# Fast and Furious Face Recognition

## Efficient metric learning for video stream data

Yoni Wexler

ORCAM

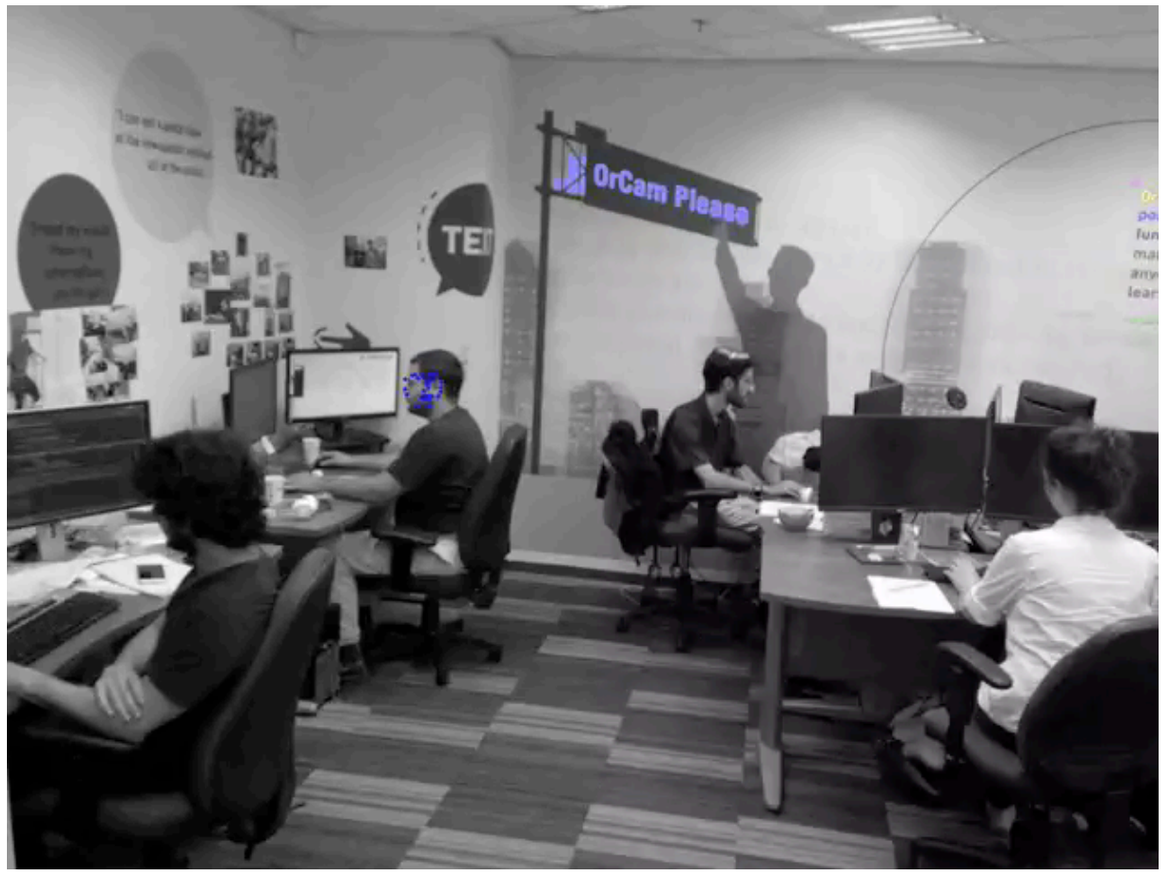
ORCAM

See for yourself

Innovative Wearable AI

# Real-time AI capabilities

- Text detection and reading (OCR)
- Language detection and modeling
- Face recognition
- Gesture recognition
- Voice recognition
- Product recognition
- Image classification



# Proper Metric Learning for Face Recognition

NIPS 2016

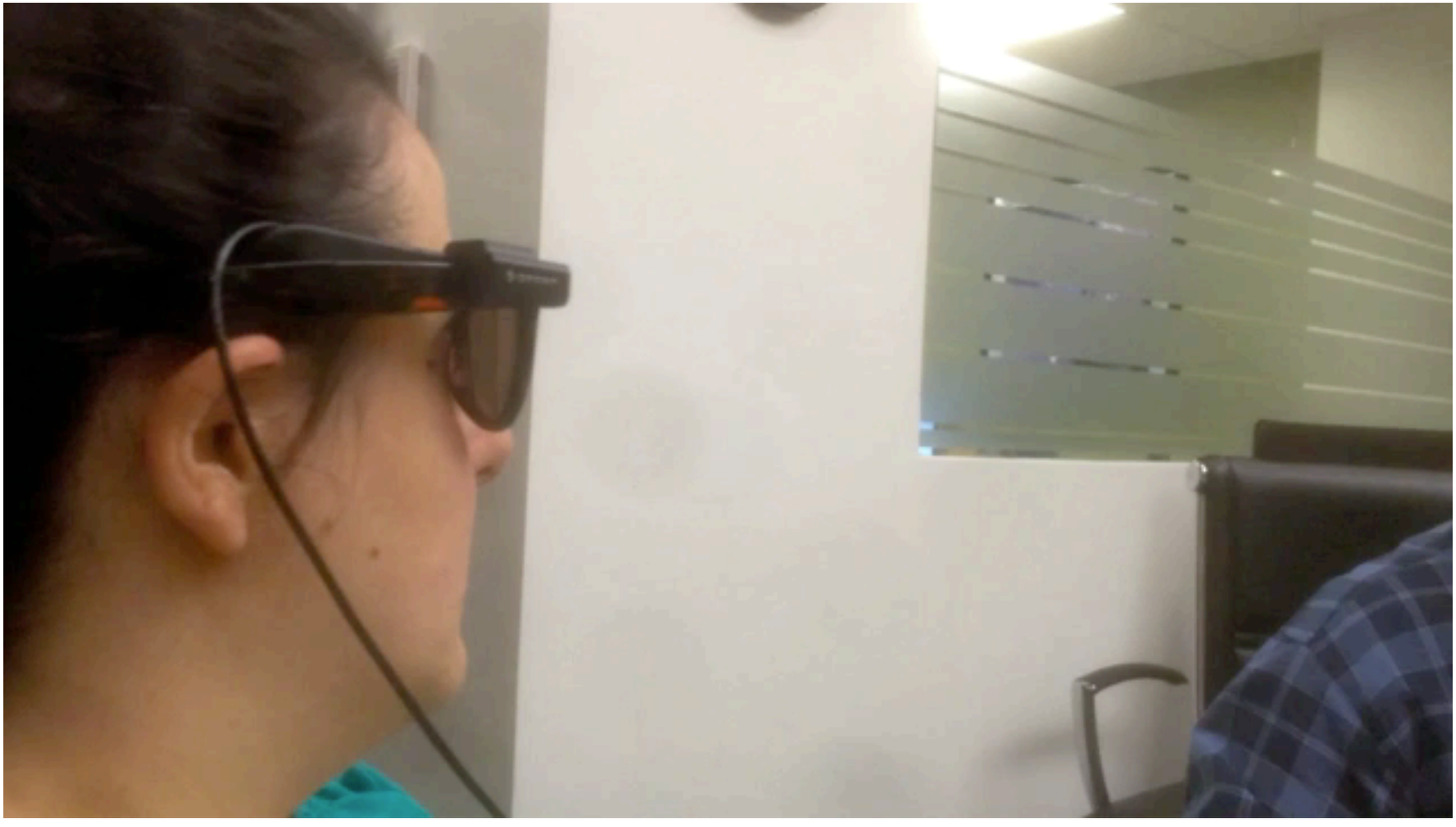
Yoni Wexler, Oren Tadmor, Tal Rosenwein,

Shai Shalev-Shwartz, Amnon Shashua

ORCAM

ORCAM

# Useful Face Recognition



ORCAM

# The Face Recognition Task

Angelina Jolie

Tom Hanks

“Don’t know”

ORCAM

# The Face Recognition Task

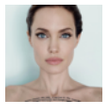
- Appearance
- Lighting
- Occlusion
- Aging



ORCAM

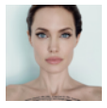
# Face Recognition Approaches

H(  ,  ) -> True

H(  ,  ) -> False

d(  ,  ) < d(  ,  )

H(  ) $\rightarrow [k]$

 $\rightarrow \vec{x}$ $||x - y||$

ORCAM

# Low-dimensional embedding

- 1987: *Low-dimensional procedure for the characterization of human faces*
  Sirovich and Kirby, “Eigenpictures”

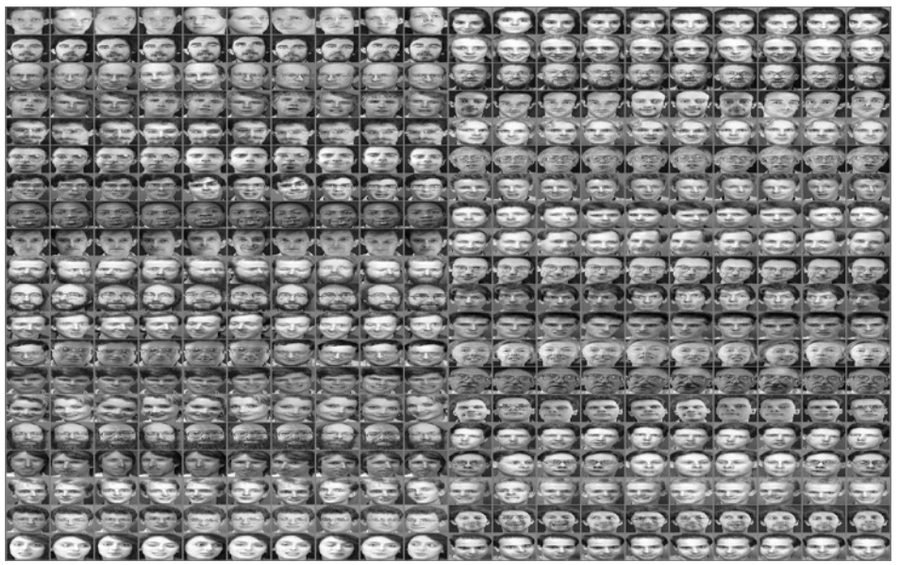


Fig. 1. Average face based on an ensemble of 115 faces. In this, as in the other plates, we have refrained from filtering out the high frequencies produced by the digitization. A pleasanter picture can be had by the usual trick of squinting or otherwise blurring the picture.

# Low-dimensional embedding

- 1987: *Low-dimensional procedure for the characterization of human faces*
  Sirovich and Kirby, “Eigenpictures”

Fig. 1. Average face based on an ensemble of 115 faces. In this, as in the other plates, we have refrained from filtering out the high frequencies produced by the digitization. A pleasanter picture can be had by the usual trick of squinting or otherwise blurring the picture.

- 1991: *Face Recognition Using Eigenfaces*
  Turk and Pentland

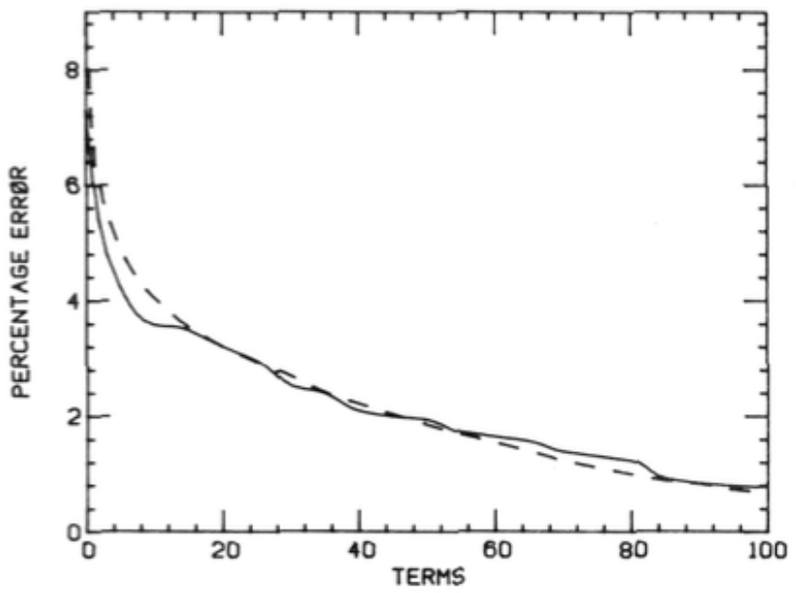
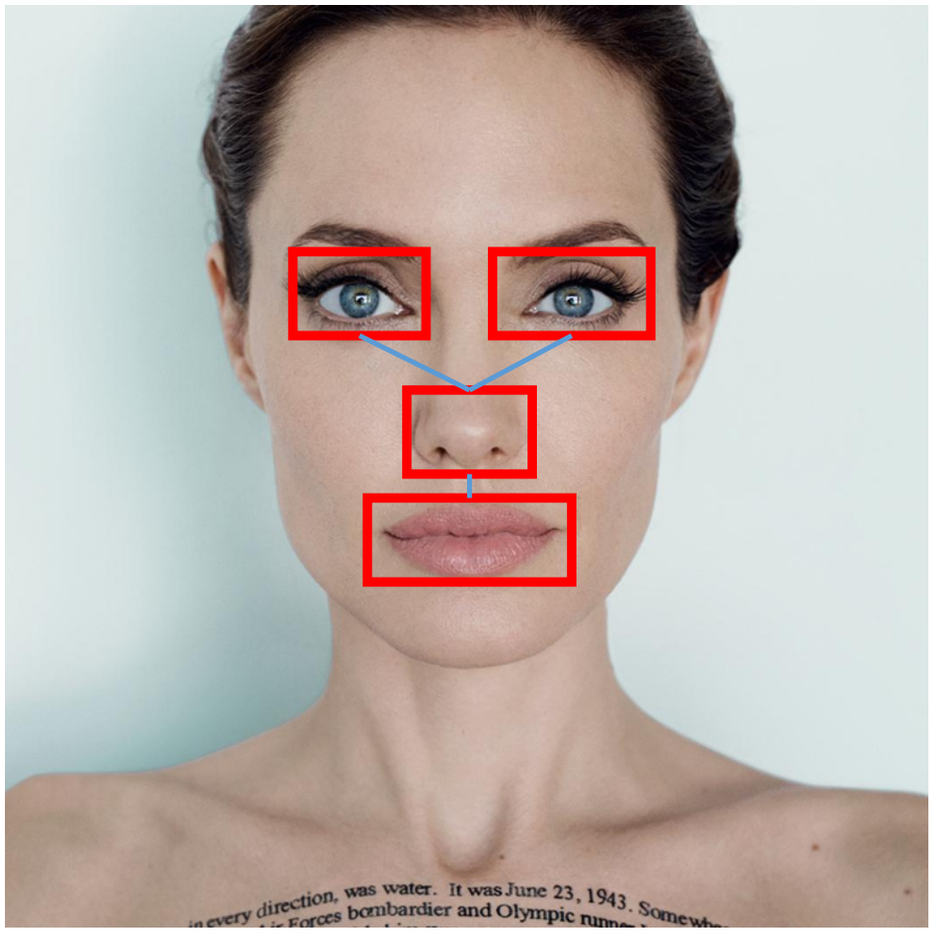


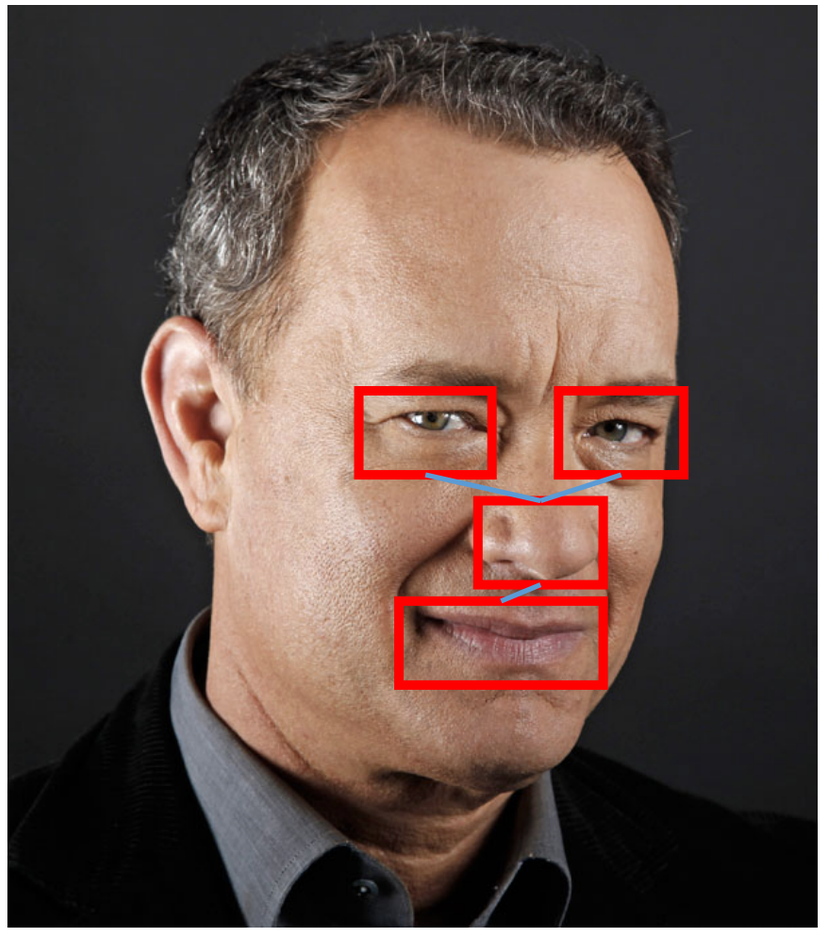

Fig. 6. Percent error versus number of eigenpictures used in the approximation. Solid curve is for picture shown in Fig. 2 (see also Fig. 5). Dashed curve is average over 10 different sample faces.

ORCAM

# Local features

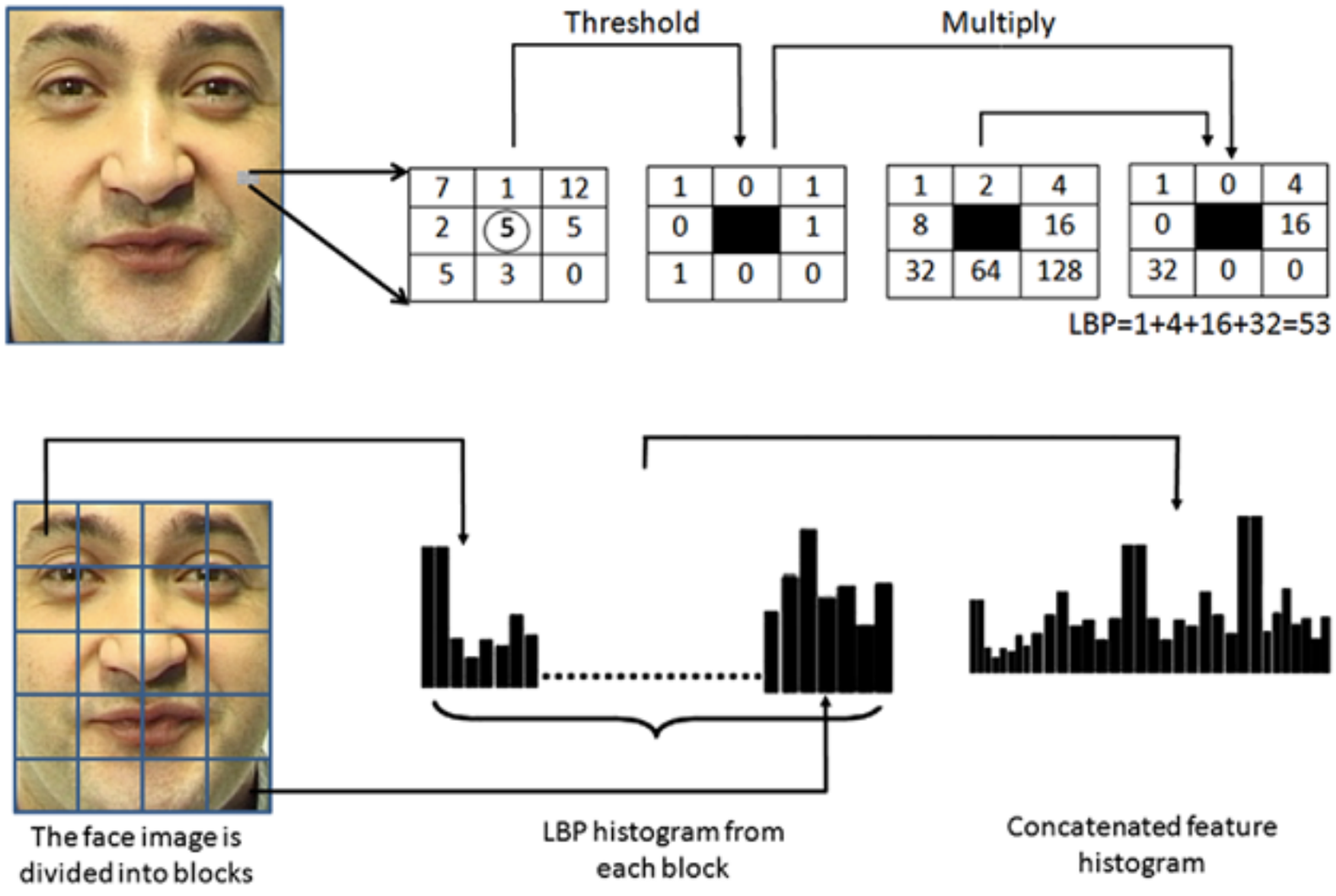
- Be robust to translation



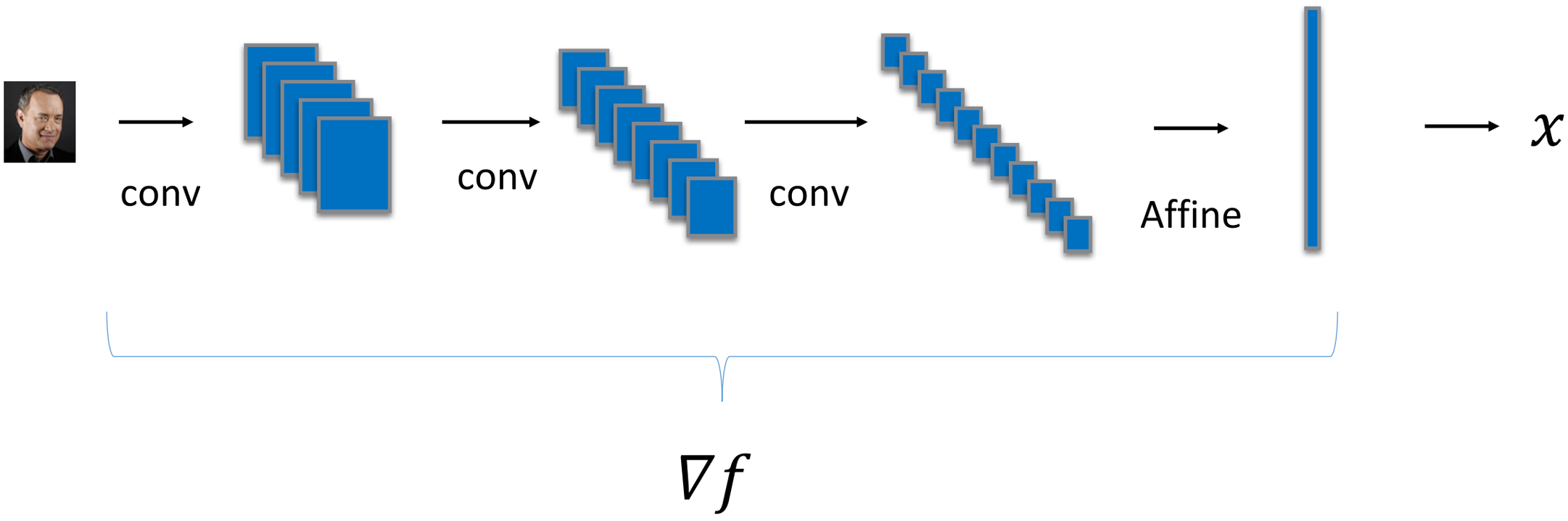



# Use Alternative Image Measurements

- LBP: Be robust to illumination



# Deep Learning



# DeepFace (Facebook)

- Taigman et al. 2014
- 3D alignment,
- 120M parameters
- 4,030 people, 1,000 photos each
- On 2.2Ghz Intel CPU: 50ms alignment, 330ms total
- LFW result: 97% with single net
  97.35% with ensemble

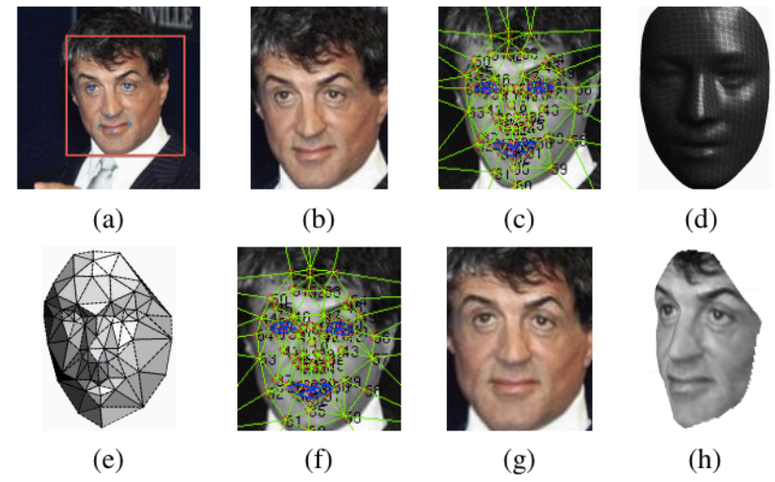



Figure 1. **Alignment pipeline.** (a) The detected face, with 6 initial fiducial points. (b) The induced 2D-aligned crop. (c) 67 fiducial points on the 2D-aligned crop with their corresponding Delaunay triangulation, we added triangles on the contour to avoid discontinuities. (d) The reference 3D shape transformed to the 2D-aligned crop image-plane. (e) Triangle visibility w.r.t. to the fitted 3D-2D camera; darker triangles are less visible. (f) The 67 fiducial points induced by the 3D model that are used to direct the piece-wise affine warpping. (g) The final frontalized crop. (h) A new view generated by the 3D model (not used in this paper).

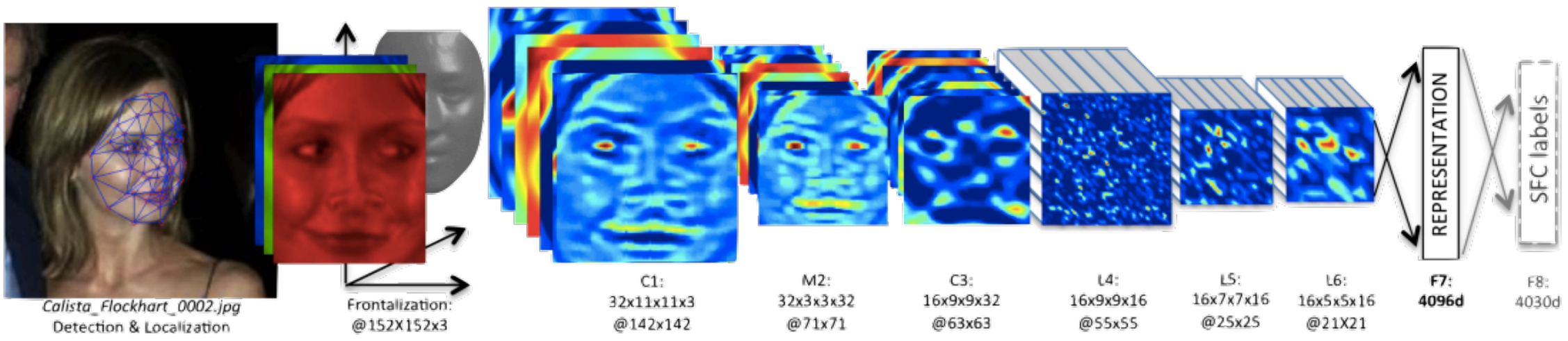



Figure 2. **Outline of the *DeepFace* architecture**. A front-end of a single convolution-pooling-convolution filtering on the rectified input, followed by three locally-connected layers and two fully-connected layers. Colors illustrate feature maps produced at each layer. The net includes more than 120 million parameters, where more than 95% come from the local and fully connected layers.

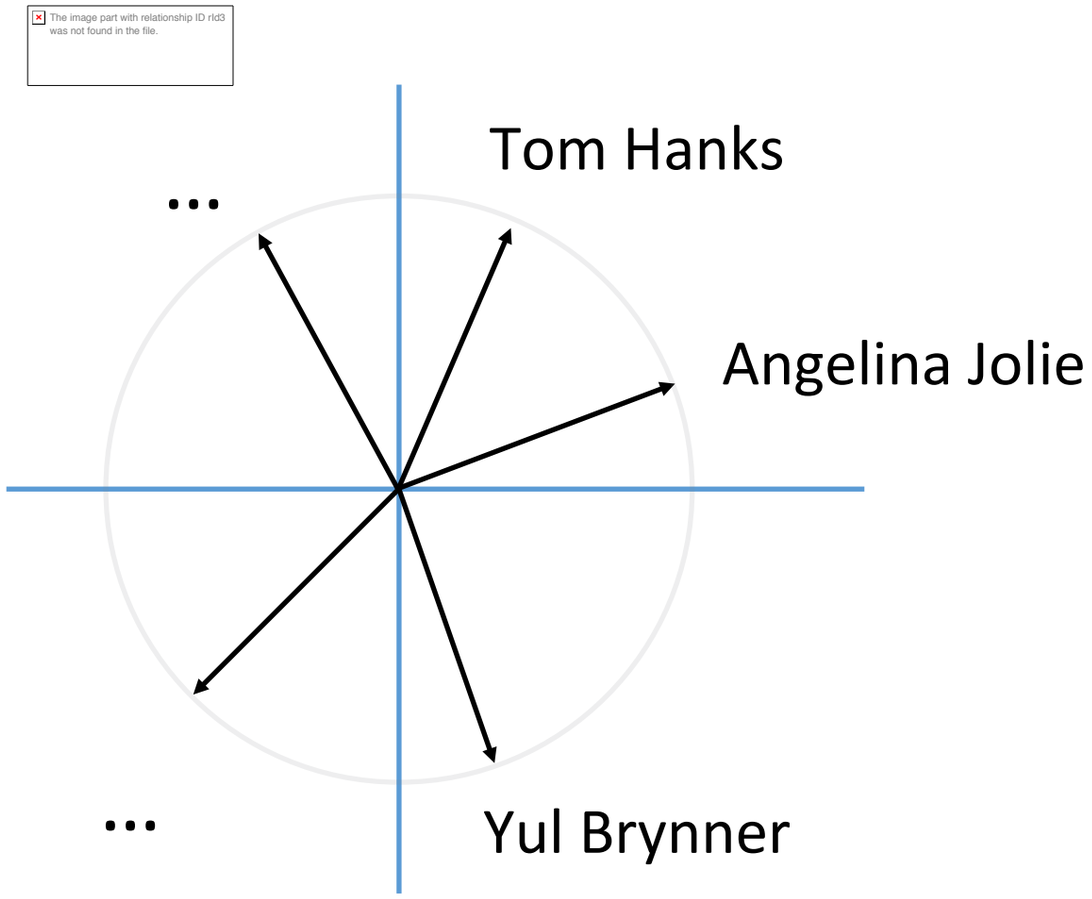
ORCAM

# DeepFace (Facebook)

- Classification: $x^\top y$
  $|x - y|_w$

- 4030 dimensions

- “75% of the representation is zero”



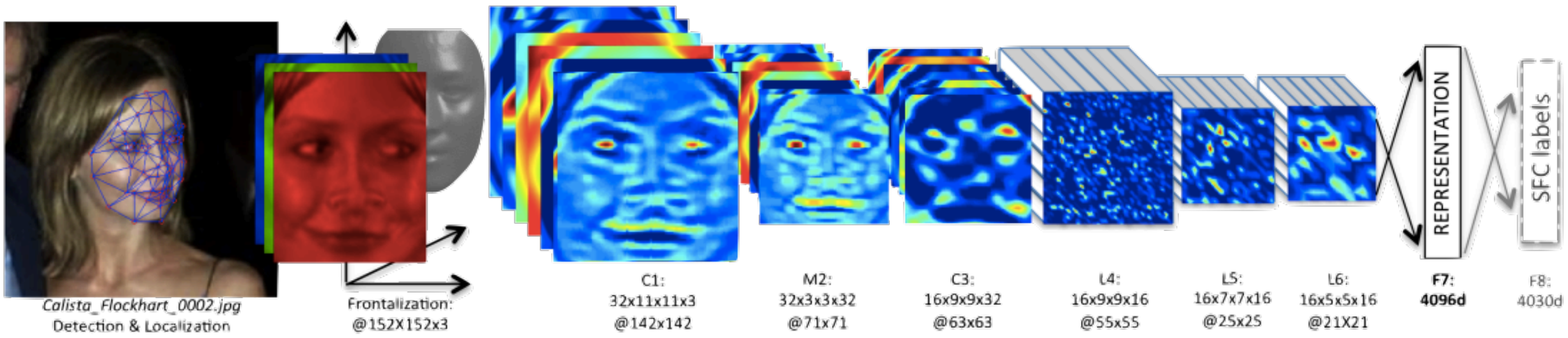
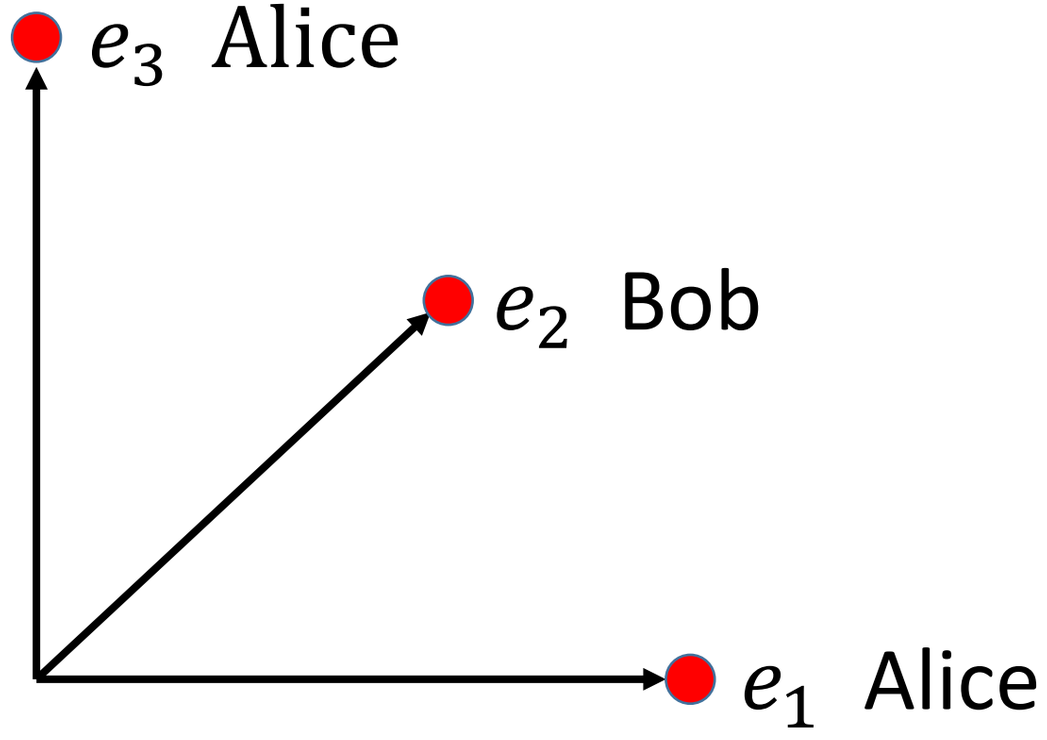




Figure 2. **Outline of the *DeepFace* architecture**. A front-end of a single convolution-pooling-convolution filtering on the rectified input, followed by three locally-connected layers and two fully-connected layers. Colors illustrate feature maps produced at each layer. The net includes more than 120 million parameters, where more than 95% come from the local and fully connected layers.

ORCAM

# Metric learning is harder than multi-class



# FaceNet (Google)

- Schroff, 2015
- No alignment
- 8M people, 260M faces total
- 140M & 7.5M parameters. 1.6B FLOPs
- "…and trained on a CPU cluster for 1,000 to 2,000 hours."
- LFW result: 98.87% no alignment
  99.63% with 2D alignment

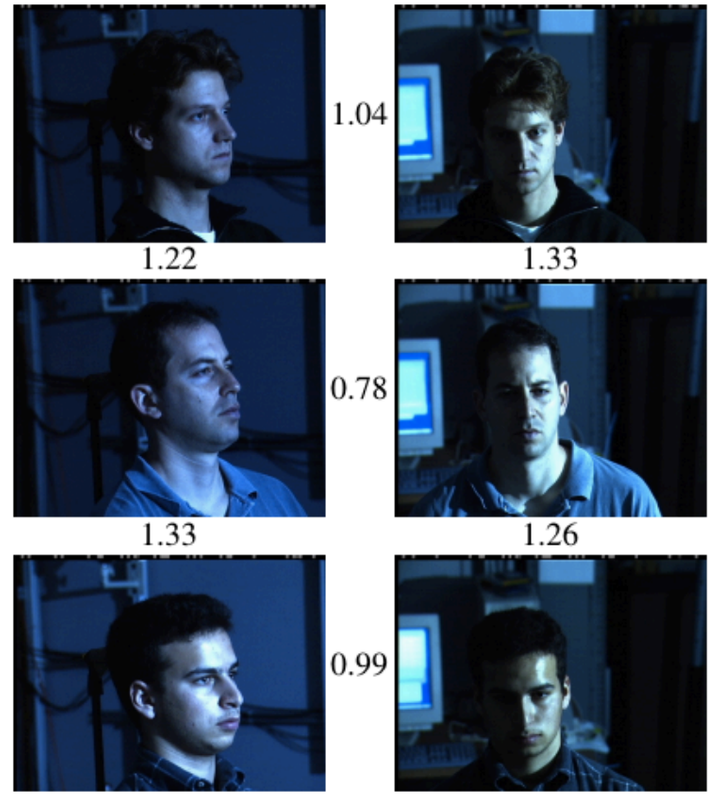



Figure 1. **Illumination and Pose invariance.** Pose and illumination have been a long standing problem in face recognition. This figure shows the output distances of FaceNet between pairs of faces of the same and a different person in different pose and illumination combinations. A distance of 0.0 means the faces are identical, 4.0 corresponds to the opposite spectrum, two different identities. You can see that a threshold of 1.1 would classify every pair correctly.

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

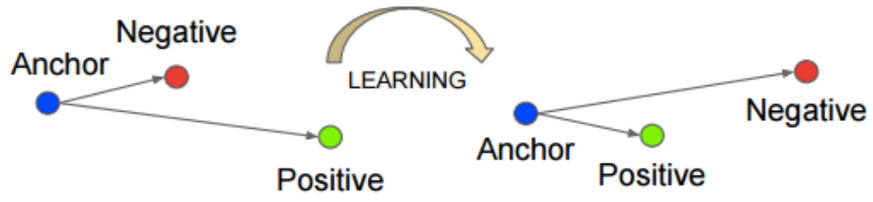
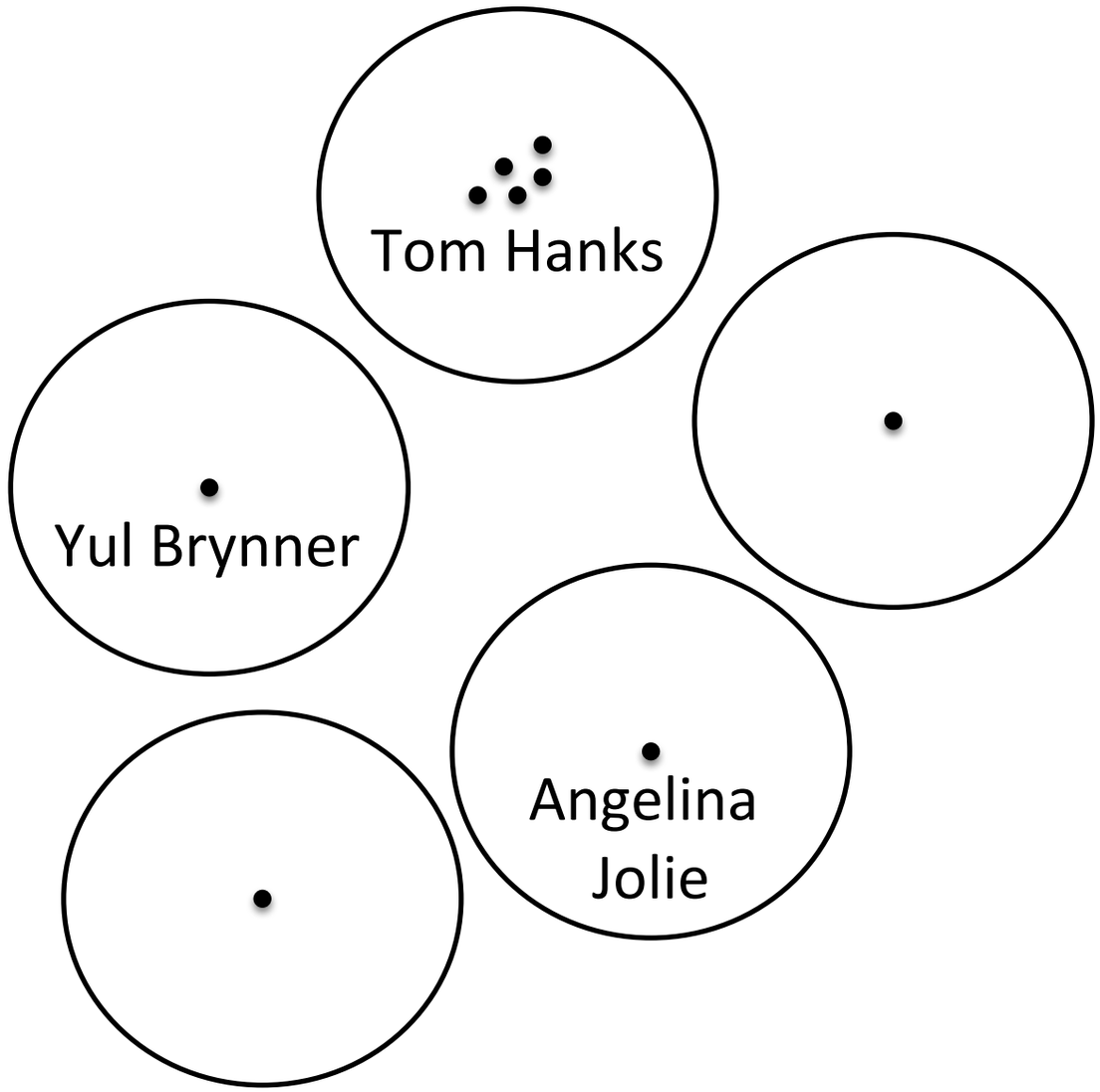



Figure 3. The **Triplet Loss** minimizes the distance between an *anchor* and a *positive*, both of which have the same identity, and maximizes the distance between the *anchor* and a *negative* of a different identity.

ORCAM

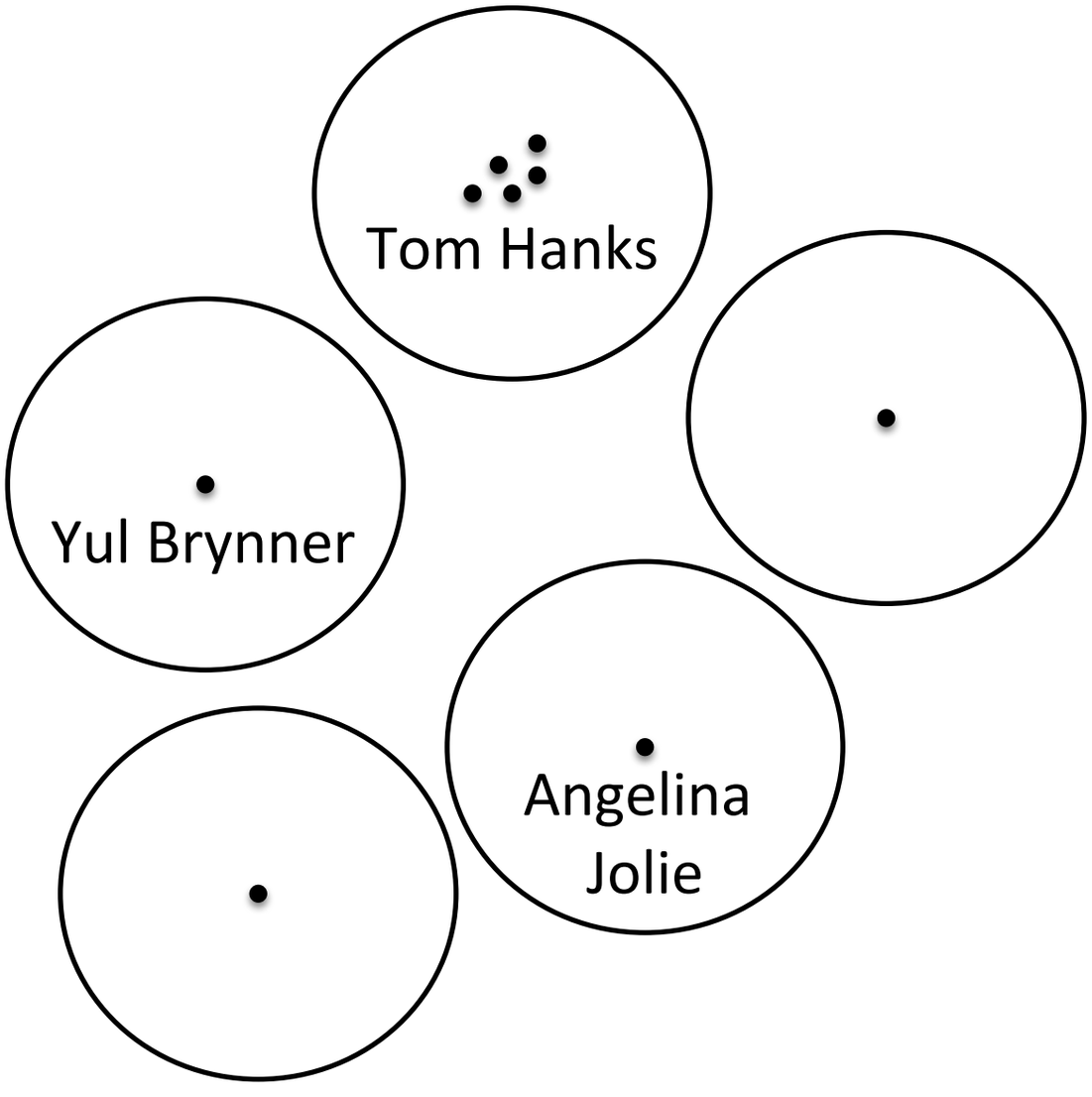
# Our Goal

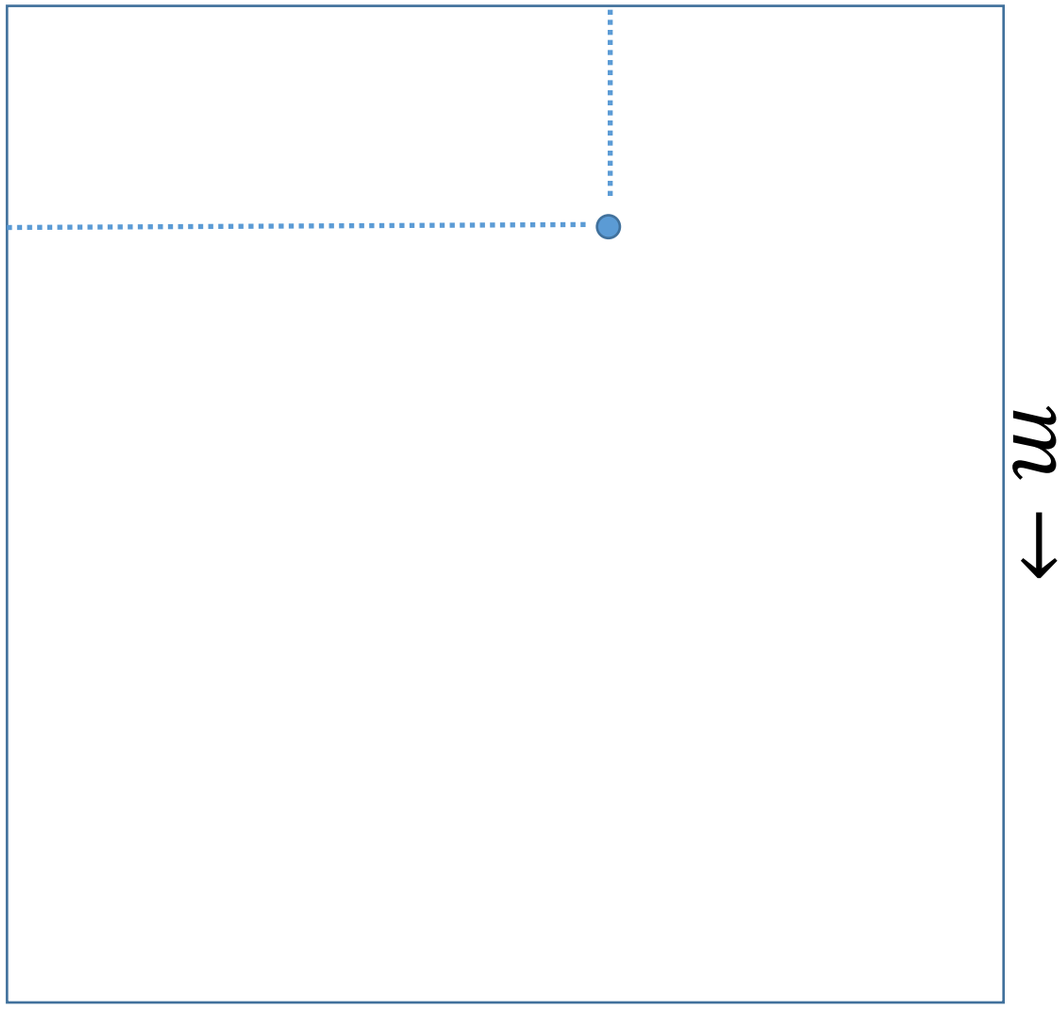



$$\text{Loss}(x_i, y_i) = \Big( 1 + y_{ij}\big(|x_i - x_j| - \theta\big)\Big)_+$$

ORCAM

# Optimization

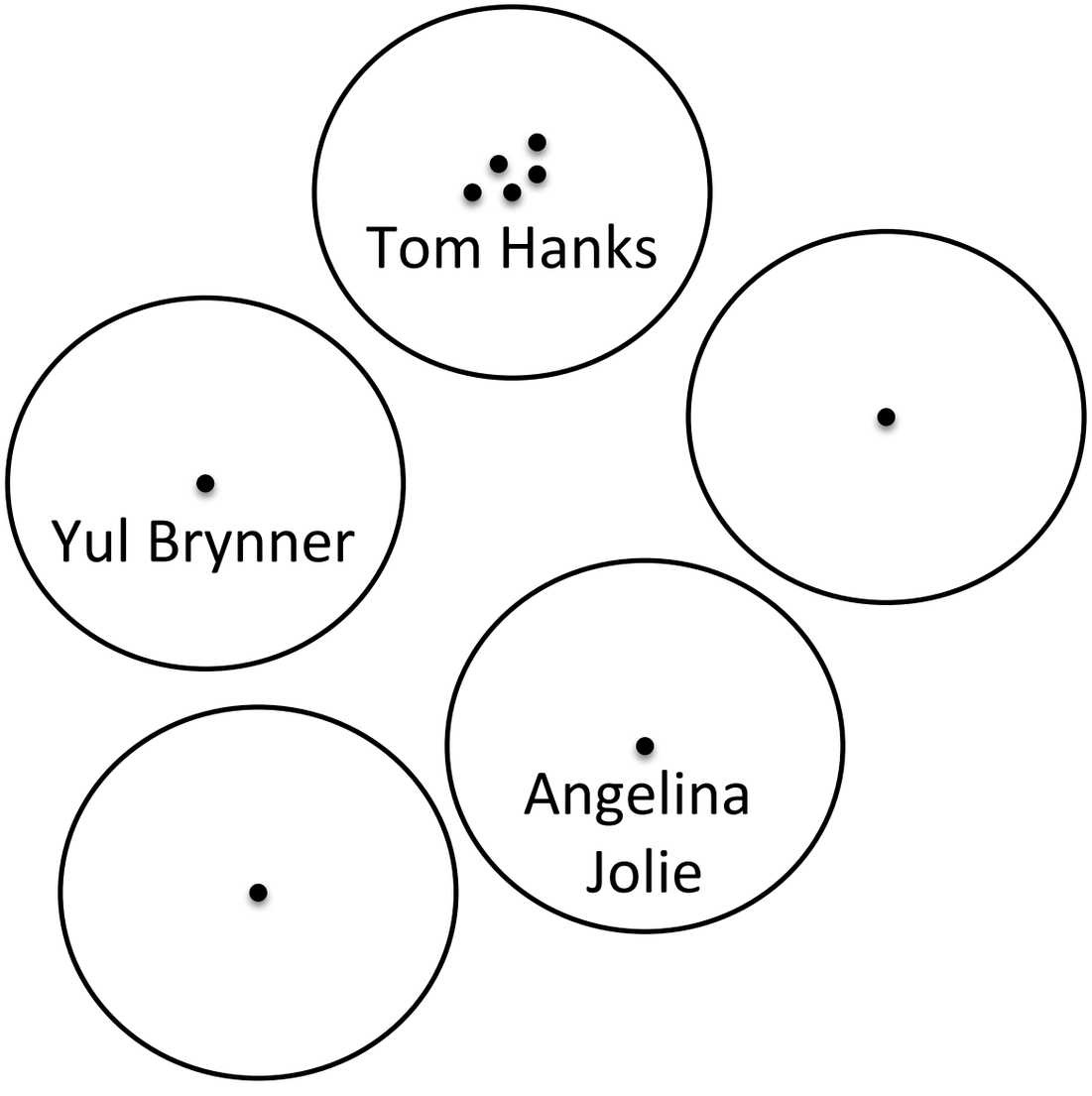


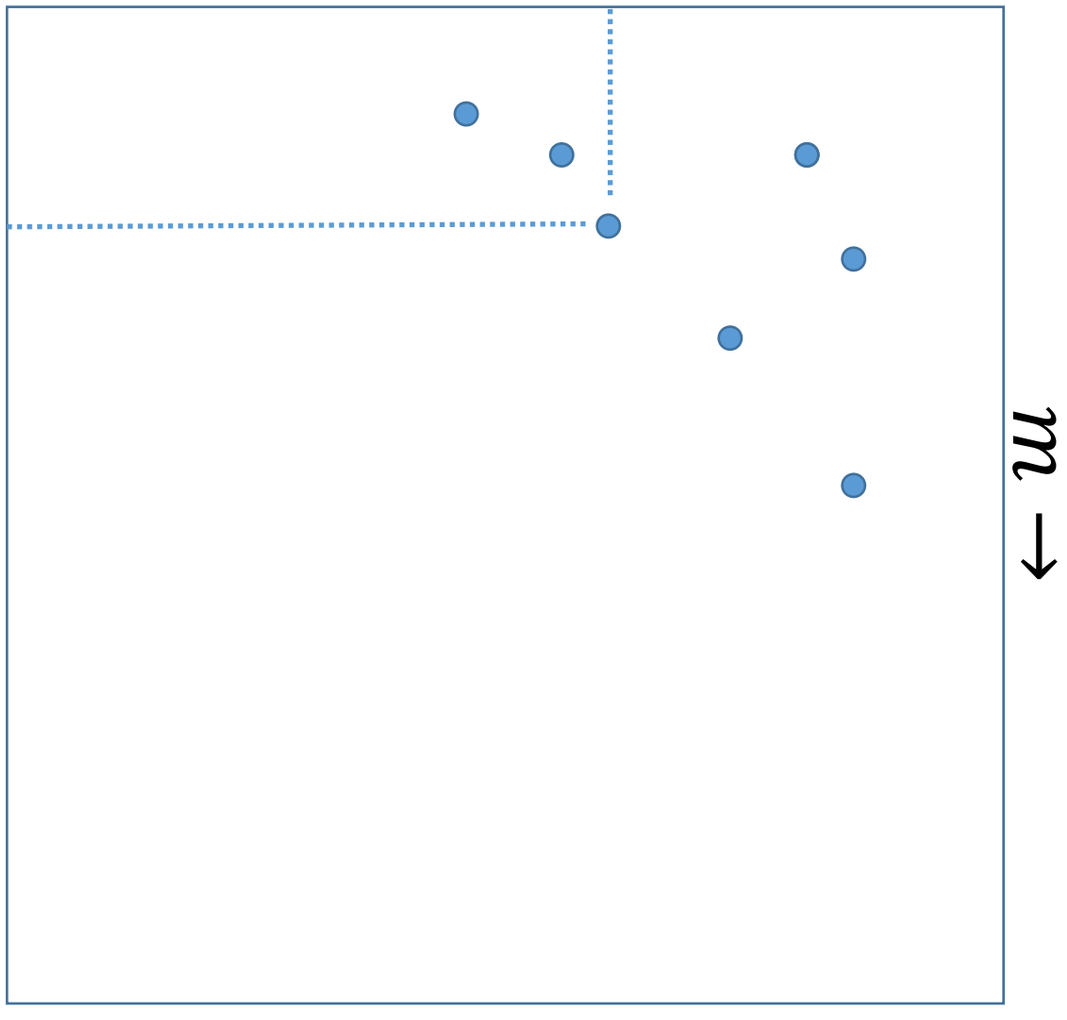




$$\mathrm{Loss}(x_i, y_i) = \Big( 1 - y_{ij}\big(\theta - |x_i - x_j|\big)\Big)_+$$

ORCAM

# Optimization – mini-batch

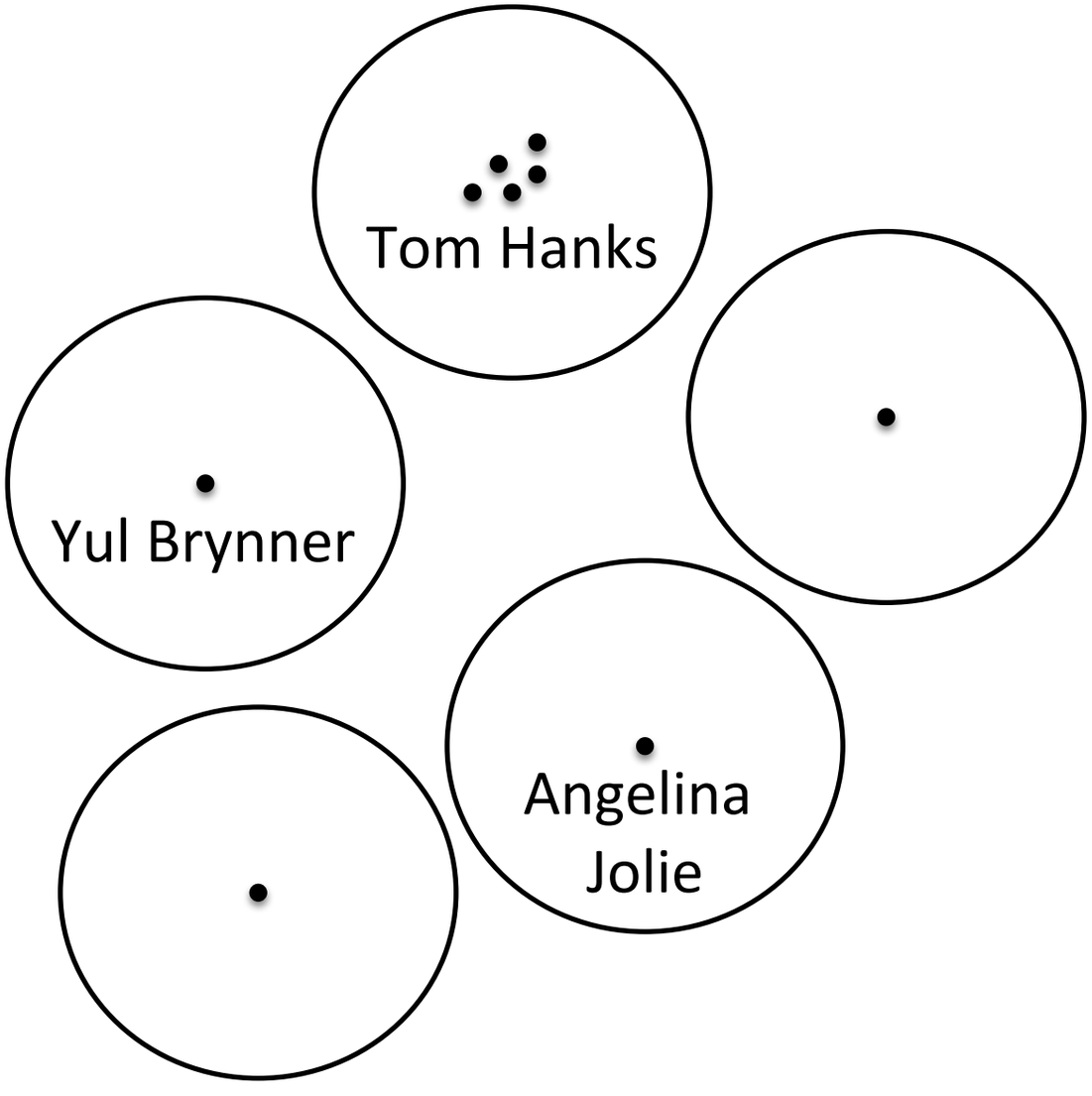


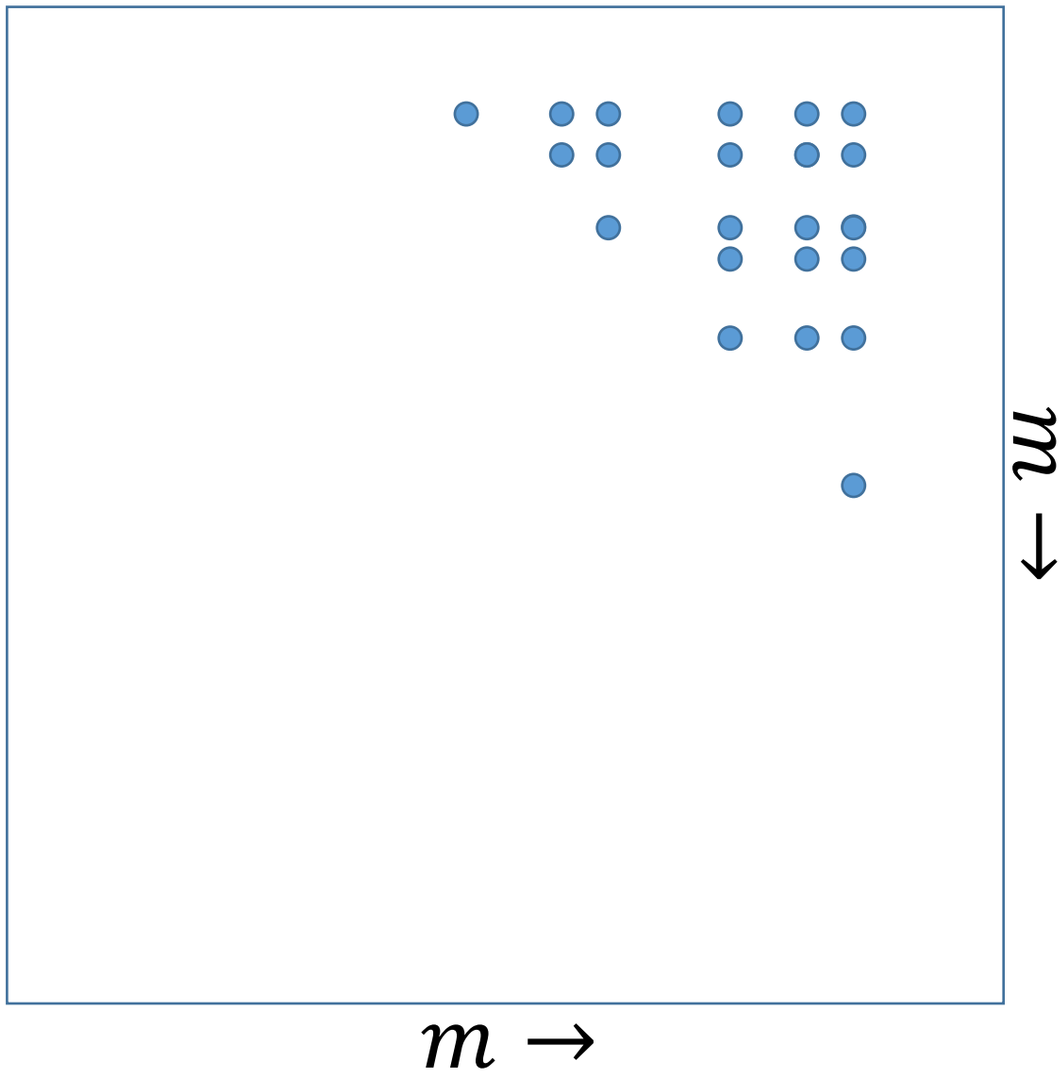




$$\text{Loss}\left(i_1, \dots i_{\frac{k}{2}}, j_1, \dots, j_{\frac{k}{2}}\right) = \sum_l \left(1 - y_{i_l j_l}\left(\theta - \left|x_{i_l} - x_{j_l}\right|\right)\right)_+$$

ORCAM

# Optimization – mini-batch

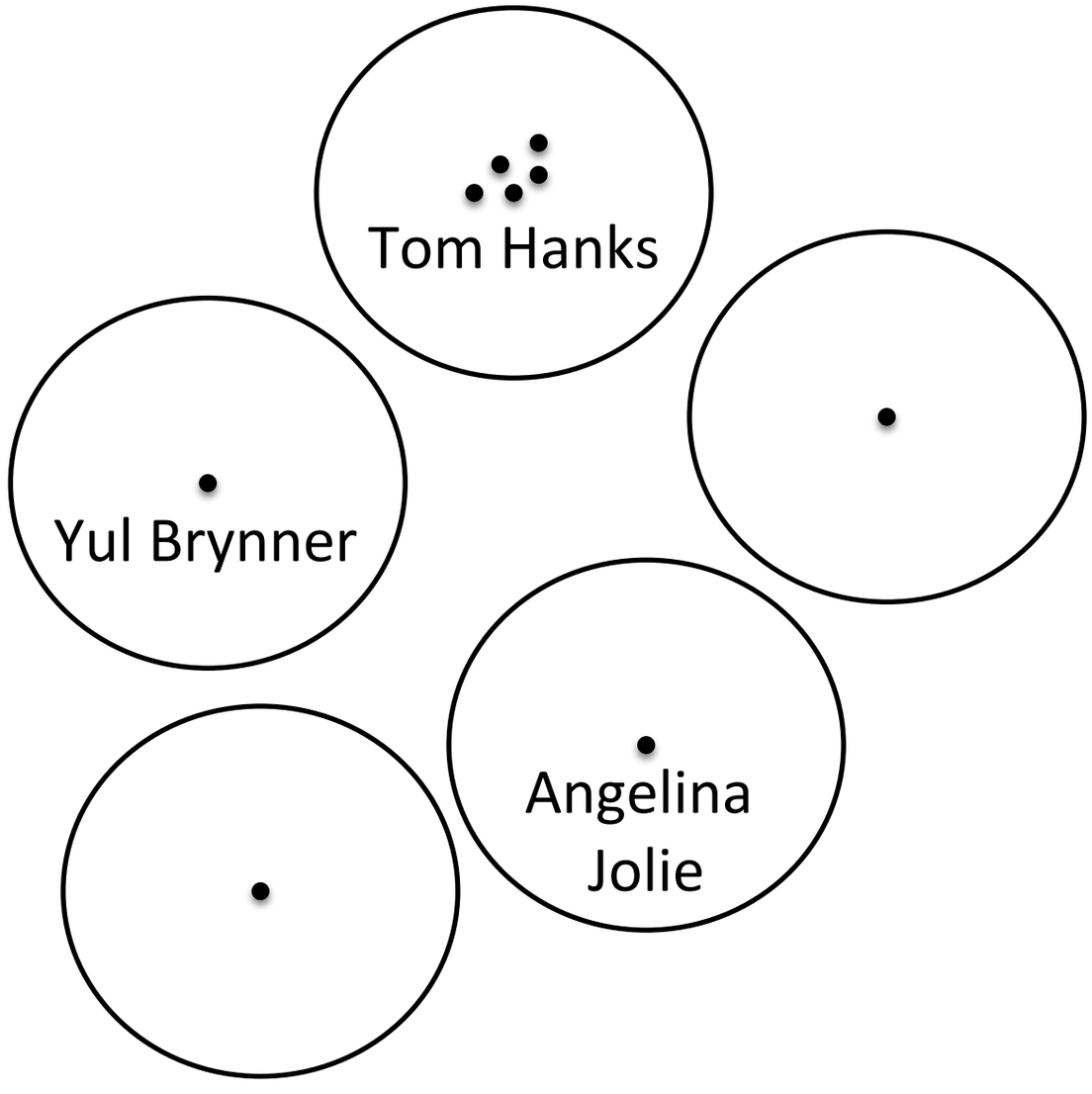


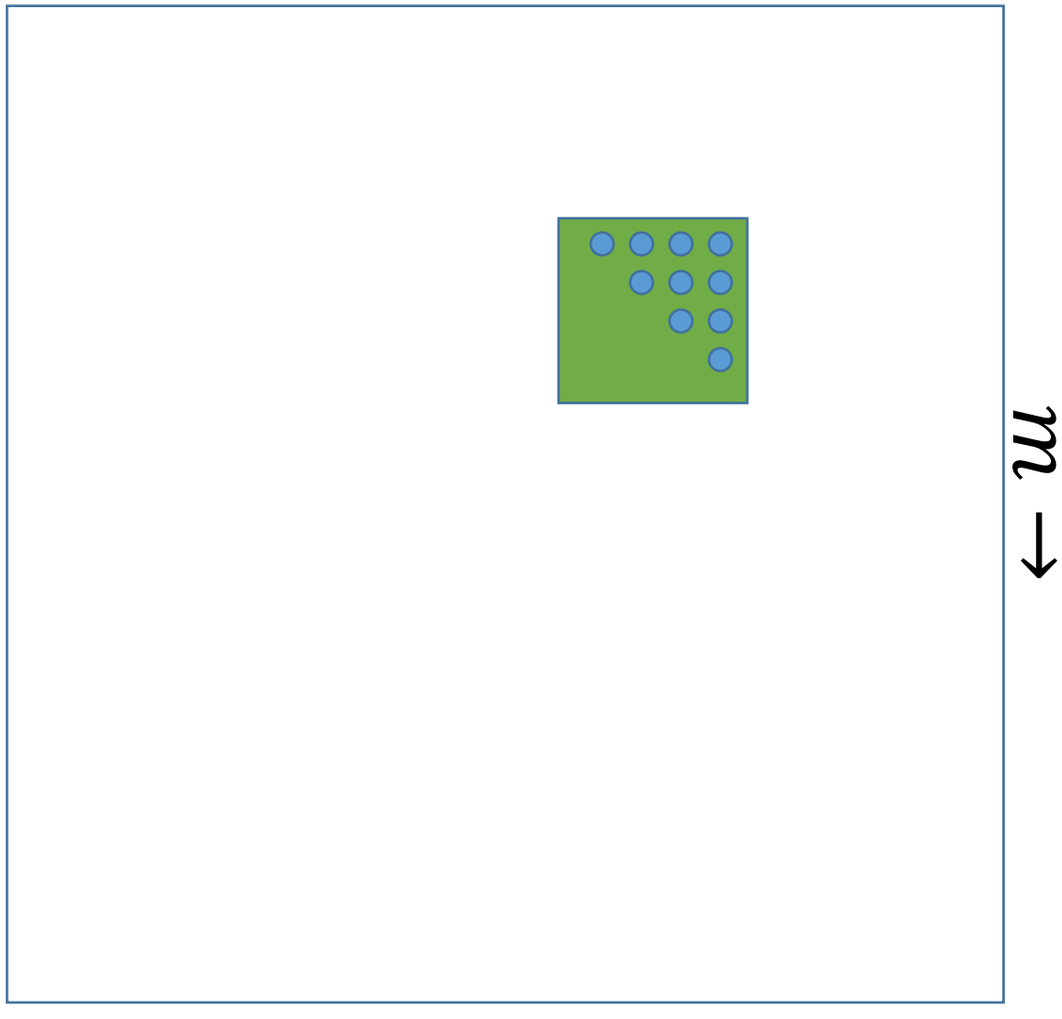




$$\text{Loss}(x_1, \ldots, x_k) = \sum_{i>j} \left(1 - y_{ij}\left(\theta - |x_i - x_j|\right)\right)_+$$

ORCAM

# Optimization – Multi-batch

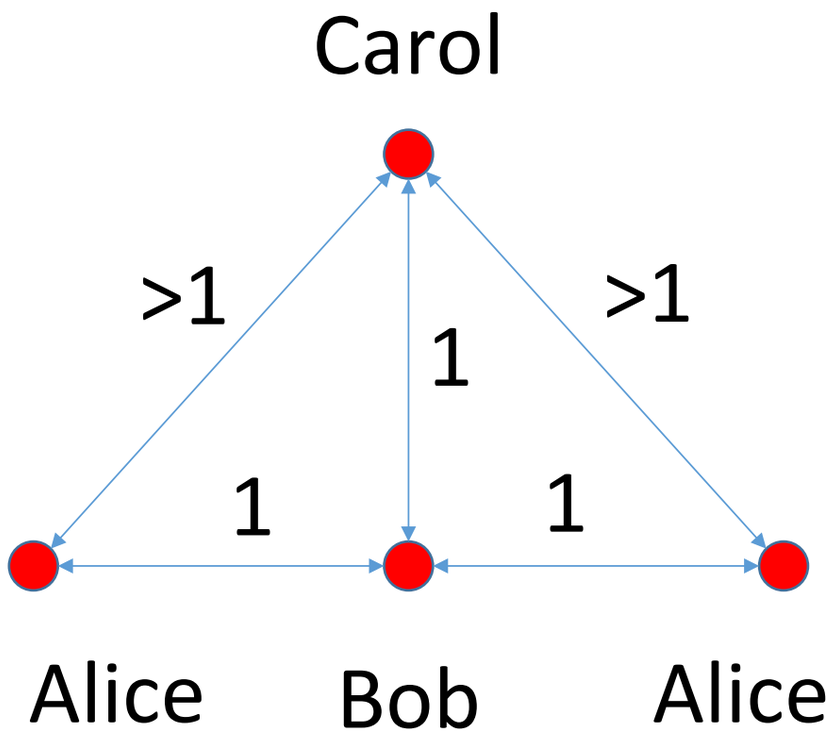



$$\text{Loss}(x_1, \ldots, x_k) = \sum_{i>j} \left(1 - y_{ij}\left(\theta - |x_i - x_j|\right)\right)_+$$

ORCAM

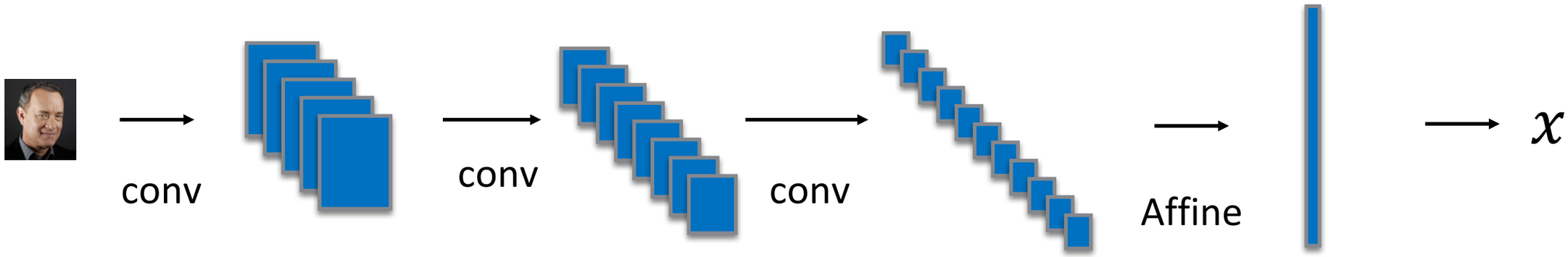
# Difficult update step




ORCAM

# Baseline Method

- Sample $\frac{k}{2}$ image pairs
- For each pair, compute $x, x'$
- Compute the loss and gradient
- Can do better?

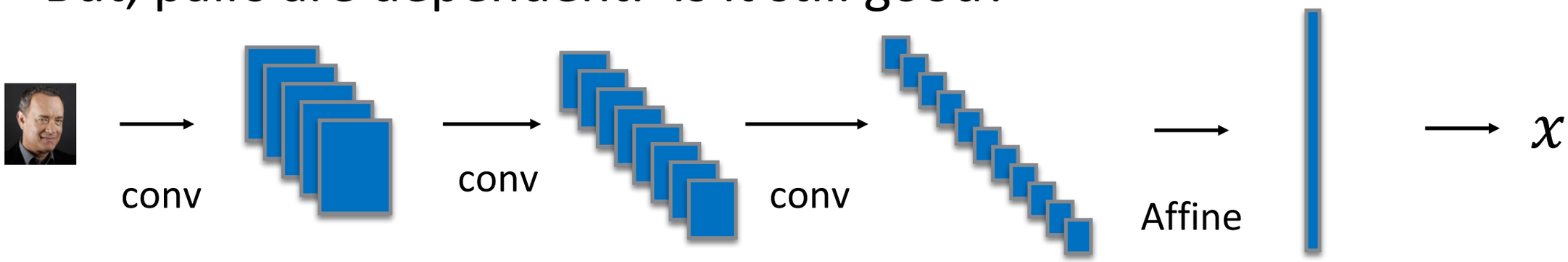



$$\text{Loss}\left(i_1, \ldots i_{\frac{k}{2}}, j_1, \ldots, j_{\frac{k}{2}}\right) = \sum_l \left(1 - y_{i_l j_l}\left(\theta - \left|x_{i_l} - x_{j_l}\right|\right)\right)_+$$

ORCAM

# Multi-batch

- The expensive part is calculating the signature
- At negligible cost, compute all pairs:
  - $\frac{k}{2} \rightarrow \frac{k^2-k}{2}$
  - $128 \rightarrow 32{,}640$ ($255x$ more pairs)
- But, pairs are dependent. Is it still good?

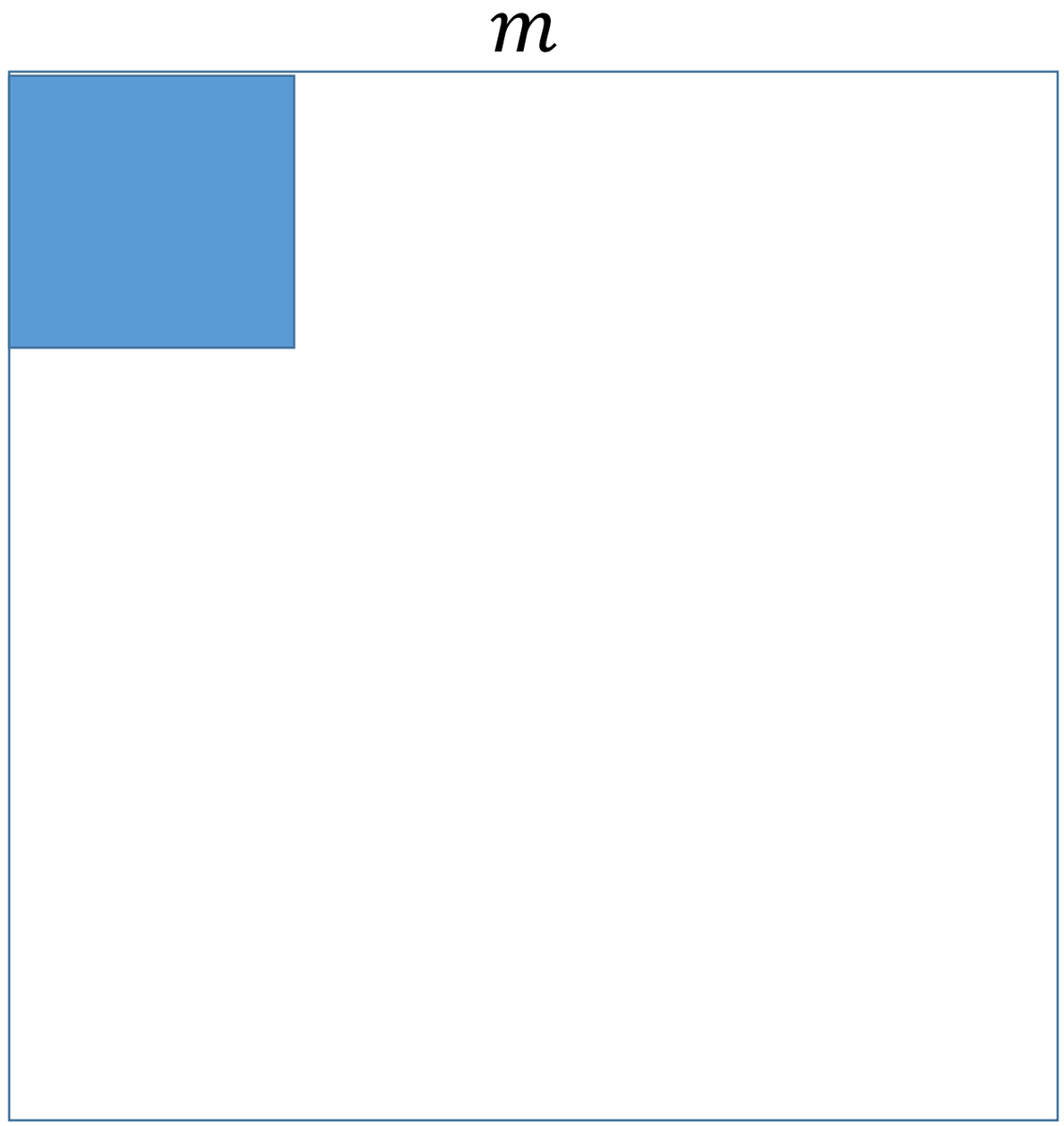



$$\text{Loss}(x_1, \ldots, x_k) = \sum_{i>j} \left(1 - y_{ij}\left(\theta - |x_i - x_j|\right)\right)_+$$

ORCAM

# Multibatch is Unbiased

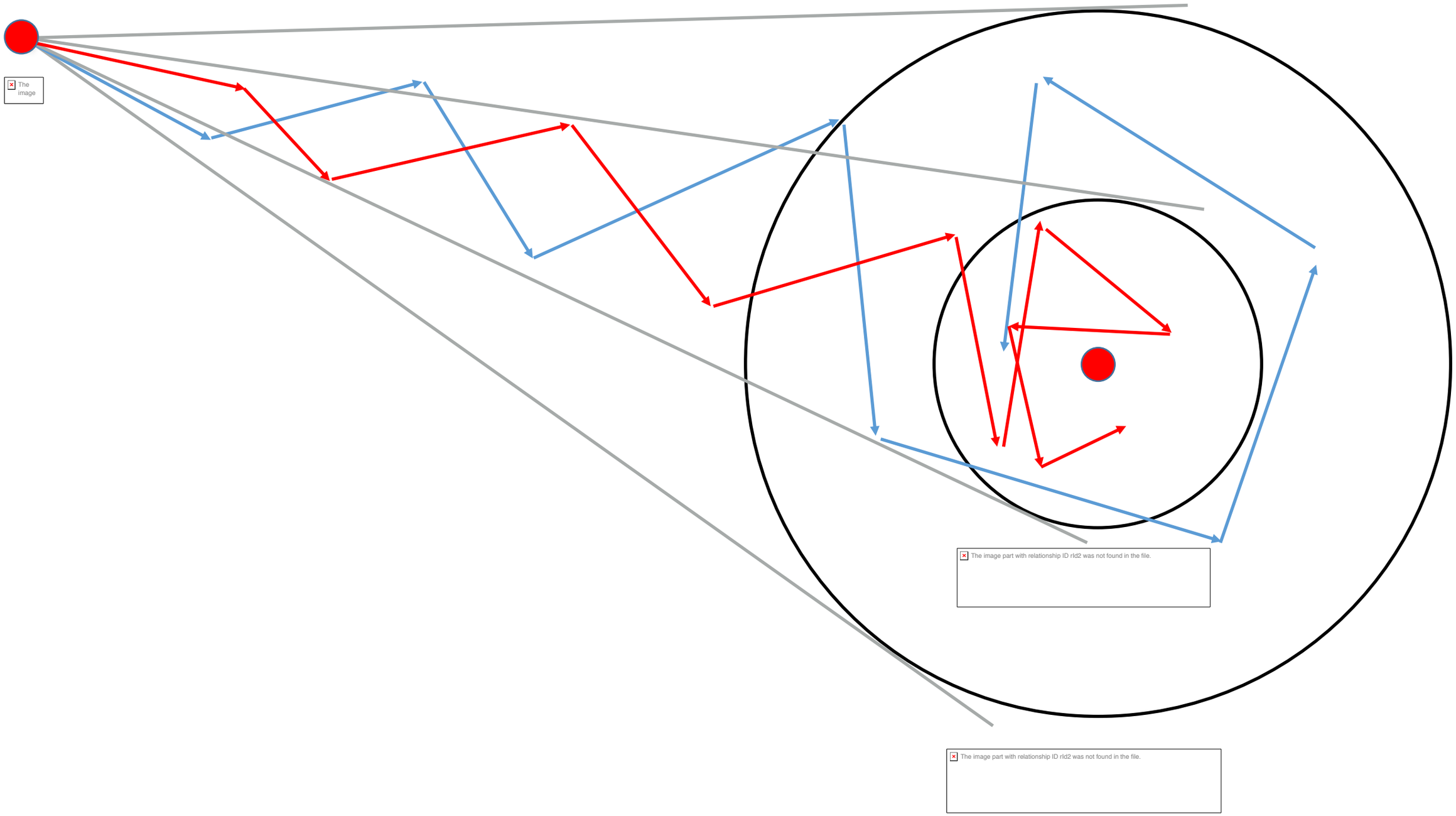
- Because of symmetry



# Multibatch reduces variance

- Claim 1: The variance of selecting $\frac{k}{2}$ pairs is $O(\frac{1}{k})$
- Claim 2: The variance of selecting all $k^2 - k$ pairs is also $O(\frac{1}{k})$
- Claim 3: In “typical cases”, the variance of selecting all $k^2 - k$ pairs is $O(\frac{1}{k^2})$

# Why low variance matters?




ORCAM

# Alignment network

| Input Size | Type | FLOPs | Parameters |
|---|---|---|---|
| 50x50x3 | Convolution with 4 kernels of $5\times5$ | 760K | 304 |
| 25x25x4 | Convolution with 12 kernels of $5\times5$ | 758K | 2K |
| 24x24x12 | Convolution with 12 kernels of $5\times5$ | 3M | 4K |
| 12x12x12 | Convolution with 12 kernels of $3\times3$ | 189K | 2K |
| 6x6x12 | Convolution with 4 kernels of $3\times3$ | 16K | 436 |
| 6x6x4 | Affine 256 | 38K | 38K |
| 256 | Affine 64 | 17K | 17K |
| 64 | Affine 3 | 192 | 192 |
| Total | | 4.8M | 69K |

# Model “A”

| Input Size | Type | FLOPs | Parameters |
|---|---|---|---|
| 112×112×3 | NIN with 32 kernels of 5×5 (stride 2) | 9M | 5k |
| 28×28×32 | NIN with 64 kernels of 3×3 | 16M | 24k |
| 14×14×64 | NIN with 128 kernels of 3×3 | 16M | 90k |
| 7×7×128 | NIN with 64 kernels of 3×3 | 4M | 80k |
| 7×7×64 | NIN with 64 kernels of 3×3 | 2M | 42k |
| 7×7×64 | Convolution with 128 kernels of 3×3 | 2M | 37k |
| 3×3×64 | Affine+ReLU 256 | 263k | 263kk |
| 256 | Affine+ReLU 256 | 66k | 66k |
| 256 | Affine 128 | 33k | 33k |
| 128 | Loss: Hinge on the all-pair distance matrix | | |
| Total: | | 49.3M | 633k |

**Table 2.** Model “A” achieves 95.5% on LFW and takes 40ms on a mobile device

# Model “B”

| Input Size | Type | FLOPs | Parameters |
|---|---|---|---|
| 112×112×3 | NIN with 32 kernels of 5×5 (stride 2) | 9M | 5k |
| 28×28×32 | NIN with 96 kernels of 3×3 | 24M | 38k |
| 14×14×96 | NIN with 128 kernels of 3×3 | 23M | 128k |
| 7×7×128 | NIN with 128 kernels of 3×3 | 9M | 165k |
| 7×7×128 | NIN with 128 kernels of 3×3 | 9M | 165k |
| 7×7×128 | Convolution with 128 kernels of 3×3 | 8M | 148k |
| 3×3×128 | Affine+ReLU 256 | 296k | 296k |
| 256 | Affine+ReLU 256 | 66k | 66k |
| 256 | Affine 128 | 33k | 33k |
| 128 | Loss: Hinge on the all-pair distance matrix | | |
| Total: | | 82.3M | 1M |

**Table 3.** Model “B” achieves 97.5% on LFW and takes 60ms on a mobile device

ORCAM

# Model “C”

| Input Size | Type | FLOPs | Parameters |
|---|---|---|---|
| $112\times112\times3$ | NIN with 64 kernels of $5\times5$ (stride 2) | 20M | 10k |
| $28\times28\times64$ | NIN with 192 kernels of $3\times3$ | 95M | 140k |
| $14\times14\times192$ | NIN with 384 kernels of $3\times3$ | 139M | 812k |
| $7\times7\times384$ | NIN with 256 kernels of $3\times3$ | 48M | 950k |
| $7\times7\times256$ | NIN with 256 kernels of $3\times3$ | 33M | 560k |
| $7\times7\times256$ | Convolution with 256 kernels of $3\times3$ | 29M | 591k |
| $3\times3\times256$ | Affine+ReLU 4096 | 10M | 10M |
| 4096 | Affine+ReLU 4096 | 17M | 17M |
| 4096 | Affine 128 | 525k | 525k |
| 128 | Loss: Hinge on the all-pair distance matrix | | |
| Total: | | 391.5M | 30M |

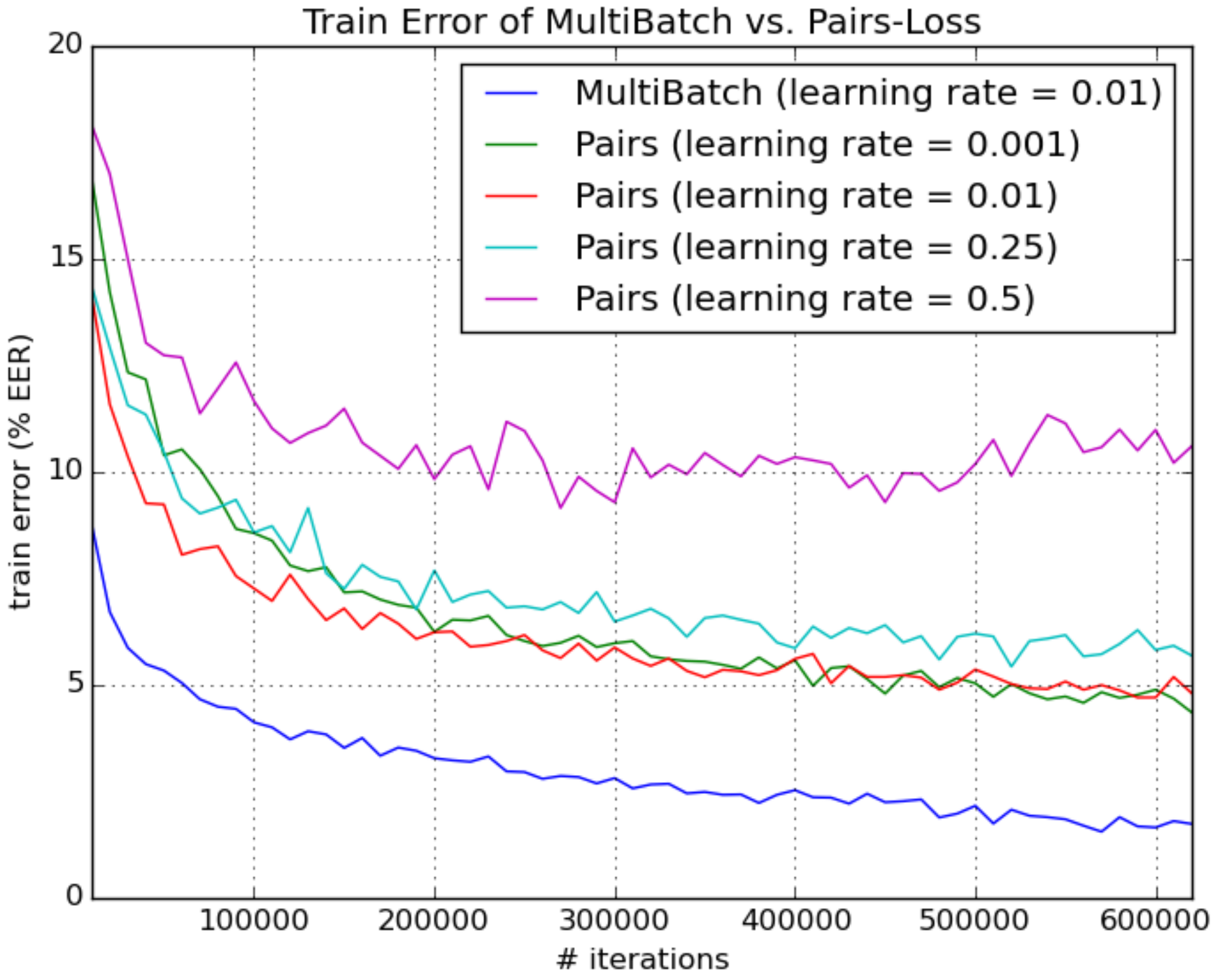
**Table 4.** Model “C” achieves 98% accuracy on LFW and takes 330ms

# Networks

| Model | FLOPs | Parameters | LFW score |
|---|---|---|---|
| “A" | 44M | 633k | 96.9% |
| “B" | 82M | 1M | 97.8% |
| “C” | 392M | 30M | 98.8% |

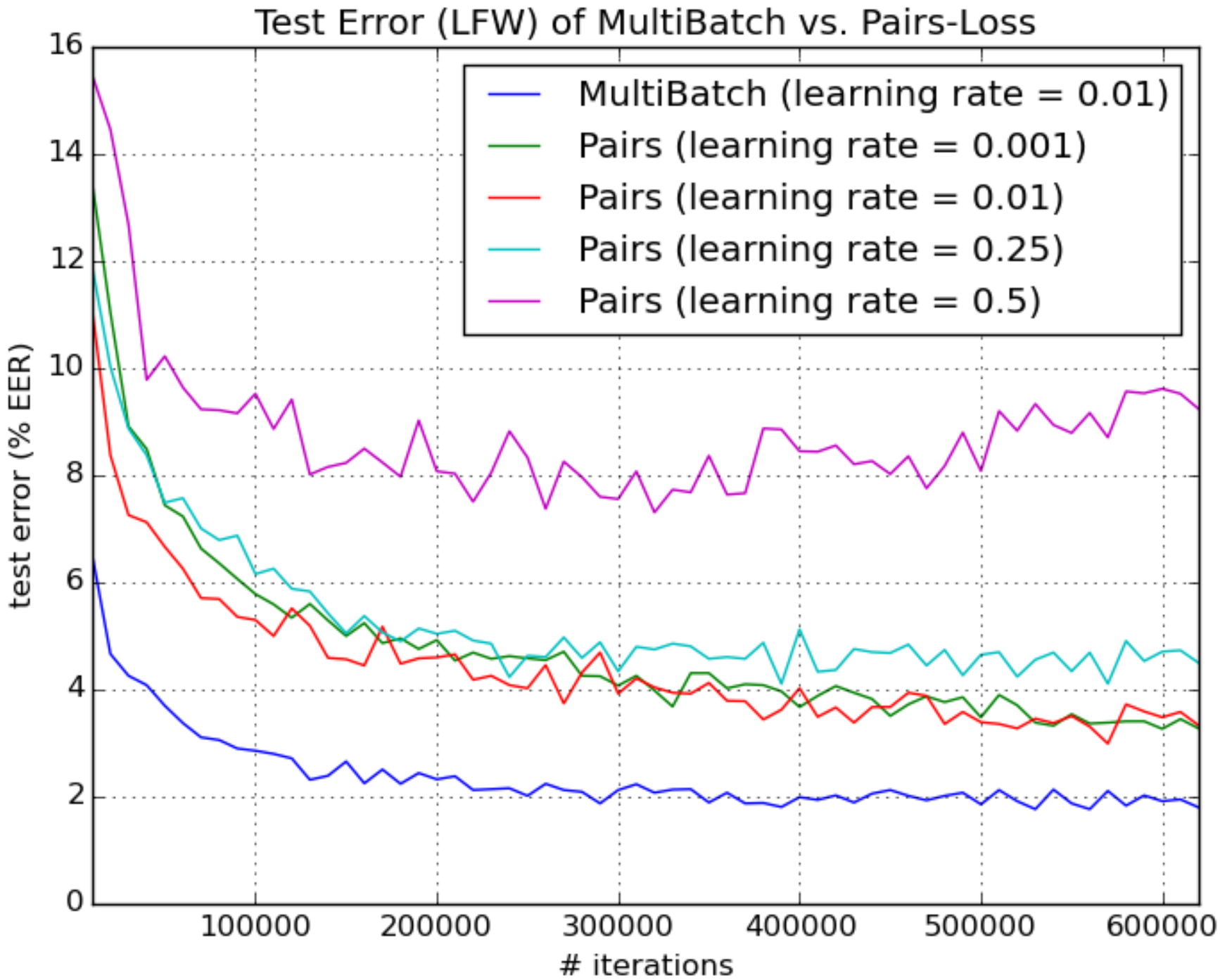
ORCAM

# Pairs vs. Multi-batch (Train)

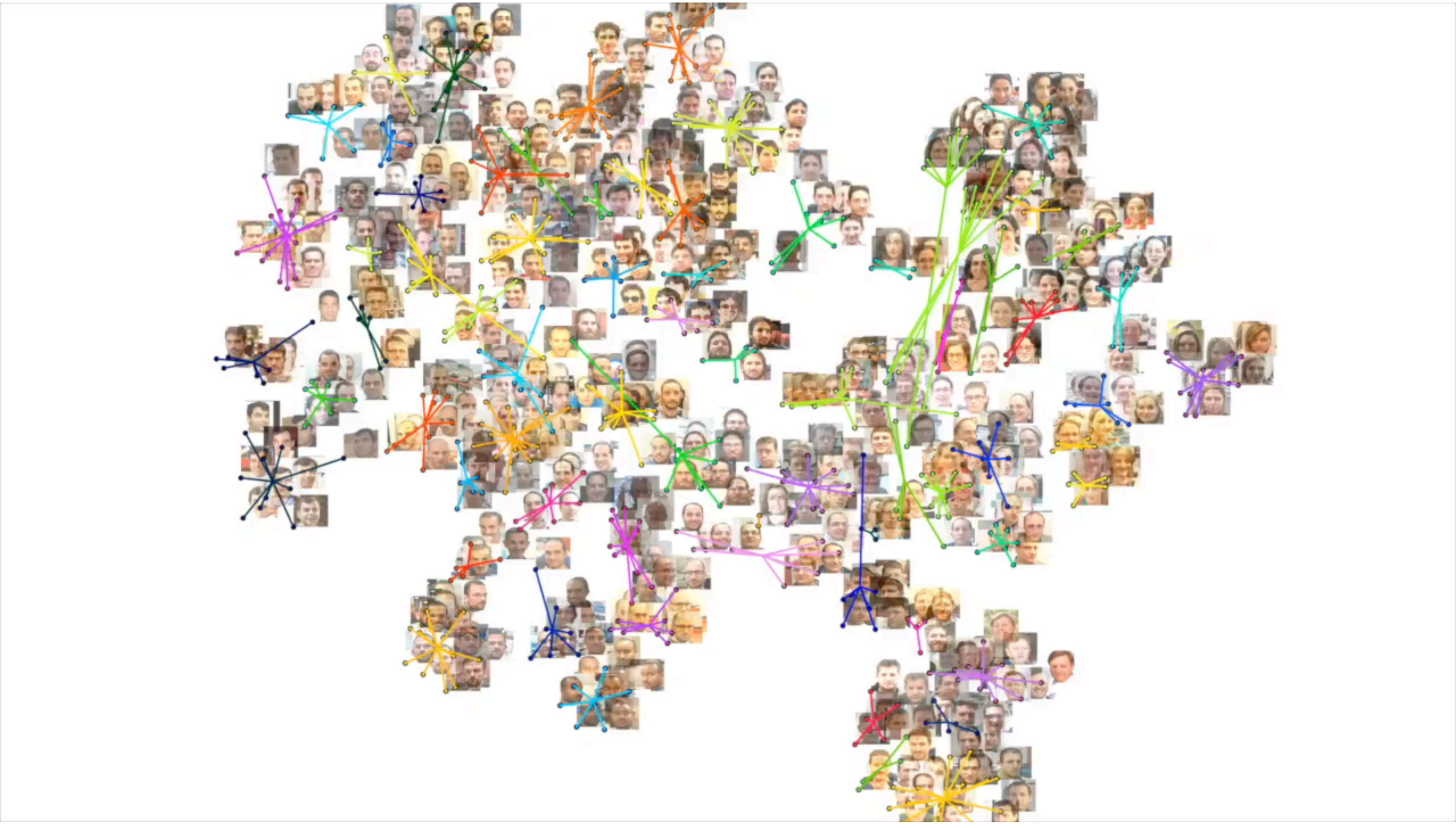



ORCAM

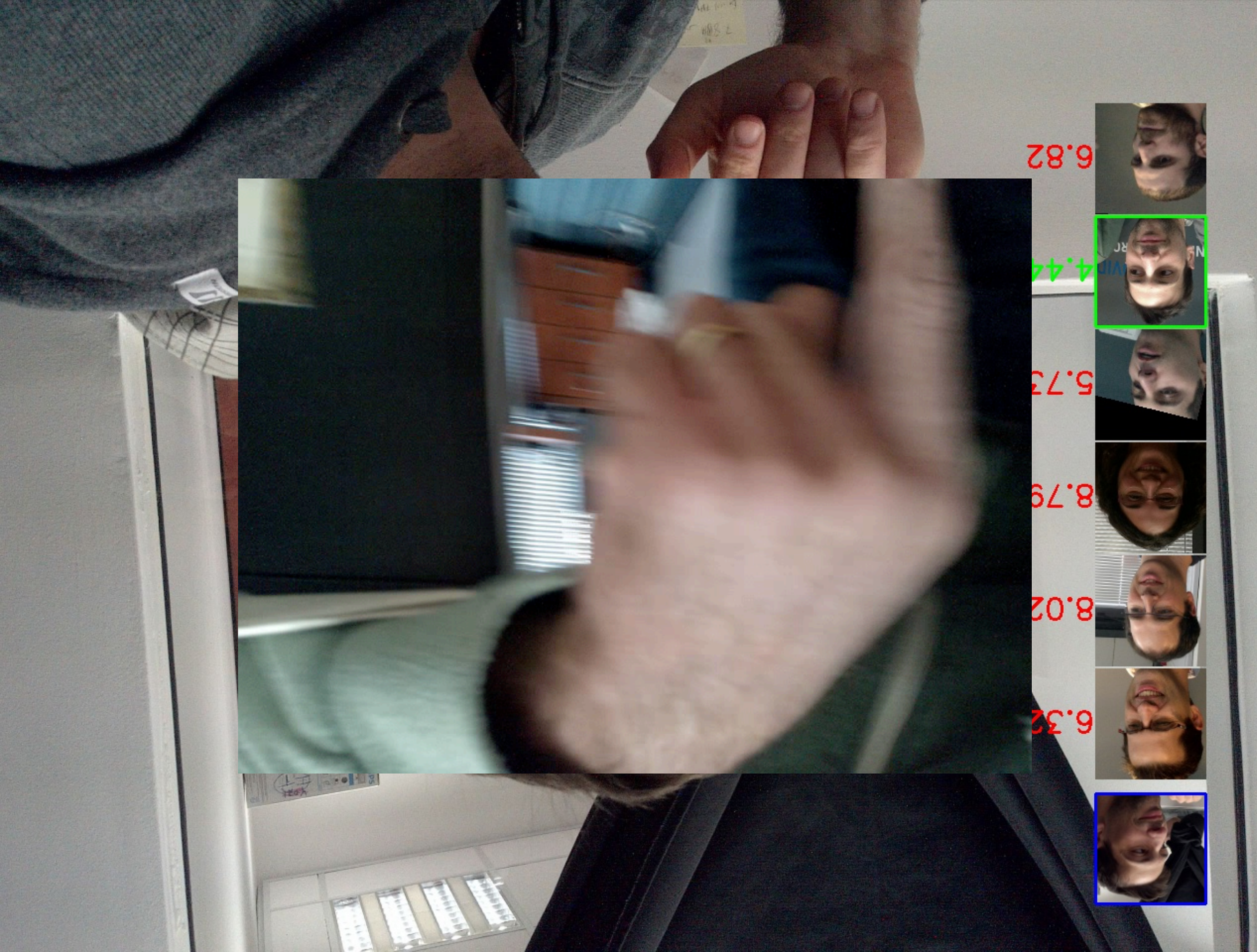
# Pairs vs. Multi-batch (Test)




ORCAM



ORCAM


6.82
4.44
5.73
8.79
8.02
6.32

# Conclusions

- Practical approach for metric learning
- Compared to FaceNet:
  - $1/100$ of the samples
  - $1/100$ of the training time
  - $1/20$ run-time

- Face alignment + recognition in 60ms on a mobile device

# Fast and Furious Face Recognition

## Efficient metric learning for video stream data

Yoni Wexler

ORCAM