INDIAN FACE DETECTION MODEL

SAKSHAM SRIVASTAVA

BATCH- B21

APRIL 21, 2021

STEP 1- SCRAPPING

TECHNOLOGY AND CODE USED-

We looked for various platforms where we could find different images with a large dataset like Google Image, Unsplash, Pinterest etc. Made scripts to scrap the images. From Google Images we were able to scrape 80 images at a time. We looked at Unsplash and we found the JSON file by which we could extract all the images in large amounts so we made a scraper for the same. Also with the scraper build a downloader by which we could download all the images.

https://github.com/saksham02112000/unsplashscraper

https://drive.google.com/drive/u/0/folders/1Hrx2ewFmqWG489PNWB_eOYepLDyVtEiY

PROBLEMS FACED-

We needed a big dataset as many of the images would be rejected by our model. Google allows only 80 images at a time. We need to scrap in bulk and make an efficient scraper.

CONTRIBUTION-

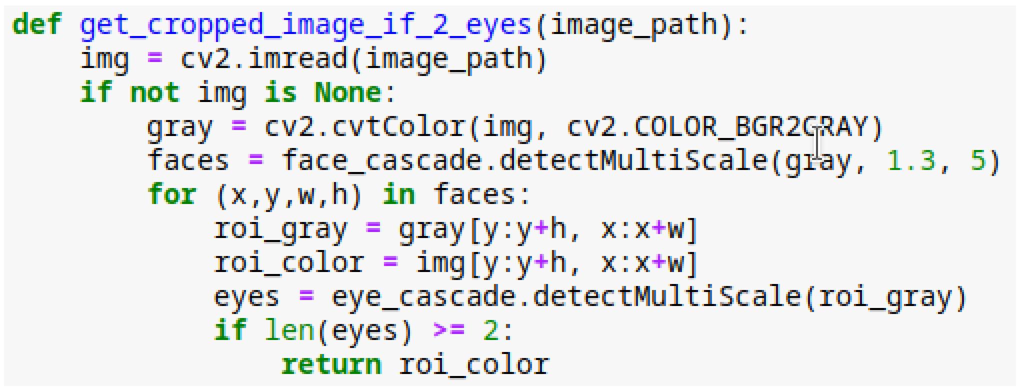
1. **Found a scraper for google and pinterest and made changes to make it work efficiently.**
2. **Suggested some alternatives like parsehub for those who don't know scrapping**
3. **Made a scraper and downloader of unsplash which scrape URLs from the json file scraped from website and downloader then downloaded all the images.**

STEP 2- FILTER ALL IMAGES OF HUMANS

TECHNOLOGY AND CODE USED-

We used OpenCV classifier to detect the face and the eyes to detect the human face and the two eyes. For detecting them we used haar cascade classifier of OpenCV. We didn’t consider the one eye faces because it is not a good dataset to put in the training part. Hence cropped down the photos and put them into folder.

SNIPPETS-



```python
def get_cropped_image_if_2_eyes(image_path):
    img = cv2.imread(image_path)
    if not img is None:
        gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
        faces = face_cascade.detectMultiScale(gray, 1.3, 5)
        for (x,y,w,h) in faces:
            roi_gray = gray[y:y+h, x:x+w]
            roi_color = img[y:y+h, x:x+w]
            eyes = eye_cascade.detectMultiScale(roi_gray)
            if len(eyes) >= 2:
                return roi_color
```

CONTRIBUTION-

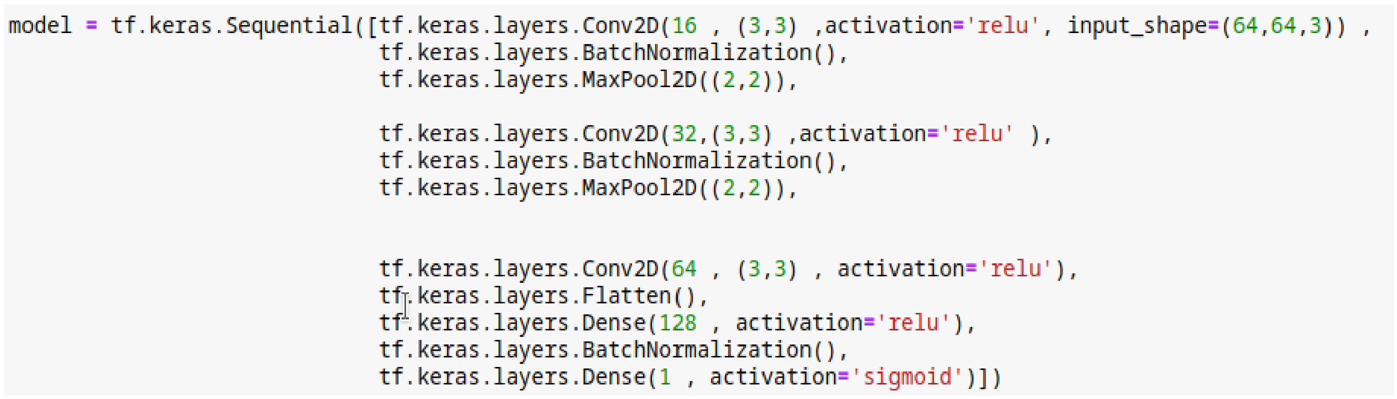
1. **Made the function which could crop the image convert it into grayscale and the detect out faces with two eyes and face using haar cascade in OpenCV**
2. **Made the part of code where we could save the images to a folder and name all the images uniquely using OS and Shutil.**

STEP 3- Train a model to classify images as indians and foreigners

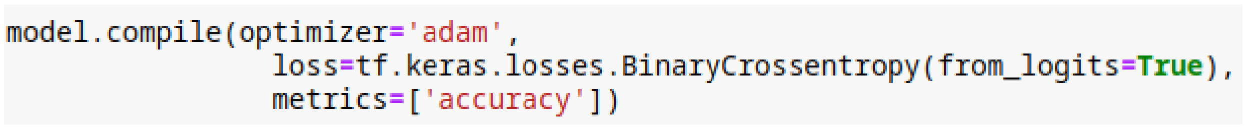
APPROACH USED AND CONTRIBUTION-

1. **First separated the images and foreigners into different folders.**
2. **Had 2731 images in training model in which 1405 were of Indians and 1326 of foreigners.**
3. **Then made a convolution neural network to train the model, used relu activation function.and for reducing error used sigmoid activation function.**
4. **Used MaxPool2D to reduce the pixel calculation.**
5. **Flattened the images so that model could train images using convolutional networks.**
6. **Dropped columns using drop() but it was giving more weird results so removed it.**
7. **Used optimizer as "adam" and to reduce error used CrossEntropy.**
8. **Finally fit the model and ran it with epochs=30.**

MODEL-



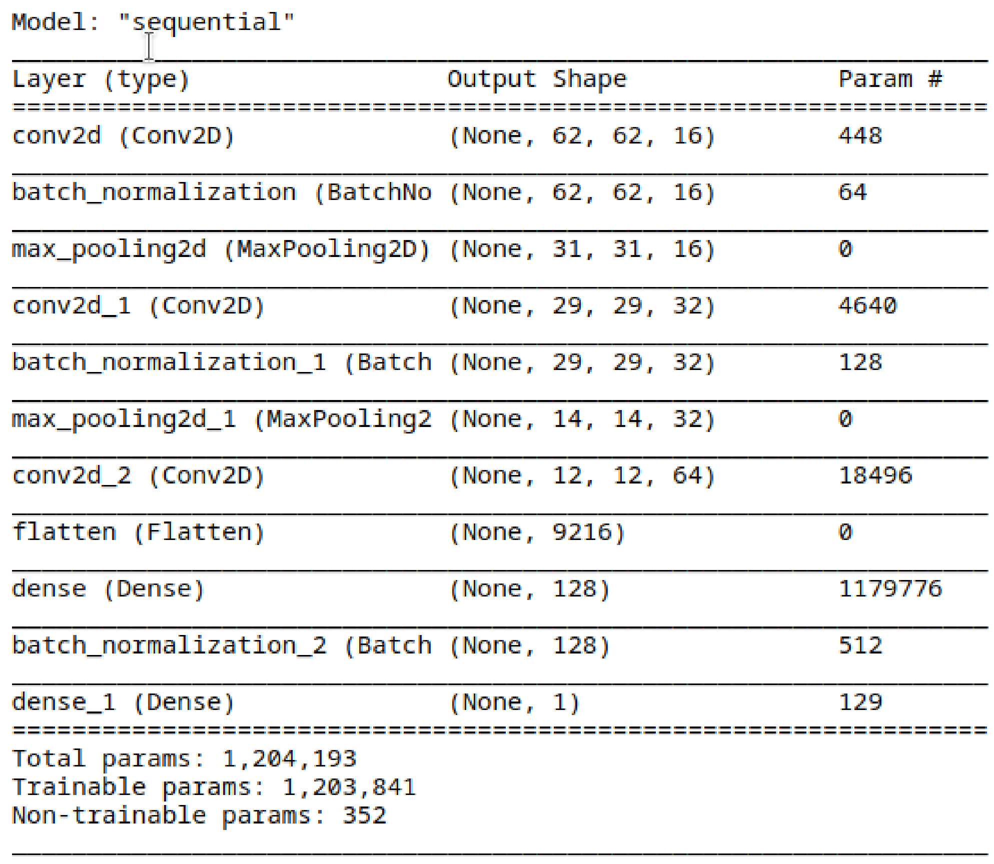
```
model = tf.keras.Sequential([tf.keras.layers.Conv2D(16 , (3,3) ,activation='relu', input_shape=(64,64,3)) ,
                             tf.keras.layers.BatchNormalization(),
                             tf.keras.layers.MaxPool2D((2,2)),

                             tf.keras.layers.Conv2D(32,(3,3) ,activation='relu' ),
                             tf.keras.layers.BatchNormalization(),
                             tf.keras.layers.MaxPool2D((2,2)),


                             tf.keras.layers.Conv2D(64 , (3,3) , activation='relu'),
                             tf.keras.layers.Flatten(),
                             tf.keras.layers.Dense(128 , activation='relu'),
                             tf.keras.layers.BatchNormalization(),
                             tf.keras.layers.Dense(1 , activation='sigmoid')])
```



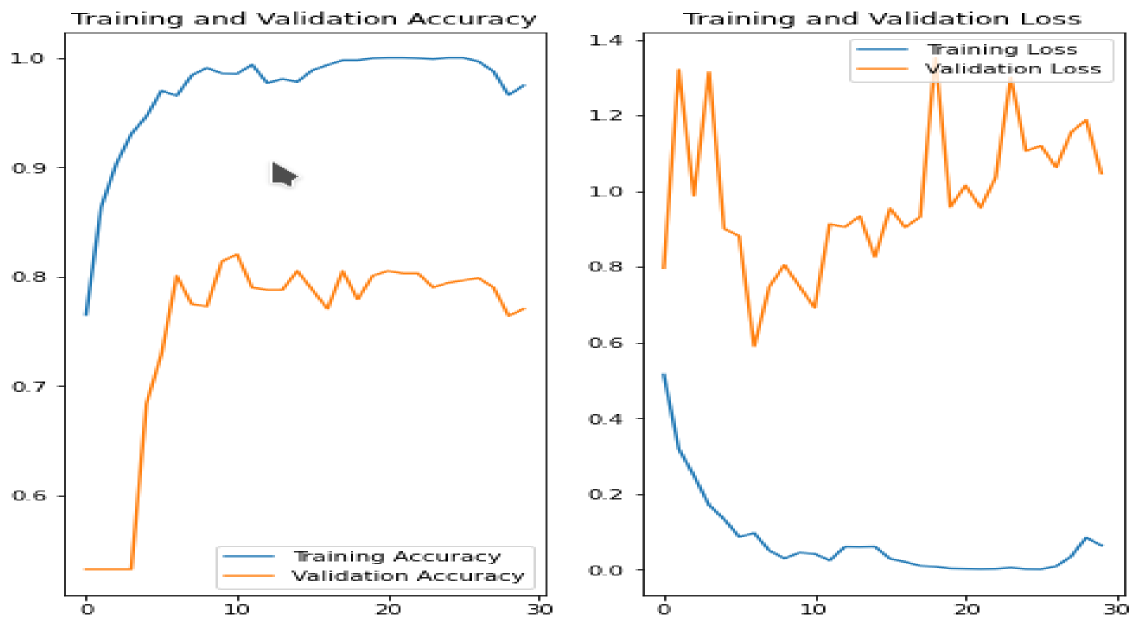
```
model.compile(optimizer='adam',
              loss=tf.keras.losses.BinaryCrossentropy(from_logits=True),
              metrics=['accuracy'])
```

ANALYSIS AND ERRORS-

Model-


```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 62, 62, 16)        448
_________________________________________________________________
batch_normalization (BatchNo (None, 62, 62, 16)        64
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 31, 31, 16)        0
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 29, 29, 32)        4640
_________________________________________________________________
batch_normalization_1 (Batch (None, 29, 29, 32)        128
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 14, 14, 32)        0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 12, 12, 64)        18496
_________________________________________________________________
flatten (Flatten)            (None, 9216)              0
_________________________________________________________________
dense (Dense)                (None, 128)               1179776
_________________________________________________________________
batch_normalization_2 (Batch (None, 128)               512
_________________________________________________________________
dense_1 (Dense)              (None, 1)                 129
=================================================================
Total params: 1,204,193
Trainable params: 1,203,841
Non-trainable params: 352
_________________________________________________________________
```

TRAINING AND TESTING ACCURACY, LOSS-



ISSUES-

There was an issue of overfitting in our model. We were getting unstable loss function which should be a decreasing graph. Also the validation accuracy was also low though training part was working right .

APPROACH USED TO FIX IT-

1. Removed the drop() in the compilation model
2. Tried to get a larger dataset.
3. Initially we were getting an accuracy of 60 since we had indians:foreigners images in ratio 9:1. E increased the dataset and our accuracy increased to 80%.
4. Still trying to make necessary changes in our model and dataset to stop overfitting.