南昌大学实验报告

姓名：Qing Liu

学号：6130116184

邮箱地址：1119637652@qq.com

专业班级：Class 164 of Computer Science and Technology

实验日期：THU. May 23rd, 2019

课程名称：Cloud Computing Technology Experiments

实验项目名称

Cloudlet

实验目的

- Understanding the concept of Cloudlet Model
- Testing the cloudlet demo in your docker environment
- Writing your own report on experiencing the demo

实验基础

- Linux
- Docker
- Cloudlet
- Virtualization

实验步骤

- Pull an Android Environment in Docker

- Pull the image `mingc/android-build-box` from Docker Hub

```
docker pull mingc/android-build-box
```

It includes the following components:

- Ubuntu 18.04
- Android SDK 16 17 18 19 20 21 22 23 24 25 26 27 28
- (Android build tools 17.0.0 18.1.1 19.1.0 20.0.0 21.1.2 22.0.1 23.0.1 23.0.2 23.0.3 24.0.0 24.0.1 24.0.2 24.0.3 25.0.0 25.0.1 25.0.2 25.0.3 26.0.0 26.0.1 26.0.2 27.0.1 27.0.2 27.0.3 28.0.1 28.0.2 28.0.3
- Android NDK r18b
- extra-android-m2repository
- extra-google-m2repository
- extra-google-google_play_services
- Google API add-ons
- Android Emulator
- Constraint Layout
- TestNG
- Python 2, Python 3
- Node.js, npm, React Native
- Ruby, RubyGems
- fastlane
- Kotlin 1.3
- Flutter 1.2.1

- Faceswap Demos

Build the server

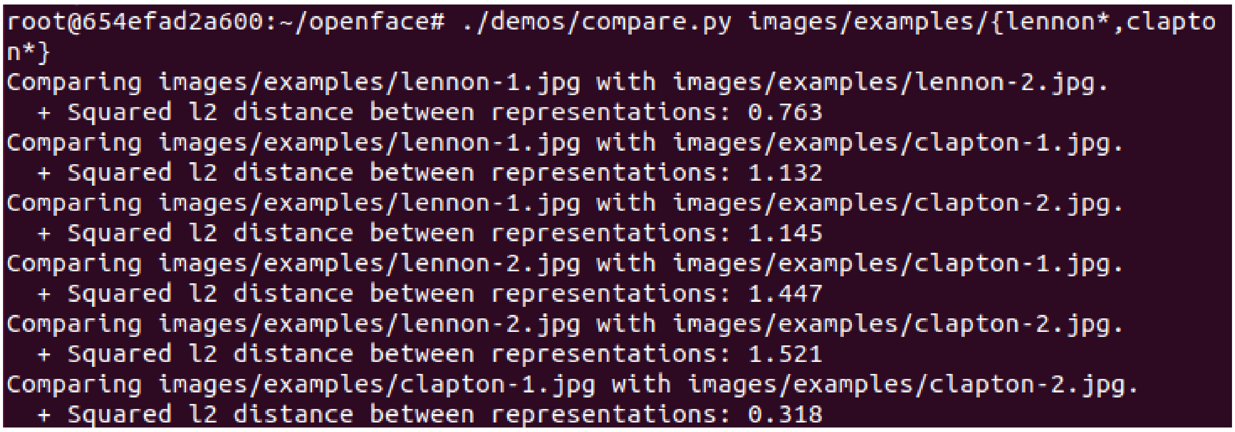
Configure in docker

1. Run openface on docker

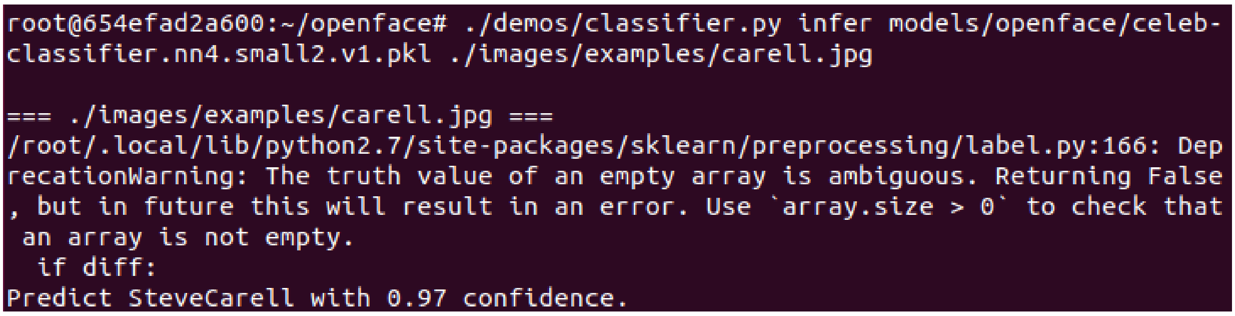
   The quickest way to getting started is to use our pre-built automated Docker build, which is available from bamos/openface. This does not require or use a locally checked out copy of OpenFace. To use on your images, share a directory between your host and the Docker container.

   ```
   docker pull bamos/openface
   docker run --name openface -p 9000:9000 -p 8000:8000 -t -i bamos/openface /b
   ```

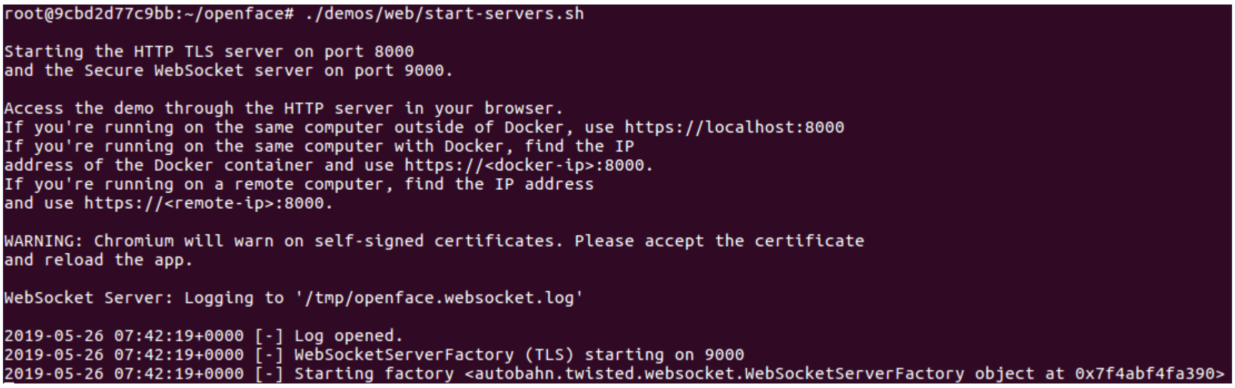
```
cd /root/openface
./demos/compare.py images/examples/{lennon*,clapton*}
./demos/classifier.py infer models/openface/celeb-classifier.nn4.small2.v1.p
./demos/web/start-servers.sh
```



```
root@654efad2a600:~/openface# ./demos/compare.py images/examples/{lennon*,clapto
n*}
Comparing images/examples/lennon-1.jpg with images/examples/lennon-2.jpg.
  + Squared l2 distance between representations: 0.763
Comparing images/examples/lennon-1.jpg with images/examples/clapton-1.jpg.
  + Squared l2 distance between representations: 1.132
Comparing images/examples/lennon-1.jpg with images/examples/clapton-2.jpg.
  + Squared l2 distance between representations: 1.145
Comparing images/examples/lennon-2.jpg with images/examples/clapton-1.jpg.
  + Squared l2 distance between representations: 1.447
Comparing images/examples/lennon-2.jpg with images/examples/clapton-2.jpg.
  + Squared l2 distance between representations: 1.521
Comparing images/examples/clapton-1.jpg with images/examples/clapton-2.jpg.
  + Squared l2 distance between representations: 0.318
```

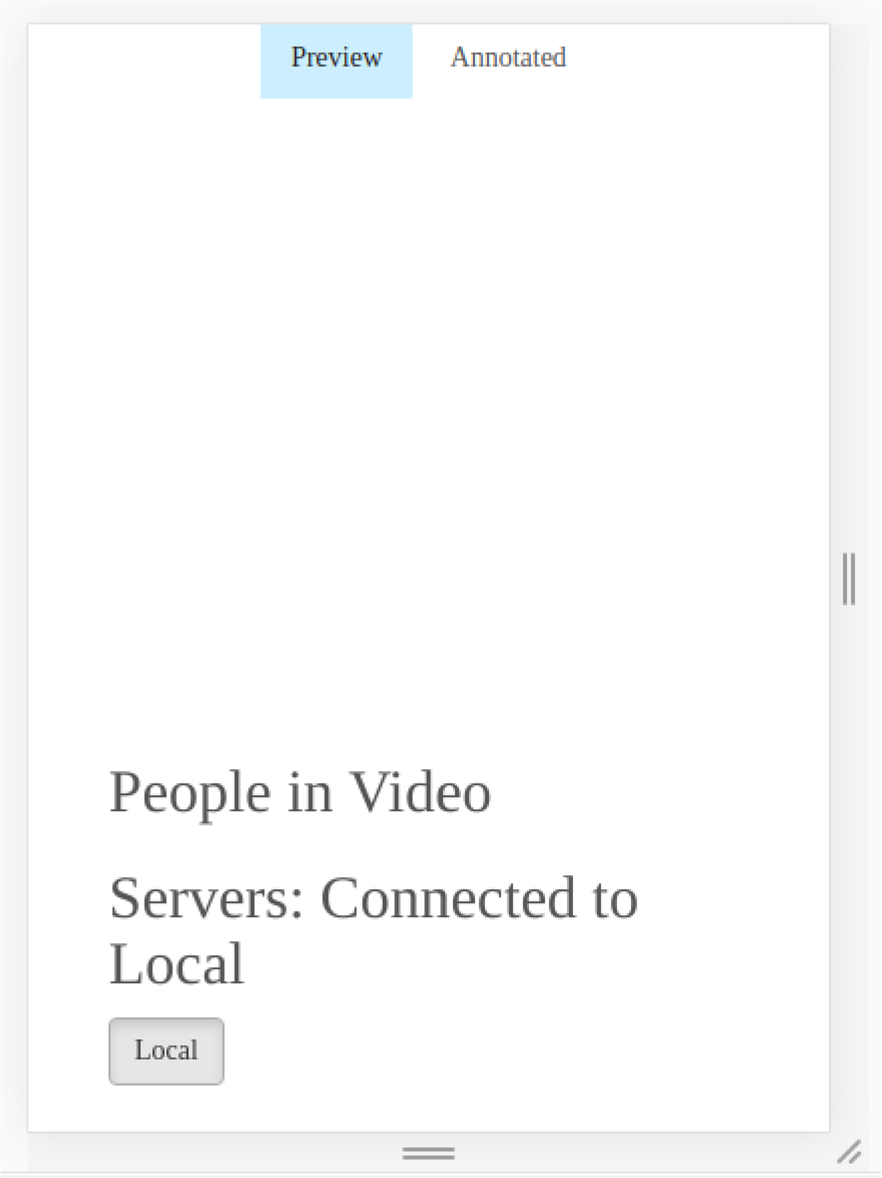


```
root@654efad2a600:~/openface# ./demos/classifier.py infer models/openface/celeb-
classifier.nn4.small2.v1.pkl ./images/examples/carell.jpg

=== ./images/examples/carell.jpg ===
/root/.local/lib/python2.7/site-packages/sklearn/preprocessing/label.py:166: Dep
recationWarning: The truth value of an empty array is ambiguous. Returning False
, but in future this will result in an error. Use `array.size > 0` to check that
 an array is not empty.
  if diff:
Predict SteveCarell with 0.97 confidence.
```



```
root@9cbd2d77c9bb:~/openface# ./demos/web/start-servers.sh

Starting the HTTP TLS server on port 8000
and the Secure WebSocket server on port 9000.

Access the demo through the HTTP server in your browser.
If you're running on the same computer outside of Docker, use https://localhost:8000
If you're running on the same computer with Docker, find the IP
address of the Docker container and use https://<docker-ip>:8000.
If you're running on a remote computer, find the IP address
and use https://<remote-ip>:8000.

WARNING: Chromium will warn on self-signed certificates. Please accept the certificate
and reload the app.

WebSocket Server: Logging to '/tmp/openface.websocket.log'

2019-05-26 07:42:19+0000 [-] Log opened.
2019-05-26 07:42:19+0000 [-] WebSocketServerFactory (TLS) starting on 9000
2019-05-26 07:42:19+0000 [-] Starting factory <autobahn.twisted.websocket.WebSocketServerFactory object at 0x7f4abf4fa390>
```

Visit the `https://localhost:8000`:



People in Video

Servers: Connected to Local

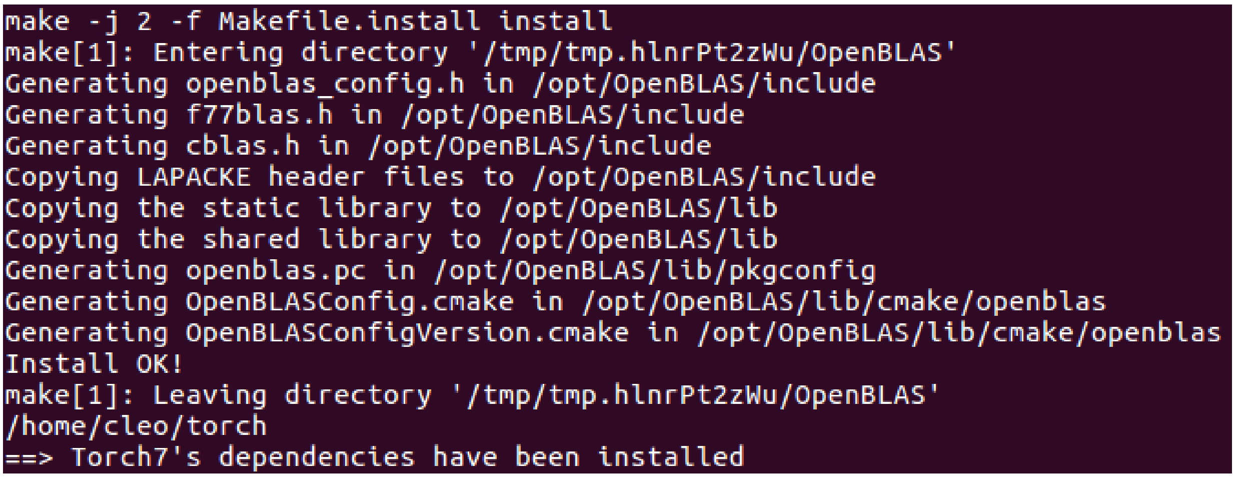
Do in Virtual Machine

1. Install Openface

```
git clone https://github.com/cmusatyalab/openface/
cd openface
python setup.py build
sudo python setup.py install
```

2. Install Torch

```
git clone https://github.com/torch/distro.git ~/torch --recursive
cd ~/torch; bash install-deps;
./install.sh
source ~/.bashrc
```



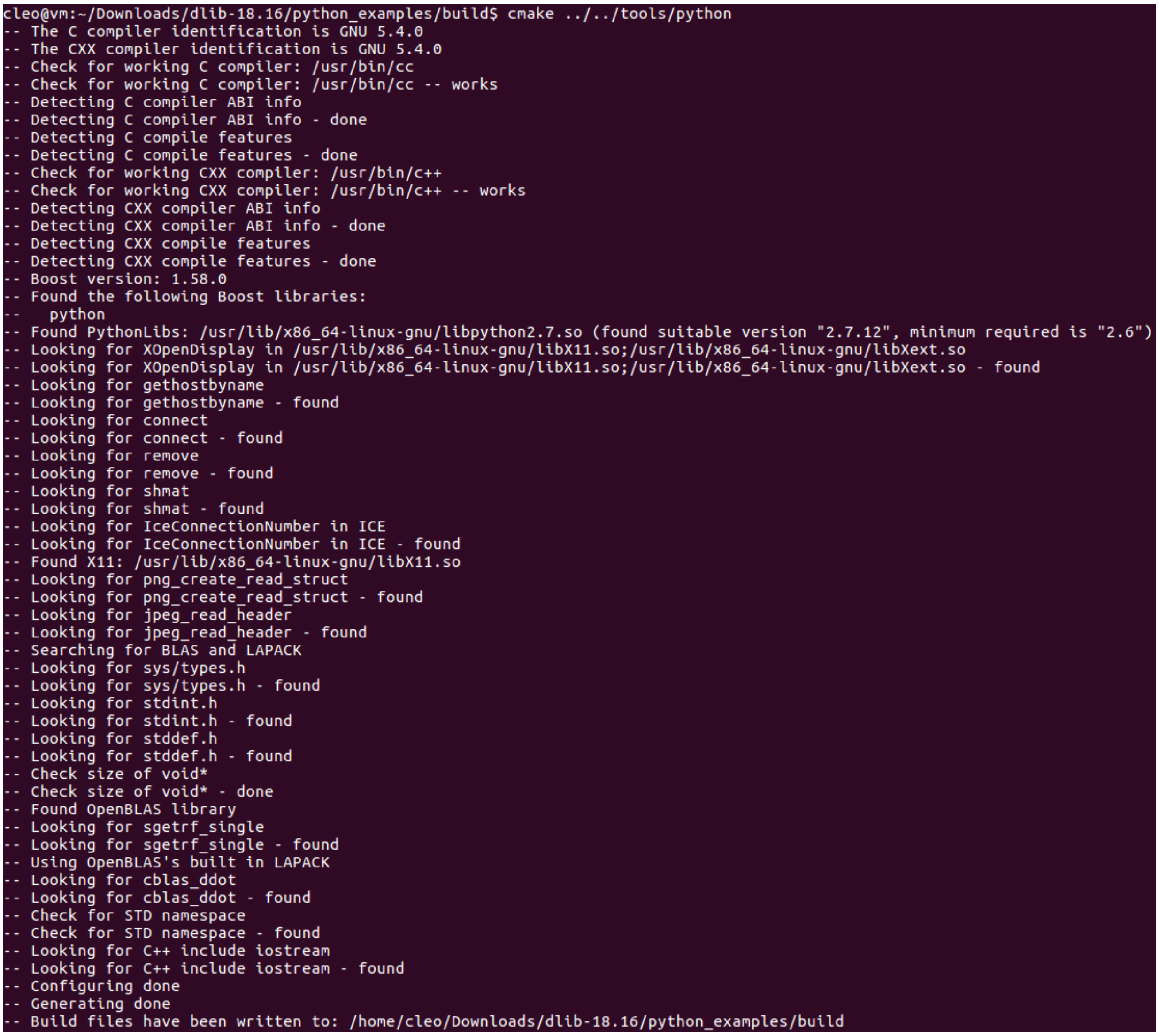
```
make -j 2 -f Makefile.install install
make[1]: Entering directory '/tmp/tmp.hlnrPt2zWu/OpenBLAS'
Generating openblas_config.h in /opt/OpenBLAS/include
Generating f77blas.h in /opt/OpenBLAS/include
Generating cblas.h in /opt/OpenBLAS/include
Copying LAPACKE header files to /opt/OpenBLAS/include
Copying the static library to /opt/OpenBLAS/lib
Copying the shared library to /opt/OpenBLAS/lib
Generating openblas.pc in /opt/OpenBLAS/lib/pkgconfig
Generating OpenBLASConfig.cmake in /opt/OpenBLAS/lib/cmake/openblas
Generating OpenBLASConfigVersion.cmake in /opt/OpenBLAS/lib/cmake/openblas
Install OK!
make[1]: Leaving directory '/tmp/tmp.hlnrPt2zWu/OpenBLAS'
/home/cleo/torch
==> Torch7's dependencies have been installed
```

3. Download this Gabriel release. Install its dependency and gabriel:

```
sudo apt-get install -y gcc python-dev default-jre python-pip pssh python-ps
sudo pip install \
        Flask==0.9 \
        Flask-RESTful \
        Jinja2==2.8 \
        MarkupSafe==0.23 \
        pycrypto \
        six \
        Werkzeug==0.11.10 &&
wget https://github.com/cmusatyalab/gabriel/archive/mobisys2016submission.zi
sudo apt-get install -y unzip &&
unzip mobisys2016submission.zip &&
cd gabriel-mobisys2016submission &&
sudo python setup.py install
```

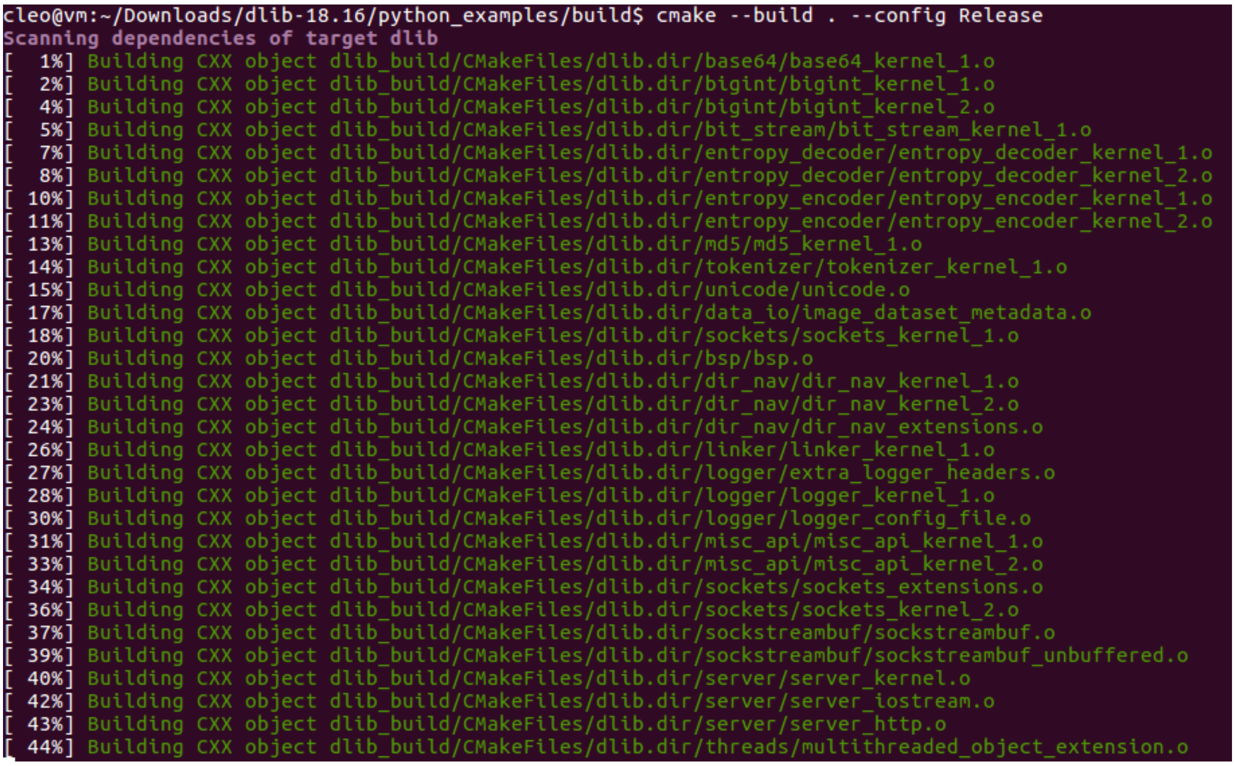
4. Install Opencv and dlib and other necessary dependency

```
pip install opencv-python
sudo apt-get install libboost-python-dev cmake

cd ~/Downloads
wget https://github.com/davisking/dlib/releases/download/v18.16/dlib-18.16.t
tar -xzvf dlib-18.16.tar.bz2
cd dlib-18.16
cd dlib-18.16/python_examples
mkdir build
cd build
cmake ../../tools/python
```



```
cleo@vm:~/Downloads/dlib-18.16/python_examples/build$ cmake ../../tools/python
-- The C compiler identification is GNU 5.4.0
-- The CXX compiler identification is GNU 5.4.0
-- Check for working C compiler: /usr/bin/cc
-- Check for working C compiler: /usr/bin/cc -- works
-- Detecting C compiler ABI info
-- Detecting C compiler ABI info - done
-- Detecting C compile features
-- Detecting C compile features - done
-- Check for working CXX compiler: /usr/bin/c++
-- Check for working CXX compiler: /usr/bin/c++ -- works
-- Detecting CXX compiler ABI info
-- Detecting CXX compiler ABI info - done
-- Detecting CXX compile features
-- Detecting CXX compile features - done
-- Boost version: 1.58.0
-- Found the following Boost libraries:
--   python
-- Found PythonLibs: /usr/lib/x86_64-linux-gnu/libpython2.7.so (found suitable version "2.7.12", minimum required is "2.6")
-- Looking for XOpenDisplay in /usr/lib/x86_64-linux-gnu/libX11.so;/usr/lib/x86_64-linux-gnu/libXext.so
-- Looking for XOpenDisplay in /usr/lib/x86_64-linux-gnu/libX11.so;/usr/lib/x86_64-linux-gnu/libXext.so - found
-- Looking for gethostbyname
-- Looking for gethostbyname - found
-- Looking for connect
-- Looking for connect - found
-- Looking for remove
-- Looking for remove - found
-- Looking for shmat
-- Looking for shmat - found
-- Looking for IceConnectionNumber in ICE
-- Looking for IceConnectionNumber in ICE - found
-- Found X11: /usr/lib/x86_64-linux-gnu/libX11.so
-- Looking for png_create_read_struct
-- Looking for png_create_read_struct - found
-- Looking for jpeg_read_header
-- Looking for jpeg_read_header - found
-- Searching for BLAS and LAPACK
-- Looking for sys/types.h
-- Looking for sys/types.h - found
-- Looking for stdint.h
-- Looking for stdint.h - found
-- Looking for stddef.h
-- Looking for stddef.h - found
-- Check size of void*
-- Check size of void* - done
-- Found OpenBLAS library
-- Looking for sgetrf_single
-- Looking for sgetrf_single - found
-- Using OpenBLAS's built in LAPACK
-- Looking for cblas_ddot
-- Looking for cblas_ddot - found
-- Check for STD namespace
-- Check for STD namespace - found
-- Looking for C++ include iostream
-- Looking for C++ include iostream - found
-- Configuring done
-- Generating done
-- Build files have been written to: /home/cleo/Downloads/dlib-18.16/python_examples/build
```

```
cmake --build . --config Release
```

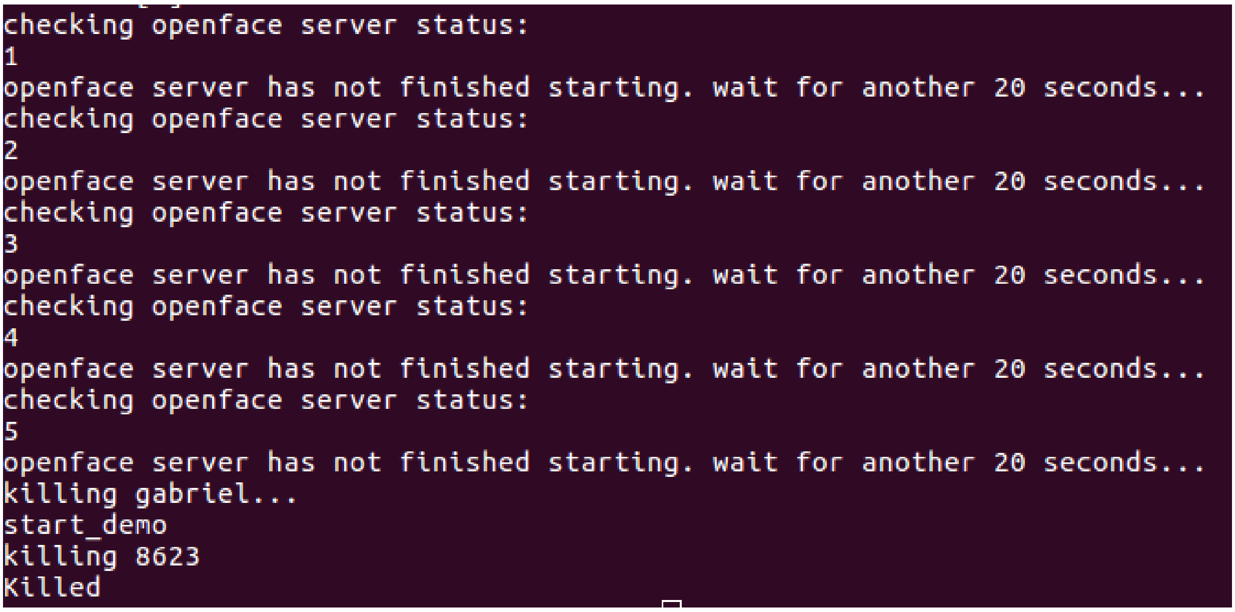


```
cleo@vm:~/Downloads/dlib-18.16/python_examples/build$ cmake --build . --config Release
Scanning dependencies of target dlib
[  1%] Building CXX object dlib_build/CMakeFiles/dlib.dir/base64/base64_kernel_1.o
[  2%] Building CXX object dlib_build/CMakeFiles/dlib.dir/bigint/bigint_kernel_1.o
[  4%] Building CXX object dlib_build/CMakeFiles/dlib.dir/bigint/bigint_kernel_2.o
[  5%] Building CXX object dlib_build/CMakeFiles/dlib.dir/bit_stream/bit_stream_kernel_1.o
[  7%] Building CXX object dlib_build/CMakeFiles/dlib.dir/entropy_decoder/entropy_decoder_kernel_1.o
[  8%] Building CXX object dlib_build/CMakeFiles/dlib.dir/entropy_decoder/entropy_decoder_kernel_2.o
[ 10%] Building CXX object dlib_build/CMakeFiles/dlib.dir/entropy_encoder/entropy_encoder_kernel_1.o
[ 11%] Building CXX object dlib_build/CMakeFiles/dlib.dir/entropy_encoder/entropy_encoder_kernel_2.o
[ 13%] Building CXX object dlib_build/CMakeFiles/dlib.dir/md5/md5_kernel_1.o
[ 14%] Building CXX object dlib_build/CMakeFiles/dlib.dir/tokenizer/tokenizer_kernel_1.o
[ 15%] Building CXX object dlib_build/CMakeFiles/dlib.dir/unicode/unicode.o
[ 17%] Building CXX object dlib_build/CMakeFiles/dlib.dir/data_io/image_dataset_metadata.o
[ 18%] Building CXX object dlib_build/CMakeFiles/dlib.dir/sockets/sockets_kernel_1.o
[ 20%] Building CXX object dlib_build/CMakeFiles/dlib.dir/bsp/bsp.o
[ 21%] Building CXX object dlib_build/CMakeFiles/dlib.dir/dir_nav/dir_nav_kernel_1.o
[ 23%] Building CXX object dlib_build/CMakeFiles/dlib.dir/dir_nav/dir_nav_kernel_2.o
[ 24%] Building CXX object dlib_build/CMakeFiles/dlib.dir/dir_nav/dir_nav_extensions.o
[ 26%] Building CXX object dlib_build/CMakeFiles/dlib.dir/linker/linker_kernel_1.o
[ 27%] Building CXX object dlib_build/CMakeFiles/dlib.dir/logger/extra_logger_headers.o
[ 28%] Building CXX object dlib_build/CMakeFiles/dlib.dir/logger/logger_kernel_1.o
[ 30%] Building CXX object dlib_build/CMakeFiles/dlib.dir/logger/logger_config_file.o
[ 31%] Building CXX object dlib_build/CMakeFiles/dlib.dir/misc_api/misc_api_kernel_1.o
[ 33%] Building CXX object dlib_build/CMakeFiles/dlib.dir/misc_api/misc_api_kernel_2.o
[ 34%] Building CXX object dlib_build/CMakeFiles/dlib.dir/sockets/sockets_extensions.o
[ 36%] Building CXX object dlib_build/CMakeFiles/dlib.dir/sockets/sockets_kernel_2.o
[ 37%] Building CXX object dlib_build/CMakeFiles/dlib.dir/sockstreambuf/sockstreambuf.o
[ 39%] Building CXX object dlib_build/CMakeFiles/dlib.dir/sockstreambuf/sockstreambuf_unbuffered.o
[ 40%] Building CXX object dlib_build/CMakeFiles/dlib.dir/server/server_kernel.o
[ 42%] Building CXX object dlib_build/CMakeFiles/dlib.dir/server/server_iostream.o
[ 43%] Building CXX object dlib_build/CMakeFiles/dlib.dir/server/server_http.o
[ 44%] Building CXX object dlib_build/CMakeFiles/dlib.dir/threads/multithreaded_object_extension.o
```

```
sudo apt-get install python-matplotlib
```

5. Download FaceSwap source code. Install its dependency and start it by invoking server/start_demo

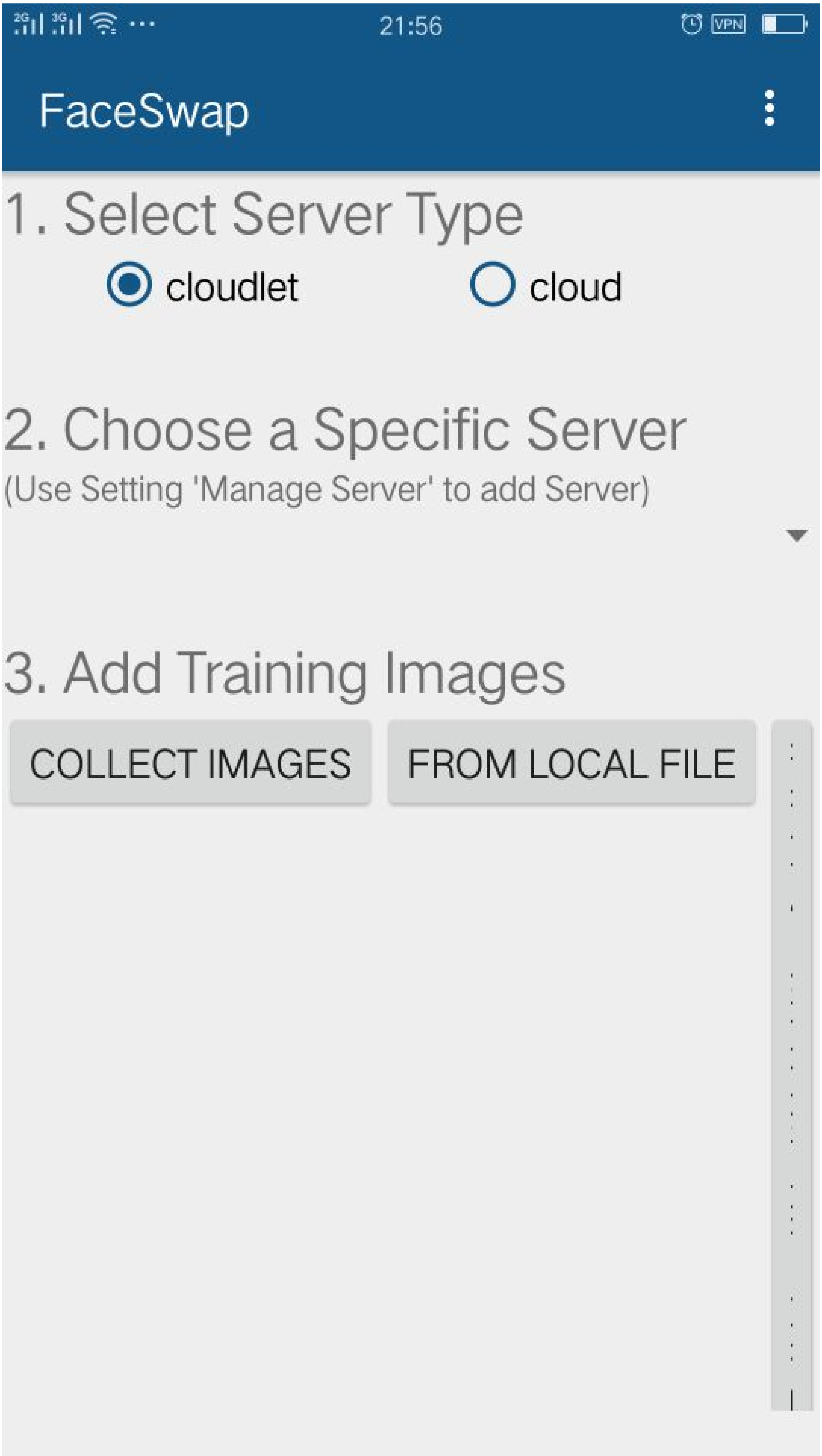
```
mkdir faceswap &&
cd faceswap &&
wget https://github.com/cmusatyalab/faceswap/archive/v1.0.zip &&
unzip v1.0.zip &&
cd ./faceswap-1.0/server/ &&
sudo pip install \
  websocket-client==0.35.0 \
  autobahn==0.10.4 \
  imagehash==1.0 \
  twisted==15.2.1 \
  scipy==0.14 \
  scikit-learn==0.17 \
  protobuf==2.5 &&
./start_demo.sh
```



```
checking openface server status:
1
openface server has not finished starting. wait for another 20 seconds...
checking openface server status:
2
openface server has not finished starting. wait for another 20 seconds...
checking openface server status:
3
openface server has not finished starting. wait for another 20 seconds...
checking openface server status:
4
openface server has not finished starting. wait for another 20 seconds...
checking openface server status:
5
openface server has not finished starting. wait for another 20 seconds...
killing gabriel...
start_demo
killing 8623
Killed
```

Build the client

I download the app from the github.



1. Select Server Type

● cloudlet ○ cloud

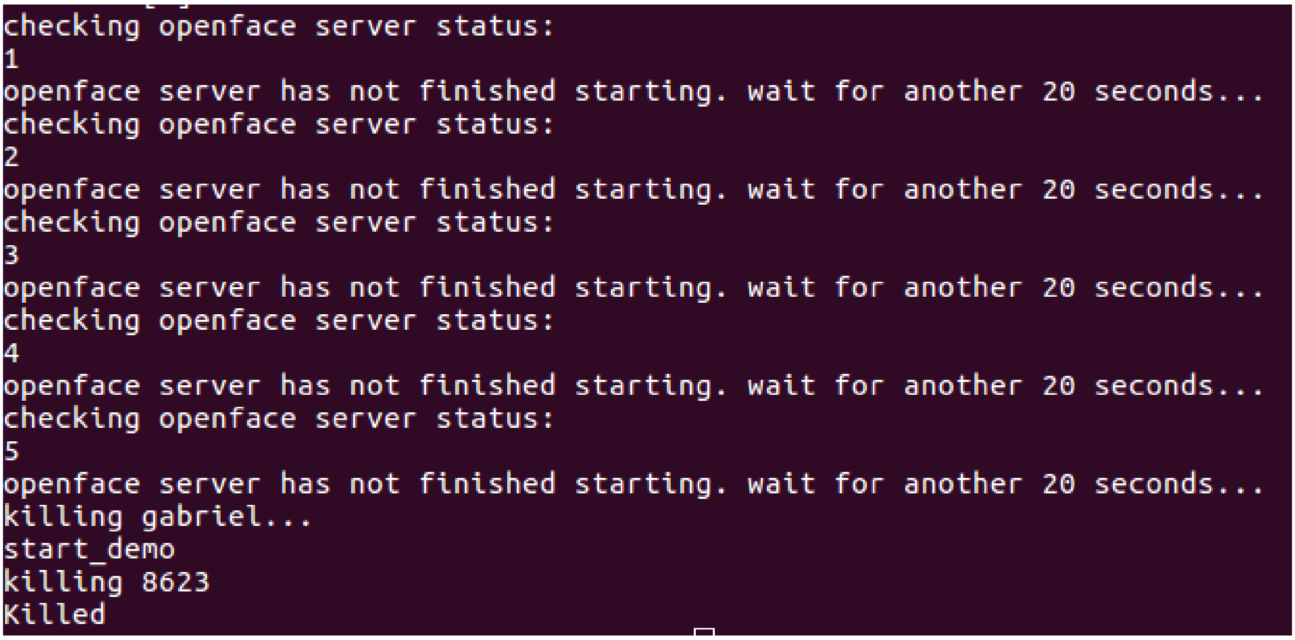
2. Choose a Specific Server

(Use Setting 'Manage Server' to add Server)

3. Add Training Images

实验数据或结果

Run the `start_demo.sh`



```
checking openface server status:
1
openface server has not finished starting. wait for another 20 seconds...
checking openface server status:
2
openface server has not finished starting. wait for another 20 seconds...
checking openface server status:
3
openface server has not finished starting. wait for another 20 seconds...
checking openface server status:
4
openface server has not finished starting. wait for another 20 seconds...
checking openface server status:
5
openface server has not finished starting. wait for another 20 seconds...
killing gabriel...
start_demo
killing 8623
Killed
```

实验思考

Cloudlet

Cloudlet is a trusted, resource-rich computer or cluster of computers that can be networked or not, allowing nearby devices to use. Cloudlet supports resource-intensive and interactive applications, and provides powerful computing resources for devices with lower latency. The term Cloudlet originated from the Mobile-Edge Computing Industry Project initiated by the European Telecommunications Standards Association.

Types

Cloudlet has two main architectural approaches.

- The first is transient cloudlet (Fig. 1). Based on the standard spoke-and-axis model, mobile users access nearby cloudlets via wireless LAN/RAN. Transient cloudlet relies on resource-rich computer infrastructure to provide data storage and computing services for mobile devices through wireless networks, mainly cellular and WLAN.

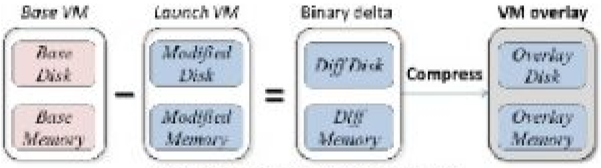
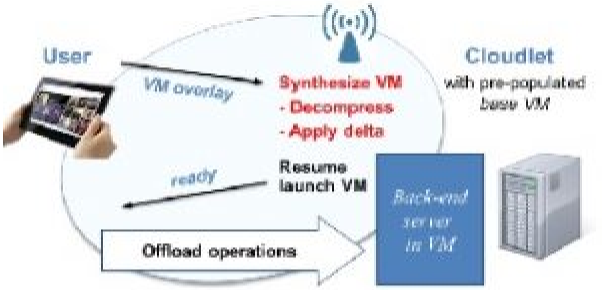
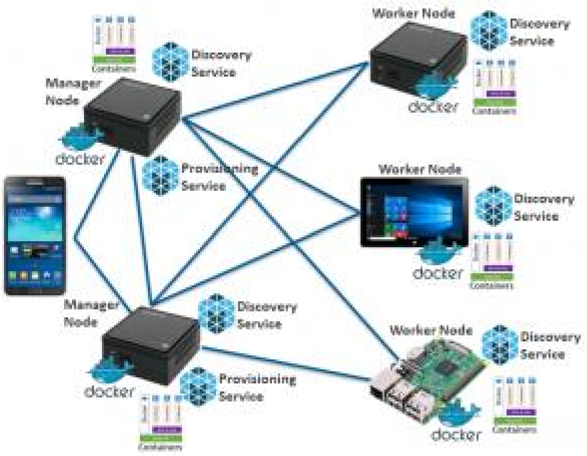


Figure 1: Creating VM overlay from Base VM



- The second type is mobile cloudlet, a group of resource rich mobile devices, called cloudlet nodes, which can be connected to each other in the mesh network to provide and use services. Mobile cloudlets rely on peer-to-peer networking, and a group of nearby mobile devices can be connected via secure Wi-Fi or Bluetooth. In this model, each mobile device acts as a node, sharing computing services in the network, using distributed computing principles.



Edge Computing

Edge computing is different from traditional centralized thinking. Its main computing nodes and applications are distributed in the data center near the terminal, which makes the service response performance and reliability higher than the traditional centralized cloud computing concept. Edge computing can be understood as an operation program that uses the edge zone near the data source. Edge computing should be a complement and optimization to cloud computing more accurately.

Deferences:

- In fact, if cloud computing is centralized large data processing, edge computing can be understood as edge large data processing. But the difference is that, just this time, the data can be resolved on the edge without having to be transmitted to distant clouds.

- Edge computing is more suitable for real-time data analysis and intelligent processing, and more efficient and secure than simple cloud computing. Both edge computing and cloud computing are actually a way of computing large data.

Edge computing has the following characteristics

- Edge computing focuses on real-time and short-period data analysis, which can better support real-time intelligent processing and execution of local services.
- Because the edge calculation is closer to the user, the data filtering and analysis are realized at the edge nodes, so the efficiency is higher.
- The combination of AI + edge computing makes edge computing more intelligent than computing.
- Cloud computing combined with edge computing costs only 39% of cloud computing alone.
- In the process of cloud transmission, some simple data processing is carried out through edge nodes, which can reduce the response time of devices and the data flow from devices to clouds.

参考资料

- https://blog.csdn.net/hongbin_xu/article/details/80223992
- https://cmusatyalab.github.io/openface/setup/
- https://cmusatyalab.github.io/faceswap/