Group Number: 13
Members: Anushka Shreyam 20050, Mayur Mankar 20169
Course Name: AAAI

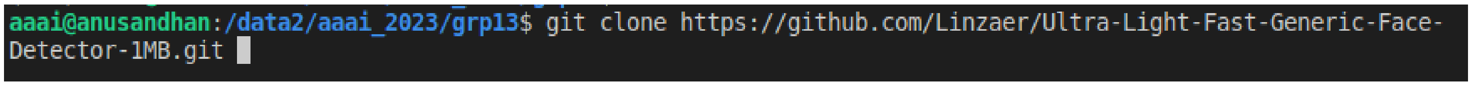
Assignment 1

Problem Statement:
1) In this assignment, we will set up a Docker environment for face detection using the "Ultra-Light-Fast-Generic-Face-Detector-1MB" repository.
2) Implement multi-threaded matrix multiplication on a (1000x1000) matrix.

Tools:
Pytorch:
PyTorch is a leading open-source deep learning framework developed by Facebook's AI Research lab (FAIR). It is renowned for its dynamic computation graph, making it highly flexible and suitable for research and development in artificial intelligence. PyTorch offers a powerful tensor computation library, automatic differentiation with Autograd, and GPU acceleration support. Its intuitive Pythonic interface and extensive neural network library have made it a popular choice for machine learning practitioners. PyTorch's active community and rich ecosystem contribute to its widespread adoption in both academia and industry.

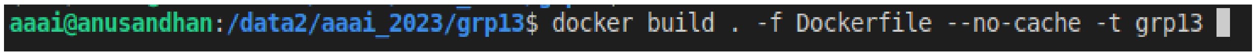
Docker:
Docker is a software platform that allows you to build, test, and deploy applications quickly. Docker packages software into standardized units called containers that have everything the software needs to run including libraries, system tools, code, and runtime. Using Docker, you can quickly deploy and scale applications into any environment and know your code will run.

Methodology:
Follow the steps below and ensure to meet the specified requirements.
1. Cloning the Github repository.



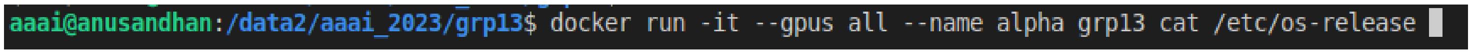
```
aaai@anusandhan:/data2/aaai_2023/grp13$ git clone https://github.com/Linzaer/Ultra-Light-Fast-Generic-Face-
Detector-1MB.git
```

2. Setup Docker environment.
a) Create Dockerfile
b) Build docker images using Dockerfile in the current directory.



```
aaai@anusandhan:/data2/aaai_2023/grp13$ docker build . -f Dockerfile --no-cache -t grp13
```

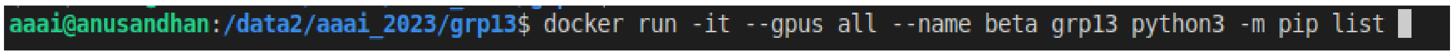
Here, the command `docker build .-f Dockerfile --no-cache -t grp13` is telling Docker to build an image using the Dockerfile in the current directory, without using cached layers, and to tag the resulting image with the name "grp13."
3. Perform environment verification.
a) Display information about the operating system within the container.



```
aaai@anusandhan:/data2/aaai_2023/grp13$ docker run -it --gpus all --name alpha grp13 cat /etc/os-release
```

Here, the command `docker run -it --gpus all --name alpha grp13 cat /etc/os-release` is telling Docker to run a new container based on the "grp13" image, allocate all available GPUs to the container, name the container "alpha," and execute the cat /etc/os-release command inside the container to display information about the operating system within the container.

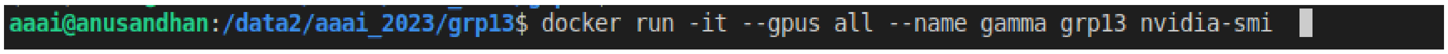
b) List installed Python packages and their versions.



```
aaai@anusandhan:/data2/aaai_2023/grp13$ docker run -it --gpus all --name beta grp13 python3 -m pip list
```

Here, the command `docker run -it --gpus all --name beta grp13 python3 -m pip` list is telling Docker to run a new container based on the "grp13" image, allocate all available GPUs to the container, name the container "beta," and execute the pip list command inside the container to list installed Python packages and their versions.

c) Display information about the NVIDIA GPUs on the host machine



```
aaai@anusandhan:/data2/aaai_2023/grp13$ docker run -it --gpus all --name gamma grp13 nvidia-smi
```

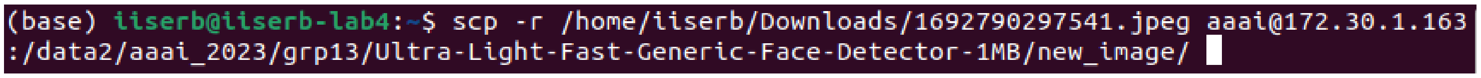
Here, the command `docker run -it --gpus all --name gamma grp13 nvidia-smi` is telling Docker to run a new container based on the "grp13" image, allocate all available GPUs to the container, name the container "gamma," and execute the nvidia-smi command inside the container to display information about the NVIDIA GPUs on the host machine. This is commonly used for checking GPU availability and status within a Docker container, especially in GPU-accelerated computing environments.

4. Detect faces from both videos and images provided.

a) Download the images from the drive and upload them in the working directory. The test image is kept in the folder “new_image” and test video is kept in the folder “new_video”.

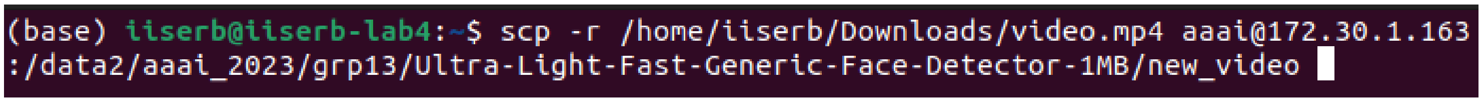
Drive link: https://drive.google.com/drive/folders/1wB-acqnSE6wJIETM7Xb980Wol3oObE5Q

* Command for uploading test image from the local system to the working directory (“new_image”).



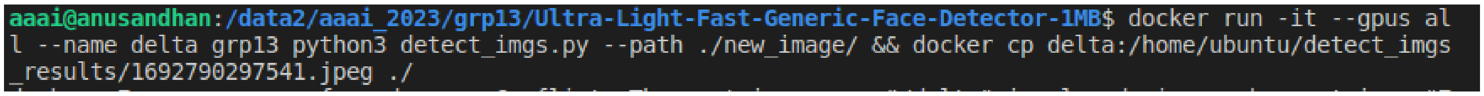
```
(base) iiserb@iiserb-lab4:~$ scp -r /home/iiserb/Downloads/1692790297541.jpeg aaai@172.30.1.163
:/data2/aaai_2023/grp13/Ultra-Light-Fast-Generic-Face-Detector-1MB/new_image/
```

* Command for uploading test video from the local system to the working directory (“new_video”).



```
(base) iiserb@iiserb-lab4:~$ scp -r /home/iiserb/Downloads/video.mp4 aaai@172.30.1.163
:/data2/aaai_2023/grp13/Ultra-Light-Fast-Generic-Face-Detector-1MB/new_video
```

b) Face detection in test image.



```
aaai@anusandhan:/data2/aaai_2023/grp13/Ultra-Light-Fast-Generic-Face-Detector-1MB$ docker run -it --gpus al
l --name delta grp13 python3 detect_imgs.py --path ./new_image/ && docker cp delta:/home/ubuntu/detect_imgs
_results/1692790297541.jpeg ./
```

* `docker run`: This is the Docker command used to run a Docker container from an image.

* `-it`: The -it options are typically used for running commands that require user input or for accessing an interactive shell.

* `--gpus all`: This option is used with Docker when you want to allocate access to all available GPUs on the host machine to the Docker container. It's specifically used for GPU-intensive workloads and requires Docker to be configured with GPU support.

* --name delta: This option is used to assign a custom name to the running container. In this case, the container is named "delta." Naming containers can make them easier to manage and reference.

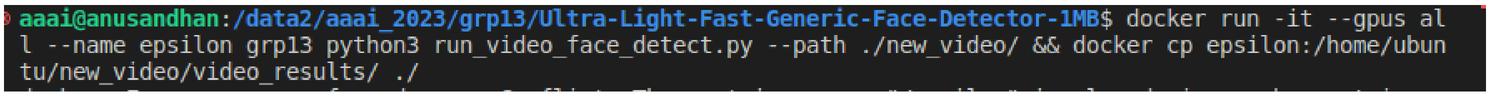
* `grp13`: This is the name of the Docker image from which the container will be created. It's specifying that the container should be based on the "grp13" Docker image.

* `python3 detect_imgs.py --path ./new_image/`: This is the command that will be executed inside the running container. It's running a Python script detect_imgs.py with the --path argument set to ./new_image/. The purpose of this script appears to be image detection or processing based on the provided path.

* `docker cp`: This is the Docker command used to copy files or directories between a running container and the host machine.

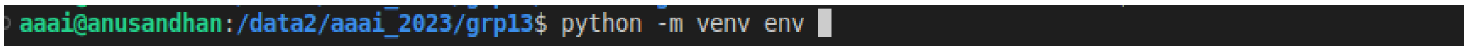
* `delta:/home/ubuntu/detect_imgs_results/1692790297541.jpeg`: This specifies the source path inside the running container (named "delta") from which the file should be copied. In this case, it's specifying

the path to a file named "1692790297541.jpeg" located in the "/home/ubuntu/detect_imgs_results/" directory inside the container.
* `./`: This specifies the destination path on the host machine where the file should be copied. In this case, it's copying the file to the current directory on the host machine.
c) Face detection in test video.



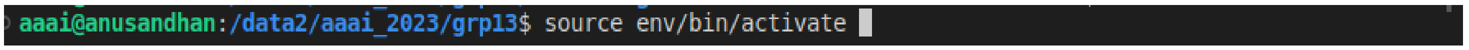
```
aaai@anusandhan:/data2/aaai_2023/grp13/Ultra-Light-Fast-Generic-Face-Detector-1MB$ docker run -it --gpus al
l --name epsilon grp13 python3 run_video_face_detect.py --path ./new_video/ && docker cp epsilon:/home/ubun
tu/new_video/video_results/ ./
```

* `python3 run_video_face_detect.py --path ./new_video/`: This is the command that will be executed inside the running container. It's running a Python script named run_video_face_detect.py and passing the --path argument as ./new_video/. This script seems to be related to video face detection based on the provided path.
* `epsilon:/home/ubuntu/new_video/video_results/`: This specifies the source path inside the running container (named "epsilon") from which the directory or files should be copied. In this case, it's specifying the path to a directory named "video_results" located in the "/home/ubuntu/new_video/" directory inside the container.
5. Implement multi-threaded matrix multiplication on a (1000x1000) matrix.
a) Create virtual environment.



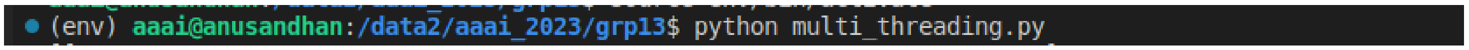
```
aaai@anusandhan:/data2/aaai_2023/grp13$ python -m venv env
```

b) Activate virtual environment.



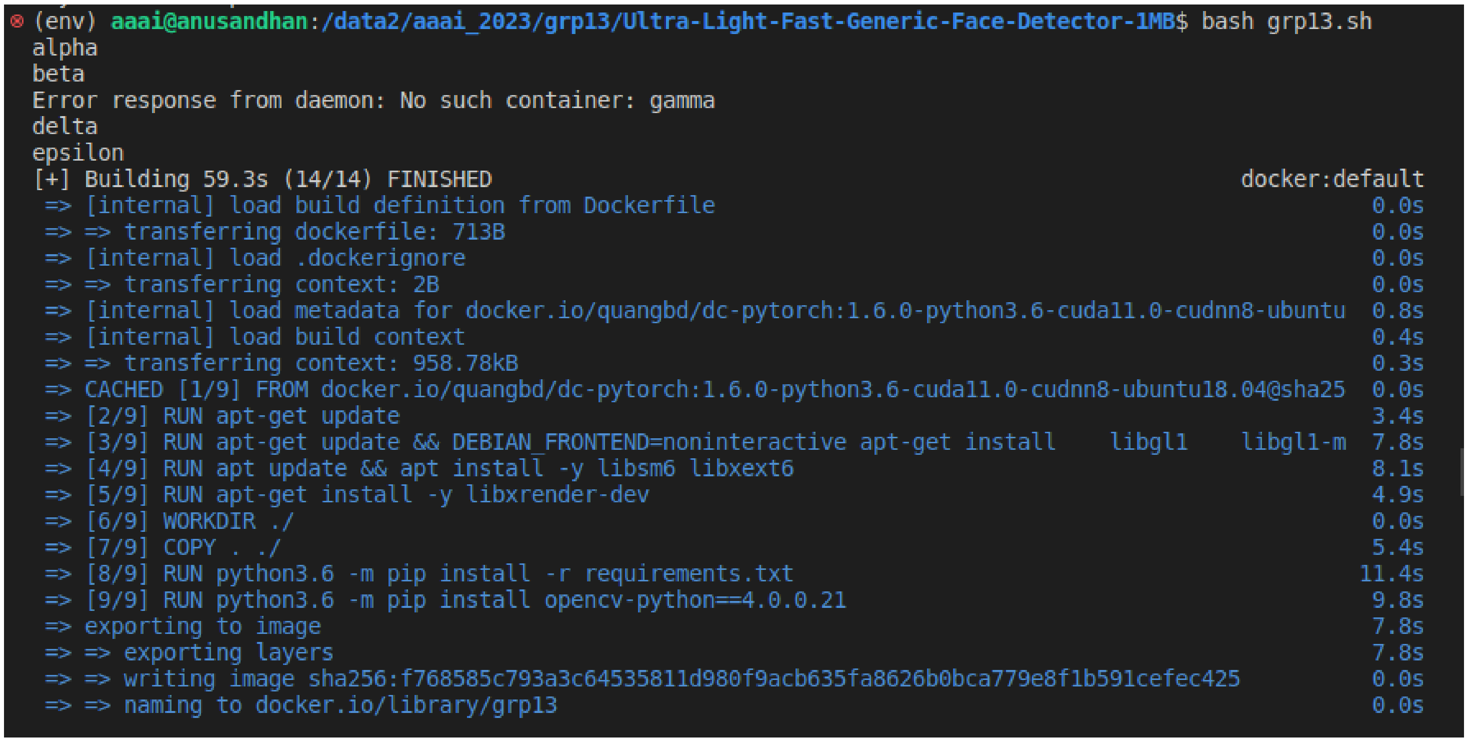
```
aaai@anusandhan:/data2/aaai_2023/grp13$ source env/bin/activate
```

c) Install required python libraries.
d) Run python script.



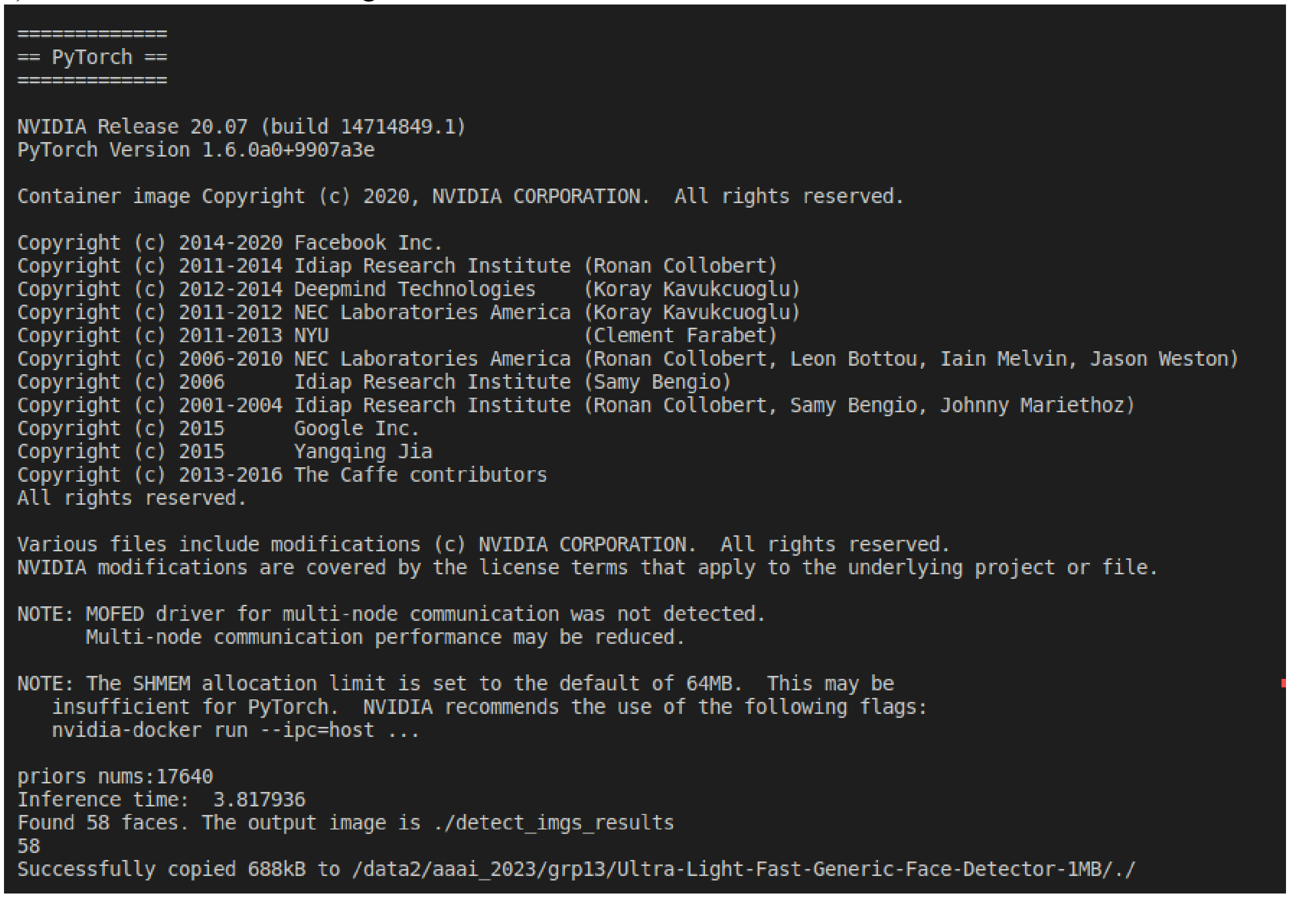
```
(env) aaai@anusandhan:/data2/aaai_2023/grp13$ python multi_threading.py
```

Results:
1. Face detection.
a) Create docker file and perform environment verification.



```
(env) aaai@anusandhan:/data2/aaai_2023/grp13/Ultra-Light-Fast-Generic-Face-Detector-1MB$ bash grp13.sh
alpha
beta
Error response from daemon: No such container: gamma
delta
epsilon
[+] Building 59.3s (14/14) FINISHED                                                     docker:default
 => [internal] load build definition from Dockerfile                                              0.0s
 => => transferring dockerfile: 713B                                                              0.0s
 => [internal] load .dockerignore                                                                 0.0s
 => => transferring context: 2B                                                                   0.0s
 => [internal] load metadata for docker.io/quangbd/dc-pytorch:1.6.0-python3.6-cuda11.0-cudnn8-ubuntu  0.8s
 => [internal] load build context                                                                 0.4s
 => => transferring context: 958.78kB                                                             0.3s
 => CACHED [1/9] FROM docker.io/quangbd/dc-pytorch:1.6.0-python3.6-cuda11.0-cudnn8-ubuntu18.04@sha25  0.0s
 => [2/9] RUN apt-get update                                                                      3.4s
 => [3/9] RUN apt-get update && DEBIAN_FRONTEND=noninteractive apt-get install    libgl1    libgl1-m  7.8s
 => [4/9] RUN apt update && apt install -y libsm6 libxext6                                        8.1s
 => [5/9] RUN apt-get install -y libxrender-dev                                                   4.9s
 => [6/9] WORKDIR ./                                                                              0.0s
 => [7/9] COPY . ./                                                                               5.4s
 => [8/9] RUN python3.6 -m pip install -r requirements.txt                                       11.4s
 => [9/9] RUN python3.6 -m pip install opencv-python==4.0.0.21                                    9.8s
 => exporting to image                                                                            7.8s
 => => exporting layers                                                                           7.8s
 => => writing image sha256:f768585c793a3c64535811d980f9acb635fa8626b0bca779e8f1b591cefec425      0.0s
 => => naming to docker.io/library/grp13                                                          0.0s
```

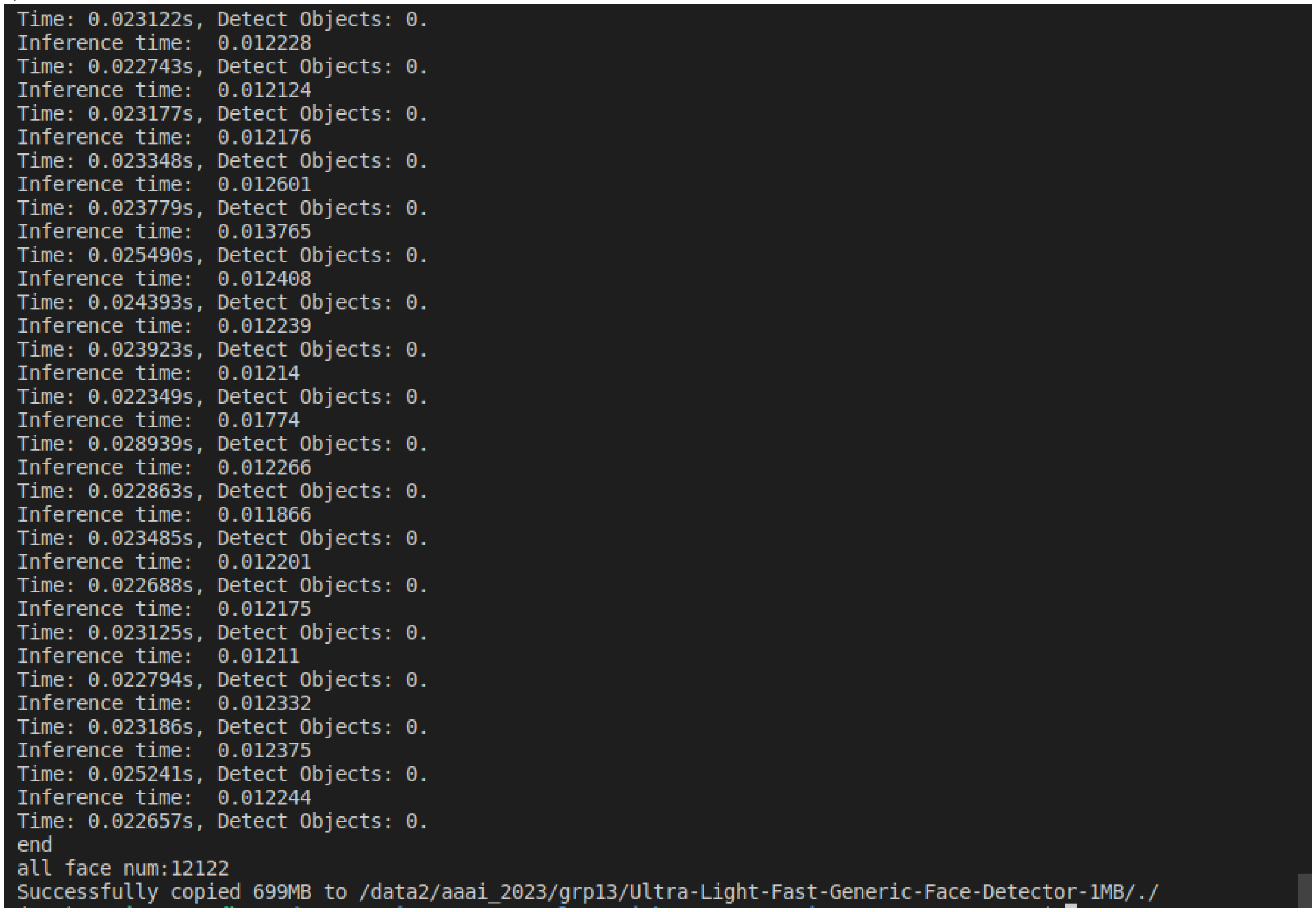
b) Face detection in test image.



```
=============
== PyTorch ==
=============

NVIDIA Release 20.07 (build 14714849.1)
PyTorch Version 1.6.0a0+9907a3e

Container image Copyright (c) 2020, NVIDIA CORPORATION.  All rights reserved.

Copyright (c) 2014-2020 Facebook Inc.
Copyright (c) 2011-2014 Idiap Research Institute (Ronan Collobert)
Copyright (c) 2012-2014 Deepmind Technologies    (Koray Kavukcuoglu)
Copyright (c) 2011-2012 NEC Laboratories America (Koray Kavukcuoglu)
Copyright (c) 2011-2013 NYU                      (Clement Farabet)
Copyright (c) 2006-2010 NEC Laboratories America (Ronan Collobert, Leon Bottou, Iain Melvin, Jason Weston)
Copyright (c) 2006      Idiap Research Institute (Samy Bengio)
Copyright (c) 2001-2004 Idiap Research Institute (Ronan Collobert, Samy Bengio, Johnny Mariethoz)
Copyright (c) 2015      Google Inc.
Copyright (c) 2015      Yangqing Jia
Copyright (c) 2013-2016 The Caffe contributors
All rights reserved.

Various files include modifications (c) NVIDIA CORPORATION.  All rights reserved.
NVIDIA modifications are covered by the license terms that apply to the underlying project or file.

NOTE: MOFED driver for multi-node communication was not detected.
      Multi-node communication performance may be reduced.

NOTE: The SHMEM allocation limit is set to the default of 64MB.  This may be
   insufficient for PyTorch.  NVIDIA recommends the use of the following flags:
   nvidia-docker run --ipc=host ...

priors nums:17640
Inference time:  3.817936
Found 58 faces. The output image is ./detect_imgs_results
58
Successfully copied 688kB to /data2/aaai_2023/grp13/Ultra-Light-Fast-Generic-Face-Detector-1MB/./
```

Found total 58 faces in the test image.

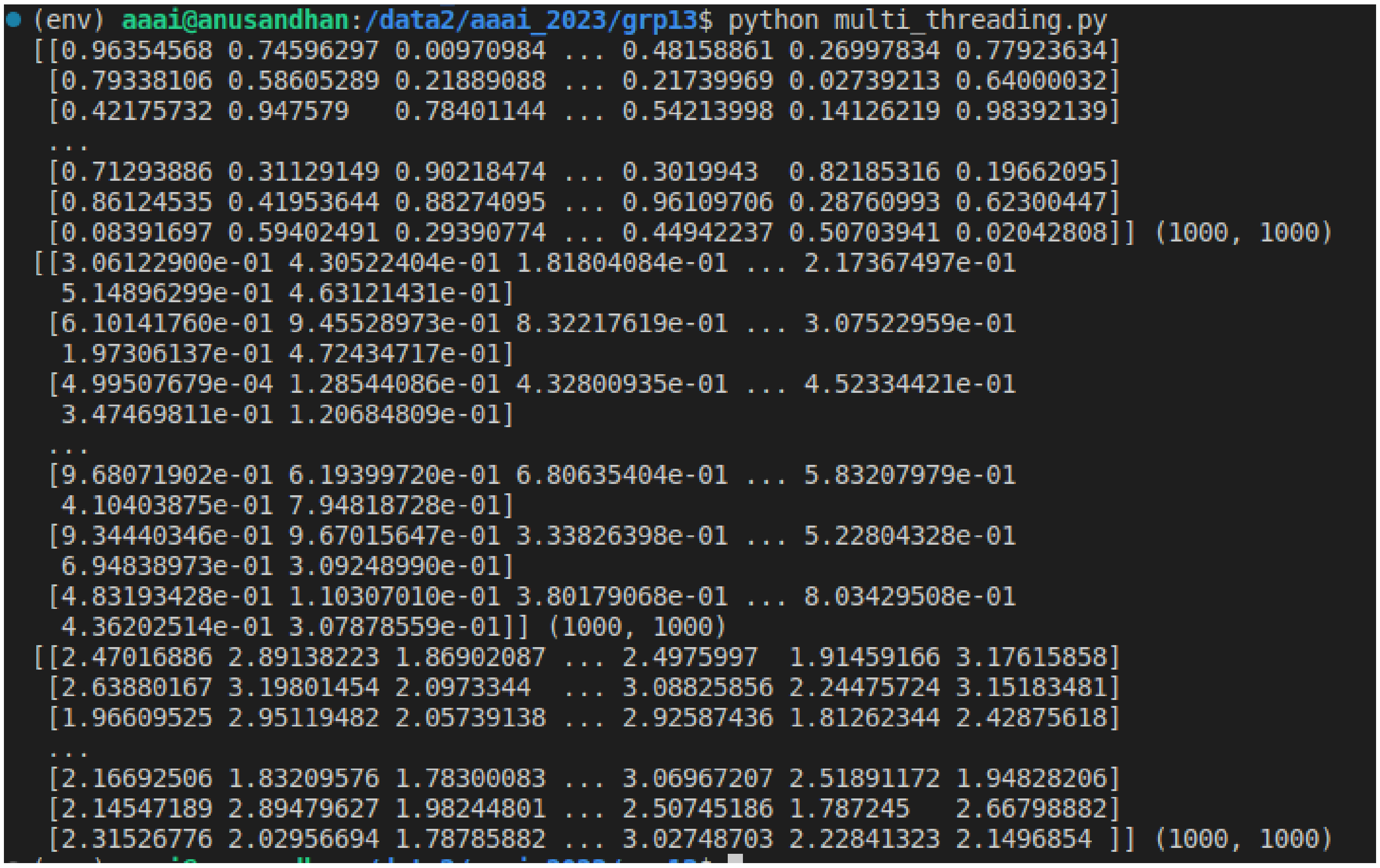
c) Face detection in test video.



```
Time: 0.023122s, Detect Objects: 0.
Inference time:  0.012228
Time: 0.022743s, Detect Objects: 0.
Inference time:  0.012124
Time: 0.023177s, Detect Objects: 0.
Inference time:  0.012176
Time: 0.023348s, Detect Objects: 0.
Inference time:  0.012601
Time: 0.023779s, Detect Objects: 0.
Inference time:  0.013765
Time: 0.025490s, Detect Objects: 0.
Inference time:  0.012408
Time: 0.024393s, Detect Objects: 0.
Inference time:  0.012239
Time: 0.023923s, Detect Objects: 0.
Inference time:  0.01214
Time: 0.022349s, Detect Objects: 0.
Inference time:  0.01774
Time: 0.028939s, Detect Objects: 0.
Inference time:  0.012266
Time: 0.022863s, Detect Objects: 0.
Inference time:  0.011866
Time: 0.023485s, Detect Objects: 0.
Inference time:  0.012201
Time: 0.022688s, Detect Objects: 0.
Inference time:  0.012175
Time: 0.023125s, Detect Objects: 0.
Inference time:  0.01211
Time: 0.022794s, Detect Objects: 0.
Inference time:  0.012332
Time: 0.023186s, Detect Objects: 0.
Inference time:  0.012375
Time: 0.025241s, Detect Objects: 0.
Inference time:  0.012244
Time: 0.022657s, Detect Objects: 0.
end
all face num:12122
Successfully copied 699MB to /data2/aaai_2023/grp13/Ultra-Light-Fast-Generic-Face-Detector-1MB/./
```

Found total 12122 faces in the test video.

2. Multi-threaded matrix multiplication.



```
● (env) aaai@anusandhan:/data2/aaai_2023/grp13$ python multi_threading.py
 [[0.96354568 0.74596297 0.00970984 ... 0.48158861 0.26997834 0.77923634]
  [0.79338106 0.58605289 0.21889088 ... 0.21739969 0.02739213 0.64000032]
  [0.42175732 0.947579   0.78401144 ... 0.54213998 0.14126219 0.98392139]
  ...
  [0.71293886 0.31129149 0.90218474 ... 0.3019943  0.82185316 0.19662095]
  [0.86124535 0.41953644 0.88274095 ... 0.96109706 0.28760993 0.62300447]
  [0.08391697 0.59402491 0.29390774 ... 0.44942237 0.50703941 0.02042808]] (1000, 1000)
 [[3.06122900e-01 4.30522404e-01 1.81804084e-01 ... 2.17367497e-01
   5.14896299e-01 4.63121431e-01]
  [6.10141760e-01 9.45528973e-01 8.32217619e-01 ... 3.07522959e-01
   1.97306137e-01 4.72434717e-01]
  [4.99507679e-04 1.28544086e-01 4.32800935e-01 ... 4.52334421e-01
   3.47469811e-01 1.20684809e-01]
  ...
  [9.68071902e-01 6.19399720e-01 6.80635404e-01 ... 5.83207979e-01
   4.10403875e-01 7.94818728e-01]
  [9.34440346e-01 9.67015647e-01 3.33826398e-01 ... 5.22804328e-01
   6.94838973e-01 3.09248990e-01]
  [4.83193428e-01 1.10307010e-01 3.80179068e-01 ... 8.03429508e-01
   4.36202514e-01 3.07878559e-01]] (1000, 1000)
 [[2.47016886 2.89138223 1.86902087 ... 2.4975997  1.91459166 3.17615858]
  [2.63880167 3.19801454 2.0973344  ... 3.08825856 2.24475724 3.15183481]
  [1.96609525 2.95119482 2.05739138 ... 2.92587436 1.81262344 2.42875618]
  ...
  [2.16692506 1.83209576 1.78300083 ... 3.06967207 2.51891172 1.94828206]
  [2.14547189 2.89479627 1.98244801 ... 2.50745186 1.787245   2.66798882]
  [2.31526776 2.02956694 1.78785882 ... 3.02748703 2.22841323 2.1496854 ]] (1000, 1000)
```

==