In [1]:

```
"""
Author: Amruth Karun M V
Date: 12-Oct-2021
"""

import os
import pandas as pd
import numpy as np
import zipfile
import tensorflow as tf
from tensorflow import keras
from tensorflow.keras.optimizers import Adam
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import (
    Dense, Activation, Dropout, Flatten,
    Conv2D, MaxPooling2D, BatchNormalization
)

from sklearn import metrics
import matplotlib.pyplot as plt
%matplotlib inline

TRAIN_PATH = "../input/cifar10/cifar10/train"
TEST_PATH = "../input/cifar10/cifar10/test"
EPOCHS = 100
BATCH_SIZE = 256
LEARNING_RATE = 0.001


def load_data(input_path, shuffle=False):
    """
    Loads input data fro directory
    Arguments:
        input_path -- input data path
        shuffle    -- whether data needs to be shuffled or not
    Returns: Data generator
    """

    data_generator = keras.preprocessing.image.ImageDataGenerator()
    data_generator = data_generator.flow_from_directory(directory=input_path, ta
rget_size=(224,224), shuffle=shuffle)

    return data_generator
```

```
def load_model():
    """
    Creates a keras VGG-16 model
    Arguments: None
    Returns: VGG-16 Model
    """

    model = Sequential()
    model.add(Conv2D(input_shape=(224,224,3),filters=64,kernel_size=(3,3),paddin
g="same", activation="relu"))
    model.add(Conv2D(filters=64,kernel_size=(3,3),padding="same", activation="re
lu"))
    model.add(MaxPooling2D(pool_size=(2,2),strides=(2,2)))

    model.add(Conv2D(filters=128, kernel_size=(3,3), padding="same", activation=
"relu"))
    model.add(Conv2D(filters=128, kernel_size=(3,3), padding="same", activation=
"relu"))
    model.add(MaxPooling2D(pool_size=(2,2),strides=(2,2)))

    model.add(Conv2D(filters=256, kernel_size=(3,3), padding="same", activation=
"relu"))
    model.add(Conv2D(filters=256, kernel_size=(3,3), padding="same", activation=
"relu"))
    model.add(Conv2D(filters=256, kernel_size=(3,3), padding="same", activation=
"relu"))
    model.add(MaxPooling2D(pool_size=(2,2),strides=(2,2)))

    model.add(Conv2D(filters=512, kernel_size=(3,3), padding="same", activation=
"relu"))
    model.add(Conv2D(filters=512, kernel_size=(3,3), padding="same", activation=
"relu"))
    model.add(Conv2D(filters=512, kernel_size=(3,3), padding="same", activation=
"relu"))
    model.add(MaxPooling2D(pool_size=(2,2),strides=(2,2)))

    model.add(Conv2D(filters=512, kernel_size=(3,3), padding="same", activation=
"relu"))
    model.add(Conv2D(filters=512, kernel_size=(3,3), padding="same", activation=
"relu"))
    model.add(Conv2D(filters=512, kernel_size=(3,3), padding="same", activation=
"relu"))
    model.add(MaxPooling2D(pool_size=(2,2),strides=(2,2)))
```

```
    model.add(Flatten())
    model.add(Dense(units=4096,activation="relu"))
    model.add(Dense(units=4096,activation="relu"))
    model.add(Dense(units=10, activation="softmax"))
    model.summary()

    opt = Adam(lr=LEARNING_RATE)
    model.compile(loss = keras.losses.categorical_crossentropy, optimizer=opt,
    metrics=['accuracy'])

    return model


def plot_curves(history):
    """
    Plots loss and accuracy and loss plots for
    training and validation datasets
    Arguments:
        history -- training history
    Returns: None
    """

    plt.plot(history.history['loss'], color='b', label="Training loss")
    plt.plot(history.history['val_loss'], color='r', label="validation loss")
    plt.xticks(np.arange(1, EPOCHS, 10))
    plt.legend()
    plt.title('Training Loss VS Validation Loss')
    plt.show()

    plt.plot(history.history['accuracy'], color='b', label="Training accuracy")
    plt.plot(history.history['val_accuracy'], color='r',label="Validation accura
cy")
    plt.xticks(np.arange(1, EPOCHS, 10))
    plt.title('Training Accuracy VS Validation Accuracy')
    plt.legend()
    plt.show()


def get_confusion_matrix(model, input_path):
    """
    Calculates the confusion matrix
    for the input data
    Arguments:
        model       -- trained model
```

```
        input_path  -- input data path
    Returns: None
    """

    data_generator = load_data(input_path)
    predictions = model.predict(data_generator, BATCH_SIZE)
    y_pred = np.argmax(predictions, axis=1)
    y_true = data_generator.classes
    class_names = ['airplane', 'automobile', 'bird', 'cat', 'deer', 'dog', 'fro
g', 'horse', 'ship', 'truck']

    print("Accuracy score = ", metrics.accuracy_score(y_true, y_pred))
    cm = metrics.confusion_matrix(y_true, y_pred)
    metrics.ConfusionMatrixDisplay(cm, display_labels=class_names).plot(cmap=plt
.cm.Blues,
                                                                        xticks_ro
tation='vertical')
    plt.show()


def train_model():
    """
    Trains VGG-16 model and saves the
    trained weights to an H5 file.
    Arguments: None
    Returns: None
    """

    train_generator = load_data(TRAIN_PATH, True)
    val_generator = load_data(TEST_PATH, True)

    # Loads VGG-16 model
    model = load_model()
    earlystop = keras.callbacks.EarlyStopping(patience=20)
    callbacks = [earlystop]

    history = model.fit(
        train_generator,
        batch_size=BATCH_SIZE,
        epochs=EPOCHS,
        validation_data=val_generator,
        validation_steps=val_generator.samples//BATCH_SIZE,
        steps_per_epoch=train_generator.samples//BATCH_SIZE,
        callbacks=callbacks)
```

```
    plot_curves(history)
    model.save_weights("model_vgg16.h5")
    print("Model saved successfully!")

    return model
```

```
2021-10-12 13:52:31.924087: I tensorflow/stream_executor/platform/d
efault/dso_loader.cc:49] Successfully opened dynamic library libcud
art.so.11.0
```

In [2]:

```
# Train the model
model = train_model()
print("Confusion matrix for train data: ")
get_confusion_matrix(model, TRAIN_PATH)
print("Confusion matrix for val data: ")
get_confusion_matrix(model, TEST_PATH)
```

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 224, 224, 64)      1792
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 224, 224, 64)      36928
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 112, 112, 64)      0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 112, 112, 128)     73856
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 112, 112, 128)     147584
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 56, 56, 128)       0
_________________________________________________________________
conv2d_4 (Conv2D)            (None, 56, 56, 256)       295168
_________________________________________________________________
conv2d_5 (Conv2D)            (None, 56, 56, 256)       590080
_________________________________________________________________
conv2d_6 (Conv2D)            (None, 56, 56, 256)       590080
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 28, 28, 256)       0
_________________________________________________________________
conv2d_7 (Conv2D)            (None, 28, 28, 512)       1180160
_________________________________________________________________
conv2d_8 (Conv2D)            (None, 28, 28, 512)       2359808
_________________________________________________________________
conv2d_9 (Conv2D)            (None, 28, 28, 512)       2359808
_________________________________________________________________
max_pooling2d_3 (MaxPooling2 (None, 14, 14, 512)       0
_________________________________________________________________
conv2d_10 (Conv2D)           (None, 14, 14, 512)       2359808
_________________________________________________________________
conv2d_11 (Conv2D)           (None, 14, 14, 512)       2359808
_________________________________________________________________
conv2d_12 (Conv2D)           (None, 14, 14, 512)       2359808
_________________________________________________________________
max_pooling2d_4 (MaxPooling2 (None, 7, 7, 512)         0
_________________________________________________________________
flatten (Flatten)            (None, 25088)             0
_________________________________________________________________
dense (Dense)                (None, 4096)              102764544
_________________________________________________________________
```

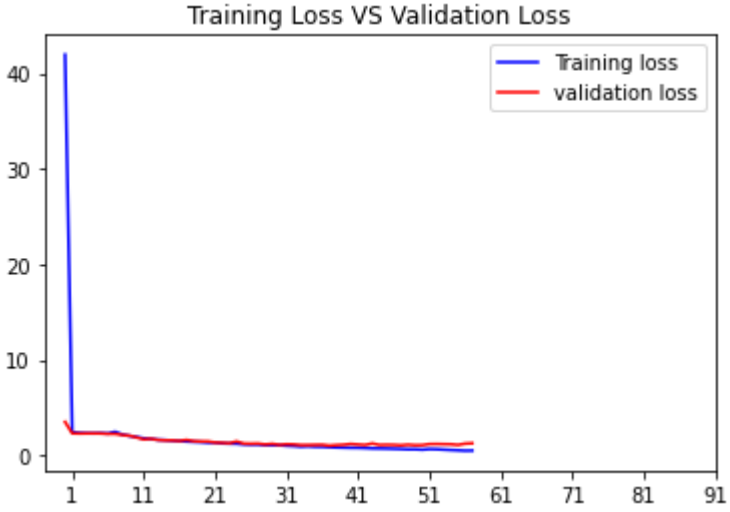
```
dense_1 (Dense)              (None, 4096)              16781312
_________________________________________________________________
dense_2 (Dense)              (None, 10)                40970
=================================================================
Total params: 134,301,514
Trainable params: 134,301,514
Non-trainable params: 0
_________________________________________________________________


2021-10-12 13:53:42.569742: I tensorflow/compiler/mlir/mlir_graph_o
ptimization_pass.cc:116] None of the MLIR optimization passes are e
nabled (registered 2)
2021-10-12 13:53:42.574621: I tensorflow/core/platform/profile_util
s/cpu_utils.cc:112] CPU Frequency: 2000185000 Hz


Epoch 1/100


2021-10-12 13:53:43.695682: I tensorflow/stream_executor/platform/d
efault/dso_loader.cc:49] Successfully opened dynamic library libcub
las.so.11
2021-10-12 13:53:44.492664: I tensorflow/stream_executor/platform/d
efault/dso_loader.cc:49] Successfully opened dynamic library libcub
lasLt.so.11
2021-10-12 13:53:44.520275: I tensorflow/stream_executor/platform/d
efault/dso_loader.cc:49] Successfully opened dynamic library libcud
nn.so.8
```

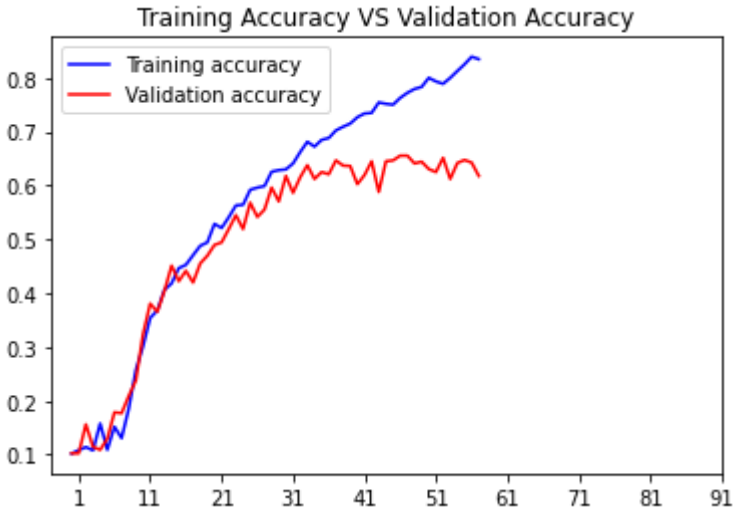
```
195/195 [==============================] - 62s 261ms/step - loss: 1
93.4352 - accuracy: 0.1000 - val_loss: 3.4999 - val_accuracy: 0.102
6
Epoch 2/100
195/195 [==============================] - 49s 251ms/step - loss:
2.5730 - accuracy: 0.1050 - val_loss: 2.3030 - val_accuracy: 0.1026
Epoch 3/100
195/195 [==============================] - 48s 247ms/step - loss:
2.3909 - accuracy: 0.1175 - val_loss: 2.2857 - val_accuracy: 0.1562
Epoch 4/100
195/195 [==============================] - 48s 245ms/step - loss:
2.3153 - accuracy: 0.1132 - val_loss: 2.2963 - val_accuracy: 0.1146
Epoch 5/100
195/195 [==============================] - 47s 242ms/step - loss:
2.2636 - accuracy: 0.1558 - val_loss: 2.3010 - val_accuracy: 0.1090
Epoch 6/100
195/195 [==============================] - 47s 240ms/step - loss:
2.3135 - accuracy: 0.1056 - val_loss: 2.2906 - val_accuracy: 0.1298
Epoch 7/100
195/195 [==============================] - 47s 240ms/step - loss:
2.2808 - accuracy: 0.1566 - val_loss: 2.2456 - val_accuracy: 0.1787
Epoch 8/100
195/195 [==============================] - 46s 237ms/step - loss:
2.3286 - accuracy: 0.1330 - val_loss: 2.2480 - val_accuracy: 0.1771
Epoch 9/100
195/195 [==============================] - 46s 237ms/step - loss:
2.2088 - accuracy: 0.1731 - val_loss: 2.1320 - val_accuracy: 0.2091
Epoch 10/100
195/195 [==============================] - 46s 237ms/step - loss:
2.0855 - accuracy: 0.2345 - val_loss: 2.0574 - val_accuracy: 0.2380
Epoch 11/100
195/195 [==============================] - 46s 234ms/step - loss:
1.9728 - accuracy: 0.2858 - val_loss: 1.8748 - val_accuracy: 0.3237
Epoch 12/100
195/195 [==============================] - 45s 233ms/step - loss:
1.8197 - accuracy: 0.3371 - val_loss: 1.6720 - val_accuracy: 0.3806
Epoch 13/100
195/195 [==============================] - 45s 232ms/step - loss:
1.7278 - accuracy: 0.3611 - val_loss: 1.7343 - val_accuracy: 0.3654
Epoch 14/100
195/195 [==============================] - 45s 231ms/step - loss:
1.6494 - accuracy: 0.3937 - val_loss: 1.6125 - val_accuracy: 0.4046
Epoch 15/100
195/195 [==============================] - 45s 232ms/step - loss:
```

```
1.6229 - accuracy: 0.4054 - val_loss: 1.5685 - val_accuracy: 0.4511
Epoch 16/100
195/195 [==============================] - 45s 232ms/step - loss:
1.5644 - accuracy: 0.4475 - val_loss: 1.5605 - val_accuracy: 0.4223
Epoch 17/100
195/195 [==============================] - 45s 231ms/step - loss:
1.4999 - accuracy: 0.4540 - val_loss: 1.5118 - val_accuracy: 0.4415
Epoch 18/100
195/195 [==============================] - 45s 230ms/step - loss:
1.4373 - accuracy: 0.4750 - val_loss: 1.6124 - val_accuracy: 0.4199
Epoch 19/100
195/195 [==============================] - 45s 230ms/step - loss:
1.4411 - accuracy: 0.4887 - val_loss: 1.4867 - val_accuracy: 0.4551
Epoch 20/100
195/195 [==============================] - 47s 241ms/step - loss:
1.3867 - accuracy: 0.4939 - val_loss: 1.4553 - val_accuracy: 0.4696
Epoch 21/100
195/195 [==============================] - 45s 230ms/step - loss:
1.3445 - accuracy: 0.5244 - val_loss: 1.4531 - val_accuracy: 0.4896
Epoch 22/100
195/195 [==============================] - 45s 230ms/step - loss:
1.3350 - accuracy: 0.5203 - val_loss: 1.3367 - val_accuracy: 0.4944
Epoch 23/100
195/195 [==============================] - 45s 230ms/step - loss:
1.2443 - accuracy: 0.5388 - val_loss: 1.3146 - val_accuracy: 0.5192
Epoch 24/100
195/195 [==============================] - 45s 230ms/step - loss:
1.2215 - accuracy: 0.5693 - val_loss: 1.2667 - val_accuracy: 0.5449
Epoch 25/100
195/195 [==============================] - 45s 230ms/step - loss:
1.2372 - accuracy: 0.5566 - val_loss: 1.4512 - val_accuracy: 0.5192
Epoch 26/100
195/195 [==============================] - 45s 230ms/step - loss:
1.1648 - accuracy: 0.5902 - val_loss: 1.2319 - val_accuracy: 0.5681
Epoch 27/100
195/195 [==============================] - 45s 230ms/step - loss:
1.1311 - accuracy: 0.5865 - val_loss: 1.2134 - val_accuracy: 0.5417
Epoch 28/100
195/195 [==============================] - 45s 230ms/step - loss:
1.1301 - accuracy: 0.5902 - val_loss: 1.2310 - val_accuracy: 0.5553
Epoch 29/100
195/195 [==============================] - 45s 230ms/step - loss:
1.0787 - accuracy: 0.6284 - val_loss: 1.1359 - val_accuracy: 0.5962
Epoch 30/100
195/195 [==============================] - 45s 230ms/step - loss:
```

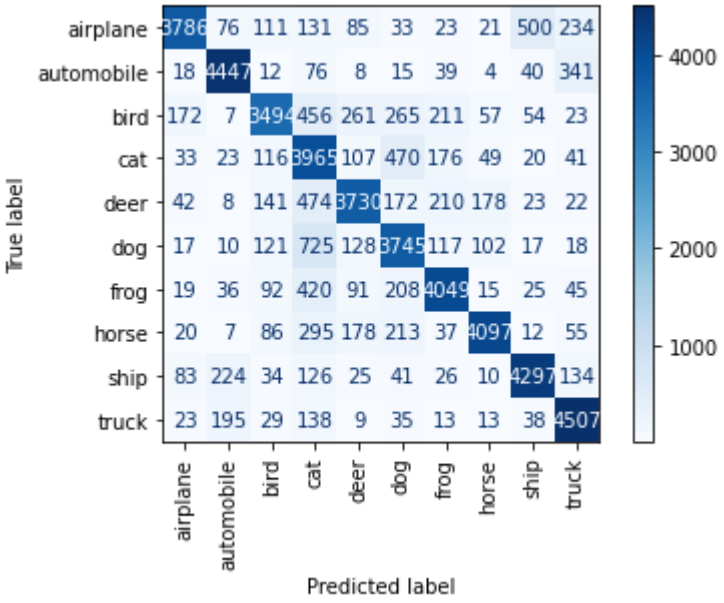
```
1.0513 - accuracy: 0.6308 - val_loss: 1.2029 - val_accuracy: 0.5705
Epoch 31/100
195/195 [==============================] - 45s 229ms/step - loss:
1.0504 - accuracy: 0.6315 - val_loss: 1.0930 - val_accuracy: 0.6178
Epoch 32/100
195/195 [==============================] - 45s 231ms/step - loss:
0.9957 - accuracy: 0.6460 - val_loss: 1.1595 - val_accuracy: 0.5865
Epoch 33/100
195/195 [==============================] - 45s 231ms/step - loss:
0.9896 - accuracy: 0.6581 - val_loss: 1.1384 - val_accuracy: 0.6154
Epoch 34/100
195/195 [==============================] - 45s 230ms/step - loss:
0.9265 - accuracy: 0.6752 - val_loss: 1.0750 - val_accuracy: 0.6378
Epoch 35/100
195/195 [==============================] - 45s 230ms/step - loss:
0.9287 - accuracy: 0.6808 - val_loss: 1.0756 - val_accuracy: 0.6122
Epoch 36/100
195/195 [==============================] - 45s 231ms/step - loss:
0.8939 - accuracy: 0.6880 - val_loss: 1.0910 - val_accuracy: 0.6250
Epoch 37/100
195/195 [==============================] - 45s 230ms/step - loss:
0.8760 - accuracy: 0.6908 - val_loss: 1.0998 - val_accuracy: 0.6210
Epoch 38/100
195/195 [==============================] - 45s 230ms/step - loss:
0.8758 - accuracy: 0.6990 - val_loss: 1.0125 - val_accuracy: 0.6466
Epoch 39/100
195/195 [==============================] - 45s 230ms/step - loss:
0.8140 - accuracy: 0.7146 - val_loss: 1.0666 - val_accuracy: 0.6370
Epoch 40/100
195/195 [==============================] - 45s 230ms/step - loss:
0.8149 - accuracy: 0.7153 - val_loss: 1.1042 - val_accuracy: 0.6362
Epoch 41/100
195/195 [==============================] - 45s 230ms/step - loss:
0.7938 - accuracy: 0.7206 - val_loss: 1.1880 - val_accuracy: 0.6026
Epoch 42/100
195/195 [==============================] - 45s 231ms/step - loss:
0.7712 - accuracy: 0.7337 - val_loss: 1.1286 - val_accuracy: 0.6194
Epoch 43/100
195/195 [==============================] - 45s 230ms/step - loss:
0.7678 - accuracy: 0.7293 - val_loss: 1.0831 - val_accuracy: 0.6450
Epoch 44/100
195/195 [==============================] - 45s 229ms/step - loss:
0.6905 - accuracy: 0.7667 - val_loss: 1.2416 - val_accuracy: 0.5881
Epoch 45/100
195/195 [==============================] - 45s 232ms/step - loss:
```

```
0.7152 - accuracy: 0.7507 - val_loss: 1.0832 - val_accuracy: 0.6450
Epoch 46/100
195/195 [==============================] - 45s 230ms/step - loss:
0.6765 - accuracy: 0.7588 - val_loss: 1.0971 - val_accuracy: 0.6466
Epoch 47/100
195/195 [==============================] - 45s 230ms/step - loss:
0.6952 - accuracy: 0.7591 - val_loss: 1.0871 - val_accuracy: 0.6554
Epoch 48/100
195/195 [==============================] - 45s 230ms/step - loss:
0.6462 - accuracy: 0.7756 - val_loss: 1.0466 - val_accuracy: 0.6554
Epoch 49/100
195/195 [==============================] - 45s 231ms/step - loss:
0.6296 - accuracy: 0.7738 - val_loss: 1.1065 - val_accuracy: 0.6410
Epoch 50/100
195/195 [==============================] - 45s 230ms/step - loss:
0.6123 - accuracy: 0.7858 - val_loss: 1.0573 - val_accuracy: 0.6442
Epoch 51/100
195/195 [==============================] - 45s 230ms/step - loss:
0.5636 - accuracy: 0.8133 - val_loss: 1.0670 - val_accuracy: 0.6306
Epoch 52/100
195/195 [==============================] - 45s 231ms/step - loss:
0.5592 - accuracy: 0.8109 - val_loss: 1.1753 - val_accuracy: 0.6250
Epoch 53/100
195/195 [==============================] - 45s 230ms/step - loss:
0.6253 - accuracy: 0.7979 - val_loss: 1.1847 - val_accuracy: 0.6514
Epoch 54/100
195/195 [==============================] - 45s 230ms/step - loss:
0.6014 - accuracy: 0.7995 - val_loss: 1.1691 - val_accuracy: 0.6122
Epoch 55/100
195/195 [==============================] - 45s 230ms/step - loss:
0.5426 - accuracy: 0.8176 - val_loss: 1.1574 - val_accuracy: 0.6418
Epoch 56/100
195/195 [==============================] - 45s 230ms/step - loss:
0.5183 - accuracy: 0.8216 - val_loss: 1.0896 - val_accuracy: 0.6474
Epoch 57/100
195/195 [==============================] - 45s 231ms/step - loss:
0.4726 - accuracy: 0.8406 - val_loss: 1.2260 - val_accuracy: 0.6426
Epoch 58/100
195/195 [==============================] - 45s 230ms/step - loss:
0.4495 - accuracy: 0.8512 - val_loss: 1.2634 - val_accuracy: 0.6178
```






```
Model saved successfully!
Confusion matrix for train data:
Found 50000 images belonging to 10 classes.
Accuracy score =  0.80234
```




```
Confusion matrix for val data:
Found 10000 images belonging to 10 classes.
Accuracy score =  0.6279
```