flatten)         (None, 1600)              0         
                                                                 
 dense_2 (Dense)             (None, 256)               409856    
                                                                 
 dense_3 (Dense)             (None, 10)                2570      
                                                                 
=================================================================
Total params: 481,162
Trainable params: 481,162
Non-trainable params: 0
_________________________________________________________________
Epoch 1/20
1563/1563 [==============================] - 69s 43ms/step - loss: 1.5568 - accuracy: 0.4339 - val_loss: 1.2736 - val
_accuracy: 0.5484
Epoch 2/20
1563/1563 [==============================] - 66s 42ms/step - loss: 1.2387 - accuracy: 0.5617 - val_loss: 1.1652 - val
_accuracy: 0.5997
Epoch 3/20
1563/1563 [==============================] - 64s 41ms/step - loss: 1.1047 - accuracy: 0.6093 - val_loss: 1.0422 - val
_accuracy: 0.6461
Epoch 4/20
1563/1563 [==============================] - 66s 42ms/step - loss: 1.0124 - accuracy: 0.6449 - val_loss: 0.9557 - val
_accuracy: 0.6714
Epoch 5/20
1563/1563 [==============================] - 66s 42ms/step - loss: 0.9401 - accuracy: 0.6703 - val_loss: 0.9604 - val
_accuracy: 0.6658
Epoch 6/20
1563/1563 [==============================] - 67s 43ms/step - loss: 0.8909 - accuracy: 0.6873 - val_loss: 0.9027 - val
_accuracy: 0.6909
Epoch 7/20
1563/1563 [==============================] - 66s 42ms/step - loss: 0.8415 - accuracy: 0.7044 - val_loss: 0.8795 - val
_accuracy: 0.6951
Epoch 8/20
1563/1563 [==============================] - 64s 41ms/step - loss: 0.8009 - accuracy: 0.7176 - val_loss: 0.8819 - val
_accuracy: 0.6965
Epoch 9/20
1563/1563 [==============================] - 64s 41ms/step - loss: 0.7635 - accuracy: 0.7322 - val_loss: 0.8665 - val
_accuracy: 0.7062
Epoch 10/20
1563/1563 [==============================] - 64s 41ms/step - loss: 0.7369 - accuracy: 0.7415 - val_loss: 0.8483 - val
_accuracy: 0.7133
Epoch 11/20
1563/1563 [==============================] - 64s 41ms/step - loss: 0.7031 - accuracy: 0.7518 - val_loss: 0.8621 - val
_accuracy: 0.7094
Epoch 12/20
1563/1563 [==============================] - 64s 41ms/step - loss: 0.6853 - accuracy: 0.7590 - val_loss: 0.8748 - val
_accuracy: 0.7042
Epoch 13/20
1563/1563 [==============================] - 64s 41ms/step - loss: 0.6600 - accuracy: 0.7684 - val_loss: 0.8997 - val
_accuracy: 0.6993
Epoch 14/20
1563/1563 [==============================] - 68s 44ms/step - loss: 0.6426 - accuracy: 0.7738 - val_loss: 0.9083 - val
_accuracy: 0.7016
Epoch 15/20
1563/1563 [==============================] - 71s 46ms/step - loss: 0.6205 - accuracy: 0.7808 - val_loss: 0.9225 - val
_accuracy: 0.6954
Epoch 16/20
1563/1563 [==============================] - 66s 42ms/step - loss: 0.6088 - accuracy: 0.7870 - val_loss: 0.8956 - val
_accuracy: 0.7091
Epoch 17/20
1563/1563 [==============================] - 59s 38ms/step - loss: 0.5884 - accuracy: 0.7925 - val_loss: 0.8884 - val
_accuracy: 0.7121
Epoch 18/20
1563/1563 [==============================] - 64s 41ms/step - loss: 0.5750 - accuracy: 0.7971 - val_loss: 0.8902 - val
_accuracy: 0.7139
Epoch 19/20
1563/1563 [==============================] - 66s 42ms/step - loss: 0.5602 - accuracy: 0.8016 - val_loss: 0.9093 - val
_accuracy: 0.7037
Epoch 20/20
1563/1563 [==============================] - 61s 39ms/step - loss: 0.5531 - accuracy: 0.8066 - val_loss: 0.9068 - val
_accuracy: 0.7164
```

In [ ]: