PRAKTIKUM

MMAI1004 – PEMBELAJARAN MESIN

PERTEMUAN I

DOSEN PENGASUH:

Prof. Dr. Taufik Fuadi Abidin, S.Si., M.Tech

Dr. Rumaisa Kruba, S.Si

Prof. Dr. Ir. Hammam Riza, M.Sc,. IPU

Praktisi Komdigi (Dr. Said Mirza Pahlevi)

ASISTEN PRAKTIKUM:

Muhammad Chaidir, S.Kom, M.Kom

Step-by-Step Guideline for Running MTCNN on FaceScrub Dataset

1. Introduction

MTCNN (Multi-task Cascaded Convolutional Networks) is a popular deep learning-based face detection algorithm. This guide explains how to use MTCNN for detecting faces from the FaceScrub dataset and saving the detected face images using Python.

---

2. Prerequisites

Before running the code, ensure you have the required libraries installed. Use the following command to install missing dependencies:

```
pip install mtcnn tensorflow opencv-python imutils tqdm
```

Ensure your system has a properly configured TensorFlow environment with GPU support if available.

---

3. Understanding the Code

The provided Python script performs the following tasks:

- Configures GPU settings for TensorFlow.
- Loads the MTCNN model.
- Iterates through images in the dataset directory.
- Detects faces in images.
- Extracts and saves the detected faces into a new directory.

---

4. Step-by-Step Execution

Step 1: Import Necessary Libraries

The following libraries are required for the implementation:

```
from mtcnn import MTCNN
import tensorflow as tf
from imutils import paths
from tqdm import tqdm
import cv
import os
```

Step 2: Configure GPU (if available)

To use GPU for processing, configure TensorFlow to allow dynamic GPU memory allocation:

```
os.environ["CUDA_VISIBLE_DEVICES"] = "0"
config = tf.compat.v1.ConfigProto()
config.gpu_options.allow_growth = True
session = tf.compat.v1.Session(config=config)
```

Step 3: Initialize MTCNN Detector

Create an instance of the MTCNN face detector:

```
detector = MTCNN()
```

Step 4: Load Image Paths

Set the directory path where images are stored and retrieve the list of image paths:

```
dirpath = "new/training"
imagePaths = sorted(list(paths.list_images(dirpath)))
```

Step 5: Process Each Image

Iterate through each image in the dataset, detect faces, and save the cropped face images:

```
for imagePath in tqdm(imagePaths):
    path_split = imagePath.split(os.sep)
    name_actor = path_split[-2]
    fn = path_split[-1]
    fn = fn.split('.')
    filename = fn[0]
    fileformat = fn[1]

dirdest = "new_mtcnn/training/" + name_actor + "/"
if not os.path.exists(dirdest):
    os.makedirs(dirdest)
    image = cv2.cvtColor(cv2.imread(imagePath), cv2.COLOR_BGR2RGB)
    result = detector.detect_faces(image)

    for i in range(len(result)):
```

```
bounding_box = result[i]['box']
keypoints = result[i]['keypoints']

bounding_box[0] = 0 if bounding_box[0] < 0 else bounding_box[0]
bounding_box[1] = 0 if bounding_box[1] < 0 else bounding_box[1]

path_save = dirdest + filename + "." + fileformat
img = image[bounding_box[1]:bounding_box[1] + bounding_box[3], \
    bounding_box[0]:bounding_box[0] + bounding_box[2]]
cv2.imwrite(path_save, cv2.cvtColor(img, cv2.COLOR_RGB2BGR))
```

Step 6: Verify the Output

Once the script completes execution, verify that the cropped face images are saved in the directory:

`new_mtcnn/training/{actor_name}/`

where {actor_name} represents the subdirectory corresponding to each person in the dataset.

---

5. Notes and Considerations

- Ensure that the dataset directory (new/training) contains correctly structured images.
- Adjust the bounding box settings if necessary to fine-tune face cropping.
- Use visualization tools like OpenCV (cv2.imshow()) to inspect detected faces before saving.
- If running on CPU, remove GPU configurations to avoid errors.

---

6. Conclusion

This guide provides a structured approach to using MTCNN for face detection on the FaceScrub dataset. By following these steps, users can effectively detect, extract, and store face images for further processing.