Face recognition using Deep Learning

Master Thesis

Author: Serra, Xavier[1]   *Advisor*: Castán, Javier[2]
Tutor: Escalera, Sergio[3]

[1]Master in Artificial Intelligence
Barcelona School of Informatics

[2]*GoldenSpear LLC*

[3]*Department of Mathematics and Computer Science*
University of Barcelona

Barcelona School of Informatics, January 2017

Outline

Outline

Uses of Face Recognition

Face Recognition has drawn plenty of *attention*

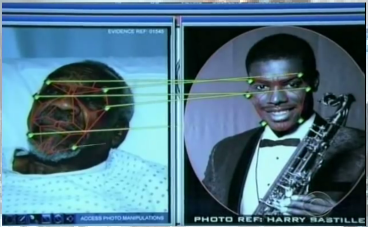
It has potential for multiple applications:

- Biometrical verification
- Search for a person through cameras
- Automatically tagging friends
- Finding similar people
- ...

So, what is actually Face Recognition?

Face Recognition in fiction

How has fiction pictured face recognition?



Actual Face Recognition

How does Face Recognition actually work?

- Eigenfaces
- Active Appearance Models
- Support Vector Machines
- Bayesian models
- Convolutional Neural Networks
- ...

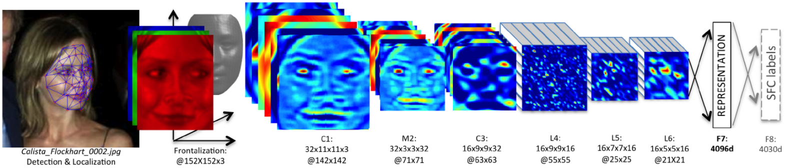


Figure: Example of a *CNN*

Goal of this master thesis

Developing a *face recognition* system so that:

- Keeps a DB of known users
- Given a new picture, determines the closest match
- Capable of on-line learning
- Usable in *uncontrolled* environments
- Reasonably fast

Outline

Face Recognition Problems

Many factors to take into account:

Face Recognition Problems

Many factors to take into account:

- Light conditions

Face Recognition Problems

Many factors to take into account:

- Light conditions
- Expression

Face Recognition Problems

Many factors to take into account:

- Light conditions
- Expression
- Face orientation

Face Recognition Problems

Many factors to take into account:

- Light conditions
- Expression
- Face orientation
- Age
- ...

Face Recognition Problems

Many factors to take into account:

- Light conditions
- Expression
- Face orientation
- Age
- ...

It can be summarized as *Intra-class variability*

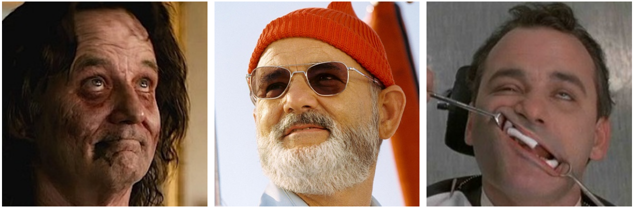


Figure: Intra-class variability

Face Recognition Problems

Inter-class similarity is also an issue:

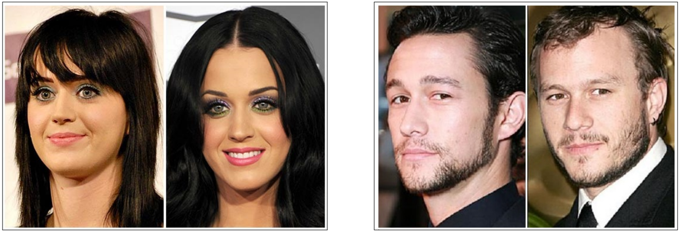


Figure: Inter-class similarity

Problems of raw images:

Common Face Recognition approach

Problems of raw images:

- Excessive *noise*

Common Face Recognition approach

Problems of raw images:

- Excessive *noise*
- Large *dimensionality*

Common Face Recognition approach

Problems of raw images:

- Excessive *noise*
- Large *dimensionality*
- *Variability* is too high

Common Face Recognition approach

Problems of raw images:

- Excessive *noise*
- Large *dimensionality*
- *Variability* is too high
- **Solution?**

Common Face Recognition approach

Problems of raw images:

- Excessive *noise*
- Large *dimensionality*
- *Variability* is too high
- **Solution?**

Convert input image into a reduced space

Common Face Recognition approach

Problems of raw images:

- Excessive *noise*
- Large *dimensionality*
- *Variability* is too high
- **Solution?**

Convert input image into a reduced space

Feature extraction

- Manually crafted
- Automatically found

- Reduces faces into more compact representations
- Uses *PCA* to produce those
- Set of *eigenvectors* from the *covariance matrix*
- Comparison by linear combination of *eigenfaces*

Figure: Set of eigenfaces

- Fits a pre-defined face shape into the image
- Iteratively improves initial estimation
- Allows finding sets of relevant points

Figure: Active Appearance Models fitting a face shape

- Successful classifier in many problems
- Finds the hyperplane separating two problems
- Can be used to determine if two images belong to same person

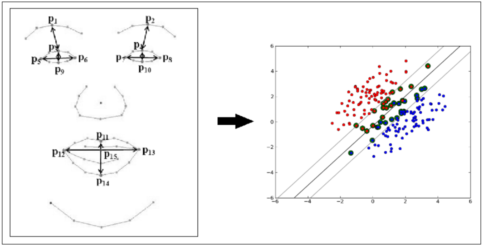


Figure: Application of Support Vector Machines

- Models each facial feature as $x = \mu + \epsilon$
- It corresponds to inter-class and intra-class variability
- Based on the full joint distribution of face image pairs

$$r(x_1, x_2) = \log \frac{P(x_1, x_2 | H_I)}{P(x_1, x_2 | H_E)}$$

- It is a type of *Artificial Neural Network*
- Works by finding increasingly *abstract* features
- Takes into account spatial relation
- High requirements in time and data
- Currently providing *state of art* results in many *CV* problems

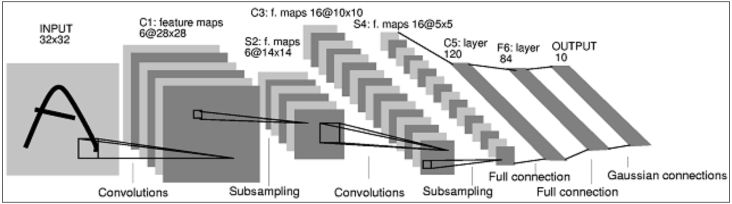


Figure: Convolutional Neural Network

Proposed approach

The proposed approach consists of 4 steps:

Proposed approach

The proposed approach consists of 4 steps:

Step 1: *Locating* the main face in the image

Proposed approach

The proposed approach consists of 4 steps:

Step 1: *Locating* the main face in the image

Step 2: *Frontalizing* the found face

Proposed approach

The proposed approach consists of 4 steps:

Step 1: *Locating* the main face in the image

Step 2: *Frontalizing* the found face

Step 3: Extracting features using a *CNN*

Proposed approach

The proposed approach consists of 4 steps:

Step 1: *Locating* the main face in the image

Step 2: *Frontalizing* the found face

Step 3: Extracting features using a *CNN*

Step 4: Performing *comparison* with stored ones

Step 1: Locate the face

Goal: Look for the *bounding box* of the most likely face

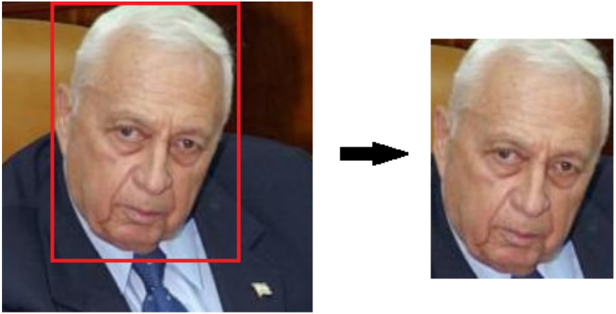


Figure: Locating the face

Benefit: Prevent erroneously located faces in next step

Step 1: Locate the face

Procedure:

- Using a region based Convolutional Neural Networks (Faster RCNN [RHGS15])
- Set of possible face locations is produced
- Most promising face is kept: *distance to center + confidence*

Figure: Selecting most likely face

Step 2: Frontalize the face

Goal: Frontalize the face so that it is looking at the camera

Figure: Frontalizing the found face

Benefits: Eliminate background noise + Equally placed faces

Step 2: Frontalize the face

Procedure:

1. Locate a set of 46 fiducial points
2. Consider the same points in a 3D pre-defined model
3. Generate a projection matrix to map from 2D input to the 3D reference
4. (Apply vertical similarity to fill in empty spots) $\leftarrow$ **Discarded**

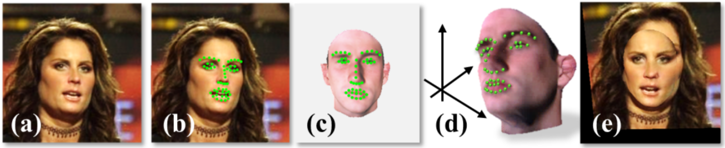


Figure: Frontalization process [HHPE15]

Step 2: Frontalize the face

Not working perfectly:

Figure: Examples of successful and unsuccessful frontalizations

Step 3: Extract relevant features

Goal: Automatically extract a set of relevant features from the face

Benefits: More efficient comparison + Reduction in variability

Step 3: Extract relevant features

Goal: Automatically extract a set of relevant features from the face

Benefits: More efficient comparison + Reduction in variability

Procedure:

- A *CNN* has been used to process each image
- Each image is compressed into a reduced representation
- A feature vector of 4096 features is generated
- Based on Facebook's *DeepFace* method [TYRW14]

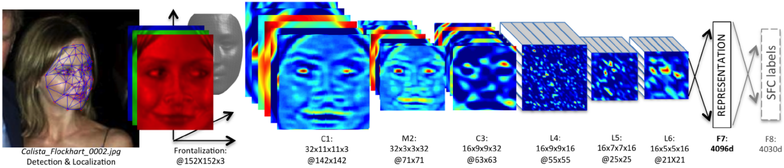


Figure: *CNN* architecture used

Step 4: Compare them

Goal: Perform the comparison with DB to look for a match

Procedure:

1. Given a generated feature vector g
2. Iterates over all people in the DB
3. Each person has N relevant feature vectors $F = f_1, f_2, ...f_N$
4. Distance comparison is performed between g and each $f_i \in F$
5. Various selection measures considered: minimum, mean, etc.

Datasets used

Three datasets considered:

- *Casia dataset:* 495,000 pictures / 10,500 people
- *CACD dataset:* 160,000 pictures / 2,000 people
- *FaceScrub:* 100,000 pictures / 500 people
- Training: 500,000 pictures / 9,351 people
- Testing: 100,000 pictures / 1,671 people

Additionally, to use as a benchmark:

- *Labeled Faces in the Wild*: 13,000 pictures / 5,700 people

From training dataset, we generated two extra:

Datasets used

Generated datasets

From training dataset, we generated two extra:

- **Augmented**:
  - Using data augmentation
  - Randomly modifying light intensity
  - Other data augmentations made not much sense – rotation, scaling, etc.
  - 1M instances

Datasets used

Generated datasets

From training dataset, we generated two extra:

- **Augmented**:
  - Using data augmentation
  - Randomly modifying light intensity
  - Other data augmentations made not much sense – rotation, scaling, etc.
  - 1M instances
- **Grayscale**:
  - Convert previous dataset to grayscale
  - Aims to make the problem easier for CNN
  - Both training and testing sets converted
  - *CNN* modified accordingly

Outline

How to evaluate their performance?

Face Recognition systems can be evaluated according to:

- *Face Verification*
- *Face Recognition*

How to evaluate their performance?

Face Recognition systems can be evaluated according to:

- *Face Verification*
- *Face Recognition*

Both intrinsically related...

How to evaluate their performance?

Face Recognition systems can be evaluated according to:

- *Face Verification*
- *Face Recognition*

Both intrinsically related...
... but differently evaluated

Face Verification

Description

Goal: Determining whether two pictures belong to same person:

- Needed on most *Face Recognition* systems
- Performance not directly related with *Face Recognition* step
- Commonly used as benchmark to compare methods
- The *Labeled Faces in the Wild* dataset has been used
- 2000 training pairs / 1000 test pairs
- Allowed for hyperparameter tuning

Face Verification

Examples

Figure: Example on test pairs

Face Verification

Procedure

Comparison performed using *Euclidean* and *Taxicab* distances

Weighted variations considered but discarded due to bad results

Training consists in:

1. Obtain distance between all train pairs
2. Find the optimal threshold placement to separate classes

Comparison performed using *Euclidean* and *Taxicab* distances

Weighted variations considered but discarded due to bad results

Training consists in:

1. Obtain distance between all train pairs
2. Find the optimal threshold placement to separate classes

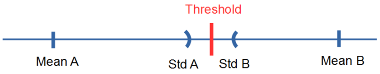


Figure: Example best case scenario

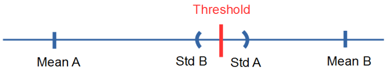


Figure: Example more difficult scenario

Method	Accuracy
Ours	0.896
Joint Bayesian	0.9242
Tom-vs-Pete	0.9330
High-dim LBP	0.9517
TL Joint Bayesian	0.9633
FaceNet	0.9963
DeepFace	0.9735
Human performance	0.9753

Table: Results state of art methods

Reasons

- Too few training data
- Further need for parameter tuning
- Improve distance metric

Results

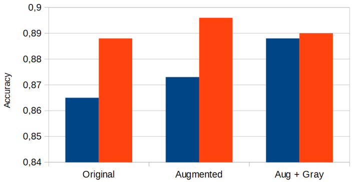


Figure: Accuracy according to dataset

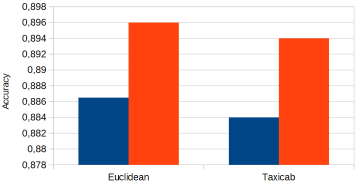


Figure: Accuracy according to distance

Goal: Determining *who* the person is:

- Select among a set of people in a DB
- Person-wise comparison $\Rightarrow$ Face Verification
- Closest match is selected
- Need to determine if there is a match at all
- Seemingly more difficult than Face Verification...
- ... empirical results prove it may not be so

Reminder:

- comparing feature vector f with all people in DB
- Each person has N feature vectors $F = f_1, f_2, ...f_N$

Comparison strategies:

Reminder:

- comparing feature vector f with all people in DB
- Each person has N feature vectors $F = f_1, f_2, ...f_N$

Comparison strategies:

1. Distance to closest feature vector in F

Reminder:

- comparing feature vector f with all people in DB
- Each person has N feature vectors $F = f_1, f_2, ...f_N$

Comparison strategies:

1. Distance to closest feature vector in F
2. Mean distance to all $f_i \in F$

Reminder:

- comparing feature vector f with all people in DB
- Each person has N feature vectors $F = f_1, f_2, ...f_N$

Comparison strategies:

1. Distance to closest feature vector in F
2. Mean distance to all $f_i \in F$
3. Product of 1 and 2

Reminder:

- comparing feature vector f with all people in DB
- Each person has N feature vectors $F = f_1, f_2, ...f_N$

Comparison strategies:

1. Distance to closest feature vector in F
2. Mean distance to all $f_i \in F$
3. Product of 1 and 2
4. Product of distance to furthest feature vector in f and 3

Reminder:

- comparing feature vector f with all people in DB
- Each person has N feature vectors $F = f_1, f_2, ...f_N$

Comparison strategies:

1. Distance to closest feature vector in F
2. Mean distance to all $f_i \in F$
3. Product of 1 and 2
4. Product of distance to furthest feature vector in f and 3

The smallest distance is chosen as a match

Each new feature vector f may be kept into the system:

Each new feature vector f may be kept into the system:

1. If less than T_1 feature vectors stored, keep it

Each new feature vector f may be kept into the system:

1. If less than T_1 feature vectors stored, keep it
2. If distance $\mathcal{M}$ between f and mean of F less than T_2, discard it

Each new feature vector f may be kept into the system:

1. If less than T_1 feature vectors stored, keep it
2. If distance $\mathcal{M}$ between f and mean of F less than T_2, discard it
3. If $\mathcal{M}$ higher than T_3, discard it (extreme outlier)

Face Recognition

Keeping Procedure

Each new feature vector f may be kept into the system:

1. If less than T_1 feature vectors stored, keep it
2. If distance $\mathcal{M}$ between f and mean of F less than T_2, discard it
3. If $\mathcal{M}$ higher than T_3, discard it (extreme outlier)
4. Select the feature vectors - F_O - far from mean (outliers)

Face Recognition

Keeping Procedure

Each new feature vector f may be kept into the system:

1. If less than T_1 feature vectors stored, keep it
2. If distance $\mathcal{M}$ between f and mean of F less than T_2, discard it
3. If $\mathcal{M}$ higher than T_3, discard it (extreme outlier)
4. Select the feature vectors - F_O - far from mean (outliers)
5. Face Verification between f and all $f_i \in F_O$

Each new feature vector f may be kept into the system:

1. If less than T_1 feature vectors stored, keep it
2. If distance $\mathcal{M}$ between f and mean of F less than T_2, discard it
3. If $\mathcal{M}$ higher than T_3, discard it (extreme outlier)
4. Select the feature vectors - F_O - far from mean (outliers)
5. Face Verification between f and all $f_i \in F_O$
6. If less than half matches, keep it (rare enough case)

Face Recognition

Keeping Procedure

Each new feature vector f may be kept into the system:

1. If less than T_1 feature vectors stored, keep it
2. If distance $\mathcal{M}$ between f and mean of F less than T_2, discard it
3. If $\mathcal{M}$ higher than T_3, discard it (extreme outlier)
4. Select the feature vectors - F_O - far from mean (outliers)
5. Face Verification between f and all $f_i \in F_O$
6. If less than half matches, keep it (rare enough case)
7. If more than T_4 feature vector stored, discard closest to mean

Each new feature vector f may be kept into the system:

1. If less than T_1 feature vectors stored, keep it
2. If distance $\mathcal{M}$ between f and mean of F less than T_2, discard it
3. If $\mathcal{M}$ higher than T_3, discard it (extreme outlier)
4. Select the feature vectors - F_O - far from mean (outliers)
5. Face Verification between f and all $f_i \in F_O$
6. If less than half matches, keep it (rare enough case)
7. If more than T_4 feature vector stored, discard closest to mean

T_1, T_2, T_3 and T_4 are hyperparameters

Face Recognition
Dataset

Self-generated dataset, from training dataset:

- 100 people (50 females / 50 males)
- 30 training images each
- 50 training images each
- Manually cleaned

Face Recognition

Results

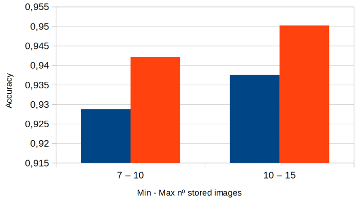


Figure: Accuracy according to number kept images

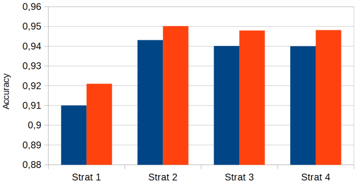


Figure: Accuracy according to comparison strategy

Face Recognition
Results

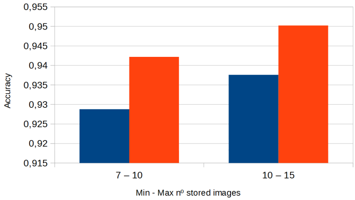


Figure: Accuracy according to number kept images

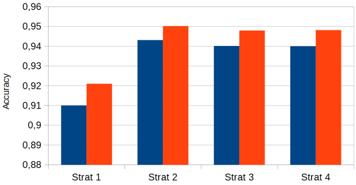


Figure: Accuracy according to comparison strategy

A 95% of accuracy was reached

Outline

To conclude...

- We have developed a functional Face Recognition System using *CNN*s
- Works in uncontrolled environment, capable of on-line learning
- Compared with state of art methods, it underperforms in Face Verification
- Quality results achieved in Face Recognition
- Exhaustive tests performed – reliable results

... or not!

Future work lines:

- Improve *CNN* performance:
  - More data
  - Better parameter tuning
- Test more comparison metrics:
  - Further try thresholding strategies
  - Different weights
- Enhance matching capabilities:
  - Use more complex strategies – apart from *min*, *mean*, etc.
  - Modify on-line learning mechanism
- Consider other alternatives for feature extraction:
  - Other existing approaches
  - Develop one on our own

References I

Tal Hassner, Shai Harel, Eran Paz, and Roee Enbar, *Effective face frontalization in unconstrained images*, IEEE Conf. on Computer Vision and Pattern Recognition (CVPR), June 2015.

Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun, *Faster r-cnn: Towards real-time object detection with region proposal networks*, Advances in Neural Information Processing Systems 28 (C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, eds.), Curran Associates, Inc., 2015, pp. 91–99.

Yaniv Taigman, Ming Yang, Marc'Aurelio Ranzato, and Lior Wolf, *Deepface: Closing the gap to human-level performance in face verification*, Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition (Washington, DC, USA), CVPR '14, IEEE Computer Society, 2014, pp. 1701–1708.

Any Question?

Thank you!

Thank you for your attention!