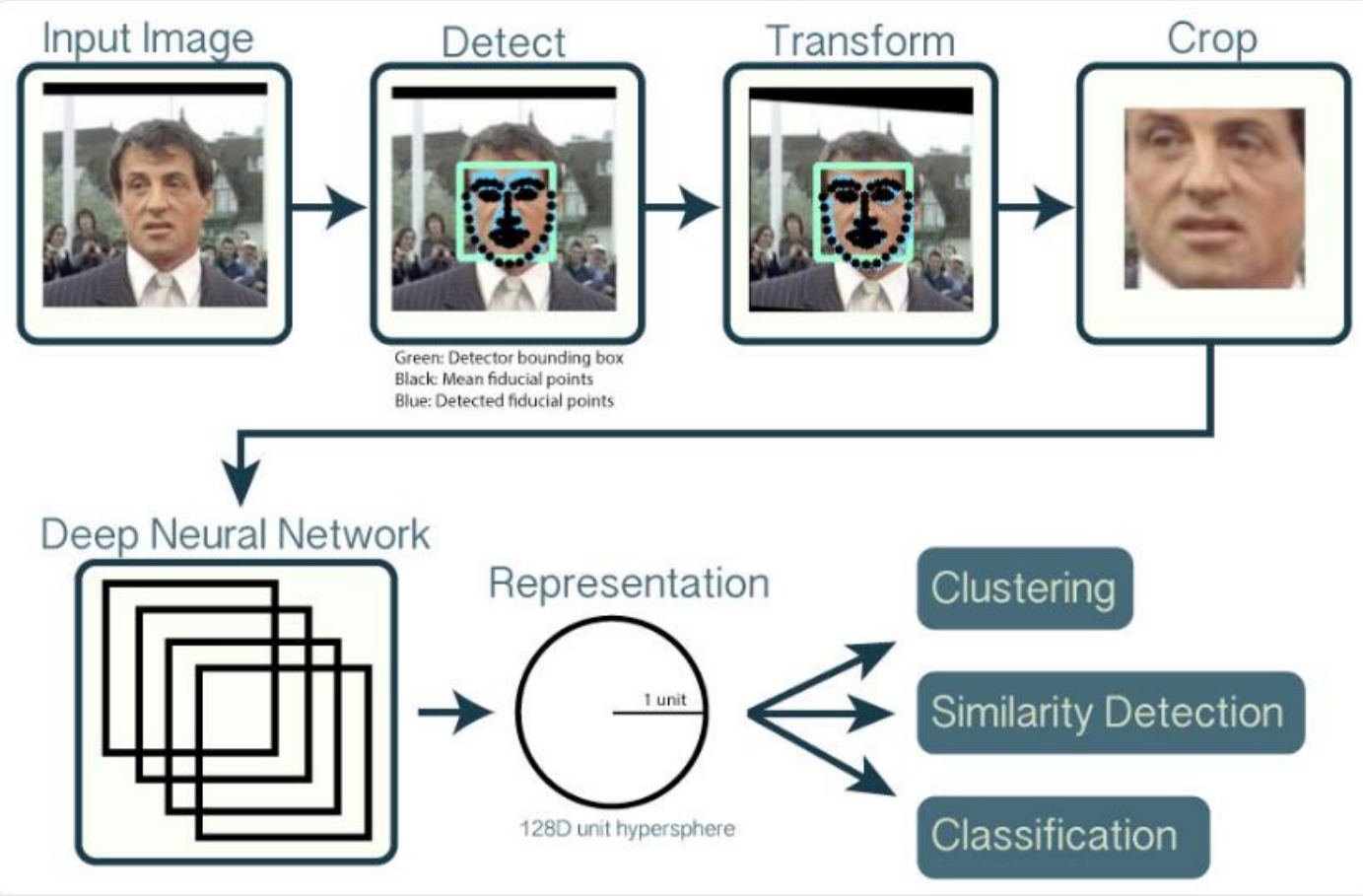
OpenFace

- Openface
  - 소개
  - 응용 예제
  - 어떻게 사용하지?
    - 설치하기.
    - 예제 돌려보기
    - pretrain 이용 학습하기
    - TorchNuralNet 새로 만들기
  - 사용 후기
  - 차후 확장

- 질의응답
- Appendixes
  - ./demo/classifier.py
  - ./training/main.lua
  - API Reference

Openface Library



papers

FaceNet: A Unified Embedding for Face Recognition and Clustering

Florian Schroff
fschroff@google.com
Google Inc.

Dmitry Kalenichenko
dkalenichenko@google.com
Google Inc.

James Philbin
jphilbin@google.com
Google Inc.

Abstract

Despite significant recent advances in the field of face recognition [10, 14, 15, 17], implementing face verification and recognition efficiently at scale presents serious challenges to current approaches. In this paper we present a system, called FaceNet, that directly learns a mapping from face images to a compact Euclidean space where distances directly correspond to a measure of face similarity. Once this space has been produced, tasks such as face recognition, verification and clustering can be easily implemented using standard techniques with FaceNet embeddings as feature vectors.

Our method uses a deep convolutional network trained to directly optimize the embedding itself, rather than an intermediate bottleneck layer as in previous deep learning approaches. To train, we use triplets of roughly aligned matching / non-matching face patches generated using a novel online triplet mining method. The benefit of our approach is much greater representational efficiency: we achieve state-of-the-art face recognition performance using only 128-bytes per face.

On the widely used Labeled Faces in the Wild (LFW) dataset, our system achieves a new record accuracy of ***99.63%****. On YouTube Faces DB it achieves* ***95.12%****. Our*

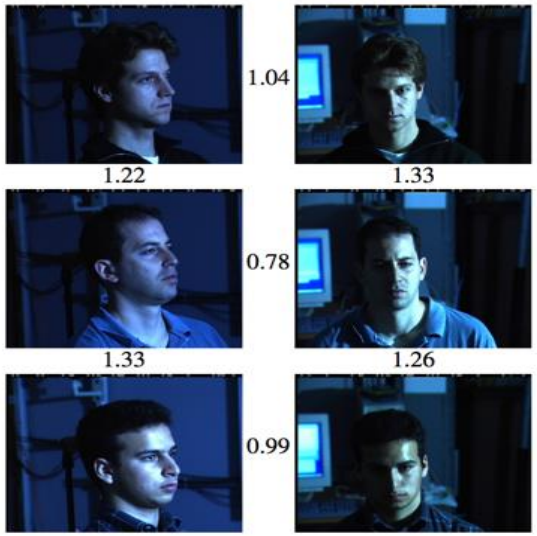


Figure 1. **Illumination and Pose invariance.** Pose and illumination have been a long standing problem in face recognition. This figure shows the output distances of FaceNet between pairs of faces of the same and a different person in different pose and il-

OpenFace: A general-purpose face recognition library with mobile applications

Brandon Amos, Bartosz Ludwiczuk[†], Mahadev Satyanarayanan

June 2016
CMU-CS-16-118

School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

[†]Poznan University of Technology

Abstract

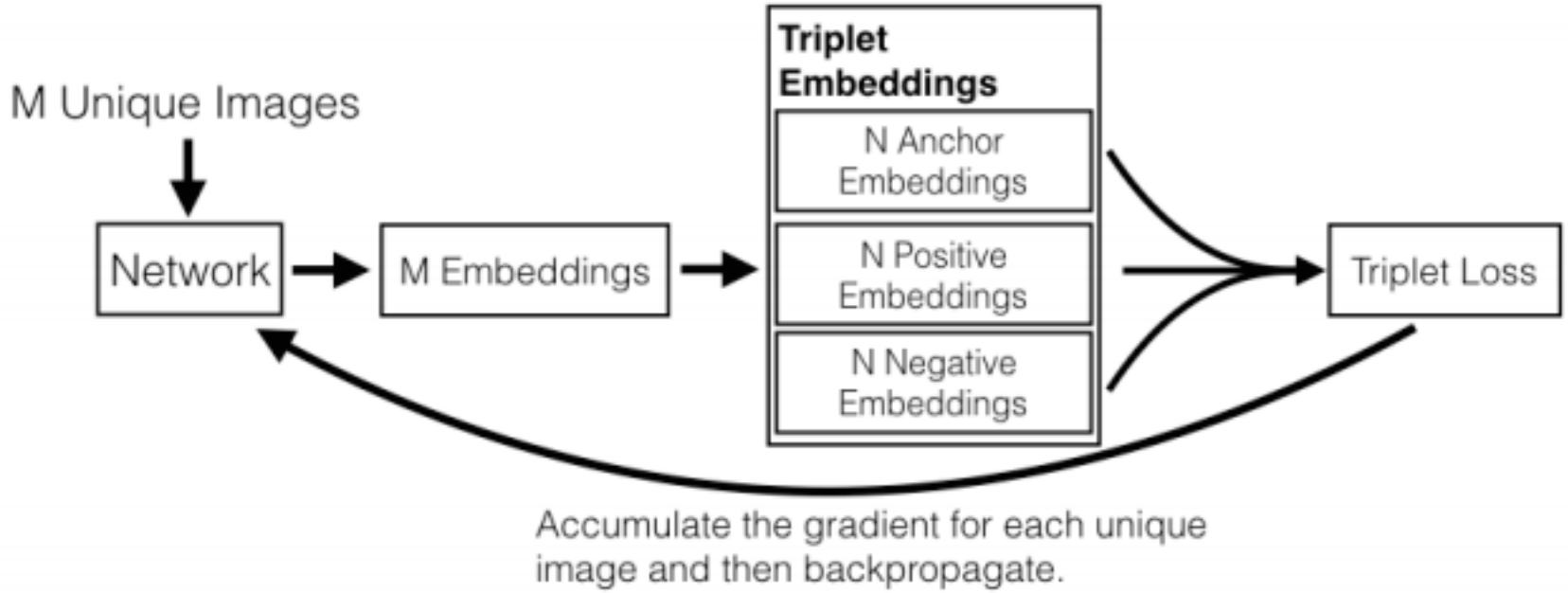
Cameras are becoming ubiquitous in the Internet of Things (IoT) and can use face recognition technology to improve context. There is a large accuracy gap between today's publicly available face recognition systems and the state-of-the-art private face recognition systems. This paper presents our OpenFace face recognition library that bridges this accuracy gap. We show that OpenFace provides near-human accuracy on the LFW benchmark and present a new classification benchmark for mobile scenarios. This paper is intended for non-experts interested in using OpenFace and provides a light introduction to the deep neural network techniques we use.

We released OpenFace in October 2015 as an open source library under the Apache 2.0 license. It is available at: `http://cmusatyalab.github.io/openface/`

Training

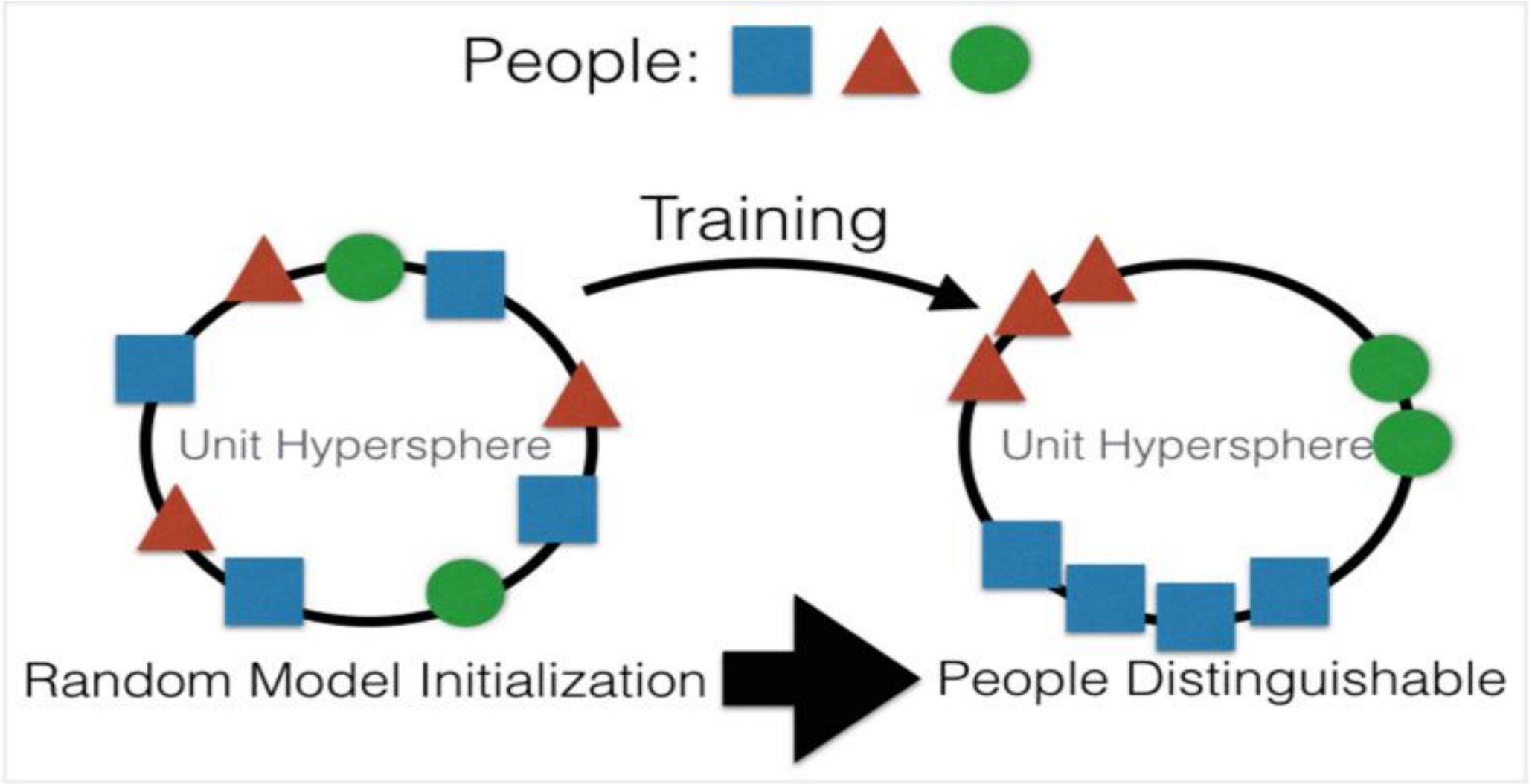
Figure 2. **Model structure.** Our network consists of a batch input layer and a deep CNN followed by L_2 normalization, which results in the face embedding. This is followed by the triplet loss during training.

Figure 3. The **Triplet Loss** minimizes the distance between an *anchor* and a *positive*, both of which have the same identity, and maximizes the distance between the *anchor* and a *negative* of a

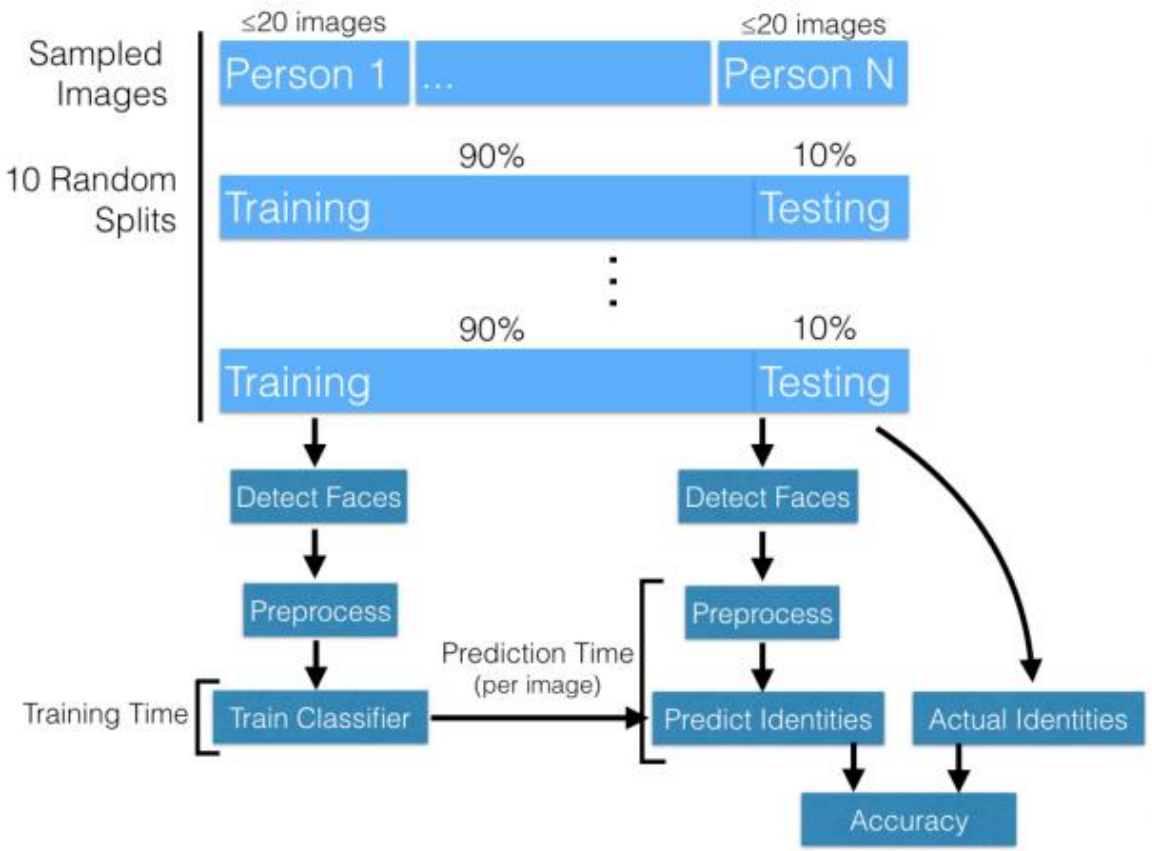


layer	size-in	size-out	kernel	param	FLPS
conv1	220×220×3	110×110×64	7×7×3, 2	9K	115M
pool1	110×110×64	55×55×64	3×3×64, 2	0	
rnorm1	55×55×64	55×55×64		0	
conv2a	55×55×64	55×55×64	1×1×64, 1	4K	13M
conv2	55×55×64	55×55×192	3×3×64, 1	111K	335M
rnorm2	55×55×192	55×55×192		0	
pool2	55×55×192	28×28×192	3×3×192, 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

- Cnn Classification example
- https://www.youtube.com/watch?v=gEfKfF9Ef3U



Bench Mark


≤20 images
≤20 images
Sampled Images
Person 1
...
Person N
10 Random Splits
90%
10%
Training
Testing
90%
10%
Training
Testing
Detect Faces
Detect Faces
Preprocess
Preprocess
Prediction Time (per image)
Training Time
Train Classifier
Predict Identities
Actual Identities
Accuracy

Accuracies

Technique	Accuracy
Human-level (cropped) [KBBN09]	0.9753
Eigenfaces (no outside data) [TP91][3]	0.6002 ± 0.0079
FaceNet [SKP15]	0.9964 ± 0.009
DeepFace-ensemble [TYRW14]	0.9735 ± 0.0025
OpenFace (ours)	0.9292 ± 0.0134

LFW

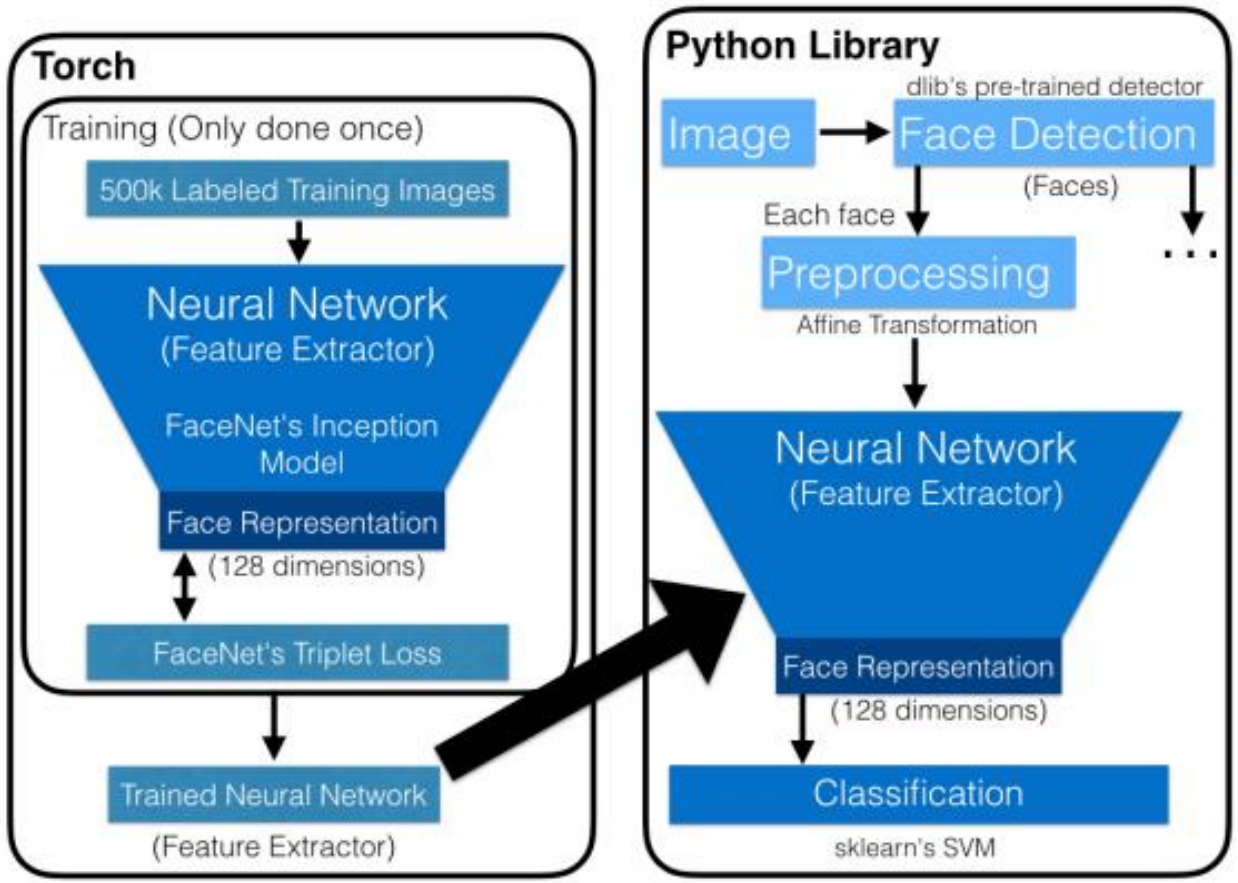
- Labeled Faces in the Wild
- http://vis-www.cs.umass.edu/lfw/index.html
  - 어떤 얼굴 인식 프로그램이든 그 성능을 확인할 수 있으려면 알고리즘을 테스트할 적절한 얼굴 데이터가 필요하다. 물론 이 데이터에는 인종, 나이, 성뿐 아니라 조명, 표정, 자세별로 다른 다양한 얼굴들이 포함돼 있어야 한다. 거기에 헤어스타일, 의상 패션, 화장에 따른 얼굴 변화도 고려돼야 한다. 현재 이런 조건을 갖춘 표준 데이터 역할을 하는 것으로 LFW(Labelled Faces in the Wild)라는 이름의 데이터세트가 있다. 이 세트에는 웹에서 수집한 약 6000명의 유명인사의 얼굴사진 1만3천여 장이 들어 있다.
  - http://www.hani.co.kr/arti/economy/it/636535.html

Openface Library

- Free and Open source Face recognition with deep learning network
- Version
  - 0.2.0
  - Accuracy (0.1.0)76.1 -> (0.2.0)92.9%
  - Base on LFW Data set

- Blog
  - https://cmusatyalab.github.io/openface/
- Api
  - http://openface-api.readthedocs.io/en/latest/
- Github
  - https://github.com/cmusatyalab/openface

구조


Torch
Training (Only done once)
500k Labeled Training Images
Neural Network
(Feature Extractor)
FaceNet's Inception Model
Face Representation
(128 dimensions)
FaceNet's Triplet Loss
Trained Neural Network
(Feature Extractor)
Python Library
dlib's pre-trained detector
Image
Face Detection
(Faces)
Each face
Preprocessing
Affine Transformation
Neural Network
(Feature Extractor)
Face Representation
(128 dimensions)
Classification
sklearn's SVM

직접 설치

- 직접 빌드
  - Git clone https://github.com/cmusatyalab/openface; python setup.py
  - And… torch, cutorch etc….
  - 복잡함.

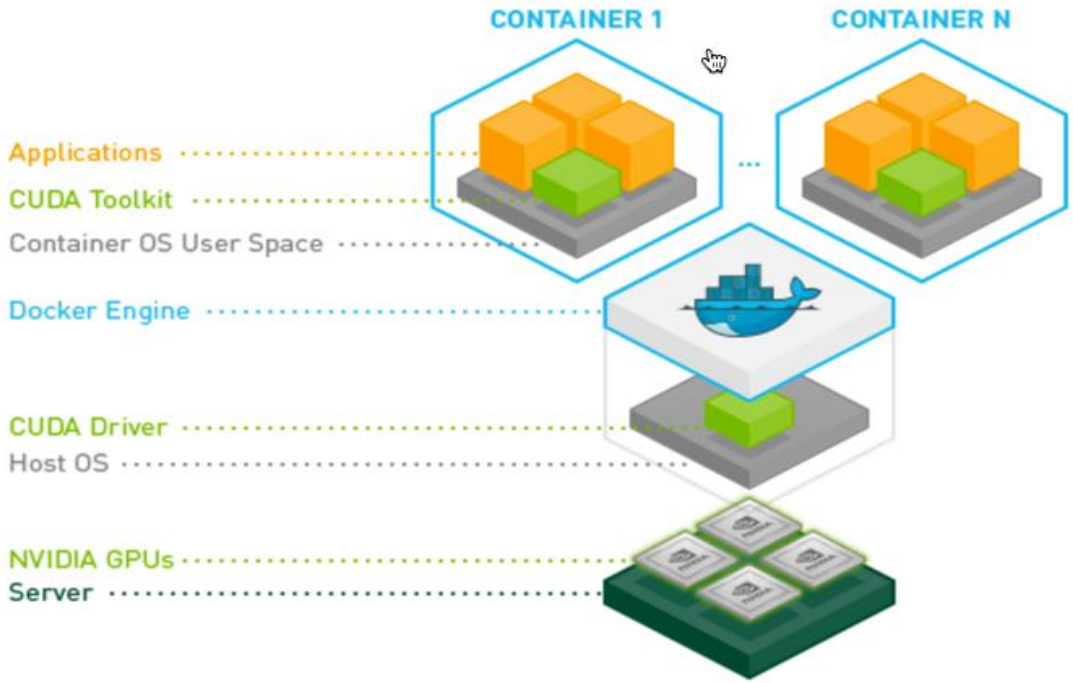
Docker를 통한 설치 1

```
docker pull bamos/openface
docker run -p 9000:9000 -p 8000:8000 -t -i bamos/openface /bin/bash
cd /root/openface
./demos/compare.py images/examples/{lennon*,clapton*}
./demos/classifier.py infer models/openface/celeb-classifier.nn4.small2.v1.pkl ./images/examples/carell.jpg
./demos/web/start-servers.sh
```

```
docker build -t openface .
docker run -p 9000:9000 -p 8000:8000 -t -i openface /bin/bash
cd /root/openface
./run-tests.sh
./demos/compare.py images/examples/{lennon*,clapton*}
./demos/classifier.py infer models/openface/celeb-classifier.nn4.small2.v1.pkl ./images/examples/carell.jpg
./demos/web/start-servers.sh
```

Nvidia-Docker를 이용한 설치1

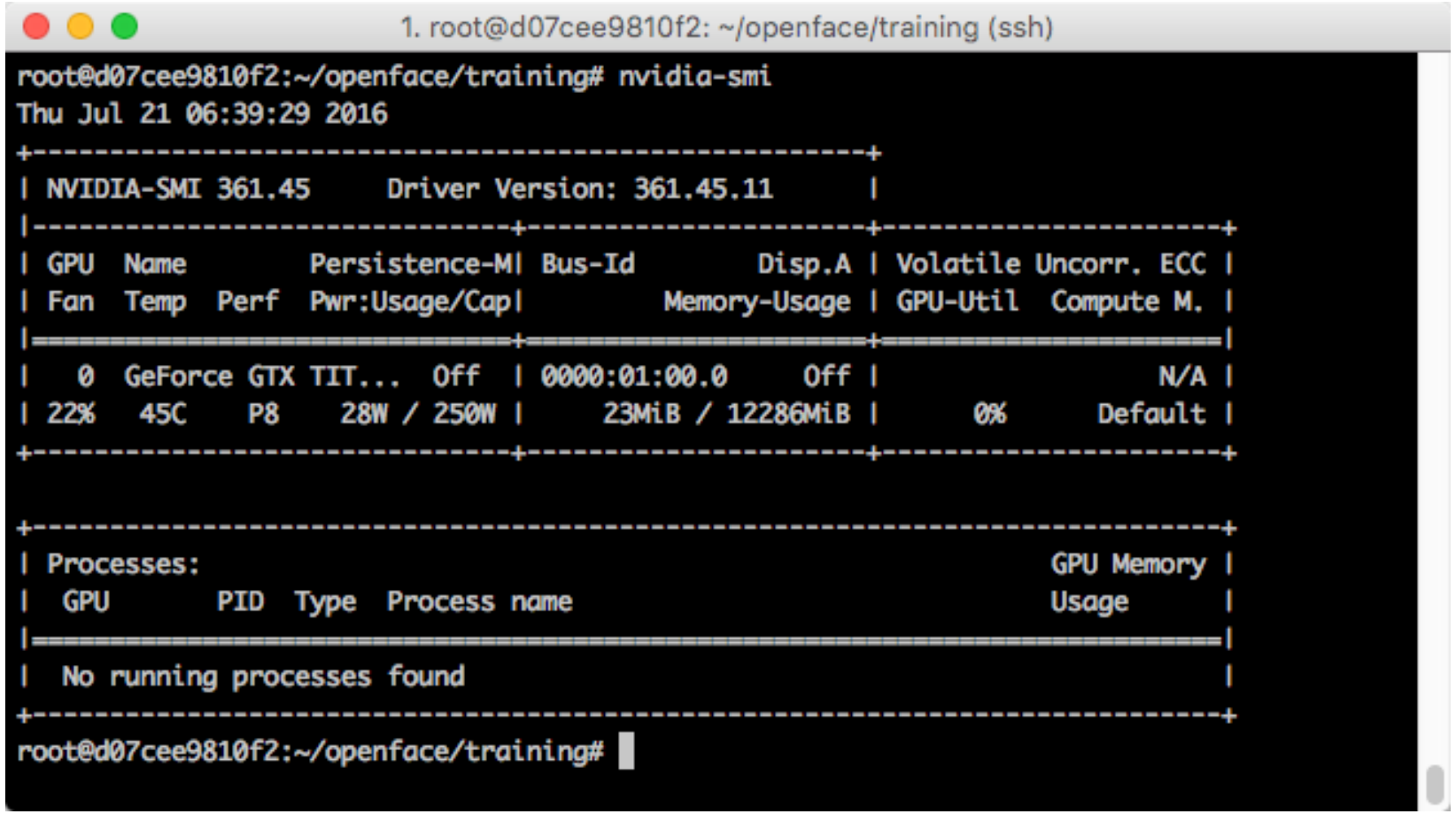
- Docker for cuda
  - Nvidia-Docker + openface
  - https://github.com/jaeho-kang/deep-learning/blob/master/openface/Dockerfile
  - Cuda 7.5, cudnn5



nvidia-docker run -p 9000:9000 –p 8000:8000 –t –I image /bin/bash

nvidia cuda 지원 docker

Nvidia-Docker를 이용한 설치2



이미지 비교

- ./demos/compare.py ./images/examples/{lennon*,clapton*}

```
root@d07cee9810f2:~/openface# ./demos/compare.py ./images/examples/{lennon*,clapton*}
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
```

lennon ₩ clapton	lennon-1	lennon-2	clapton-1	clapton-2
lennon-1	-	0.763	1.132	1.145
lennon-2	0.763	-	1.447	1.521
clapton-1	1.132	1.447	-	0.318
clapton-2	1.145	1.521	0.318	-

분류예제

- ./demos/classifier.py infer models/openface/celeb-classifier.nn4.small2.v1.pkl ./images/examples/carell.jpg

```
/usr/local/lib/python2.7/dist-packages/sklearn/lda.py:4: DeprecationWarning: lda.LDA has been moved to disc
riminant_analysis.LinearDiscriminantAnalysis in 0.17 and will be removed in 0.19
  "in 0.17 and will be removed in 0.19", DeprecationWarning)

=== ./images/examples/carell.jpg ===
Predict SteveCarell with 0.97 confidence.
```

- class, confidence 출력.

pretrain 이용 학습하기

1. 데이터셋 준비하기
2. 전처리
3. Representations 생성
4. 분류기 학습
5. 분류기 활용

데이터셋 준비하기

오른쪽과 같은 구조의 데이터를 생성
인식하는 확장자는 jpg,png 소문자

```
$ tree data/mydataset/raw
person-1
├── image-1.jpg
├── image-2.png
...
└── image-p.png

...

person-m
├── image-1.png
├── image-2.jpg
...
└── image-q.png
```

전처리

```
root@d07cee9810f2:~/openface# for N in {1..8}; do ./util/align-dlib.py <path-to-raw-data> align outerEyesAn
dNose <path-to-aligned-data> --size 96 & done.
```

- 위의 코드를 이용 전처리.

. ./util/prune-dataset.py

- 전처리 된 데이타셋에서 이미지가 임계 수량 보다 작을 경우 가치 치기를 수행

Representation 작성

```
root@d07cee9810f2:~/openface# ./batch-represent/main.lua ./data/imgs/test1/feature/ -data ./data/imgs/test1
/aligned/
```

위의 코드를 이용하여 전처리된 이미지를 representation을 작성
labels.csv, reps.csv 작성

```
drwxr-xr-x 2 root root    4096 Jul 20 05:49 .
drwxr-xr-x 5 root root    4096 Jul 21 07:07 ..
-rw-r--r-- 1 root root 2544989 Jul 20 05:47 classifier-dnn.pkl
-rw-r--r-- 1 root root 2544978 Jul 20 05:49 classifier.pkl
-rw-r--r-- 1 root root   55100 Jul 19 06:39 labels.csv
-rw-r--r-- 1 root root 2572965 Jul 19 06:39 reps.csv
```

```
5,../../data/imgs/aligned/person2/person2_0021.png
5,../../data/imgs/aligned/person2/person2_0112.png
5,../../data/imgs/aligned/person2/person2_0163.png
5,../../data/imgs/aligned/person2/person2_0080.png
5,../../data/imgs/aligned/person2/person2_0150.png
5,../../data/imgs/aligned/person2/person2_0095.png
5,../../data/imgs/aligned/person2/person2_0060.png
5,../../data/imgs/aligned/person2/person2_0029.png
5,../../data/imgs/aligned/person2/person2_0020.png
```

```
1102 0.063714526593685,0.11789680272341,-0.093560397624969,0.032627712935209,-0.0062868464738131,0.056935355067253,0.11773211508989,0.041251167654991,-0.011534295044839,0.047863774001598,0.10600770264864,0.096401482820511,0.085638433694839,-0.11223154515028,-0.11457435786724,0.065393127501011,0.01583961583674,-0.036754757165909,-0.14794178307056,0.10977283120155,0.021689906716347,-0.14182053506374,0.028591526672244,0.096884869039059,-0.064116597175598,-0.10214360058308,-0.031717784702778,-0.0048588947393,0.066314533352852,0.053924586623907,-0.068141132593155,0.079813174903393,-0.0019122110679746,0.10452831536531,-0.20215983688831,-0.090987369418144,-0.02733806706965,0.081981986761093,-0.026652898639441,0.11389617621899,-0.01528012752533,-0.1104848459363,0.0328284278512,-0.089450165629387,-0.0012109241215512,-0.12196105718613,0.027869530022144,-0.031088063493371,-0.070204883813858,0.15090116858482,0.074378088116646,0.019232016056776,-0.1574028134346,0.031664222478867,-0.010649780742824,0.015186873264611,-0.020671071484685,0.070981882512569,-0.048278354108334,0.09680962562561,-0.10080373287201,-0.015751145780087,0.10165838897228,-0.18572653830051,0.0081141926348209,0.0010242903372273,-0.057982217520475,-0.071272373199463,-0.22229632735252,-0.042046993970871,-0.061778835952282,0.048077546060085,-0.083433762192726,0.20655515789986,0.095840029418468,0.03051073290407,0.19234138727188,0.021167842671275,-0.039720688015223,0.0076642083004117,-0.0812723711133,-0.16499789059162,0.043903950601816,0.037677526473999,-0.062675207853317,0.14623844623566,-0.0022741830907762,-0.080301485955715,0.05777120962739,-0.022749949246645,-0.037322908639908,-0.18885242938995,-0.038078527897596,0.060444414615631,0.086178064346313,0.0018444207962602,-0.033869054168463,-0.01482202578336,0.0010018871398643,0.061683602631092,-0.029023490846157,-0.034093454480171,-0.072982870042324,0.24302186071873,0.026123873889446,0.077146023511887,-0.023779122158885,0.14770500361919,-0.015426262281835,-0.1404864937067,-0.10716092586517,0.10866080224514,0.074286043643951,0.060720302164555,0.012869196943939,0.010410184971988,0.060524236410856,0.14047771692276,0.10674763470888,0.21389852464199,-0.055304884910583,-0.023309495300055,-0.026703080162406,0.076132692396641,-0.025276876986027,0.059581231325865,-0.068015038967133,-0.039311941713095
```

분류기 학습 및 활용

- 분류기 학습

```
root@d07cee9810f2:~/openface# ./demos/classifier.py train ./data/imgs/test1/feature/
/usr/local/lib/python2.7/dist-packages/sklearn/lda.py:4: DeprecationWarning: lda.LDA has been moved to
discriminant_analysis.LinearDiscriminantAnalysis in 0.17 and will be removed in 0.19
  "in 0.17 and will be removed in 0.19", DeprecationWarning)
Loading embeddings.
Training for 5 classes.
Saving classifier to './data/imgs/test1/feature//classifier.pkl'
```

- 분류기 테스트

```
root@d07cee9810f2:~/openface# ./demos/classifier.py infer ./data/imgs/test1/feature/classifier.pkl ./da
ta/imgs/test1/raw/person-3/13269414_1309786465717161_717462499_n.jpg
/usr/local/lib/python2.7/dist-packages/sklearn/lda.py:4: DeprecationWarning: lda.LDA has been moved to
discriminant_analysis.LinearDiscriminantAnalysis in 0.17 and will be removed in 0.19
  "in 0.17 and will be removed in 0.19", DeprecationWarning)

=== ./data/imgs/test1/raw/person-3/13269414_1309786465717161_717462499_n.jpg ===
Predict person3 with 0.94 confidence.
```

pretrain 이용 학습 정리

--preprocess

for N in {1..8}; do ./util/align-dlib.py <path-to-raw-data> align outerEyesAndNose <path-to-aligned-data> --size 96 & done.

--representation generate

./batch-represent/main.lua -outDir <feature-directory> -data <path-to-aligned-data>

-- classification training

./demos/classifier.py train <feature-directory>

-- classification test

./demos/classifier.py infer ./models/openface/celeb-classifier.nn4.small2.v1.pkl images/examples/{carell,adams,lennon}*n <feature-directory>

TorchNuralNet 새로 만들기

1. 데이터셋 준비하기
2. 전처리
3. Model 학습
4. Model 평가
5. Model 활용

데이터셋 준비하기

오른쪽과 같은 구조의 데이터를 생성
인식하는 확장자는 jpg,png 소문자

```
$ tree data/mydataset/raw
person-1
├── image-1.jpg
├── image-2.png
...
└── image-p.png

...

person-m
├── image-1.png
├── image-2.jpg
...
└── image-q.png
```

전처리

```
root@d07cee9810f2:~/openface# for N in {1..8}; do ./util/align-dlib.py <path-to-raw-data> align outerEyesAn
dNose <path-to-aligned-data> --size 96 & done.
```

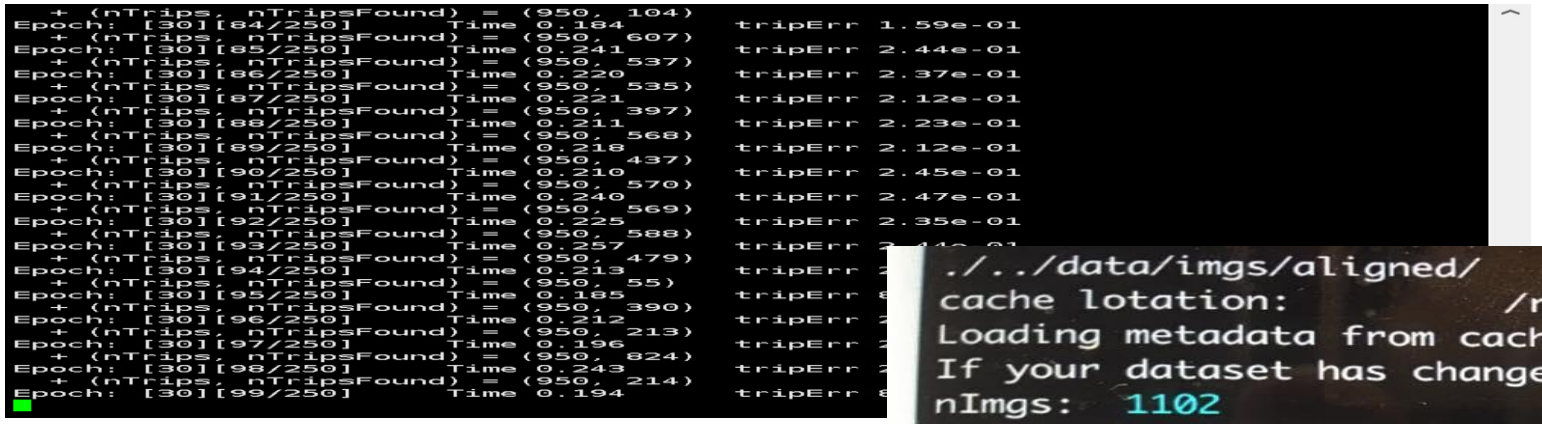
- 위의 코드를 이용 전처리.

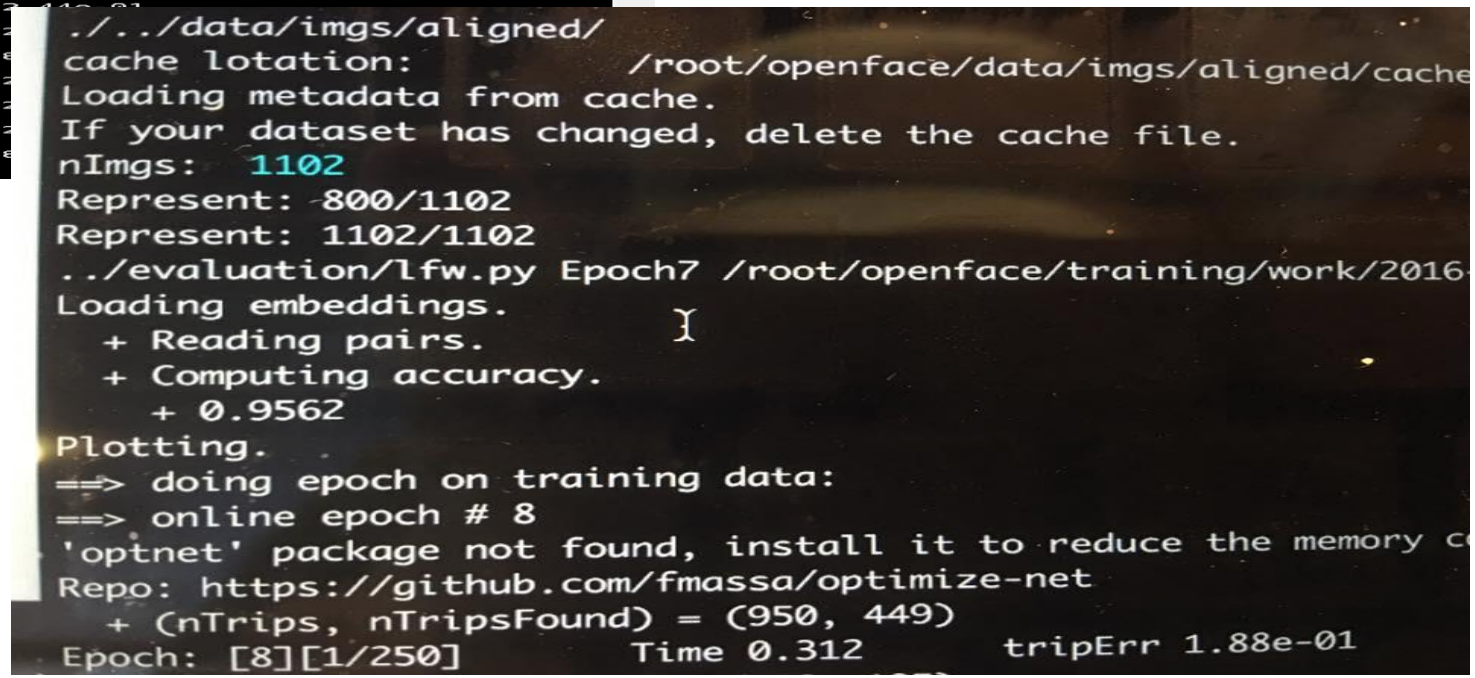
. ./util/prune-dataset.py

- 전처리 된 데이타셋에서 이미지가 임계 수량 보다 작을 경우 가치 치기를 수행

Model 학습

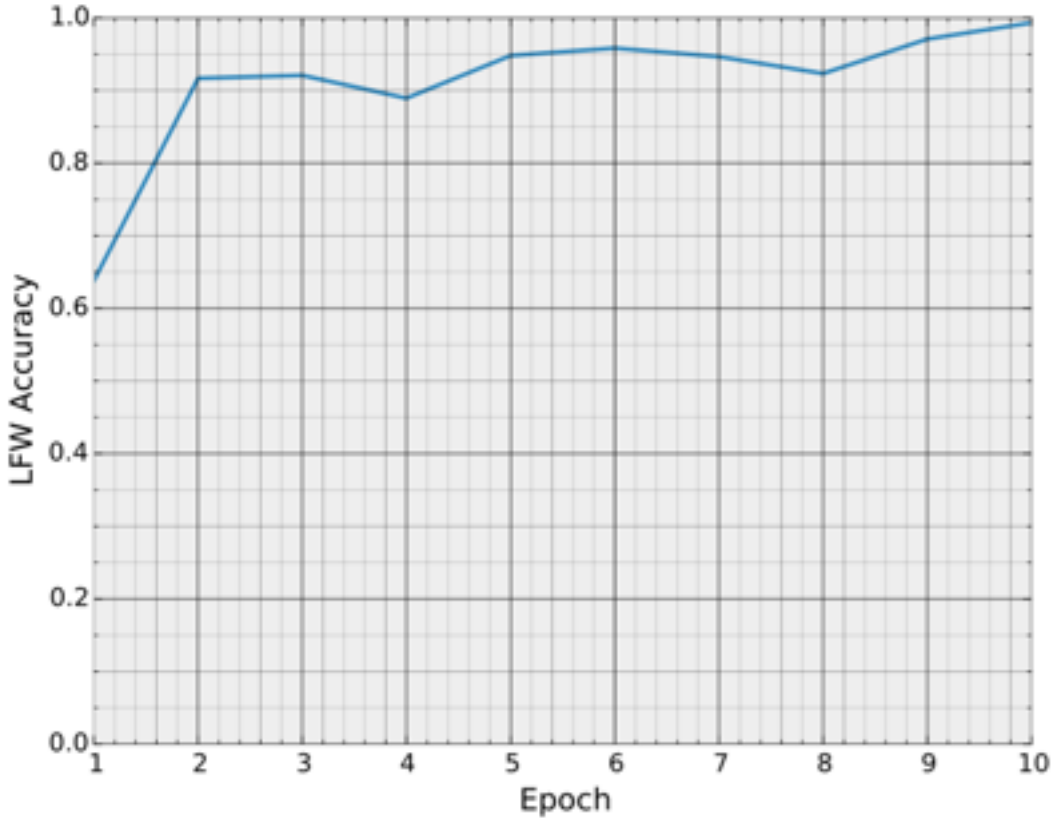
./training/main.lua –data <DataDir> -lfwDir <LfwDir> -peoplePerBatch 15 –imagesPerPerson 20

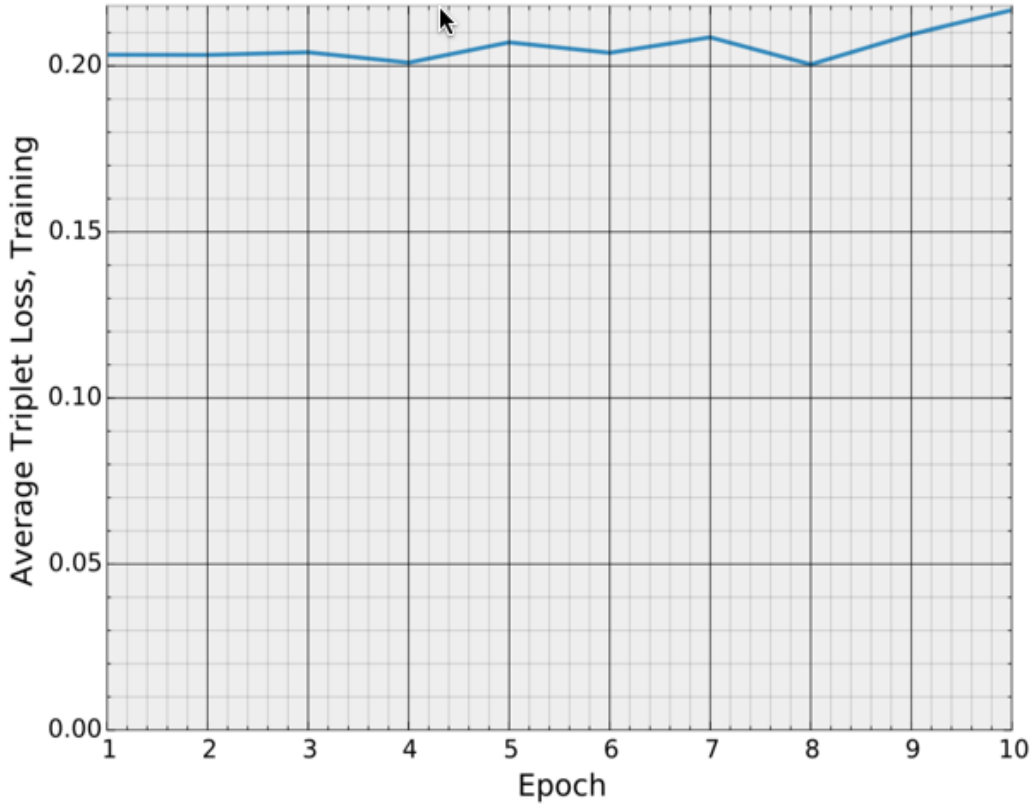




Model 분석

./training/plot-loss.py wordDirs [WordDir]





Model 활용

```
root@d07cee9810f2:~/openface# ./demos/classifier.py train ./data/imgs/test1/feature/
/usr/local/lib/python2.7/dist-packages/sklearn/lda.py:4: DeprecationWarning: lda.LDA has been moved to
discriminant_analysis.LinearDiscriminantAnalysis in 0.17 and will be removed in 0.19
  "in 0.17 and will be removed in 0.19", DeprecationWarning)
Loading embeddings.
Training for 5 classes.
Saving classifier to './data/imgs/test1/feature//classifier.pkl'
```

./demos/classifier.py –networkModel [TrainModel] train [Feature Path]

networkModel : ./training/work/dir/model_n.t7 파일 DNN Weight 파일

```
root@d07cee9810f2:~/openface# ./demos/classifier.py --networkModel ./training/work/2016-07-18_16-38-23/
model_10.t7 infer ./data/imgs/test1/feature/classifier-dnn.pkl ./data/imgs/test1/raw/*/*.jpg
```

./demos/classifier.py –networkModel [TrainModel] infer [TrainedModel] DataSource

networkModel : ./training/work/dir/model_n.t7 파일 DNN Weight 파일

TrainModel : classifier.py 결과물을 사용

preTrain VS DNN

```
root@d07cee9810f2:~/openface# ./demos/classifier.py --networkModel ./training/work/2016-07-18_16-26-49/model_10.t7 infer ./data/imgs/test1/feature/classifier.pkl ./data/imgs/test1/
aligned/person2/person2_0019.png
/usr/local/lib/python2.7/dist-packages/sklearn/lda.py:4: DeprecationWarning: lda.LDA has been moved to discriminant_analysis.LinearDiscriminantAnalysis in 0.17 and will be removed
in 0.19
  "in 0.17 and will be removed in 0.19", DeprecationWarning)

=== ./data/imgs/test1/aligned/person2/person2_0019.png ===
Predict person5 with 0.57 confidence.
[ 0.1   0.22  0.08  0.04  0.57]
```

DNN

```
LabelEncoder()
root@d07cee9810f2:~/openface#
root@d07cee9810f2:~/openface# vi ./demos/classifier.py
root@d07cee9810f2:~/openface#
root@d07cee9810f2:~/openface#
root@d07cee9810f2:~/openface# ./demos/classifier.py train ./data/imgs/test1/feature/
/usr/local/lib/python2.7/dist-packages/sklearn/lda.py:4: DeprecationWarning: lda.LDA has been moved to discriminant_analysis.LinearDiscriminantAnalysis in 0.17 and will be removed
in 0.19
  "in 0.17 and will be removed in 0.19", DeprecationWarning)
Loading embeddings.
Training for 5 classes.
Saving classifier to './data/imgs/test1/feature//classifier.pkl'
root@d07cee9810f2:~/openface#
root@d07cee9810f2:~/openface# ./demos/classifier.py infer ./data/imgs/test1/feature/classifier.pkl ./data/imgs/test1/aligned/person2/person2_0019.png
/usr/local/lib/python2.7/dist-packages/sklearn/lda.py:4: DeprecationWarning: lda.LDA has been moved to discriminant_analysis.LinearDiscriminantAnalysis in 0.17 and will be removed
in 0.19
  "in 0.17 and will be removed in 0.19", DeprecationWarning)

=== ./data/imgs/test1/aligned/person2/person2_0019.png ===
Predict person2 with 0.74 confidence.
[ 0.05  0.74  0.01  0.02  0.18]
```

preTrain

Dnn 학습 정리

--preprocess

for N in {1..8}; do ./util/align-dlib.py <path-to-raw-data> align outerEyesAndNose <path-to-aligned-data> --size 96 & done.

--representation generate

./batch-represent/main.lua -outDir <feature-directory> -data <path-to-aligned-data>

--training dnn neural network

./training/main.lua –data <DataDir> -lfwDir <LfwDir> -peoplePerBatch 15 –imagesPerPerson 20

-- classification training

./demos/classifier.py train <feature-directory> --networkModel <torch networkModel>

-- predict classification

./demos/classifier.py infer --networkModel <torch networkModel> <Train Result.pkl> images/examples/{carell,adams,lennon}*n <feature-directory>

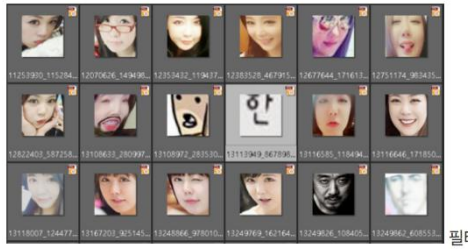
사용후기

1. GPU는 필수...
   - 이미지 1102개 DNN 학습 소요 시간

	CPU	GPU
소요시간	3시간20분	15분

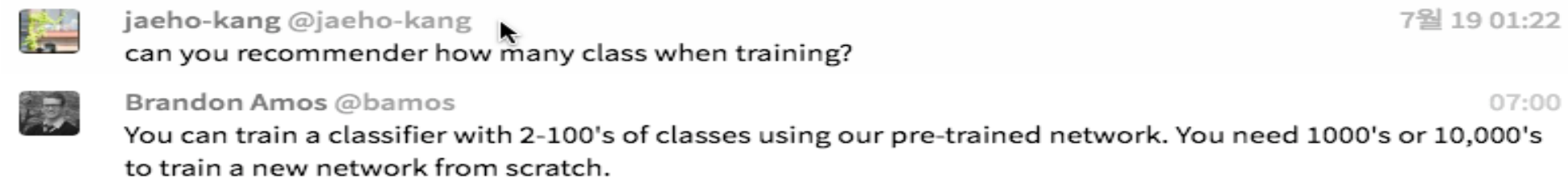
2. 양질의 데이타 필수
   - 전 처리 단계에서 필터링을 하지만 다 거르지는 못함.


원본 데이타 일부


필터링된 데이타 일부

3. 분류할 클래스 수량에 따라 방법론를 다르게

분류 클래스 < 1000 : preTrain.
분류 클래스 >= 1000 : DNN 학습

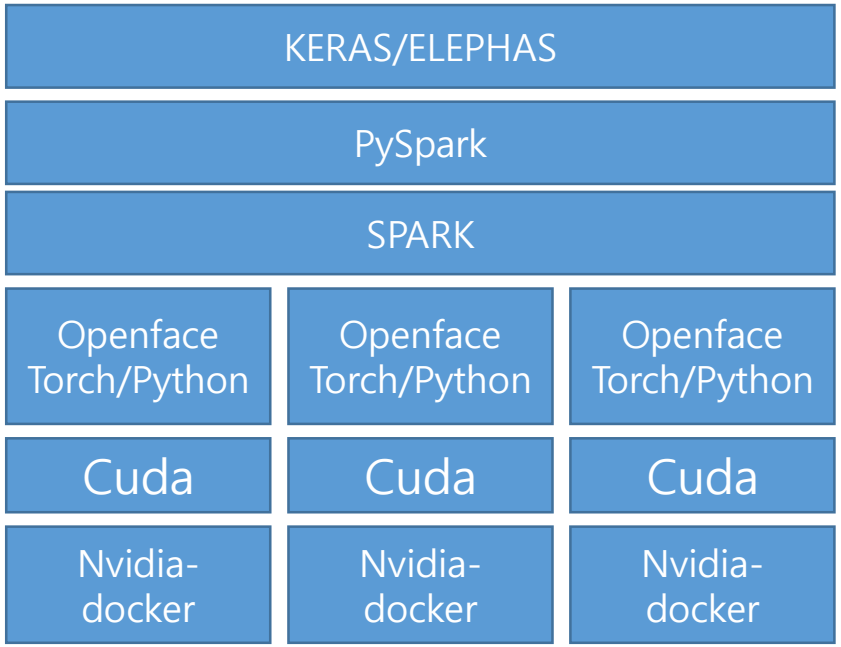


4. Linux 쉘 스크립트 공부 필요.
   1. 특히 파일명 순차적으로 바꿀때...

차후 확장

1. DNN 학습을 위한 이미지 클래스 및 이미지 수량 추가.
    1. 오버피팅 없는 모델 작성.

2. SPARK 기반 분산 플랫폼화.
    1. SPARK + KERAS + ELEPHAS



Q&A

Appendix Library Reference

./demos/classifier.py 공통

Classifier.py [infer|train] 두개의 옵션을 통해 기계학습 모델을 학습, 예측하는 모듈.

인자값	비고	생략가능
--dlibFacePredictor	얼굴을 인식하는 라이브러리 지정. 얼굴의 랜드마크적인 부분을 통해서 얼굴을 인식하게 처리. 기본값 : Shape_predictor_68_face_landmarks.dat	생략가능
--networkModel	Torch network Model 토치로 구성된 DNN Network Model 경로. DNN을 학습 했을 경우 이 인자값을 학습된 network 파일로 지정 기본값 : nn4.small2.v1.t7	
--imgDim	기본 얼굴 이미지 크기 지정 기본값 : 96	
--cuda	Cuda 라이브러리 사용여부 기본값 : store_true * 시스템에 cuda 라이브러리가 없을 경우는 사용하지 않음.	
--verbose	중간 중간 메시지 출력 여부 지정. 기본값 : store_true	

./demos/classifier.py

모드	인자값	비고	생략가능
train	--ldaDim	차원축소 값. 기본값 : -1	생략가능
	--classifier	L2 값을 이용해서 학습시키기 위한 분류기를 지정. LinearSvm, GMM 두가지가 선택 가능. 기본값 : LinearSvm	
	workDir	Representation 값이 있는 경로	
infer	classifierModel	train 단계에서 작성된 pkl 파일을 입력.	
	imgs	분류기를 통해 예측을 수행할 이미지 혹은 이미지가 포함된 경로	

./demos/classifier.py train <feature-directory> --networkModel <torch networkModel>
./demos/classifier.py infer --networkModel <torch networkModel> <Train Result.pkl> images/examples/{carell,adams,lennon}*n <feature-directory>

./training/main.lua

Torch DNN을 학습시키는 프로그램

인자 구분	인자값	비고	기본값	인자 구분	인자값	비고	기본값
general	-cache	캐시 디렉토리	Work	Data	-nDonkeys	데이터 로딩 쓰레드	2
	-save	학습된 결과를 저장할 경로	-	Training	-nEpochs	학습을 하기 위한 Epoch 카운트	1000
	-data	이미지 데이터 경로			-epochSize	한 epoch당 실행할 batch size	250
	-manualSeed	랜덤 시드 넘버			-epochNumber	Epoch Number 학습을 중단 후 재시작시 재 시작에 해당하는 값	1
	-cuda	Cuda 사용 여부	2		-peoplePerBatch	미니배치 수행시 읽어 들일 인물 클래스의 수량	15
	-device	사용할 cuda device 번호 지정.	True		-peoplePerPerson	미니배치 수행시 읽어 들일 클래스내에서의 이미지 수량	20
	-nGPU	최대 n개의 GPU 사용	1		-testing	이 값이 true 일 경우 LFW 테스트 수행	True
	-cudnn	Cudnn 사용 여부	True		-testBatchSize	Test 수행시 한번에 읽어 들일 이미지 크기	800
	-cudnn_bench	Cudnn fast option 사용 여부 이 옵션을 사용하면 메모리 사용량이 증가함.	false		-lfwDir	Test 수행시 일어 들일 lfw 데이터 위치	../data/lfw/aligned

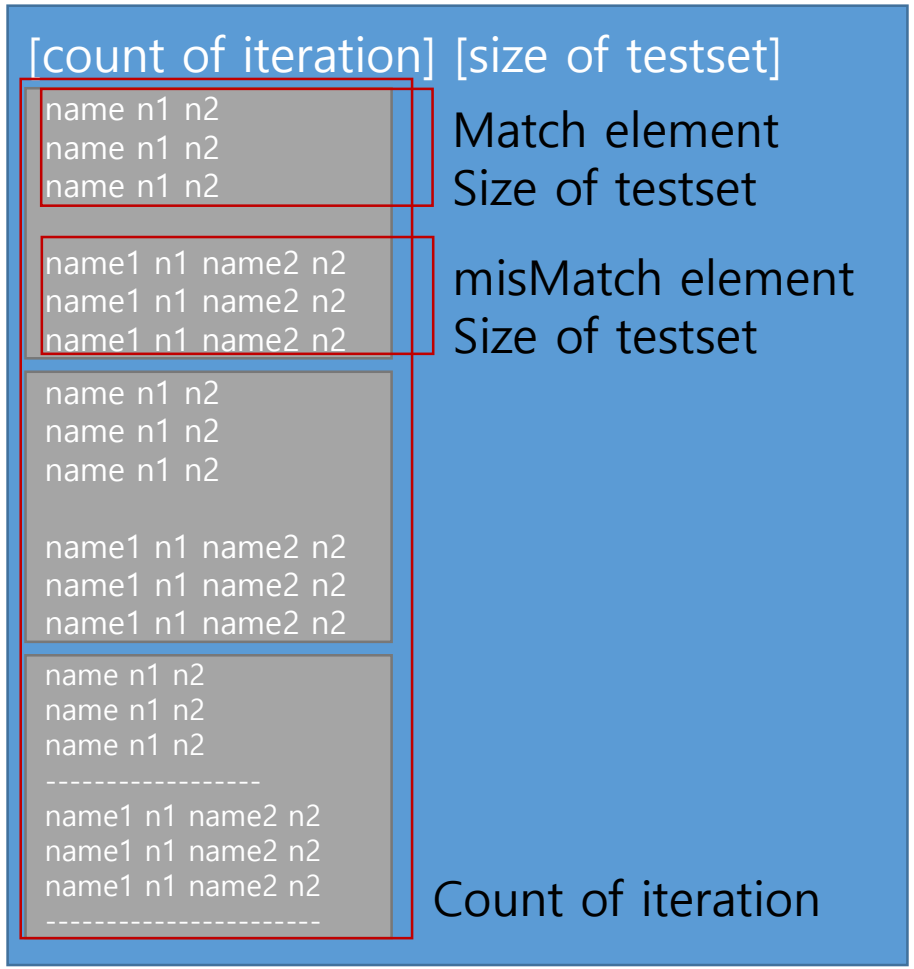
./training/main.lua

Torch DNN을 학습시키는 프로그램

인자 구분	인자값	비고	기본값
model	-retrain	재학습을 위한 모델의 경로	
	-modelDef	Torch DNN 모델이 정의된 파일의 경로	../models/openface/nn.def.lua
	-imgDim	Torch DNN 학습을 위한 이미지 크기	94
	-embSize	Torch DNN 을 통해 생성된 이미지의 embedding	128
	-alpha	TripletLoss 마진	2

Appendix LFW Pairs.txt

- LFW 데이타셋을 이용하여 정확률을 계산하기 위한 데이타 셋
- http://vis-www.cs.umass.edu/lfw/README.txt
  - 3.c pairs.txt format 참고.
- /root/openface/data/imgs/test1/aligned/pair_gen.py 작성
  - Count of iteration : 10
  - Size of testset : 300
  - Total line :
    - 1(head) + 10(size of iteration) * 300(match element) + 10(size of iteration) * 300(mismatch element)



Openface API Reference

package	class	Method	parameter	Parameter description	Return	비고
Openface	AlignDLib	contructor	facePredictor	Dlib 라이브러리 경로		생성자
		align	imgDim rgbImg Bb landmarkIndices skipMulti	이미지 크기, int 이미지, numpy,ndarray 얼굴을 인식할 영역,dlib.rectangle 얼굴을 인식하는 방법, list of ints 다수의 이미지 인식 처리 여부, bool	Rgb image Shape(imgDim ,imgDim,3)	이미지와 얼굴을 인식하는 방법을 입력받아 인식된 얼굴을 출력

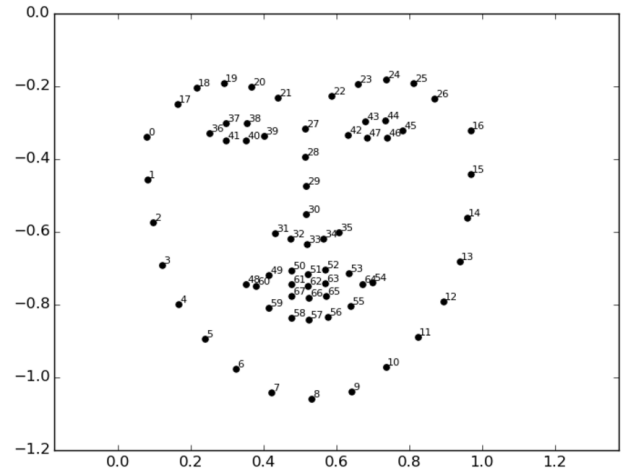
landmarkIndices

이 값은 입력된 이미지를 오른쪽과 같이 맵핑 시켜 특정 지점의 랜드마크값을 이용하여 얼굴을 인식함

기본적으로 정의된 값

INNER_EYES_AND_BOTOOM_LIP = [39.42.57]

OUTER_EYES_AND_NOSE = [36,45,33]



Openface API Reference

package	class	Method	parameter	Parameter description	Return	비고
Openface	AlignDLib	findLandmarks	rgbImg bb	이미지, numpy.ndarray 영역, dlib.rectangle	List of (x,y) tuple	이미지와 영역을 입력, 얼굴로 인식된 영역을 반환
		getAllFaceBoundingBoxed	rgbImg	이미지, numpy.ndarray	Dlib.rectangles	이미지에서 얼굴로 인식된 모든 영역을 반환
		getLargestFaceBoundingBox	rgbImg skipMulti	이미지, numpy.ndarray 불린값, false	Dlib.rectangles	입력된 이미지에서 얼굴로 인식된 영역 중 가장 큰 영역을 반환,

Openface API Reference

package	class	Method	parameter	Parameter description	Return	비고
Openface	TorchNueralNet	constructor	model imgDim Cuda	Torch 모델 파일 경로, string 이미지 크기, int Cuda 사용 여부, bool		Torch 모델을 이용한 DNN 서비스를 실행
		forward	rgbImg	이미지, numpy.ndarray	Numpy.ndarray	이미지를 DNN에 적용시켜 나온 결과값
		forwardPath	imgPath	이미지 경로, string	Numpy.ndarray	입력된 경로의 이미지를 DNN에 적용시켜 나온 결과값

Openface API Reference

package	class	Method	parameter	Parameter description	Return	비고
Openface	data	Image	Class Name Path	이미지의 클래스,string 이미지의 이름, string 이미지의 경로, string	Image 객체 생성	클래스, 이름, 경로를 통해 특정 이미지를 생성
		getBGR()			numpy.ndarray	이미지 객체에서 BGR 포맷의 데이터를 획득
		getRGB()			numpy.ndarray	이미지 객체에서 RGB 포맷의 데이터를 획득
		iterImgs	directory	이미지를 포함하는 디렉토리 위치,string	Iterator of images	디렉토리 내부에서 이미지를 읽어 들이고 그것들을 메모리 로딩, 이터레이터를 반환
	helper	mkdirP	path	디렉토리 경로,string		특정 디렉토리를 생성, 이미 있을 경우 스킵

Predict process