Facial Identification Using A Multilayer Perceptron

David Lavy MS ENG ‘16
Sean Matuszak CAS ‘16
Sean Smith CAS ‘17

Introduction

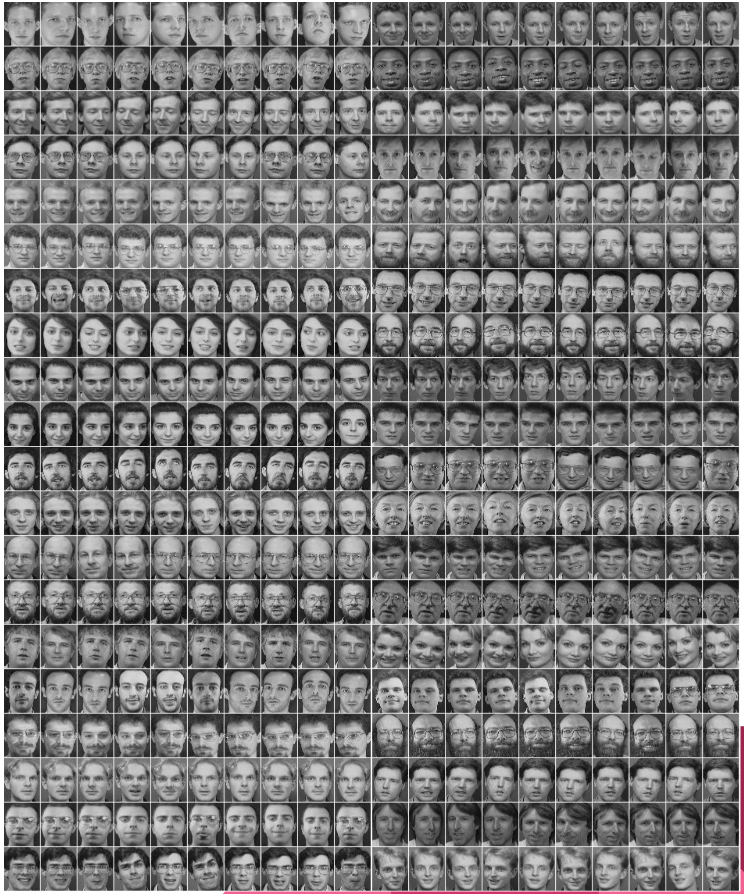
- Why Face Recognition?
  - Facebook
  - Surveillance
  - Crime fighting
- Why Neural Networks?
  - State-of-the-art pattern recognition
  - “Bionic Image Analysis” - mimics biology

Goals

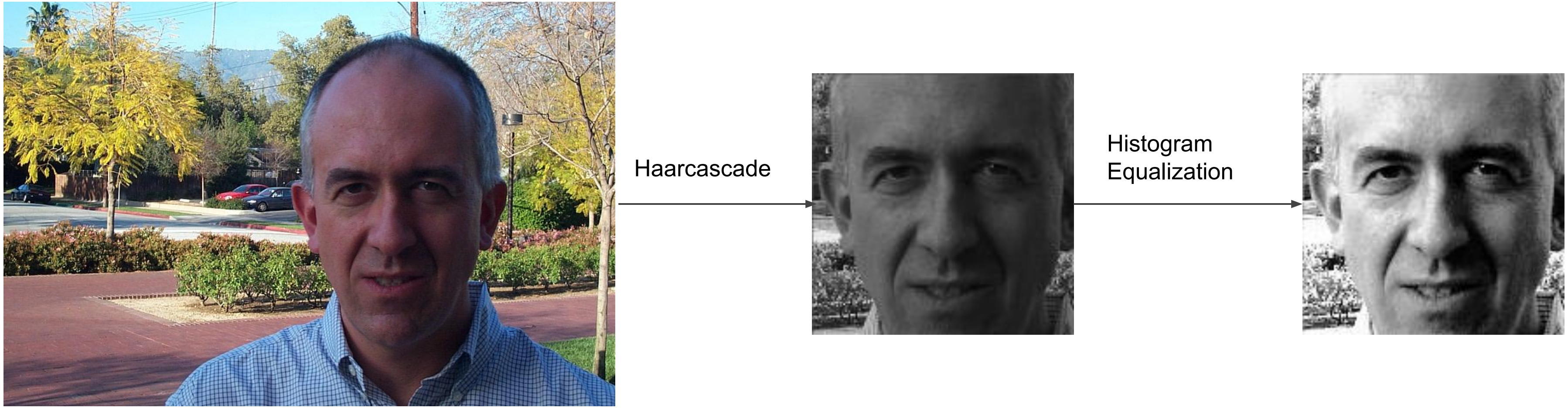
1. Implement and train a multilayer perceptron (MLP)
2. Train the MLP on pixel values
3. Improve results by using computer vision techniques to pre-process
4. Approach State-of-the-Art detection rates

Data Sets

1. ATT Dataset (right)
    - 10 pictures of each of 40 subjects
    - Cropped and Scaled
    - Greyscale
    - Uniformly lit
2. Caltech Frontal Face Dataset
    - Full-size color images
    - 10 pictures of 19 subjects
    - Preprocessesing



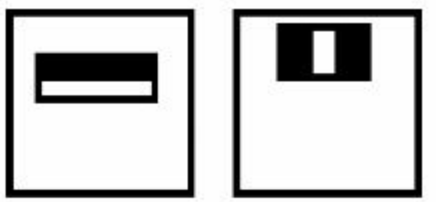
Caltech Frontal Face Data Set - preprocessing



Haarcascades for Facial Segmentation

- Edge, Line, Four-rectangle features detected
- Pre-trained system detects features that are face-like
- For each window create “stages” (groups of classifiers with smaller numbers of features in the earlier stages)
    - if a stage fails, reject the window as a face for efficiency
    - a window that passes all stages (i.e. all features are regarded as face-like), a face is identified
- Method proposed by Paul Viola and Micheal Jones in “Rapid Object Detection using a Boosted Cascade of Simple Features” (2001)
- Used OpenCV library call and a haarcascade xml (to get the trained coefficients)

Haarcascades



Example of Haarcascades filters

Haarcascades filters applied to image

→ Haar features (.xml)

image from openCV documentation

Histogram Equalization for Lighting Correction

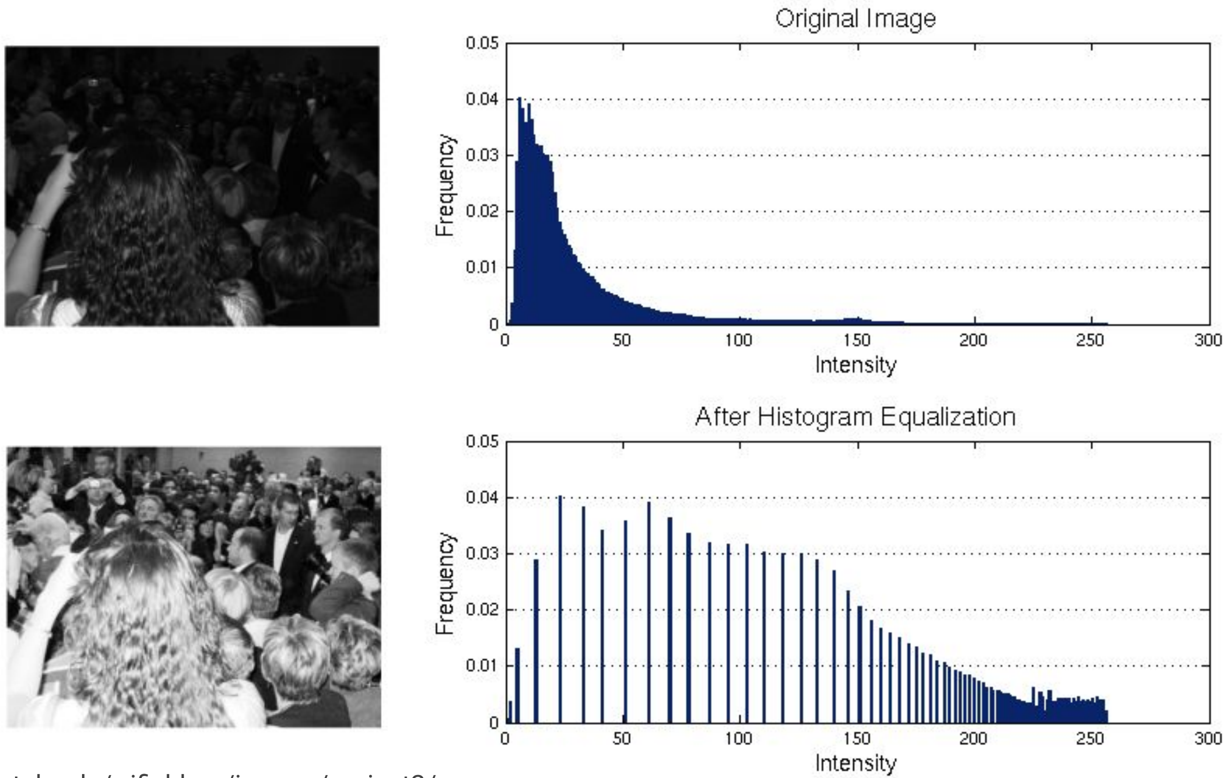


Image from http://www.cs.utah.edu/~jfishbau/improc/project2/

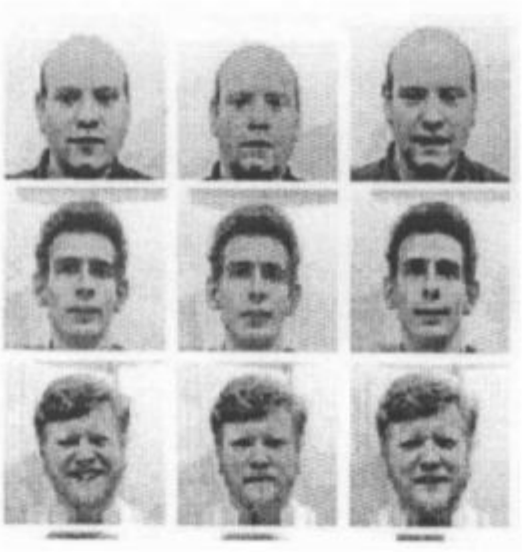
HQ in our set of images

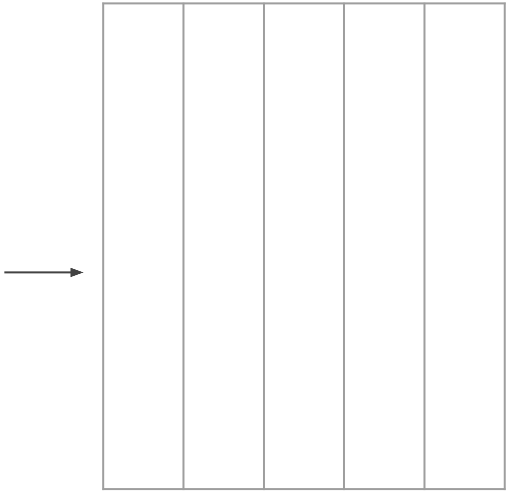
BEFORE HQ

HQ in our set of images

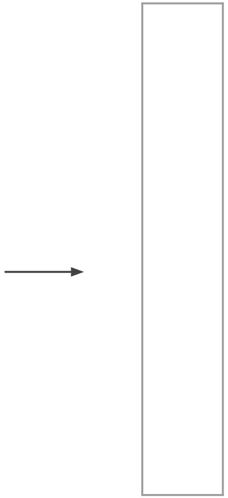
AFTER HQ

Eigenfaces: Calculation





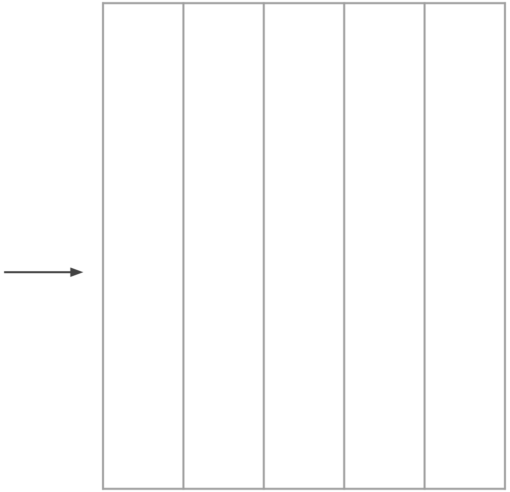
Represent face as a vector Γ_i



$$\Psi = \frac{1}{M}\sum_{i=1}^{M}\Gamma_i$$

Calculate average face

Eigenfaces: Calculation



Substrace the
average face
(normalized)

$$\Phi_i = \Gamma_i - \Psi$$

Correlation matrix

$$C = \frac{1}{M} \sum_{i=1}^{M} \Phi_i \Phi_i^T \quad (N^2 x N^2) \text{ matrix}$$

$$A = [\Phi_1 \ \Phi_2 \ldots \Phi_M] \quad (N^2 x M) \text{ matrix}$$

Γ_i

Eigenfaces: Calculation

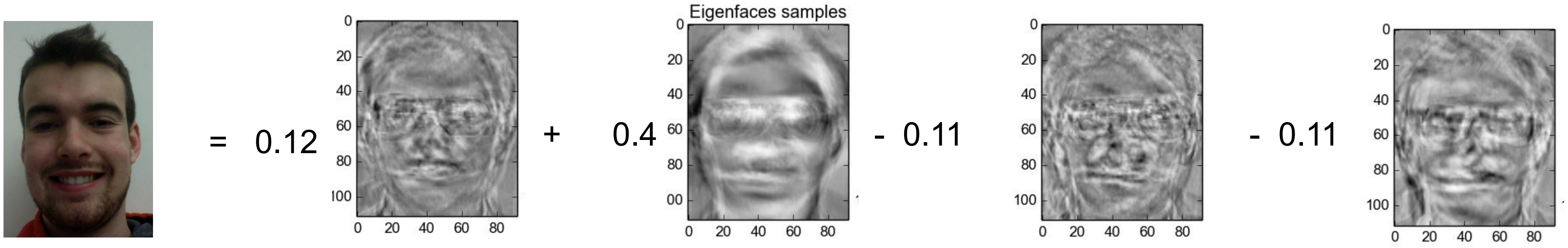
→ Calculate eigenvectors and eigenvalues → Keep the best eigenvectors that correspond to the largest eigenvalues (K values) → Normalize the K largest eigenvectors →

u_i, v_i

→ $$\hat{\Phi}_i - mean = \sum_{j=1}^{K} w_j u_j \quad , w_j = u_j^T \Phi_i$$

Eigenfaces examples

- Faces turned into a vector of coefficients which represent a face’s summation of several generated “eigenfaces”
- Commonly used method in facial recognition



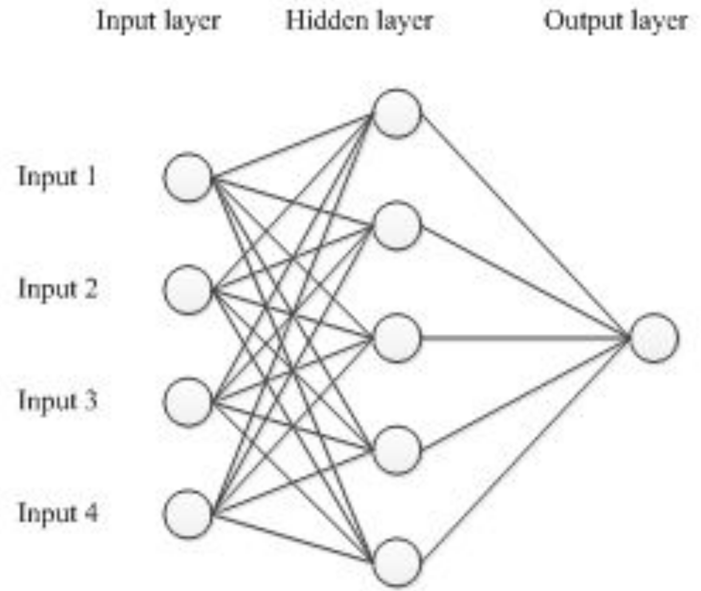
A sample of actual eigenfaces generated and Sean Matuszak as an example.
Note - coefficients are not the real results for Sean’s face

Multilayer Perceptron

- Neural Network
- Implemented with the PyBrain Library
- Sigmoid Function

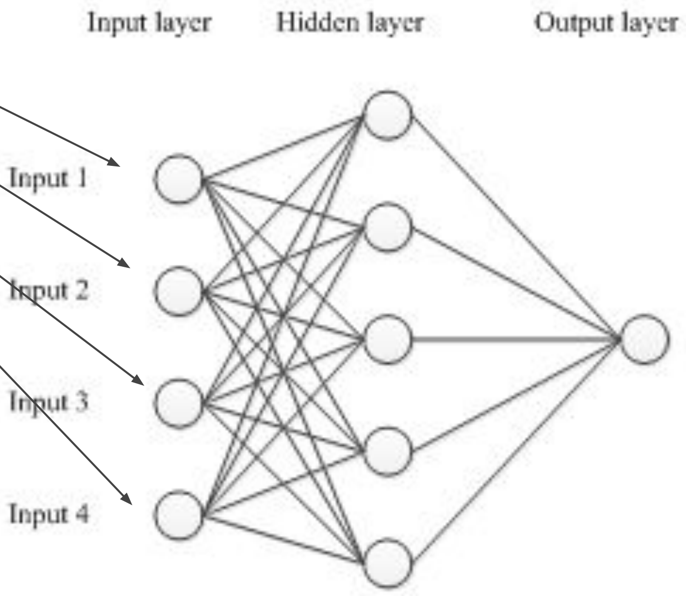
$$S(t) = \frac{1}{1 + e^{-t}}.$$

- Input: eigenvectors or pixel values
- Hidden layer: sigmoid curve
- Output layer: Binary array where the index corresponds to



Concrete Example

[-1109.92254531,
3822.22955533,
-115.90735887,
-1806.63734274,
-877.09674426,
-1835.81884568,
263.87997572,
403.22660642,
447.07778044,
-110.09288836,
-278.75246454,
221.73020268,
...
120.89521511,
-110.33682746]



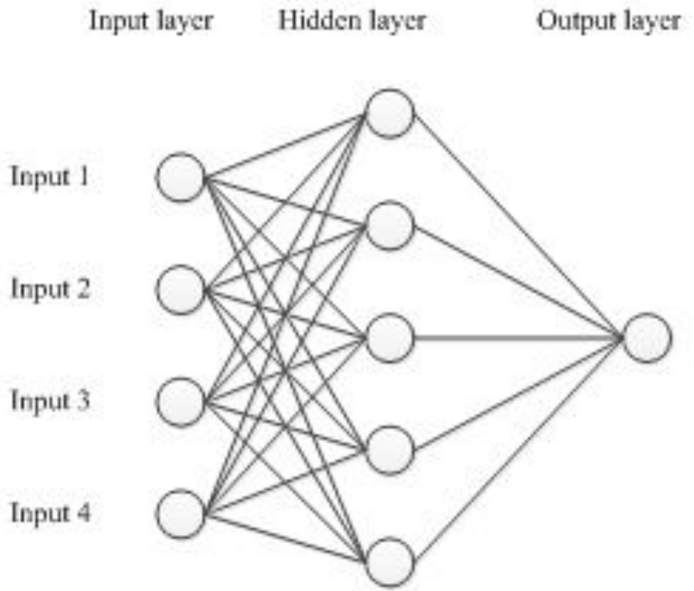
hidden layer has 150 nodes

Labeled Outcome:
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ,0, 1, 0]

Actual Outcome:
[0.01199941, 0.0056677
0.00703534, 0.03423871
0.00178826, 0.01630305
0.00516428, 0.00117239
0.00124584, 0.00368228
0.01062241, 0.00692866
0.00647613, 0.05230364
0.00095177, 0.15660527
0.28097068, 0.00765588
0.3891883]

MLP - Training: Backward Propagation of Errors

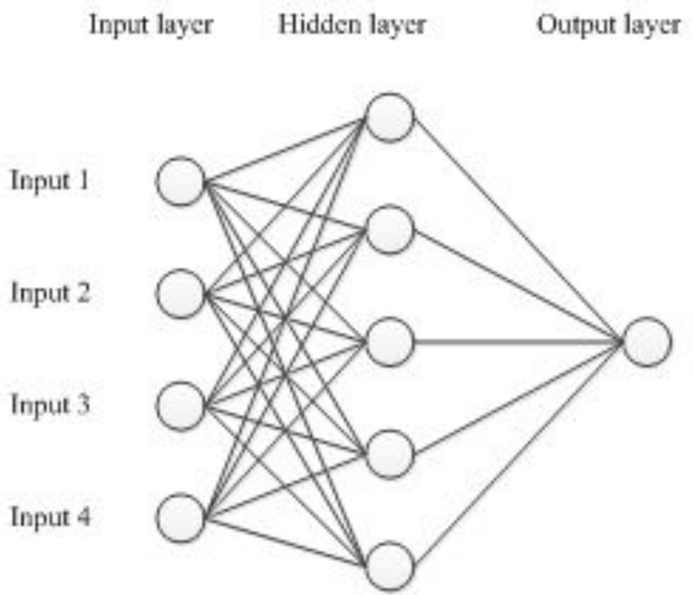
- Phase One - Propagation
  - Runs input forward through the MLP, saving values at each neuron
  - Runs result backwards through MLP, saving values that “should have” been generated at each neuron based on the current weights
  - Calculates the difference at each neuron and saves it as “delta”



Description adapted from https://en.wikipedia.org/wiki/Backpropagation

MLP - Training: Backward Propagation of Errors

- Phase Two - Updates
  - At each "synapse" between a weight and a neuron, multiply each output delta by the input activation to get a gradient
  - Generate a correction based on the learning rate (a percentage of the gradient) to subtract from current weight values
    - high learning rate/high percentage will train faster, but be less accurate
    - low learning rate will require more training iterations, but will be more accurate



Description adapted from https://en.wikipedia.org/wiki/Backpropagation

3 Phases of Testing

1. Naive approach
2. ATT dataset
3. Caltech frontal dataset

Testing Phase One - Naive Approach

- Input: All Pixel Values
- Hidden Layer: 3 Neurons
- Output: One output neuron, outputting number corresponding to the individual tested (i.e. “subject 1 - 40”)
- 200 rounds of training

Testing Phase One - Results

- Very low correct identification (true positive) rate
- MLP consistently returning ~20, why?
- Backpropagation Algorithm lowers deltas for each individual
  - Ideally, this would minimize by teaching the neural net which values give correct values to ID faces
  - Realistically with the number of input and hidden nodes, we were minimizing the sum of squares by setting the output to the average output between 1 and 40
- Conclusions
  - Too many input nodes
  - Too few hidden nodes
  - Too many classes (individuals to be identified)

Testing Phase Two - Eigenfaces

- Input Layer: eigenface values (input = 300, 75% of the number of images)
- Hidden Layer: 150 nodes (half of 300, based on our trial and error)
- Output Layer: 1 node outputs a vector of size 40 (number of classes)
  - max index selected, corresponding to the subject's index
- Training Rounds: 500
- Results: 82.5% correct identification rate

Testing Phase Two - Confusion Matrix

Truth Value

Hypothesized Value

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40
1	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
8	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
12	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
13	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
14	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
17	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
18	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
19	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
20	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
21	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
22	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
23	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
24	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
25	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
26	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0
27	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
28	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0
29	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0
30	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0
31	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0
32	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0
33	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0
34	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0
35	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1
36	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
37	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0
38	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0
39	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0
40	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2

Testing Phase Three - Facial extraction, brightening, and eigenfaces

- Faces segmented using Haarcascades
- Faces were normalized using histogram equalization
- Faces were turned into eigenfaces
- 92.1% correct identification rate

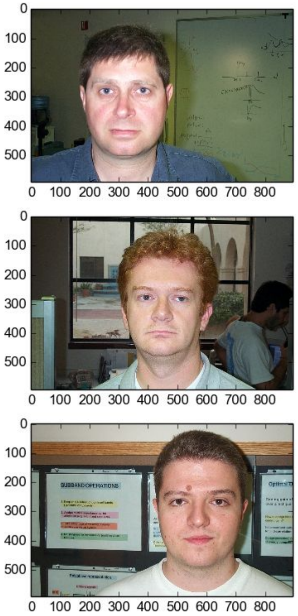
Testing Phase Three - Confusion Matrix

Truth Values

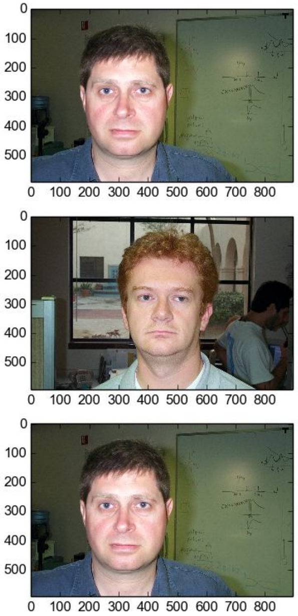
Hypothesized Values

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
1	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0
6	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0
12	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0
13	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0
14	0	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0
15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0
16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0
17	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0
18	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0
19	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2

Visual Results



Actual Data



Predicted Data

True Positive Rate with Different Splits

	50%	60%	70%	80%	90%
Training Set	5	6	7	8	9
Testing Set	5	4	3	2	1
True Positive Rate	70.53	78.95	82.46	92.11	94.74

Training set	Number of images per person								
Training set	Number of images per person								
1	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	
Testing set	*9*	*8*	*7*	*6*	*5*	*4*	*3*	*2*	*1*
Eigenfaces/MLP	32.5%	39.0%	48.0%	54.5%	59.5%	66.0%	73.0%	82.0%	86.0%

Published work: Using MLP and RBF Neural Networks for Face Recognition: an Insightful Comparative Case Study" published by Hala M. Ebeid in 2011

Interesting Result - Robust to Lighting Conditions

- Testing Histogram Equalization against unadjusted input did not result in a significant difference
- Eigenfaces have a method where they subtract the “average face”, making the eigenfaces robust to lighting conditions

State-of-the-Art

- Current MLP SOTA:
  - “Using MLP and RBF Neural Networks for Face Recognition: an Insightful Comparative Case Study” published by Hala M. Ebeid in 2011 used an MLP and eigenfaces to detect individuals from a set of 40, and achieved a correct identification rate of 82.0% for a 80/20% training/testing split
- Other Facial detection methods:
  - The RBF (radial basis function neural network) results were 83.0%, slightly better than MLP
  - Facebook, an industry leader in facial identification, uses euclidean distance between facial features to identify faces. Facebook correct identification data was not found.
- Our results exceed current SOTA!
  - For our 40-person test, our correct identification rate were 82.5%, marginally higher than current MLP data!
  - For our 19-person test, our correct identification rate 92.1%, impressive considering this was segmented out of a whole image

Limitations

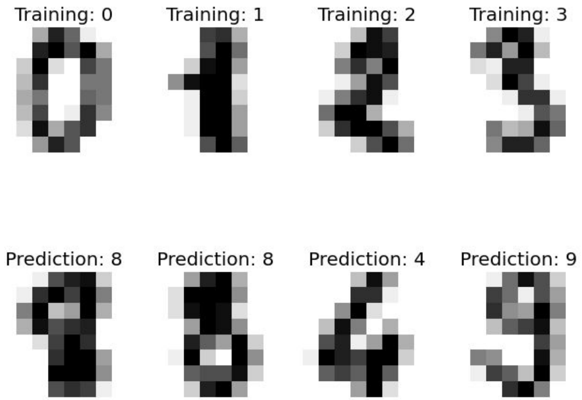
- Inherent limitation of training: cannot detect faces which are untrained
- Input size should be scaled to a manageable size
- Larger numbers of classes give worse results
- Training/testing is time-consuming
- hidden nodes must be configured manually
- Detections are binary, not a level of certainty

Conclusions

- Haarcascades is a fast and reliable method to detect faces
- Multilayer perceptrons are effective in classifying faces, but with ~20% error there is lots of progress to be made
- Eigenfaces are valuable in reducing the size of the input and in finding key facial features, and are robust to lighting conditions
- PyBrain is useful but not well-documented
  - Machine learning libraries are not well-documented in general:
    - Fann is written in C++ has python wrappers (worse documented)
    - Tensor Flow (mostly for convolutional NN)
- Ideas for future study
  - Euclidean Distance - face recognition “golden standard” used by Facebook
  - Convolutional Neural Network - more complicated neural network structure

Extra - Integer Dataset

- Proof of Concept
- Input: 64 pixels
- Hidden: 8 nodes (squareroot usage for hidden nodes is standard)
- Output: The integer that the picture corresponds to
- 200 rounds of training



Extra - Results

- Reliable Results
  - Only 2% error in test cases
  - TODO - get confusion matrix and correct identification rate
  - Certain patterns were unique, such as zero which had a 100% detection rate
  - Some numbers were easily confusable such as 8s and 1s
- Conclusions
  - Small number of inputs yields good results
  - Small classification size yields good results
  - Square root as the number of hidden nodes yields good results (based on research)
  - Similar patterns were confused more than others, predicting that some faces will be more distinct than others