American International University Bangladesh

FLOWER CLASSIFICATION WITH CNN

Submitted to: DR. HOSSAIN MD. SHAKHAWAT

AUGUST 11, 2022

GROUP B

Section B

Group Members:

Name	Id
MD. RASHEDUL ISLAM	19-40559-1
MAHMUD HASAN	17-33881-1
ASADUR RAHMAN	18-37447-1
SHAZZAD HOSSAIN SHOVON	17-34884-2
SK. MOHIUDDIN AHMED	19-40126-1

Contents

Purpose

Flower Identification is an image processing technique using machine learning algorithm. The algorithm identify what kind of flower is given as an input and the give us the class of the flower. There are lots of flower species in the earth it is quite hard to memorize every flower in human brain. This system will help to detect what kind of flower it is.

Introduction

There are many flower species in the world. Some species have many colors, such as roses. It is hard to remember all flower names and their information. Furthermore, someone may be confused with similar flower species. For example, white Champak a and Champak have similar names and petal shapes but they have different colors and petal lengths.

At this time, it is almost impossible to identify particular flowers or flower species in any other way but to seek information based on personal knowledge and experience of experts. Availability of such experts may be a barrier to such information seeking. Searching for such information on the Internet is, today, very much restricted to key word searching; text processing. Even in this the searcher needs to provide sufficiently useful keywords, which they cannot do, which is the crux of the matter.

Project Literature

Earlier flower recognition research was proposed by Das et al. [1]. However, only a color-based algorithm was developed. It is hard to classify different flower species based purely on color; many different flowers and specifies have similar colors, and many flowers of the same species have different colors. Saitoh and Kaneto [2] developed an automatic recognition system for wild flowers. There are two input images utilised; flower and leaf. The requirement for both flower and leaf features to be analysed is a restriction of this approach

Proposed Methodology

CNN

A convolutional neural network (CNN) is a type of artificial neural network used in image recognition and processing. A Convolutional neural network (CNN) is a neural network that has one or more convolutional layers and are used mainly for image processing, classification, segmentation and also for other auto correlated data. A convolution is essentially sliding a filter over the input. The most common use for CNNs is image classification, for example identifying satellite images that contain roads or classifying hand written letters and digits. There are other quite mainstream tasks such as image segmentation and signal processing, for which CNNs perform well at. CNNs have been used for understanding in Natural Language Processing (NLP) and speech recognition, although often for NLP Recurrent Neural Nets (RNNs) are used.

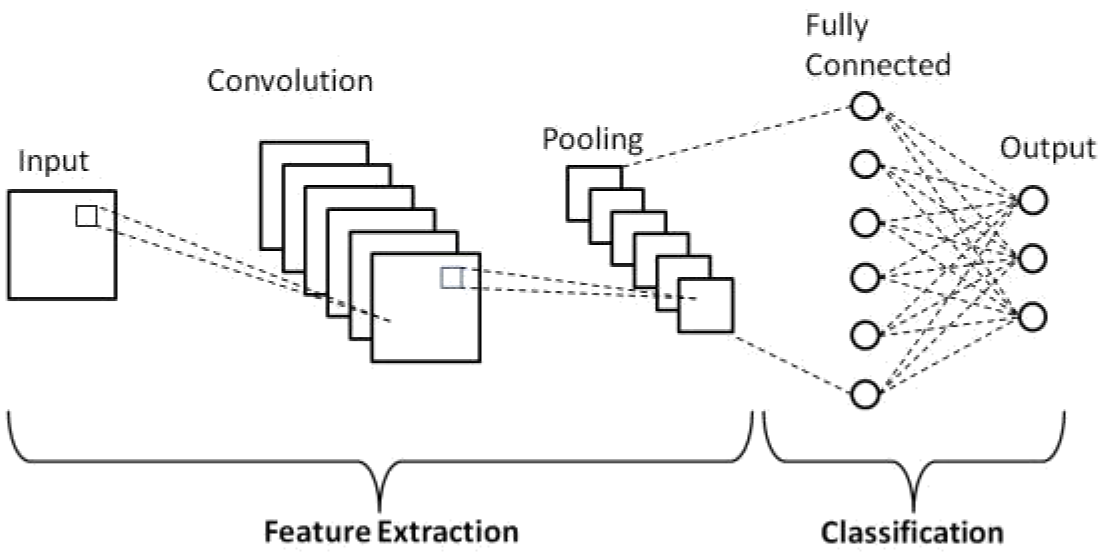

Fig.1

Dataset

Link

https://drive.google.com/file/d/1h6l1ilhVxPj60iwcsh2ZcUYgwhdVhxmy/view

Pre-processing

We use the TensorFlow, NumPy library for our implementation. We take 4 class of images (Daisy, Dandelion, Roses and sunflowers). We take total 477 images to make compile time short. We use NumPy because of we need array operation to convert the images in array. There are different sizes of images in dataset so make it all a constant size which is 224X224. We take the batch size = 64, that’s mean each epoch work with 64 images. When the dataset is small preprocessing make multiple images from one single image by share, horizontal flip and zoom.

Simulation

Code

```
import tensorflow as tf
import numpy as np

base_dir=r"C:\Users\Asus\flower"

IMAGE_SIZE=224
BATCH_SIZE=64

#pre=processing
train_datagen=tf.keras.preprocessing.image.ImageDataGenerator( rescale=1./255,

    shear_range=0.2,
    zoom_range=0.2,
    horizontal_flip=True,
    validation_split=0.1
    )

test_datagen=tf.keras.preprocessing.image.ImageDataGenerator( rescale=1./255,

     validation_split=0.1
)

train_datagen=train_datagen.flow_from_directory(
    base_dir,
    target_size=(IMAGE_SIZE,IMAGE_SIZE),
    batch_size=BATCH_SIZE,
    subset='training'
)
```

```
test_datagen=test_datagen.flow_from_directory(
    base_dir,
    target_size=(IMAGE_SIZE,IMAGE_SIZE),
    batch_size=BATCH_SIZE,
    subset='validation'
)
```

Found 431 images belonging to 4 classes.

Found 46 images belonging to 4 classes.

```
cnn=tf.keras.Sequential()
cnn.add(tf.keras.layers.Conv2D(filters=64,padding='same',strides=2,kernel_size=3,activation='relu',input_shape=(224,224,3)))
cnn.add(tf.keras.layers.MaxPool2D(pool_size=2,strides=2))

cnn.add(tf.keras.layers.Conv2D(filters=32,padding='same',strides=2,kernel_size=3,activation='relu'))
cnn.add(tf.keras.layers.MaxPool2D(pool_size=2,strides=2))

cnn.add(tf.keras.layers.Conv2D(filters=32,padding='same',strides=2,kernel_size=3,activation='relu'))
cnn.add(tf.keras.layers.MaxPool2D(pool_size=2))

cnn.add(tf.keras.layers.Flatten())
cnn.add(tf.keras.layers.Dense(4,activation='softmax'))
cnn.compile(optimizer=tf.keras.optimizers.Adam(),loss='categorical_crossentropy',metrics=['accuracy'])
cnn.fit(train_datagen,epochs=20,validation_data=test_datagen)
```

Out Put

Epoch 1/20

7/7 [==============================] - 11s 1s/step - loss: 1.3794 - accuracy: 0.2651 - v
al_loss: 1.3118 - val_accuracy: 0.3478
Epoch 2/20
7/7 [==============================] - 10s 1s/step - loss: 1.3009 - accuracy: 0.4346 - v
al_loss: 1.2331 - val_accuracy: 0.5000
Epoch 3/20
7/7 [===============================] - 9s 1s/step - loss: 1.1951 - accuracy: 0.5116 - va
l_loss: 1.1588 - val_accuracy: 0.5435
Epoch 4/20
7/7 [===============================] - 9s 1s/step - loss: 1.1460 - accuracy: 0.5605 - va
l_loss: 1.0668 - val_accuracy: 0.5435
Epoch 5/20
7/7 [===============================] - 9s 1s/step - loss: 1.0143 - accuracy: 0.6052 - va
l_loss: 0.9973 - val_accuracy: 0.5652
Epoch 6/20
7/7 [===============================] - 9s 1s/step - loss: 0.9333 - accuracy: 0.6118 - va
l_loss: 0.9125 - val_accuracy: 0.6304
Epoch 7/20
7/7 [===============================] - 9s 1s/step - loss: 0.8894 - accuracy: 0.6447 - va
l_loss: 0.8881 - val_accuracy: 0.6087
Epoch 8/20
7/7 [===============================] - 9s 1s/step - loss: 0.8674 - accuracy: 0.6572 - va
l_loss: 0.8956 - val_accuracy: 0.5870
Epoch 9/20
7/7 [===============================] - 9s 1s/step - loss: 0.8417 - accuracy: 0.6544 - va
l_loss: 0.8512 - val_accuracy: 0.5652
Epoch 10/20
7/7 [===============================] - 9s 1s/step - loss: 0.8416 - accuracy: 0.6812 - va
l_loss: 0.8376 - val_accuracy: 0.6522
Epoch 11/20

7/7 [==============================] - 9s 1s/step - loss: 0.7969 - accuracy: 0.6783 - va

l_loss: 0.7975 - val_accuracy: 0.6522

Epoch 12/20

7/7 [=============================] - 10s 1s/step - loss: 0.7958 - accuracy: 0.6829 - v

al_loss: 0.7957 - val_accuracy: 0.6087

Epoch 13/20

7/7 [==============================] - 9s 1s/step - loss: 0.7956 - accuracy: 0.6736 - va

l_loss: 0.8205 - val_accuracy: 0.5652

Epoch 14/20

7/7 [==============================] - 9s 1s/step - loss: 0.7630 - accuracy: 0.7203 - va

l_loss: 0.7816 - val_accuracy: 0.6304

Epoch 15/20

7/7 [==============================] - 9s 1s/step - loss: 0.7911 - accuracy: 0.6716 - va

l_loss: 0.9267 - val_accuracy: 0.5000

Epoch 16/20

7/7 [==============================] - 9s 1s/step - loss: 0.8356 - accuracy: 0.6589 - va

l_loss: 0.7974 - val_accuracy: 0.6304

Epoch 17/20

7/7 [==============================] - 9s 1s/step - loss: 0.7675 - accuracy: 0.6848 - va

l_loss: 0.7210 - val_accuracy: 0.7174

Epoch 18/20

7/7 [==============================] - 9s 1s/step - loss: 0.7513 - accuracy: 0.6847 - va

l_loss: 0.7319 - val_accuracy: 0.6957

Epoch 19/20

7/7 [=============================] - 10s 1s/step - loss: 0.7115 - accuracy: 0.7075 - v

al_loss: 0.7439 - val_accuracy: 0.6739

Epoch 20/20

7/7 [=============================] - 10s 1s/step - loss: 0.7318 - accuracy: 0.6883 - v

al_loss: 0.7286 - val_accuracy: 0.7174

<tensorflow.python.keras.callbacks.History at 0x2ec975c8648>

Conclusion

The flower classification based on the image processing takes real time images any smart phone or digital camera for analysis and identification the class of a flower. Therefore, the system is highly mobile, and can be used ‘in the field’ by researchers and laymen alike. In the proposed system, the original flower image is resized for faster processing. Using CNN where have several layers. The accuracy of this system is more than 70%. Our validation accuracy is not good because we have face some problem and have limitation in doing the system. This system can be further improved to yield more accuracy by combining other features,

References

[1] M. Das, R. Manmatha, and E. M. Riseman, “Indexing flower patent images using domain knowledge”, IEEE Intell. Syst., vol 14, pp. 24-33, 1999.

[2] T. Saitoh and T. Kaneko, “Automatic recognition of wild flowers”, System and Computer in Japan, vol. 34, no. 10, 2003.