Australian National University

ASSIGNMENT COVER SHEET

ANU College of Engineering and Computer Science
Australian National University
Canberra ACT 0200 Australia
www.anu.edu.au

+61 2 6125 5254

Submission and assessment is anonymous where appropriate and possible. Please do not write your name on this coversheet.

This coversheet must be attached to the front of your assessment when submitted in hard copy. If you have elected to submit in hard copy rather than Turnitin, you must provide copies of all references included in the assessment item.

All assessment items submitted in hard copy are due at 5pm unless otherwise specified in the course outline.

Student ID	Hanwen Bi (U6292748) and Chaoyun Gong (U6329142)
Team members	
Course Code	Engn 8535
Course Name	Data Analysis
Assignment Number	Project Report (Topic C. Learn embeddings for face recognition and object retrieval)
Due Date	
Date Submitted	Extension Granted

I declare that this work:

- ☐ upholds the principles of academic integrity, as defined in the ANU Policy: Code of Practice for Student Academic Integrity;
- ☐ is original, except where collaboration (for example group work) has been authorised in writing by the course convener in the course outline and/or Wattle site;
- ☐ is produced for the purposes of this assessment task and has not been submitted for assessment in any other context, except where authorised in writing by the course convener;
- ☐ gives appropriate acknowledgement of the ideas, scholarship and intellectual property of others insofar as these have been used;
- ☐ in no part involves copying, cheating, collusion, fabrication, plagiarism or recycling.

Signature

Real-time Face Detection and Recognition Based on FaceNet

1. Introduction

Faces, containing lots of important information, is an essential part in computer vision applications. In this paper, we introduce main methods of FaceNet, and present the implementation of FaceNet method in real-time face recognition. The FaceNet was developed in 2015 which created a new record on Labeled Faces in the Wild (LFW) dataset with 99.63% accuracy at that time [1]. FaceNet can be used to do face verification, face recognition, and face clustering. The main idea of FaceNet is using the deep convolutional network to learn the embedding which projects the image into a. feature space $\mathbb{R}^d$. By doing that, the European distance in the embedding space can directly indicate the face similarity: small distance means that faces come from the same person, and a large distance indicates that faces belong to different persons [1]. Another breakthrough in FaceNet is that the author creates triplet loss which finds an optimal margin between faces from one person and faces from other people.

In our implementation, we use the trained FaceNet model as a feature extractor. Firstly, we take photos of Chaoyun and I, and use MTCNN to extract the face region. After that, these face images are inputted to FaceNet model to get 128 dimensions embeddings and use these embedding to train the SVM classifier. In the real-time face recognition part, we access the camera in the laptop and input each frame of the vide to MTCNN to get the face region, and then these face images will be classified by the trained SVM model. Finally, the computer screen shows the face region and the name of the face, Chaoyun, Hanwen, or others.

The remainder of this report is organized as follows: in section 2 we will review the literature in this field, 3.1 describes the main method in FaceNet, and 3.2 describes the method of our real time face recognition. Finally, in section 4, real time face recognition results will be presented and analyzed.

2. Related work

For the architecture of deep convolution network, Szegedy *et al* [2] created GooLeNet which parallelly runs multiple different convolutional and pooling layers and connect their responses. This architecture can improve the networks, especially in the computer vision area, and it was used to win ImageNet 2014.

For the recent implementation of FaceNet, *Heath & Guibas* [3] created a real-time algorithm to track people's trajectories and get their frontal face images. In their approach, FaceNet is used to evaluate the frontal face images they acquired on real-time data.

J. Bhattacharya et al [4] employed a real-time face verification tool aiming to help visually impaired people to recognize people who enter their view region. They use two variants of FaceNet to extract the features of face images and compares performances of two FaceNet variants.

Z. Ming et al [5] use the intra/inter- class distance metric learning method to develop a simple class-wise triplet loss which greatly reduces the number of

learned triplets, without reducing the overall performance of FaceNet method.

Wu, Liu &d Z. Su [6] employed deep learning based real-time recognition from video stream method. In their method, they use MTCNN as face detector and use FaceNet as a feature extractor. After getting feature vectors from FaceNet, they use SVM classifier to quickly identify face images.

Wan & Lee [7] proposed a novel face sketch image recognition method based FaceNet, which firstly transfer images to gallery sketch style face images and compare the transferred image with sketch images. FaceNet is used to compare the L2 distances between a transferred image and a sketch image and do the recognition.

Boka &Morris [8] developed a complete system for monitoring by using facial recognition. In their project, FaceNet is used as a face recognition method before doing FRT access visualization.

X. Zhao et al [9] employed the JADE-MTCNN-FaceNet algorithm which combining JADE, MTCNN and FaceNet algorithm for blind source face images separation, and their simulation results show that this algorithm can increase the accuracy rate on the LFW database.

W. Chu & W. Li [10] presented a Manga FaceNet method to detect and recognize mange faces based on deep neural network, which is an interesting area of face detection and recognition.

3. Methodology

In this section, we will briefly discuss the main methods in FaceNet and the methods we used to implement real-time face recognition. The FaceNet method section contains the description of the structure of FaceNet, triplet loss, triplet selection, and deep convolutional networks structure. In the methods of our approach, we present our implementation step by step.

3.1 Methodology for FaceNet

Fig. 1. Structure of FaceNet [1]

Figure 1 shows the structure of FaceNet, where lies the input image batch layer followed by deep convolutional network and L2 normalization, before the embedding step and triplet loss. The triplet loss is a kind of loss function defined by the author of FaceNet which can minimize the distance between two face images from the same person and maximize the distance between two face images from different identities. The formula of triplet loss is shown below

$$L = \sum_{i}^{N} \left[\| f(x_i^a) - f(x_i^p) \|_2^2 - \| f(x_i^a) - . f(x_i^n) \|_2^2 + \alpha \right]_+$$
$$\forall (f(x_i^a), f(x_i^p), f(x_i^n)) \in \mathcal{T}$$

Where $f(x) \in \mathbb{R}^d$ embed the image into a d-dimensional feature vector which is on the d-dimensional hypersphere by making $\| f(x) \|_2 = 1$. x_i^a (anchor) is the target face image. x_i^p (positive) is the face image from the same identity with the target image. x_i^n (negative) is the face image from different identity of the target image. $\mathcal{T}$ is the possible triplet set.

For the selection of triplets, as there are so many possible triplets in the training set, it is important to choose hard triplets which means choosing positive face

image with large distance with the anchor and negative face image with small distance with the anchor. The author uses online generate method to select triplets by choosing the hard positive and negative faces in a mini-batch. During the experiment, the author found that using all anchor-positive pairs in the mini-batch can result in a converging fast and stably. Rather than choosing the hardest negative face, the author finally selected the hard anchor-negative pair with larger distance than positive-anchor pairs. The equation below shows that

$$\| f(x_i^a) - f(x_i^p) \|_2^2 < \| f(x_i^a) - f(x_i^n) \|_2^2$$

This triplet selection strategy can make the triplet loss converge stably and fast.

Stochastic Gradient Descent (SGD) is used to train the deep convolutional networks of FaceNet. Two network architectures were used in FaceNet. One is based on the model of Zeiler&Fergus [11], and the table 1 shows the structure details of this model. The other model is based on GoogLeNet [2], and figure 2 shows the structure of this model. The two model have around 6.6 million to 7.5 million parameters and 500 million to 1.6 billion FLOPS.

Table 1. structure of **NN1** model [1]

layer	size-in	size-out	kernel	param	FLPS
conv1	220×220×3	110×110×64	7×7×3, 2	9K	115M
pool1	110×110×64	55×55×64	3×3×64, 2	0	
rnorm1	55×55×64	55×55×64		0	
conv2a	55×55×64	55×55×64	1×1×64, 1	4K	13M
conv2	55×55×64	55×55×192	3×3×64, 1	111K	335M
rnorm2	55×55×192	55×55×192		0	
pool2	55×55×192	28×28×192	3×3×192, 2	0	
conv3a	28×28×192	28×28×192	1×1×192, 1	37K	29M
conv3	28×28×192	28×28×384	3×3×192, 1	664K	521M
pool3	28×28×384	14×14×384	3×3×384, 2	0	
conv4a	14×14×384	14×14×384	1×1×384, 1	148K	29M
conv4	14×14×384	14×14×256	3×3×384, 1	885K	173M
conv5a	14×14×256	14×14×256	1×1×256, 1	66K	13M
conv5	14×14×256	14×14×256	3×3×256, 1	590K	116M
conv6a	14×14×256	14×14×256	1×1×256, 1	66K	13M
conv6	14×14×256	14×14×256	3×3×256, 1	590K	116M
pool4	14×14×256	7×7×256	3×3×256, 2	0	
concat	7×7×256	7×7×256		0	
fc1	7×7×256	1×32×128	maxout p=2	103M	103M
fc2	1×32×128	1×32×128	maxout p=2	34M	34M
fc7128	1×32×128	1×1×128		524K	0.5M
L2	1×1×128	1×1×128		0	
total				140M	1.6B

Fig. 2. Structure of GoogLeNet model [2]

3.2 Real-time face recognition

This project tries to apply the FaceNet to do face Recognition in real time by using the camera. The structure of implement steps is below:

1. Use camera to take video in real time. and extract each frame as input image.
2. Apply MTCNN to detect and extract faces in images.
3. Train the FaceNet neural network by using triplet loss, then save the trained model.
4. Put the extracted faces into the trained FaceNet neural network model and get the embedding feature vectors.
5. Apply SVM to classify the embedding vectors for different peoples.
6. Box the face in monitor of camera in real time and label it by the predict name.

Firstly, the prepared work is to download and initialize the neural network for the project. Downloading MTCNN and FaceNet neural network from Github. For FaceNet model, this project will call src/FaceNet.py program. And this project will use an pre-trained model for FaceNet because of the lack of the GPU in computer. Copy FaceNet.py file in FaceNet to MTCNN-Tensorflow-master/test in MTCNN. Then, this project will create a new .py file named RealtimeIdentification.py in MTCNN-Tensorflow-master/test to implement the purpose- face recognition in real time.
In RealtimeIdentification.py, there are some packages should be imported. The list is below:

```
import sys
sys.path.append("../")
import tensorflow as tf
import numpy as np
import argparse
import facenet
import os
import sys
import math
import pickle
from scipy import misc
from Detection.MtcnnDetector import MtcnnDetector
from Detection.detector import Detector
from Detection.fcn_detector import FcnDetector
from train_models.mtcnn_model import P_Net, R_Net, O_Net
import cv2
from sklearn.svm import SVC
```

Fig. 3. The import packages of RealtimeIdentification.py

Secondly, construct the face dataset. In this project, we build two persons dataset (the two team members: Chaoyun and Hanwen). We take several photos for each person in different angles and facial expressions and save it in one file named by the person's name. Thus, there are two files (Chaoyun and Hanwen) in the data_face file.

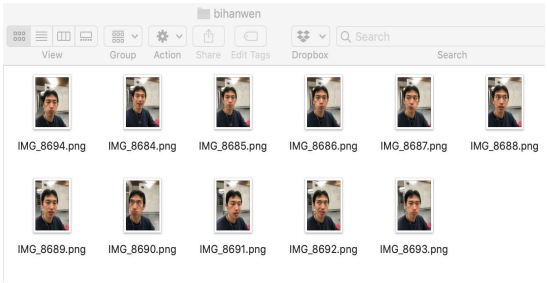

Fig. 4. The file of Hanwen's dataset

Figure 4 shows the photos in Hanwen file. However, the size of input image of FaceNet neural network model is 160*160*3. Thus, this project will apply MTCNN neural network to capture the face in photo and resize it to 160*160*3. The program named align_dataset_mtcnn.py in FaceNet-master/src/align. The input of this program is that: the input file path, the output file path --image_size 160 --margin 32 --random_order. After that, the output file will have 160*160*3 sampled face images and have the same number of

input photos. The result outputs of the model by inputting Hanwen's photos are below:

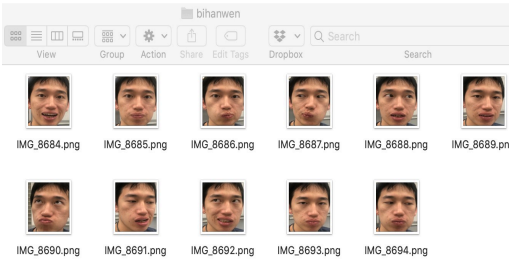

Fig. 5. The file of Hanwen's tailored faces

From figure 5, the performance of MTCNN neural network is very well. Each photo of Hanwen has been cropped the face correctly.

When the face dataset is ready, FaceNet model can be used to construct the face database by applying the face dataset to FaceNet model and getting the embedding feature vectors of them. The database stores the embedding feature vectors for each face of people (each face will be represented by a vector with 1*128 features).

In RealtimeIdentification.py, the function *face2database()* has been used to implement the whole second step. Finally, the function *face2database()* will construct the database of different peoples' faces and save it.

Thirdly, train the SVM classifier. Now, the stored database in second step can be used to train the SVM classifier. The SVM classifier in RealtimeIdentification.py named *ClassifyTrainSVC()*. We input the database of embedding faces. And it outputs the SVM classifier model (SVCmodel.pkl) and saves it in SVCpath (which path has been defined in main function). In SVM classifier, it will output the nominalized similar score for each class (from 0 to 1). Then choosing the class which has the highest score as the predict class.

Fourthly, implement the real time face recognition. In RealtimeIdentification.py, the function *RTrecognization()* is used to implement this it. The algorithm is that:

1. Load the FaceNet model which has been download and SVM classifier model which is stored in third step.
2. Occupy the input image and output embedding vector for FaceNet model by using TensorFlow.
3. Call camera in computer and get the frame of camera video as image in real time.
4. Apply MTCNN to detect the faces for each image.
5. Apply lines to box the face in each frame of video.
6. Put the faces to FaceNet model then apply SVM model to classify the embedding vector (the output of FaceNet model).
7. Label the SVM model result in box of face in video frame.

Finally, apply main function to define all the paths and call the functions.

4. Results

4.1 FaceNet

This project is highly based on the FaceNet. The further implement is based on the FaceNet model. One of the core algorithms of FaceNet is the loss function, Triplet loss. In FaceNet, the original author has trained the dataset by using triple loss and softmax separately to

compare the difference. If the data set is small, softmax is easier to converge. However, when the data set is very large (includes many different individuals, over 100,000), The output of last softmax layer will be very large, but the training by using Triple loss can still work effectively.

In the F. Schroff, D. Kalenichenko, J. Philbin's paper, they apply six neural network models to train the data set and compare the accuracy by using validation. And the validation rate is 10e-3.

The first neural network NN1 is from Zeiler&Fergus [11]. It adds 1*1*d convolution layer in the beginning several convolution layers. It reduces the number of parameters effectively. However, compare with GoogleNet, the number of parameters is still large.

The NN2 is GoogleNet [2]. The differences are that:

1. Use L2 pooling instead of max pooling.
2. The convolution kernel of pooling layer is always 3*3 (except the last average pooling). And it is parallel to the convolution module in each input module.
3. If there is a dimension drop after 1*1, 3*3, 5*5 pooling, then connect them as the final output.

NN3 and NN4 are also GoogLeNet, the same structure of NN2. However, the input image size is 160*160 and 96*96 respectively. It reduces the calculation time for CPU.

NNS1 and NNS2 are the Cropped GoogLeNet. In order to make the model embed in mobile devices, the model is also cropped in FaceNet paper. NNS1 requires only 26M parameters and 220M floating point operations overhead. NNS2 requires only 4.3M parameters and 20M floating point operations overhead.

Then the accuracies of the six neural networks are below:

Table 3. The accuracy of different neural network [1]

architecture	VAL
NN1 (Zeiler&Fergus 220×220)	87.9% ± 1.9
NN2 (Inception 224×224)	89.4% ± 1.6
NN3 (Inception 160×160)	88.3% ± 1.7
NN4 (Inception 96×96)	82.0% ± 2.3
NNS1 (mini Inception 165×165)	82.4% ± 2.4
NNS2 (tiny Inception 140×116)	51.9% ± 2.9

From Table 3, it is obvious that NN2 (GoogleNet) has the highest accuracy 89.4%. However, the input size of image is too large, the neural network will be very complex. NN3 just has half size of NN2 approximately, and the accuracy is 88.3%, just 1% lower than NN2. And also, from the Table2, it shows the effect of input image quality of model accuracy. 25,600 pixels is corresponding to NN3, and 65,536 is corresponding to NN2. The accuracies of them are also similar. Thus, in this project, we just apply NN3. It can make the model be more effective. So, before training for FaceNet neural network, the image will be extracted the face by MTCNN and resize to 160*160*3.

Table 4. The accuracy of image pixels [1]

#pixels	val-rate
1,600	37.8%
6,400	79.5%
14,400	84.5%
25,600	85.7%
65,536	86.4%

The LFC accuracies of the FaceNet neural network model are shown in table 5:

Table 5. The LFC accuracies of the FaceNet neural network model [12]

Model name	LFW accuracy	Training dataset	Architecture
20180408-102900	0.9905	CASIA-WebFace	Inception ResNet v1
20180402-114759	0.9965	VGGFace2	Inception ResNet v1

There are two training data set for this neural network (CASIA-WebFace and VGGFace2). The LFC accuracies are 99.05% and 99.65% respectively. They are very high accuracies.

4.2 Face Recognition

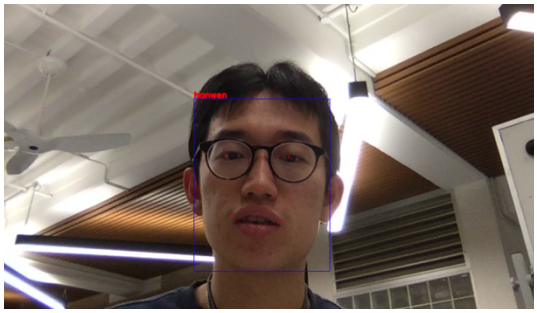

Fig. 6. Face recognition in real time of Hanwen

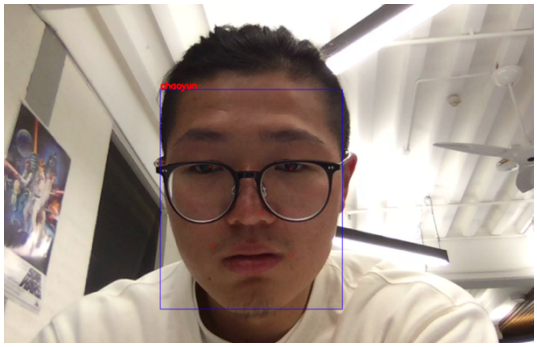

Fig. 7. Face recognition in real time of Chaoyun

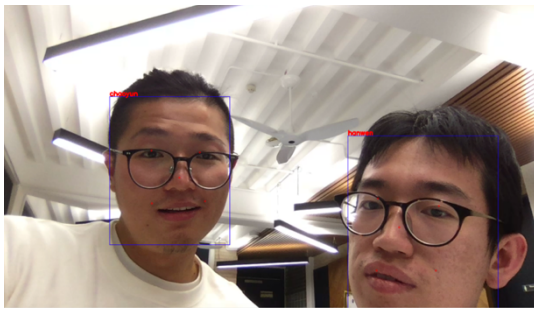

Fig. 8. Face recognition in real time of both

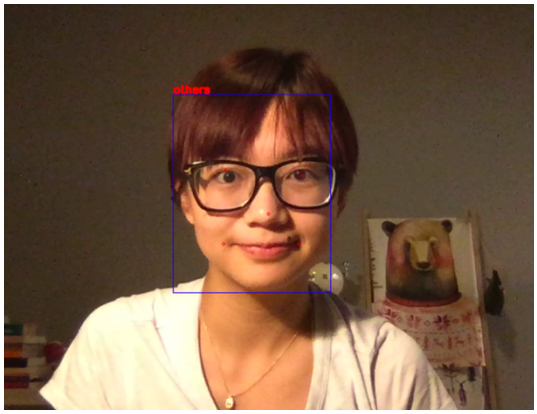

Fig. 9. Face recognition in real time of others

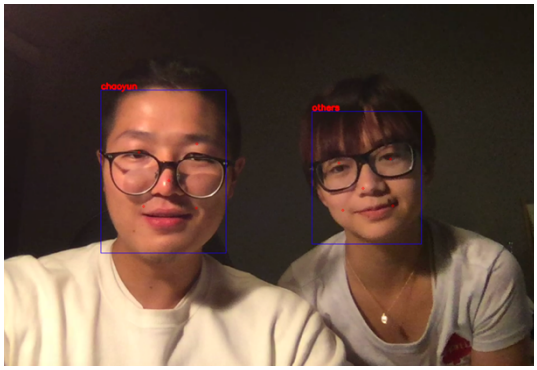

Fig. 10. Face Recognition in real time of Chaoyun and others

Figure 6,7,8,9 and 10 are the results of face recognition in real-time. The person in figure 7 is Chaoyun, and he has been recognized correctly. The person in figure 6 is Hanwen, and he has also been recognized correctly. In figure 8,

Chaoyun and Hanwen are all in one frame. Their faces have been boxed respectively and have been labeled by corresponding names. In figure 9, it shows someone not in the two classes and has been labeled as others. In the algorithm, if the highest score from SVM classifier is smaller than a threshold, it will be identified as others. The threshold in this project is 0.4. Figure 10 shows that the model distinguishes the class person Chaoyun and the others, which is also correct.

Then this project tries to calculate the approximate accuracy of the face recognition in real time. We have recorded one-minute video from the computer camera and apply this face recognition model to recognize the face in video. In the one minute, we do expressions and add some interferences. Then compare with the ground truth (which is labeled by human eyes). Calculate the duration with error in the one minute. Finally, the error duration lasts about 8 seconds. Thus, the accuracy of the real time face recognition model is approximately 86.67%.

In this paper, there are just three figures can be show. However, from the real time face recognition in monitor, it is obvious that the accuracy of this face recognition model is very high. However, the recognition speed is a little bit slow. The real time video has some delay because of the complex calculation. If the model needs to be applied in some real time areas, the problem must be solved. At the beginning, this project tries to apply MTCNN to extract the face and apply FaceNet on it to get the embedding vector. Then compare the vector with the other vectors in database and use the label of the minimum distance's face to label the detected face. However, this method is slower than the original one. And when the database becomes larger, the calculation will be more complex, and the speed will reduce. It will not be real time face recognition. Therefore, to increase the recognition speed, we may apply other more simply classify model, use more powerful computer, or simplify the neural network.

5. Conclusion

For FaceNet, through CNN, the face is mapped to a feature vector in European space, and the distance of the feature vector of different face image is calculated. We can apply the prior knowledge to train the neural network that the distance of the same people's face is always smaller than the distance between different peoples' faces. Only the face features need to be calculated during the test, and then the distance threshold can be used to determine whether the two face photos belong to the same individual.

In this project, FaceNet neural network model has been discussed and applied for face recognition in real time. And the MTCNN neural network is used to detect and extract the face in monitor in real time and classify it by trained SVM classifier. The accuracy of the face recognition in real time is good. However, There are more future work can be done to improve to performance of the model, for example, the calculation speed, more classes of database.

References

[1] Florian Schroff, Dmitry Kalenichenko, lames Philbin, "FaceNet: A Unified Embedding for Face Recognition and Clustering", Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition 2015

[2] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. CoRR, abs/1409.4842, 2014. 2, 3, 4, 5, 6, 10

[3] K. Heath and L. Guibas, "FaceNet: Tracking people and acquiring canonical face images in a wireless camera sensor network," in 2007, . DOI: 10.1109/ICDSC.2007.4357514.

[4] J. Bhattacharya et al, "Feeding a DNN for face verification in video data acquired by a visually impaired user," in 2017, DOI: 10.23919/MIPRO.2017.7973585.

[5] Z. Ming et al, "Simple triplet loss based on Intra/Inter-class metric learning for face verification," in 2017, DOI: 10.1109/ICCVW.2017.194.

[6] W. Wu, C. Liu and Z. Su, "Novel real-time face recognition from video streams," in 2017, . DOI: 10.1109/ICCSEC.2017.8446960.

[7] W. Wan and H. J. Lee, "FaceNet based face sketch recognition," in 2017, . DOI: 10.1109/CSCI.2017.73.

[8] A. Boka and B. Morris, "Person recognition for access logging," in 2019, DOI: 10.1109/CCWC.2019.8666483.

[9] X. Zhao et al, "Blind source separation for face image based on deep leaming," in 2018, . DOI: 10.1109/AUTEEE.2018.8720780.

[10] W. Chu and W. Li, "Manga FaceNet: Face detection in manga based on deep neural network," in 2017, . DOI: 10.1145/3078971.3079031.

[11] M. D. Zeiler and R. Fergus. Visualizing and understanding convolutional networks. CoRR, abs/1311.2901, 2013. 2, 3, 4, 6

[12] D. Sandberg. Face recognition using. TensorFlow. GitHub. April 2018.