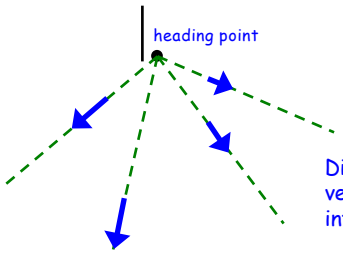
Observer motion problem

From image motion, compute:

- Observer translation
  $(T_x\ T_y\ T_z)$
- Observer rotation
  $(R_x\ R_y\ R_z)$
- Depth at every location
  $Z(x,y)$

1-1

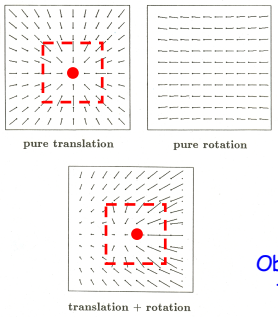
Observer translates toward hearing point



But... simple strategy doesn't work if observer also rotates

1-2

Observer motion problem, revisited



From image motion, compute:

- Observer translation
  $(T_x\ T_y\ T_z)$
- Observer rotation
  $(R_x\ R_y\ R_z)$
- Depth at every location
  $Z(x,y)$

Observer undergoes both translation + rotation

1-3

Equations of observer motion

Translation	Rotation	Depth
(T_x, T_y, T_z)	(R_x, R_y, R_z)	$Z(x,y)$

$$V_x = (-T_x + xT_z)/Z + R_x xy - R_y(x^2+1) + R_z y$$

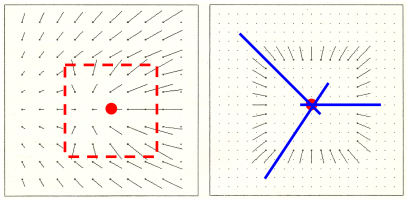
$$V_y = (-T_y + yT_z)/Z + R_x(y^2+1) - R_y xy - R_z x$$

Translational Component

Rotational Component

1-4

Longuet-Higgins & Prazdny



- Along a depth discontinuity, *velocity differences* depend only on observer translation
- Velocity differences point to the observer's heading point (FOE of the translational component of motion)

1-5

What is a chair?

Alignment methods

Find an object model and geometric transformation that *best match* the viewed image

- V viewed object (image)
- M_i object models
- T_{ij} allowable transformations between viewed object and models
- F measure of fit between V and the expected appearance of model M_i under the transformation T_{ij}

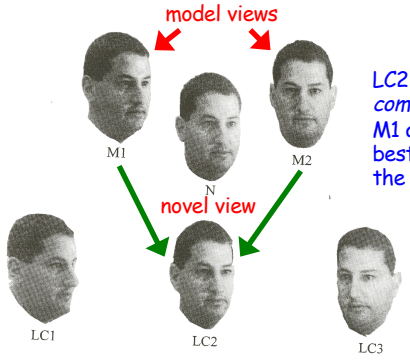
GOAL: Find a combination of M_i and T_{ij} that maximizes the fit F

Alignment method: recognition process

(1) Find best transformation T_{ij} for each model M_i (optimizing over possible views)

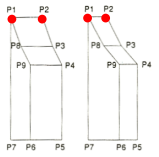
(2) Find M_i whose best T_{ij} gives the best match to image V

Recognition by linear combination of views



LC2 is a *linear combination* of M1 and M2 that best matches the novel view

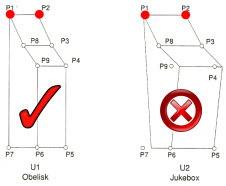
Predicting object appearance



I_1 I_2

two known views of obelisk

I_0 I_0



$X_{P_1I_0} = \alpha X_{P_1I_1} + \beta X_{P_1I_2}$

$X_{P_2I_0} = \alpha X_{P_2I_1} + \beta X_{P_2I_2}$

Recognition process:

(1) compute α, β that predict P1 and P2

(2) use α, β to predict other points

(3) evaluate fit of model to image

Why is face recognition hard?

changing pose

changing illumination

changing expression

aging

clutter

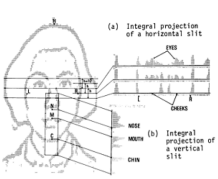
occlusion

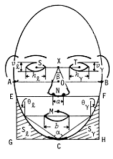
familiar vs. unfamiliar faces

It all began with Takeo Kanade (1973)…

PhD thesis, *Picture Processing System by Computer Complex and Recognition of Human Faces*

- Special purpose algorithms to locate eyes, nose, mouth, boundaries of face
- ~ 40 geometric features, e.g. ratios of distances and angles between features

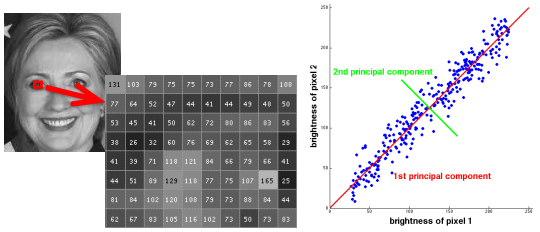




Eigenfaces for recognition (Turk & Pentland)
Principal Components Analysis (PCA)

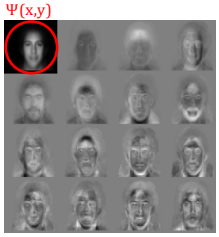
Goal: reduce the dimensionality of the data while retaining as much information as possible in the original dataset

PCA allows us to compute a linear transformation that maps data from a high dimensional space to a lower dimensional subspace



Eigenfaces for recognition (Turk & Pentland)

$\Psi(x,y)$



Perform ***PCA*** on a large set of training images, to create a set of *eigenfaces*, $E_i(x,y)$, that span the data set

First components capture most of the variation across the data set, later components capture subtle variations

$\Psi(x,y)$: average face (across all faces)

http://vismod.media.mit.edu/vismod/demos/facerec/basic.html

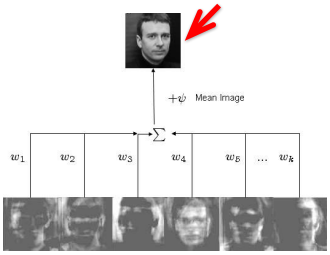
Each face image F(x,y) can be expressed as a weighted combination of the eigenfaces $E_i(x,y)$:

$$F(x,y) = \Psi(x,y) + \Sigma_i\, w_i * E_i(x,y)$$

Representing individual faces

Each face image F(x,y) can be expressed as a weighted combination of the eigenfaces $E_i(x,y)$:

$$F(x,y) = \Psi(x,y) + \Sigma_i\, w_i * E_i(x,y)$$



Recognition process:

(1) Compute weights w_i for novel face image

(2) Find image m in face database with most similar weights, e.g.

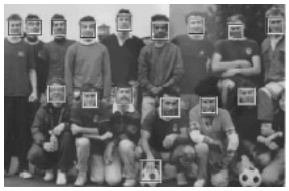
$$\min \sum_{i=1}^{k} (w_i - w_i^m)^2$$

Face detection: Viola & Jones

Multiple view-based classifiers based on simple features that best discriminate faces vs. non-faces

Most discriminating features ***learned*** from thousands of samples of face and non-face image windows

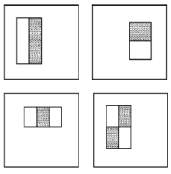
Attentional mechanism: cascade of increasingly discriminating classifiers improves performance



Viola & Jones use simple features

Use simple *rectangle features:*

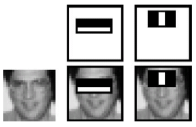
Σ I(x,y) in gray area – Σ I(x,y) in white area



within 24 x 24 image sub-windows

- initially consider 160,000 potential features per sub-window!
- features computed very efficiently

Which features best distinguish face vs. non-face?



Learn most discriminating features from thousands of samples of face and non-face image windows

Learning the best features

weak classifier using one feature:

$$h(x, f, p, \theta) = \begin{cases} 1 & \text{if } pf(x) < p\theta \\ 0 & \text{otherwise} \end{cases}$$

x = image window
f = feature
p = +1 or -1
θ = threshold

$(x_1, w_1, 1)$ ... $(x_n, w_n, 0)$

n training samples, equal weights, known classes

$$C(x) = \begin{cases} 1 & \sum_{t=1}^{T} \alpha_t h_t(x) \geq \tau \\ 0 & \text{otherwise} \end{cases}$$

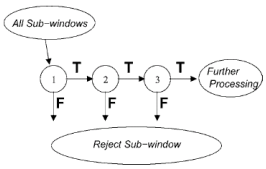
normalize weights → find next best weak classifier $\epsilon_t = \min_{f,p,\theta} \sum_i w_i\, |h(x_i, f, p, \theta) - y_i|$ → final classifier

AdaBoost

use classification errors to update weights

~ 200 features yields good results for "monolithic" classifier

"Attentional cascade" of increasingly discriminating classifiers



Early classifiers use a few highly discriminating features, low threshold

•1st classifier uses two features, removes 50% non-face windows

•later classifiers distinguish harder examples

- Increases efficiency
- Allows use of many more features

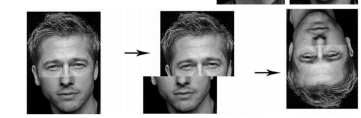
→ Cascade of 38 classifiers, using ~6000 features

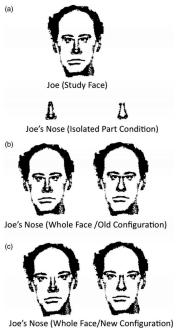
1-19

Feature based vs. holistic processing

Tanaka & Simonyi (2016), Sinha et al. (2006)

- composite face effect
- face inversion effect
- whole-part effect





Feature based vs. holistic processing

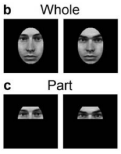
composite face effect

Aligned Misaligned

- identical top halves seen as different when aligned with different bottom halves
- when misaligned, top halves perceived as identical

whole-part effect

Study



Identification of "studied" face is significantly better in whole vs. part condition

face inversion effect

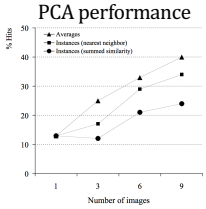
Upright

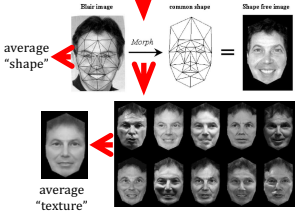
Inverted

- inversion disrupts recognition of faces more than other objects
- prosopagnosics do not show effect

The power of averages, Burton et al. (2005)

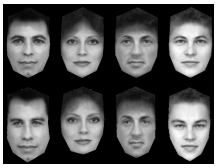
PCA performance



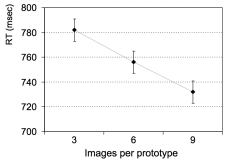


Human recognition of average faces

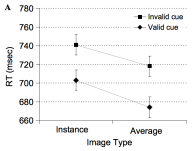
Burton et al. (2005)

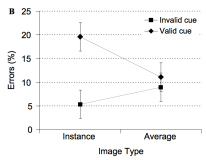


Performance: shape-free images



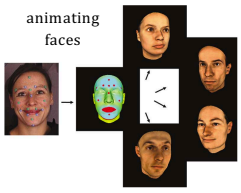
Performance: texture + shape images

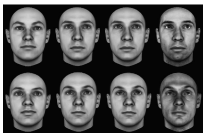


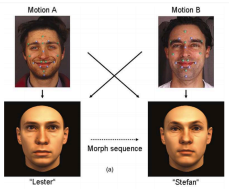


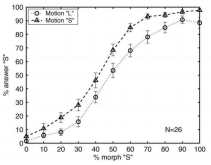
Role of facial motion, Knappmeyer et al. 2003

animating faces

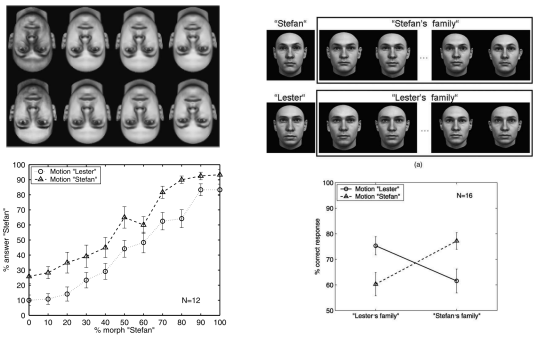




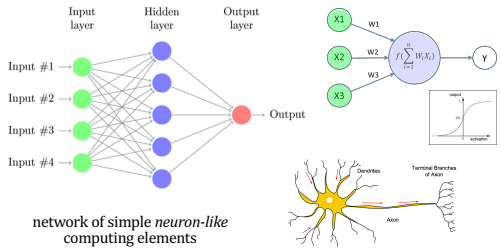




Role of facial motion, Knappmeyer et al. 2003

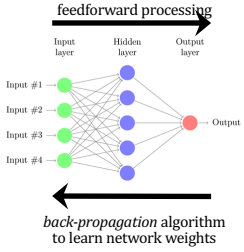


Artificial Neural Networks



network of simple *neuron-like* computing elements

Learning to Recognize Input Patterns



back-propagation algorithm to learn network weights

network weights can be ***learned*** from training examples (mapping from inputs to correct outputs)

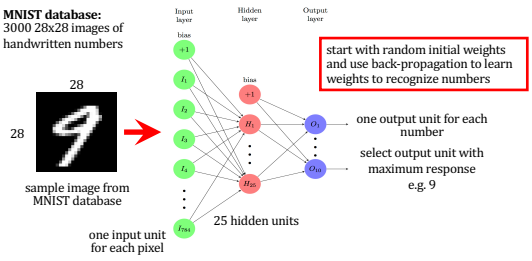
back-propagation:
iterative algorithm that progressively reduces error between computed and desired output until performance is satisfactory

on each iteration:

- compute output of current network and assess performance
- compute weight adjustments from hidden to output layer that can reduce output errors
- compute weight adjustments from input to hidden units that can enhance hidden layer
- change network weights, using rate parameter

Example: Learning Handwritten Digits

MNIST database:
3000 28x28 images of handwritten numbers



start with random initial weights and use back-propagation to learn weights to recognize numbers

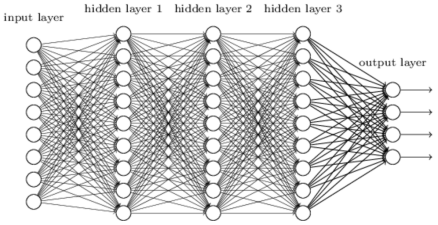
Results: Learning Handwritten Digits

correct examples | *wrong* examples | confusion matrix

overall classification accuracy: 87.1%

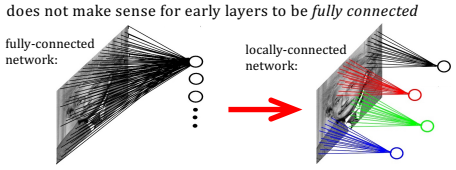
"Deep" neural networks

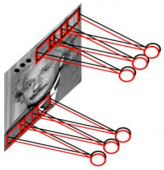
- early work extended simple neural networks to have multiple, highly-connected hidden layers
- ***if*** such networks could be trained, they would be much more powerful than "shallow" neural nets
- ***but*** multi-layer networks are much harder to train!!



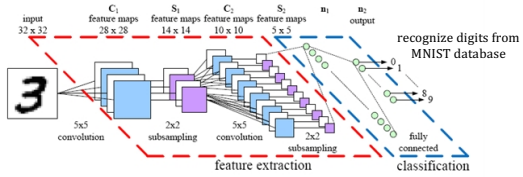
Deep convolutional networks

- for image processing, if network inputs are individual pixels, it does not make sense for early layers to be *fully connected*



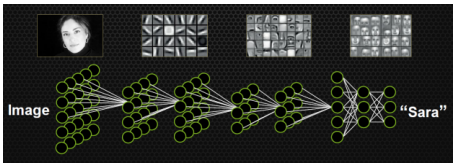


- in *convolutional networks*, early layers perform a convolution of their inputs
- multiple convolution operators (e.g. red & black)
- weights in convolution operators are *learned*
- convolution layers followed by *pooling layers*
  - *max-pooling:* max activation in 2x2 neighborhoods



recognize digits from MNIST database

LeCun, Bottou, Bengio, Haffner (1998)



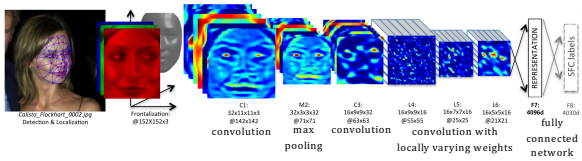
unsupervised learning of hierarchical representations

Lee, Grosse, Ranganath, Ng (2009)

Facebook's DeepFace system

Taigman et al., 2014

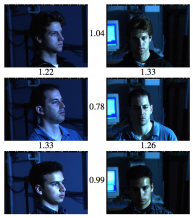
- detect face
- 2D align face in crop window using 6 fiducial points
- align to 3D shape model using 67 fiducial points
- use 3D model + image to generate frontal view



Google's FaceNet system

Schroff et al., 2015

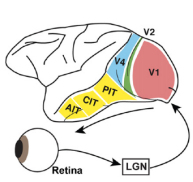
FaceNet also uses a deep convolutional network

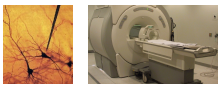


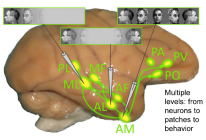
threshold = 1.1 classifies pairs correctly (smaller value means more similar)

- learns mapping from images to a space where distance between images captures similarity
- training data: triplets of face thumbnails two same ID, one different ID
- learning process: minimize distance between anchor & positive images (same ID), maximize distance between anchor & negative images

Neural Processing in the Ventral Visual Pathway







HMAX model