BOSCH's Age and Gender Detection

The Project aims to detect the age and gender of people from given low-quality CCTV surveillance video. For the same purpose, we have broken the problem statement into three parts:

1. Detecting Persons from the surveillance video and extracting the faces from it.
2. Enhancing the photo thus obtained through GAN models.
3. Predict the gender and the age from the super-resolution images obtained.

The above components are named M1, M2, and M3 respectively.

The codes are in the Google Colabotary notebook.

Pre-requisites for running the code:

- The colab must be connected to GPU runtime.
- The version of the libraries should be the following:
  - numpy : 1.21.5
  - openCV : 4.1.2
  - torch : 1.10.0+cu111
  - matplotlib : 3.2.2
  - Pillow : 7.1.2
  - pytube : 12.0.0
  - gdown : 4.2.2
  - basicsr>=1.3.3.11
  - facexlib>=0.2.0.3
  - gfpgan>=0.2.1

Code

- First two cells of the notebook install the necessary libraries needed for stage 1.

```
!pip install --upgrade --no-cache-dir gdown
```

```
!pip install pytube
```

- The following three cells download the required data, including pre-trained model weights and other essential data.

```
!gdown --id '10NaxFCpitXjtLX0rZ4M02FV0rOGRpLKI' #weights_dummy.pt
```

```
!gdown --id '1rJI17Y2u0MDmqv2qkcM6ScOYZpGE_9dd' #weights.caffemodel
```

```
!gdown --id '1M38YE0S9ZztaKK9JJ1GeBbUqIunAimRV' #deploy.prototext file for caffemodel
```

- **Kindly run these cells individually as it might take a few seconds to load the large files into the workspace and also ensure that weights_dummy.pt, weights.caffemodel and deploy.prototext gets imported in the Colab workspace.**

After doing so, you can run the rest of the code. The training is skipped, and the model is loaded with pre-trained weights. To train stage 1 of the model, uncomment the cells just below the provided heading "Uncomment the following cells to train the model again," and, comment out or do not run the "Training is skipped, and model is loaded with pre-trained weights' section." and comment out the next section.

> **Note:** It might take some time to clone some repositories in the notebook.

- `im_convert(tensor)` converts a torch.tensor to image.
- get_bbox(frame,model,x,k) takes the input image, model as parameters. The x and k is for the purpose of saving the output file. This function saves the cropped persons images as in the folder 'm1_cropped'. (The file names contain a particular format.)
- In the get_bbox function, there are two parameters involved.

```
num_people = len(scores[scores > 0.7])
```

Here those persons are selected where the confidence is more significant than 0.7. This value can be changed as per the user's requirement.

```
  indices =   torch.ops.torchvision.nms(torch.Tensor(boxes),torch.Tensor(scores),0.25).tolist()
```

Here those boxes are chosen where the IOU is less than 25%. Again this threshold can be changed as per the user's requirement. * videoDataProcessing function takes the model and the path of the input video file. K is taken for the purpose of saving a file in a particular name. This function iterates over each frame of the video and captures the frame after every 20th millisecond and from that frame it extracts the person's body and saves it in the folder.

In the function `videoDataProcessing` we are capturing frames after every 20 milliseconds,

```
while cap.isOpened():
        ret, frame = cap.read()
        if ret and i%20==0:
            get_bbox(frame,model,i/20,k)
            print(i)
        elif not ret:
            break
        i+=1
        if cv2.waitKey(20) & 0xFF == ord('q'):
            break
```

This value can be changed by replacing some other value in place of 20.

- `get_from_YouTube` function downloads the video with a given link in the colab workspace.
- After running these functions, the program will pause and ask the user for the input.

```
Enter 0 for uploading a video file.
Enter 1 for uploading images.
```

  - The user has the flexibility to provide the input either as images or videos.
  - The user can provide the video input in two formats, i.e., a youtube video link or uploading the video from the local system.
  - For providing video inputs, first, you need to give 0 as input, and again you will be asked for input. Enter 0 if you want to enter a youtube video link;

```
Enter 0 for uploading a video file.
Enter 1 for uploading images.
0
Enter 0 for uploading youtube video.
Enter 1 for uploading local files.
0
Enter youTube video link.
```

    otherwise, it will ask to upload files from the system.
  - For providing image inputs, the user needs to enter 1 and choose the images **only from the local system**.

> **Note:** It might take some time to upload the images/videos into the Google Colab workspace depending upon the speed/bandwidth of the network used.
>
> **Note:** Kindly refrain from using a YouTube video link whose content length is more than 5 minutes, as it might not be processed by pytube library.

Once the inputs are provided, the model will automatically detect the persons and crop the person's body and the faces and save them in different folders in the Colab workspace, namely m1_cropped and faces, respectively.

These images thus obtained will be fed to Real ESRGAN (M2), which will super-resolve the images.

- The super-resolved body images obtained from the previous step are saved in the folder body_images in the Colab workspace.
- The super-resolved face images are saved in the folder face_images in the Colab workspace.
- The model M3 performs age and gender prediction. We have divided the M3 into three models. The first model predicts via superimposed body images whereas the second model predicts via superimposed face images and the third model concatenates the features of both the models and adds two additional age and gender classification layers.
- The 'Data loading' section loads the training data.
- `class ConvNeuralNet` and the scratch model of CNN architecture is implemented on it to get the model of prediction of the body images only. We stored the weights of the preivious training model and the validation accuracy comes out to be approximately 66%. We can use this model to predict the age and gender.
- `class vgg16` is trained from scratch. We stored the weights of the preivious training model and the validation accuracy comes out to be approximately 23%. We can use this model to predict the age and gender.
- Combined Model uses both the above models to predict age and gender using the approch of concatenation of features of both the models by removing few last layers.
- We stored the weights of the combined model. We can use this model to predict the age and gender.

Test Function is there which predicts the output of the custom input by the user (video/images). The final output is stored in submission.csv file which includes: * path of the face image which encodes both the frame and the person in the image. * path of the body image which also encodes both the frame and the person in the image. * bb_xmin which represents the starting X-Coordinate of the bounding box. * bb_ymin which represents the starting Y-Coordinate of the bounding box. * bb_height which represents the height (bb_xmax- bb_xmin) of the bounding box. * bb_width which represents the width (bb_ymax- bb_ymin) of the bounding box. * age_min represents the predicted minimum age. * age_max represents the predicted maximum age. * gender represents the predicted gender.