



IMAGE CLASSIFICATION

Submitted by:

Lahari S.K

ACKNOWLEDGMENT

I would like to thank each and every one who helped me through out the completion of this project.

INTRODUCTION

Image classification is growing and becoming a trend among technology developers especially with the growth of data in different parts of industry such as e-commerce, automotive, healthcare, and gaming.

The most obvious example of this technology is applied to Facebook. Facebook now can detect up to 98% accuracy in order to identify face with only a few tagged images and classified it into Facebook's album.

The technology itself almost beats the ability of human in image classification or recognition. One of the dominant approaches for this technology is deep learning.

Deep learning falls under the category of Artificial Intelligence where it can act or think like a human. The system itself will be set with hundreds or maybe thousands of input data in order to make the 'training' session to be more efficient and fast. It starts by giving some sort of 'training' with all the input data.

Machine learning is also the frequent systems that has been applied towards image classification. However, there are still parts that can be improved within machine learning. Therefore, image classification is going to be occupied with deep learning

The project goes out in 2 phases








1. Image Collection
2. Image Collection

Image Collection

In this particular project we have to collect data of minimum 200 images of categories like

- a) Saree
- b) Pants(Men)
- c) Trousers(Men) from the amazon website.

Libraries Used for Image Collection

-  Pandas
-  Numpy
-  Selenium
-  Shutil
-  Requests
-  OS
-  Time

Steps followed in Image Classification

1.Creating directory:

```
# function to create a directory

def directory(folder):
    img_dir = os.path.join(os.getcwd(), folder)
    if not os.path.exists(img_dir):
        os.makedirs(img_dir)

    return folder
```

“directory” is a function which is used to create a directory. `img_dir` variable contains the path of the directory. If `img_dir` exist return the folder. If the directory path does not exist then create one in the path using the function `os.makedirs`.

```
[ ] # Calling the above function

folder = directory("/content/drive/MyDrive/Projects/Image Processing/data")
```

Calling directory function and passing the folder path as the parameter.

2.Code Snippet for scraping Images

```
[ ] # Function for scraping, downloading and storing the required images/data.

def images(url, folder, labels, label):

    # Opening drive
    amazon_url = driver.get(url)

    time.sleep(3)

    # Locating the search bar
    search_bar = driver.find_element_by_id("twotabsearchtextbox")
    search_bar.clear()
    category = input('Search Category : ')
    search_bar.send_keys(category)

    #locating the button and clicking it to search the category
    button = driver.find_element_by_id('nav-search-submit-button')
    button.click()

    time.sleep(3)

    # Creating empty list fto store data
    image_urls = []

    # Loop for iterating the pages and thus scraping the images
    for page in range(1, 6):
        print('\nPage', page)

        time.sleep(3)

        # Giving path of the images to be scrapped
        images = driver.find_elements_by_xpath("//img[@class = 's-image']")

        # Getting the source/link of the images.
        for img in images:
            source = img.get_attribute('src')
            image_urls.append(source)

        # Downloading the images and saving the same in the respective directory.
        for i, image_link in enumerate(image_urls):
            response = requests.get(image_link)

            # Downloading the images
            image_name = folder + '/' + category + '_' + str(i+1) + '.jpg'
            with open(image_name, 'wb') as file:
                file.write(response.content)

        time.sleep(3)

        print("Downloads of images from Page", page, "is completed! \n")

        time.sleep(3)
```

“images” is the function that we are creating to scrap all the images we are defining 4 parameters inside the function namely

- 1.url=url of the website to be scrapped
- 2.folder=folder name where we need images to be saved
- 3.labels=variable to specify output constraint
- 4.label=assigning different values to different categories

amazon_url contains the url that has to be searched. search_bar is used to send the keys of the input . button is used to search the entered product.

We are scrapping first 6 pages of the website.image names is created by concatenating the folder name,category ,number and jpg extension.Now the images is saved file.

3.Calling images function

Saree

```
[ ] # Calling the function and scraping the 'Sarees (women)' category images.  
images("https://www.amazon.com/", folder, labels, 0)
```

Jeans

```
[ ] # Calling the function and scraping the 'Jeans(men)' category images.  
images("https://www.amazon.com/", folder, labels, 1)
```

Trousers

```
[ ] # Calling the function and scraping the 'Trousers(men)' category images.  
images("https://www.amazon.com/", folder, labels, 2)
```

Saree is labelled as 0.









Jeans is labelled as 1.

Trousers is labelled as 2.

labels is now converted into dataframe and saved as df variable. df is then saved in csv .

Image Classification

Libraries used

-  tensorflow
-  keras
-  PIL
-  Image
-  kerastuner
-  RandomSearch
-  Hyperparameters
-  matplotlib

Images are read from the folder using Image.open. Since all the images are of different sizes, resize is used to normalize or scale all the images into same size.

The data is loaded read from the csv file. We are using simple cnn architecture with keras tuner to solve the problem.

Sequential layer is created which is consisted of 14 layers. One input layer, 1 output layer, 4 convolution layers, 4 maxpooling layer, 5 dropout layer to overcome overfitting.

The code snippet is as follows

```
[ ] def build_model(hp):
    model=keras.Sequential([

        #FIRST CONVOLUTION LAYER
        keras.layers.Conv2D(
            filters=hp.Int('conv_1_filter',min_value=32,max_value=128,step=16),
            kernel_size=hp.Choice('conv1_kernal',values=(3,5)),
            activation='relu',
            input_shape=X.shape[1:],

        ),
        #MAXPOOLING
        keras.layers.MaxPooling2D(pool_size=(2,2)),

        #DROP OUT
        keras.layers.Dropout(0.2),

        #SECOND CONVOLUTION LAYER
        keras.layers.Conv2D(
            filters=hp.Int('conv_1_filter',min_value=32,max_value=128,step=16),
            kernel_size=hp.Choice('conv1_kernal',values=(3,5)),
            activation='relu',
        ),

        #MAXPOOL LAYER
        keras.layers.MaxPooling2D(pool_size=(2,2)),

        #DROPOUT LAYER
        keras.layers.Dropout(0.2),

        #THIRD CONVOLUTION LAYER
        keras.layers.Conv2D(
            filters=hp.Int('conv_1_filter',min_value=32,max_value=128,step=16),
            kernel_size=hp.Choice('conv1_kernal',values=(3,5)),
            activation='relu',
        ),

        #MaXPOOL LAYER
        keras.layers.MaxPooling2D(pool_size=(2,2)),

        #DROPOUT LAYER
        keras.layers.Dropout(0.2),

        #FOURTH CONVOLUTION LAYER
        keras.layers.Conv2D(
            filters=hp.Int('conv_1_filter',min_value=32,max_value=128,step=16),
            kernel_size=hp.Choice('conv1_kernal',values=(3,5)),
            activation='relu',
        ),

        #MaXPOOL LAYER
        keras.layers.MaxPooling2D(pool_size=(2,2)),

        #DROPOUT LAYER
        keras.layers.Dropout(0.2),

        #FLATTEN LAYER
        keras.layers.Flatten(),
```



```

#MAXPOOL LAYER
keras.layers.MaxPooling2D(pool_size=(2,2)),

#DROPOUT LAYER
keras.layers.Dropout(0.2),

#FLATTEN LAYER
keras.layers.Flatten(),

#DROPOUT LAYER
keras.layers.Dropout(0.5),

#OUTPUT LAYER
keras.layers.Dense(10,activation="softmax")

])

model.compile(optimizer=keras.optimizers.Adam(hp.Choice('learning_rate',values=[1e-2,1e-3])),
              loss="sparse_categorical_crossentropy",
              metrics=["accuracy"])

return model

```

After finding the best parameters for number of conv filters. With the best model parameters we build the model

```
[ ] tuner_search=RandomSearch(build_model,objective="val_accuracy",max_trials=5,directory="output1",project_name="Image Processing1")
```

```
[ ] tuner_search.search(X_train,y_train,epochs=3,validation_split=0.1)
```

```

Trial 5 Complete [00h 10m 10s]
val_accuracy: 0.6301369667053223

Best val_accuracy So Far: 0.6301369667053223
Total elapsed time: 00h 41m 35s
INFO:tensorflow:Oracle triggered exit

```

```
[ ] model=tuner_search.get_best_models(num_models=1)[0]
```

The results areas follows

Training accuracy: 71.14 with loss: 0.572

Validation accuracy: 78.8 with loss:0.58

Test data accuracy: 0.75 with loss:0.57

