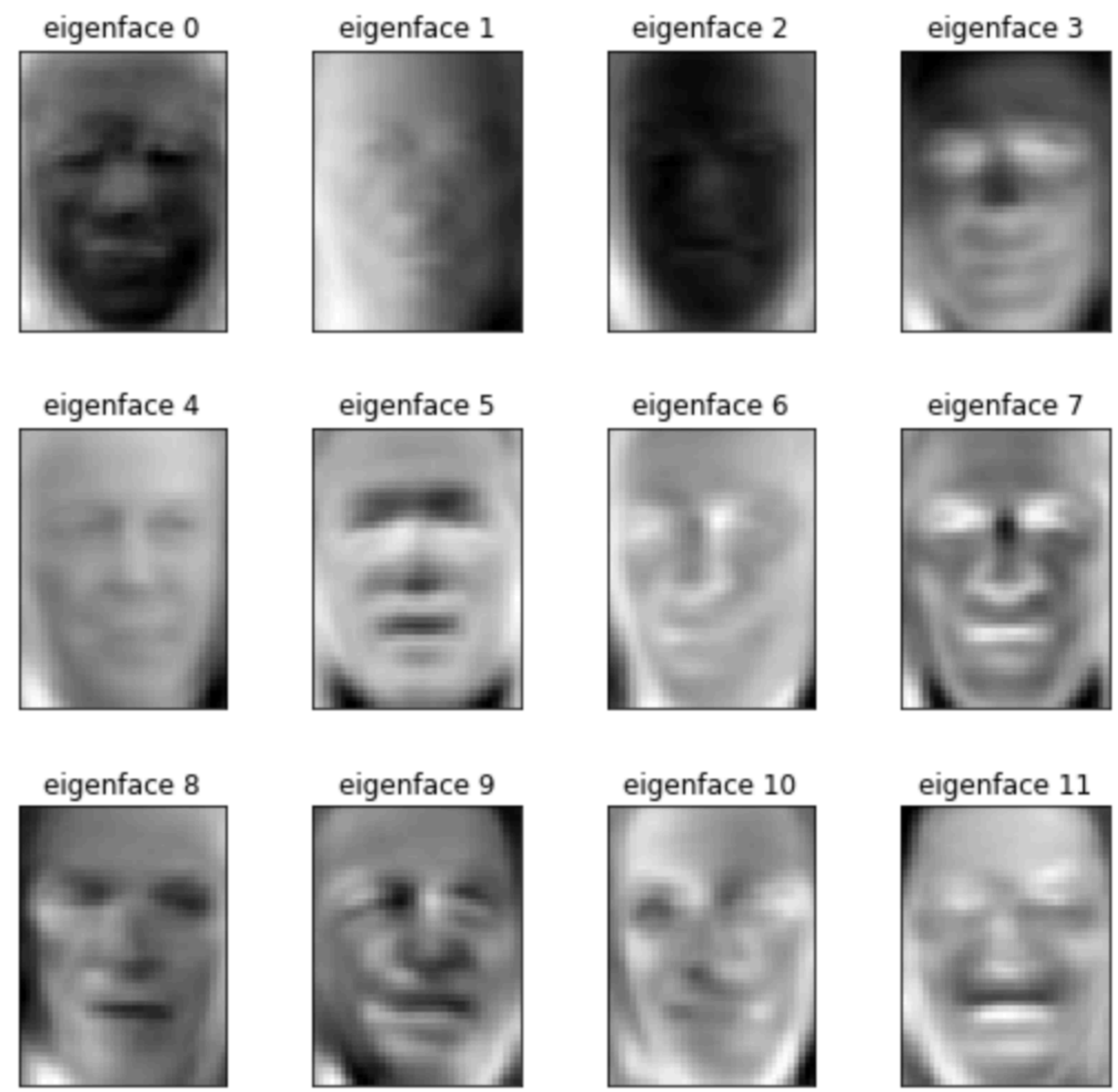
Deep Face Recognition: A Survey

Mei Wang, Weihong Deng

Eigenface

1990s

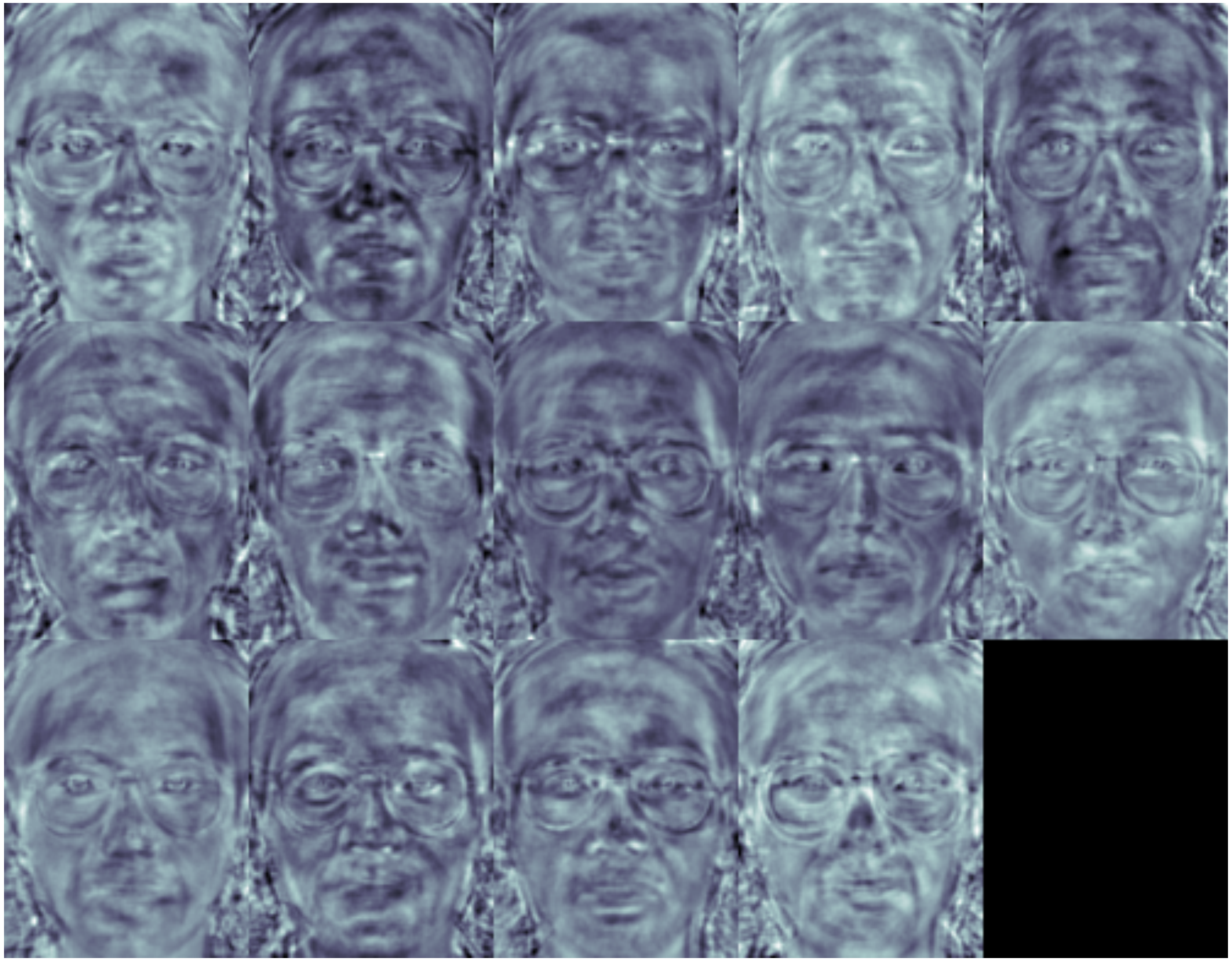
Principal Component Analysis (PCA)



ref : https://github.com/gouravaich/faces-recognition-pca/blob/master/PCA%20Mini-Project.ipynb

Fisherface 1997

Linear Discriminator Analysis (LDA)



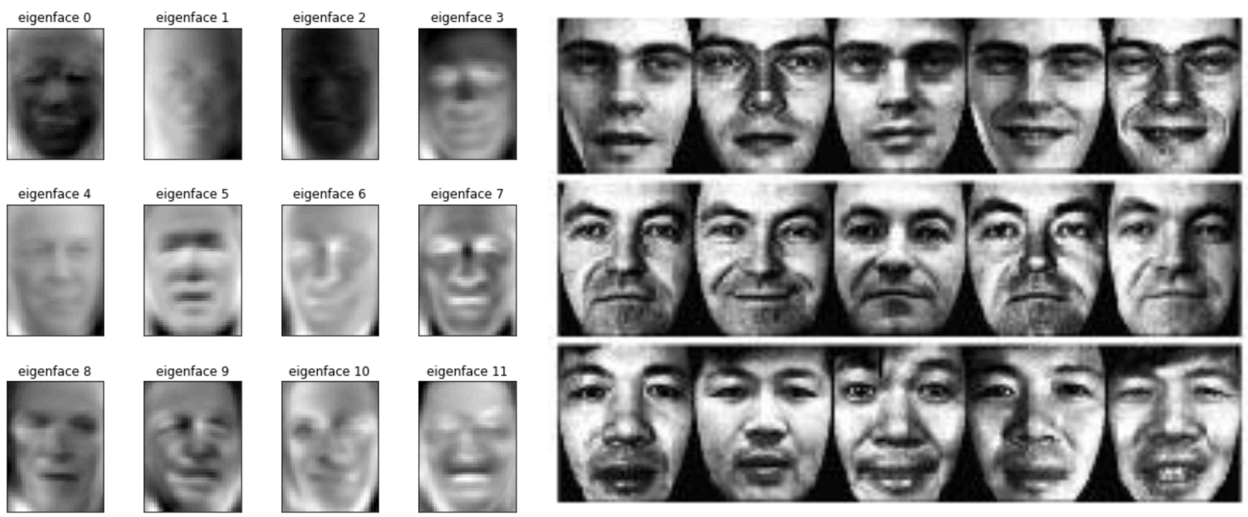
ref : https://github.com/gouravaich/faces-recognition-pca/blob/master/PCA%20Mini-Project.ipynb

Fisherface 1997

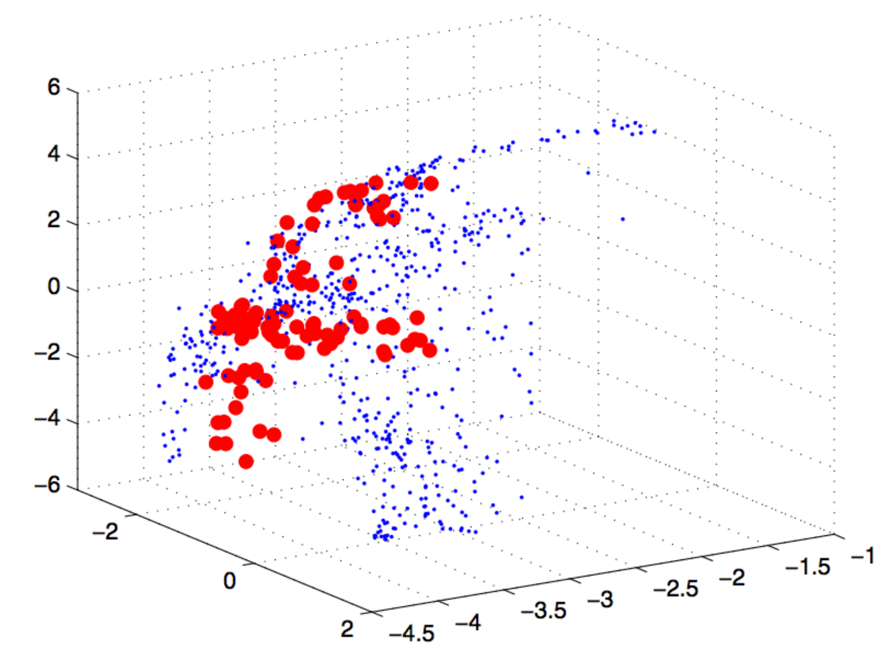
Linear Discriminator Analysis (LDA)

ref : https://keshavchinna.blogspot.com/2012/05/recognition-face-foremost.html

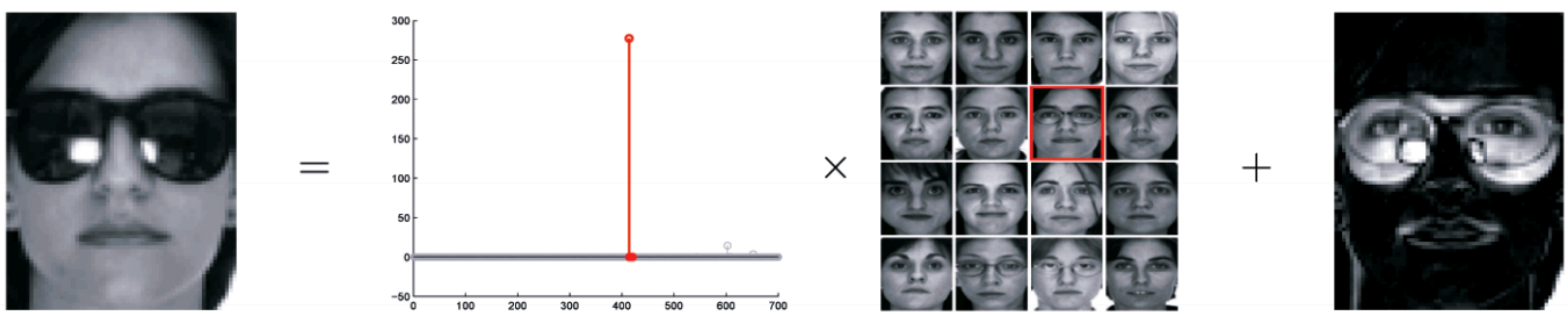
Linear subspace



Manifold

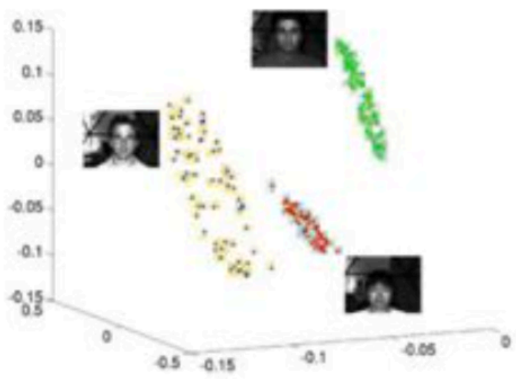


Sparse representation



Sparse representation

Linear subspace model for images of same face under varying illumination:



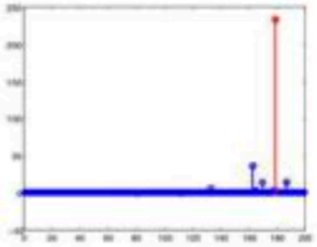
$$A_i = [\;|\;|\;\ldots\;] \in \mathbb{R}^{m \times n_i}$$

If test image $\boldsymbol{y} \in \mathbb{R}^m$ is also of subject i, then $\boldsymbol{y} = A_i \boldsymbol{x_i}$ for some $\boldsymbol{x}_i \in \mathbb{R}^{n_i}$.

Can represent *any* test image wrt the *entire training set* as $\boldsymbol{y} = A\boldsymbol{x} + \boldsymbol{e}$

$=$

$\times$



$+$

$\boldsymbol{y} \in \mathbb{R}^m$ Test image

$A = [A_1 \mid A_2 \mid \ldots \mid A_k] \in \mathbb{R}^{m \times n}$ Combined training dictionary

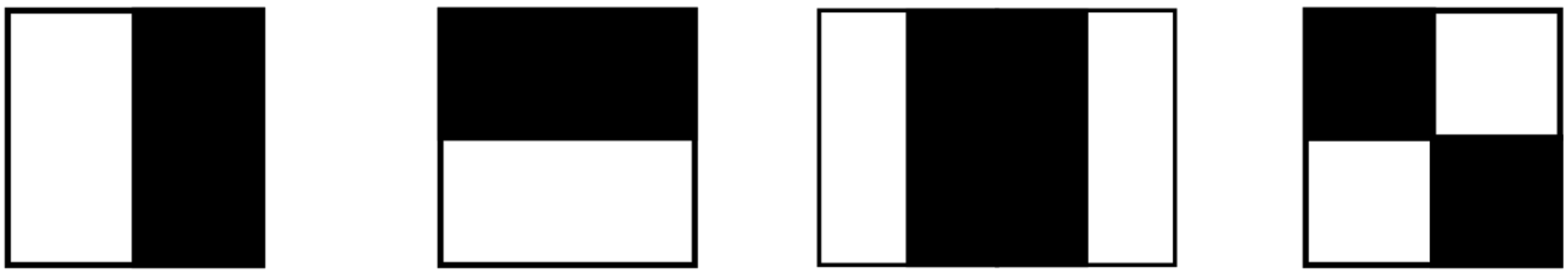
$\boldsymbol{x} \in \mathbb{R}^n$ coefficients

$\boldsymbol{e} \in \mathbb{R}^m$ corruption, occlusion

ref : https://slideplayer.com/slide/4839441/

Haar Feature 2002

(Face Detection)



(1) (2) (3) (4)

-1	-1	5
-1	-1	5
-1	-1	5

(1)

5	5	5
-1	-1	-1
-1	-1	-1

(2)

-1	5	-1
-1	5	-1
-1	5	-1

(3)

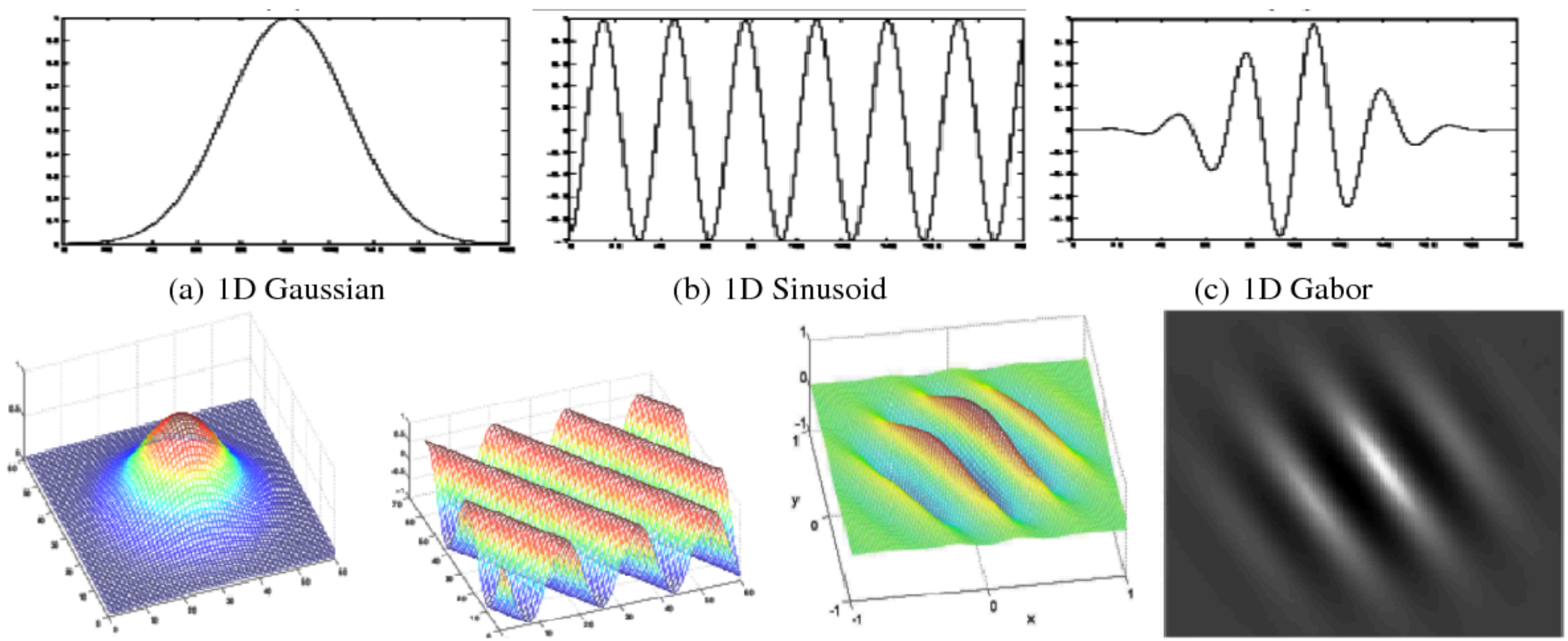
5	-1	-1
-1	5	-1
-1	-1	5

(4)

ref : https://github.com/gouravaich/faces-recognition-pca/blob/master/PCA%20Mini-Project.ipynb

Gabor Filter

2000s



(a) 1D Gaussian (b) 1D Sinusoid (c) 1D Gabor

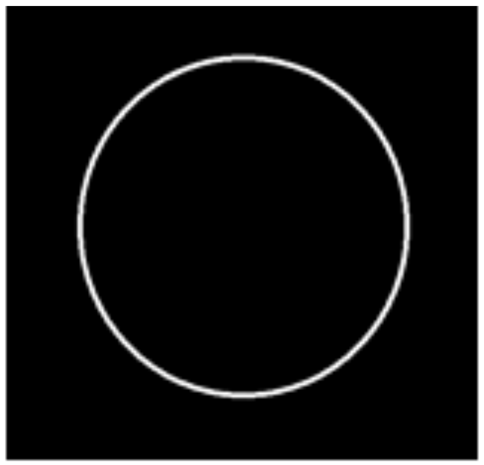
(d) 2D Gaussian (e) 2D Sinusoid (f) 2D Gabor (g) 2D Gabor Filter

Figure 6: Conventional 1 and 2 Dimensional Gabor Filter (Prasad and Domke, 2007).

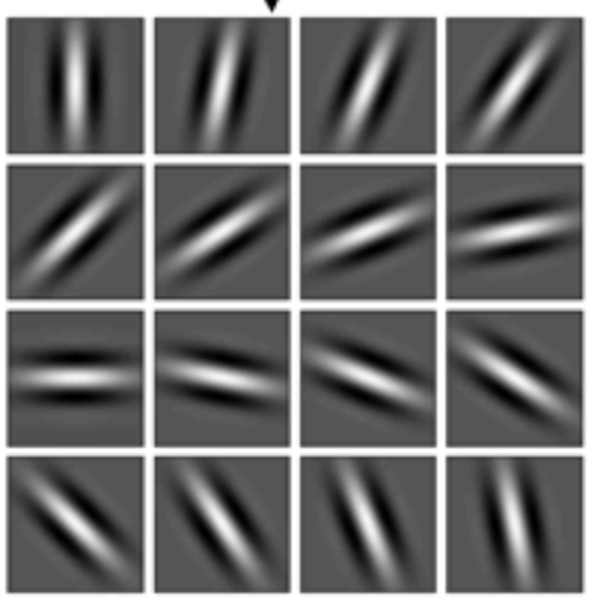
ref : https://www.semanticscholar.org/paper/Finger-Knuckle-Print-ROI-Extraction-using-Curvature-Nigam-Gupta/3bc557d594033907d011f01a4cc01b3595e8bb64

Gabor Filter

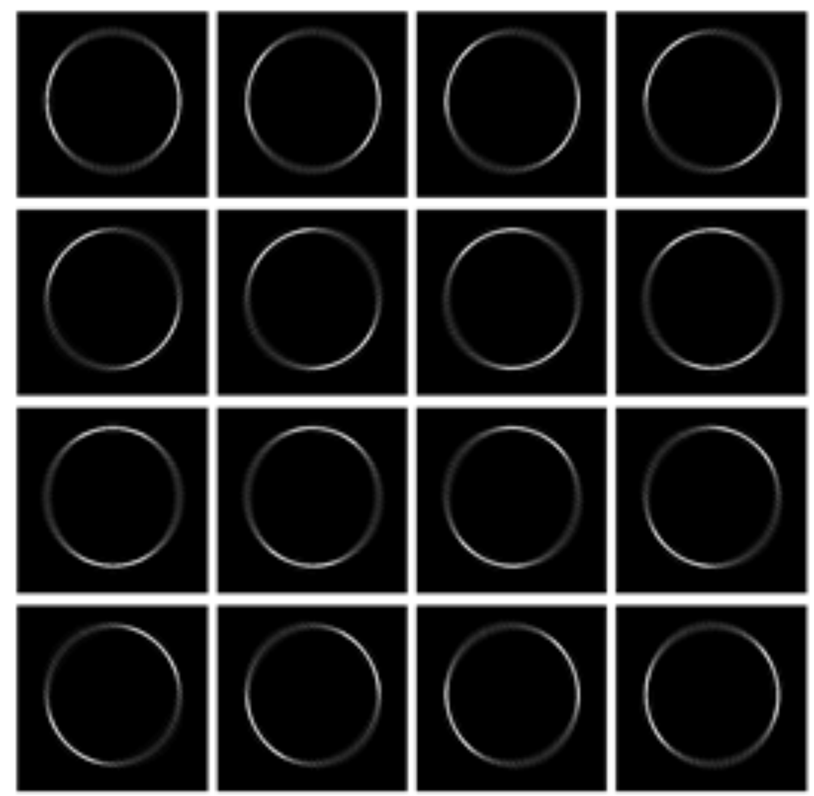
2000s



Input Image of
a circle



A bank of 16 Gabor Filters

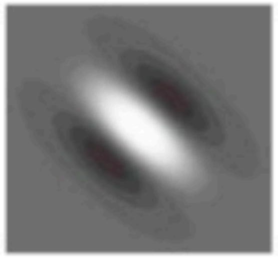


The output circle as seen when pass
through individual Gabor filter

ref : https://medium.com/@anuj_shah/through-the-eyes-of-gabor-filter-17d1fdb3ac97

Gabor Filter

2000s

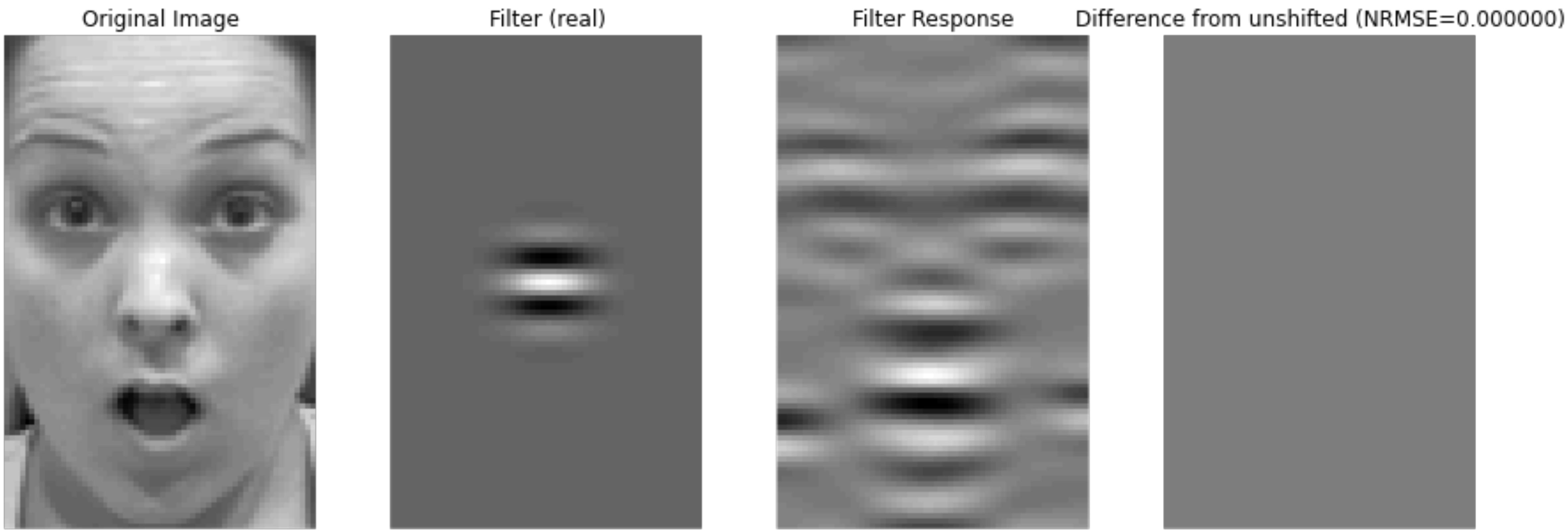


Move this Gabor filter with different orientations along the fingerprint

ref : https://www.slideshare.net/ankitnayan3/gabor-filtering-for-fingerprint-image-enhancement

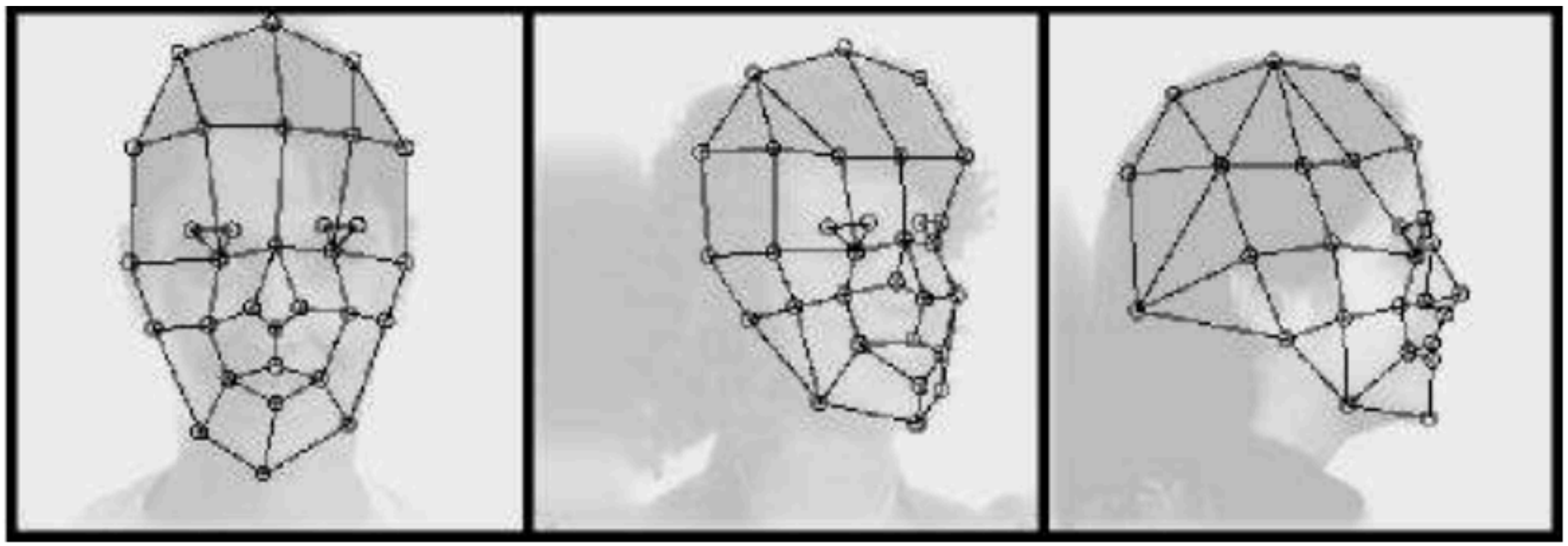
Gabor Filter

2000s



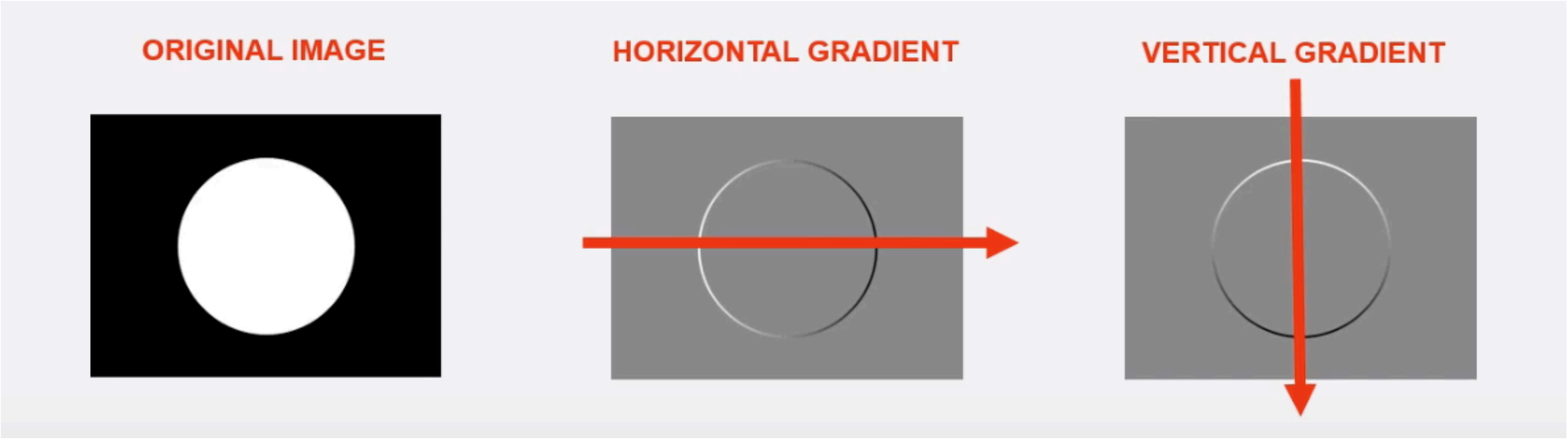
ref : http://ttsuchi.github.io/2015/08/26/gaborfilters.html

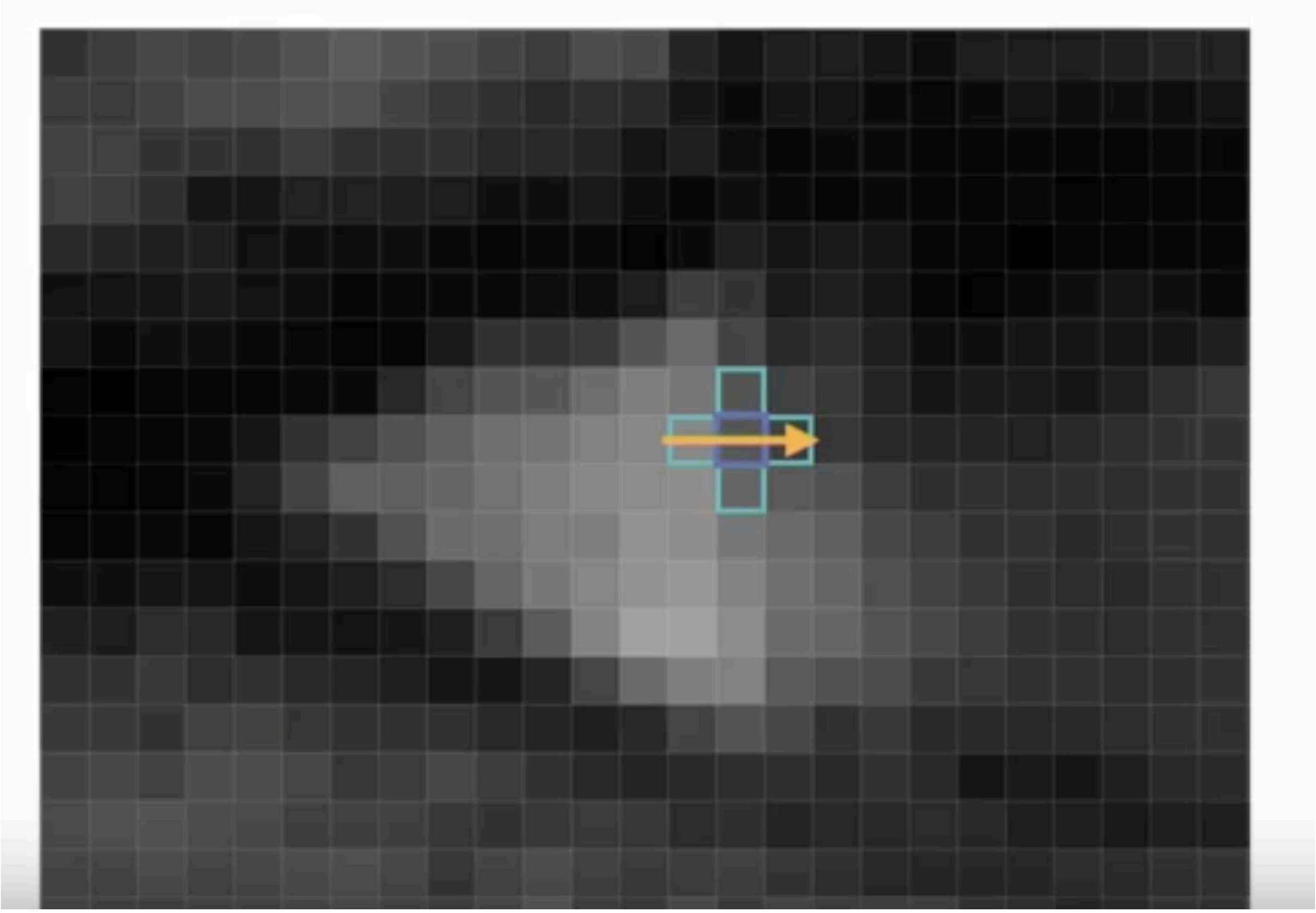
Elastic Bunch Graph Matching (EBGM)



ref : https://pdfs.semanticscholar.org/de86/a9f484addcfee57a6f5a9224aa77bd23345b.pdf

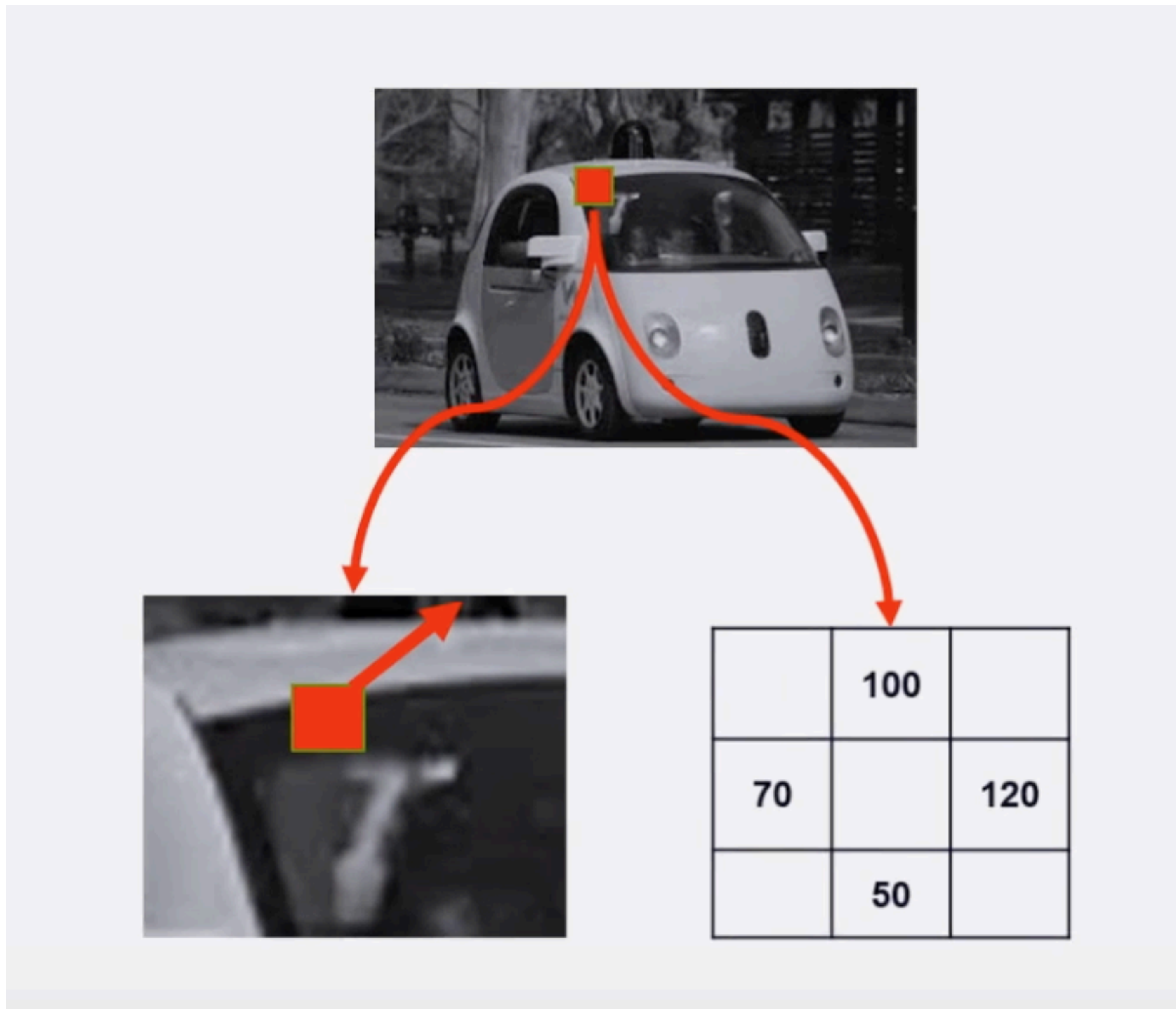
HOG (Histogram of oriented gradients) 2005





ref : https://medium.com/@rehan_ahmad/a-take-on-h-o-g-feature-descriptor-e839ebba1e52

HOG (Histogram of oriented gradients) 2005

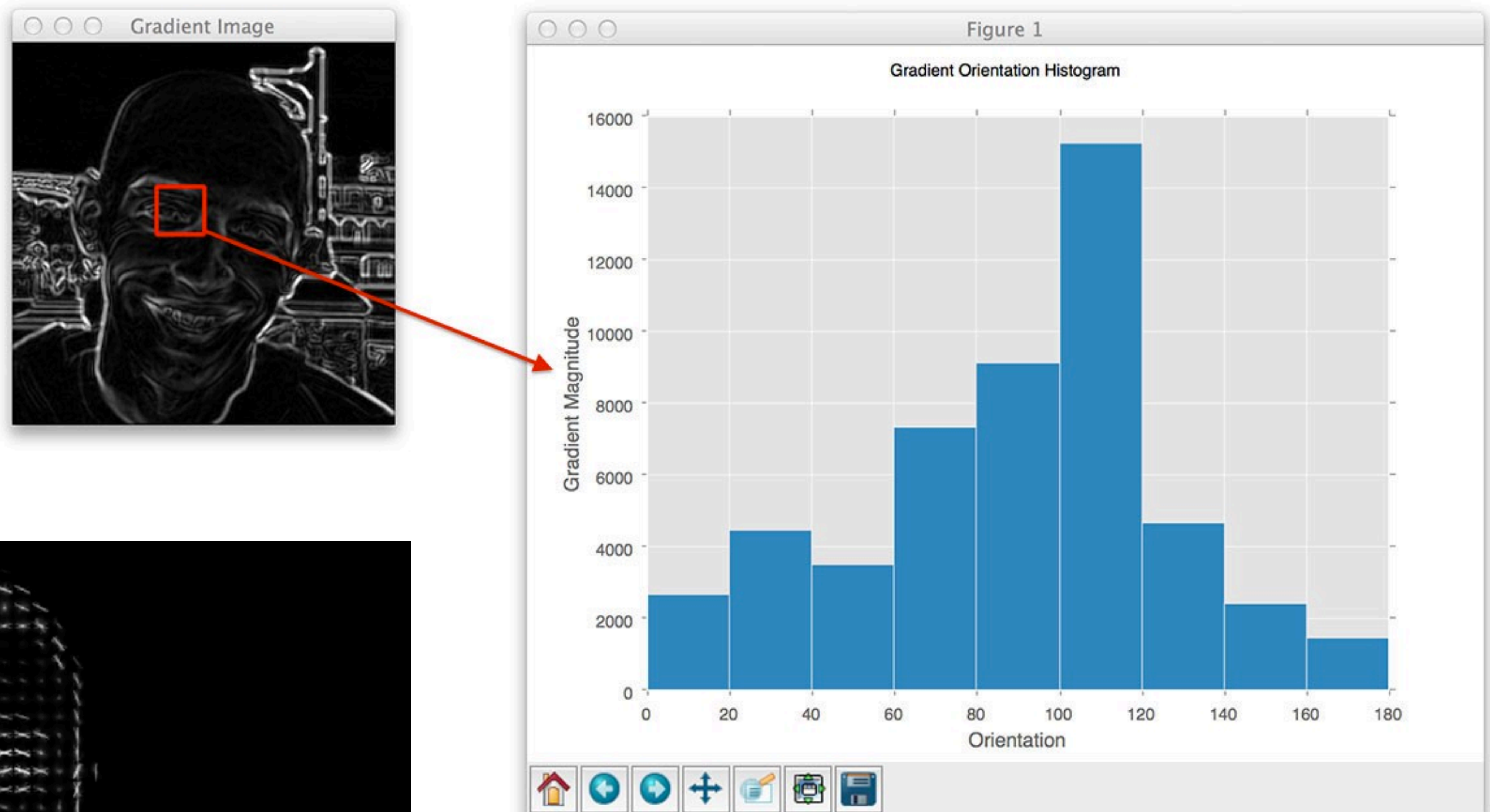


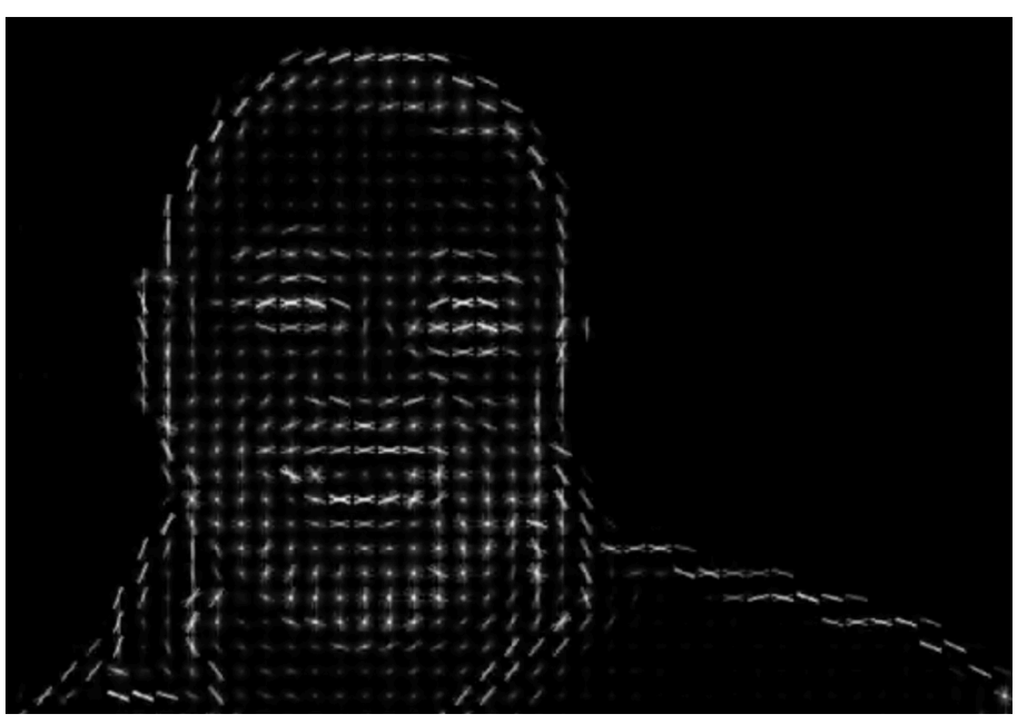
Gradient Magnitude = $\sqrt{(50)^2 + (50)^2} = 70.1$

Gradient Angle = $\tan^{-1}(50/50) = 45°$

ref : https://medium.com/@rehan_ahmad/a-take-on-h-o-g-feature-descriptor-e839ebba1e52

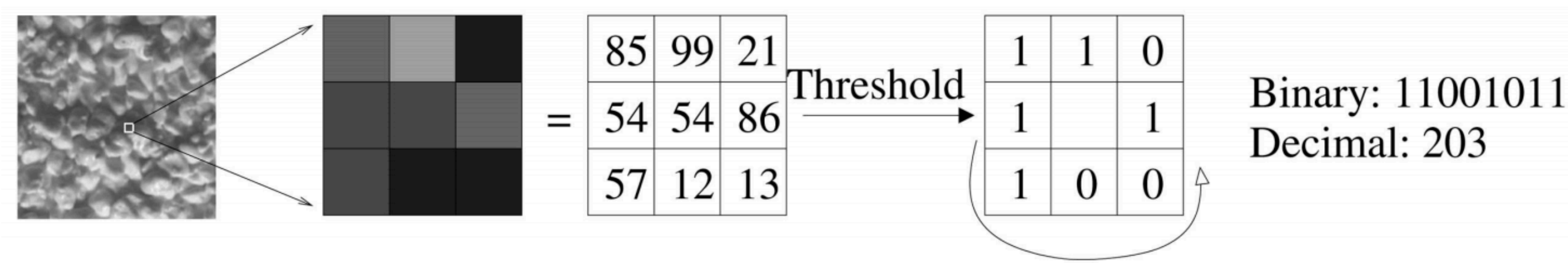
HOG (Histogram of oriented gradients) 2005





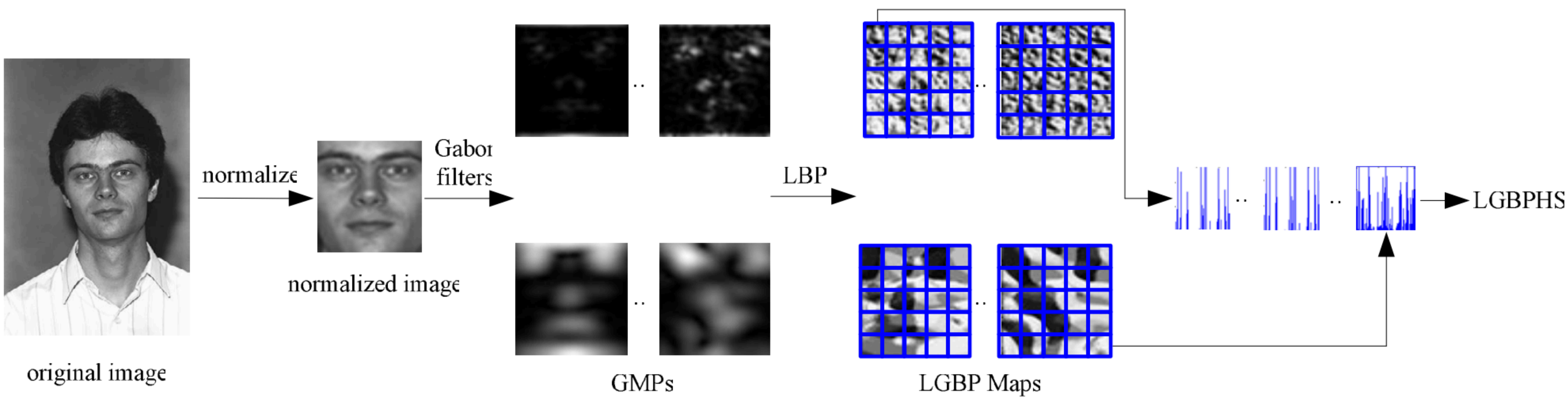
ref : https://medium.com/@rehan_ahmad/a-take-on-h-o-g-feature-descriptor-e839ebba1e52

LBP (Local Binary Patterns) 2006

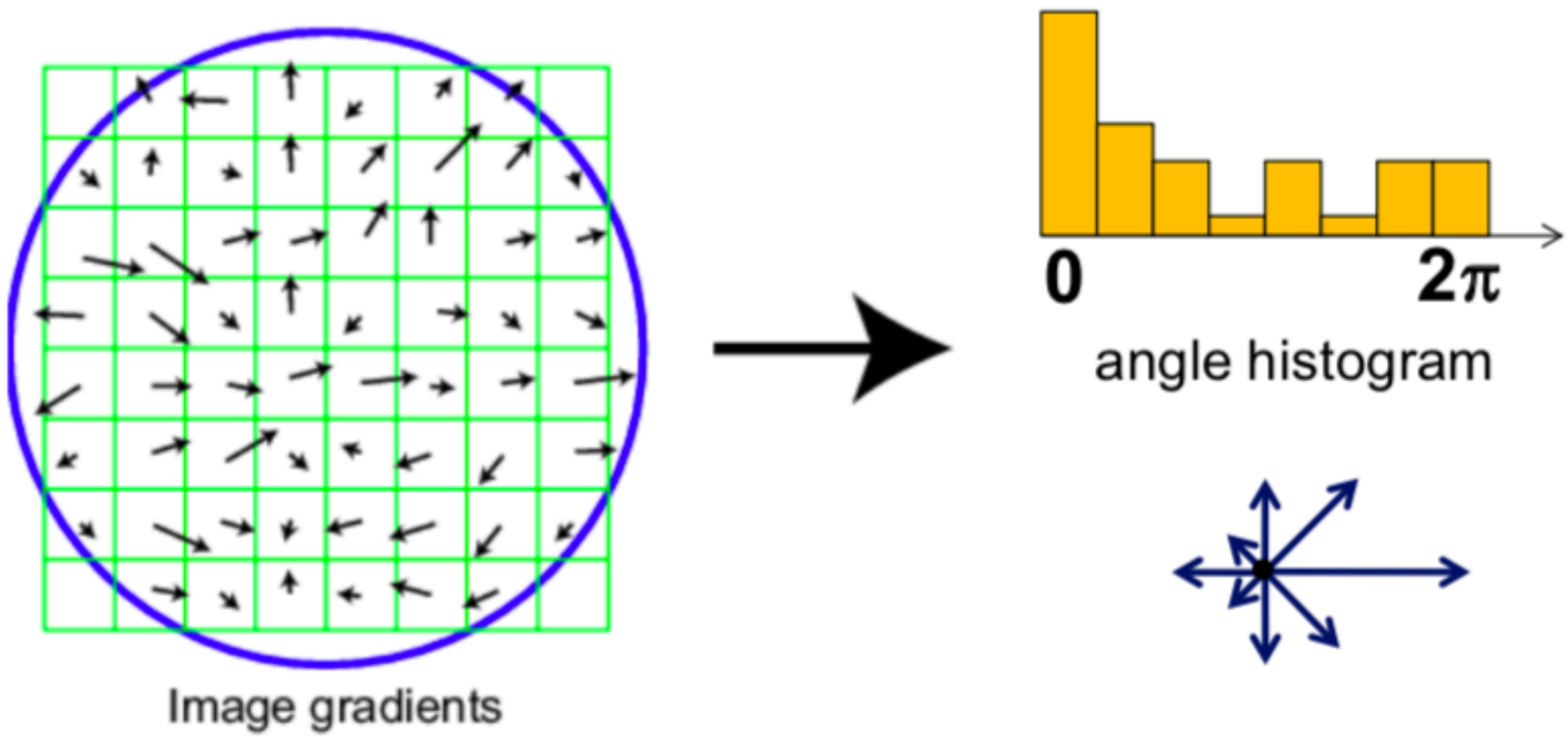


ref : https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=1717463

Local Gabor Binary Pattern Histogram Sequence (LGBPHS)

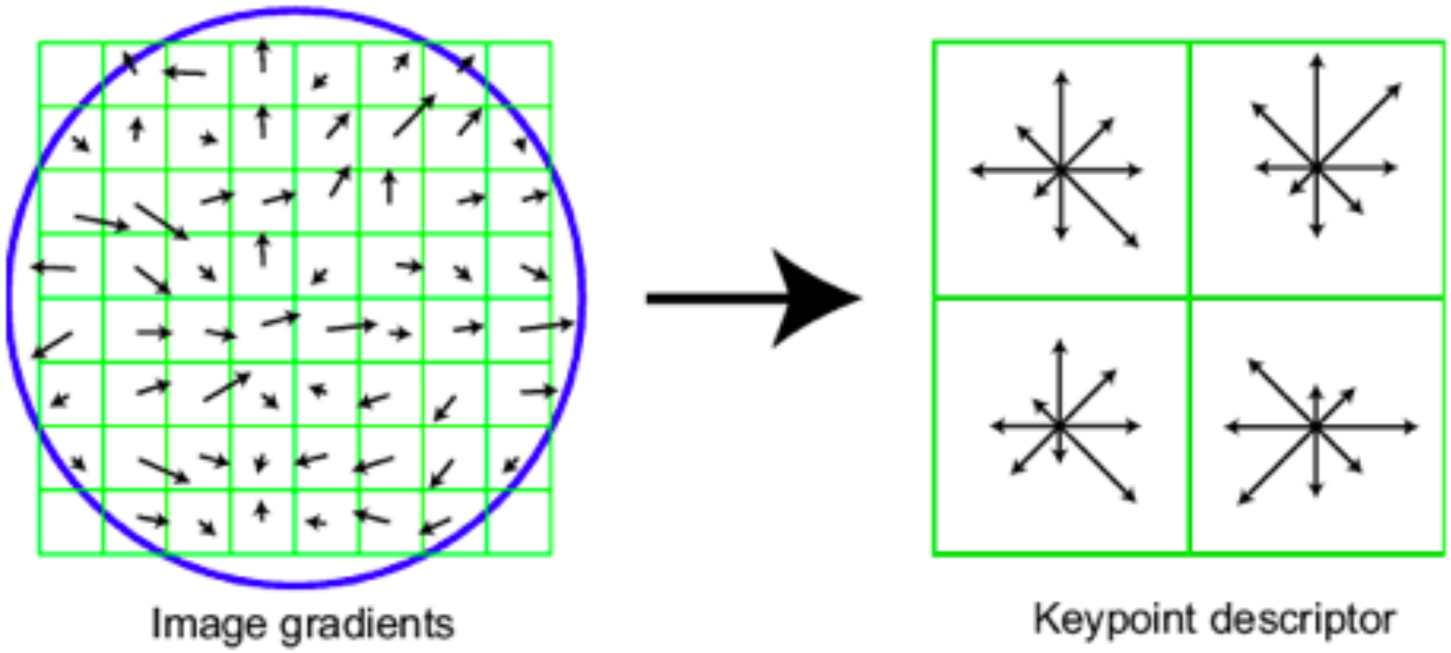


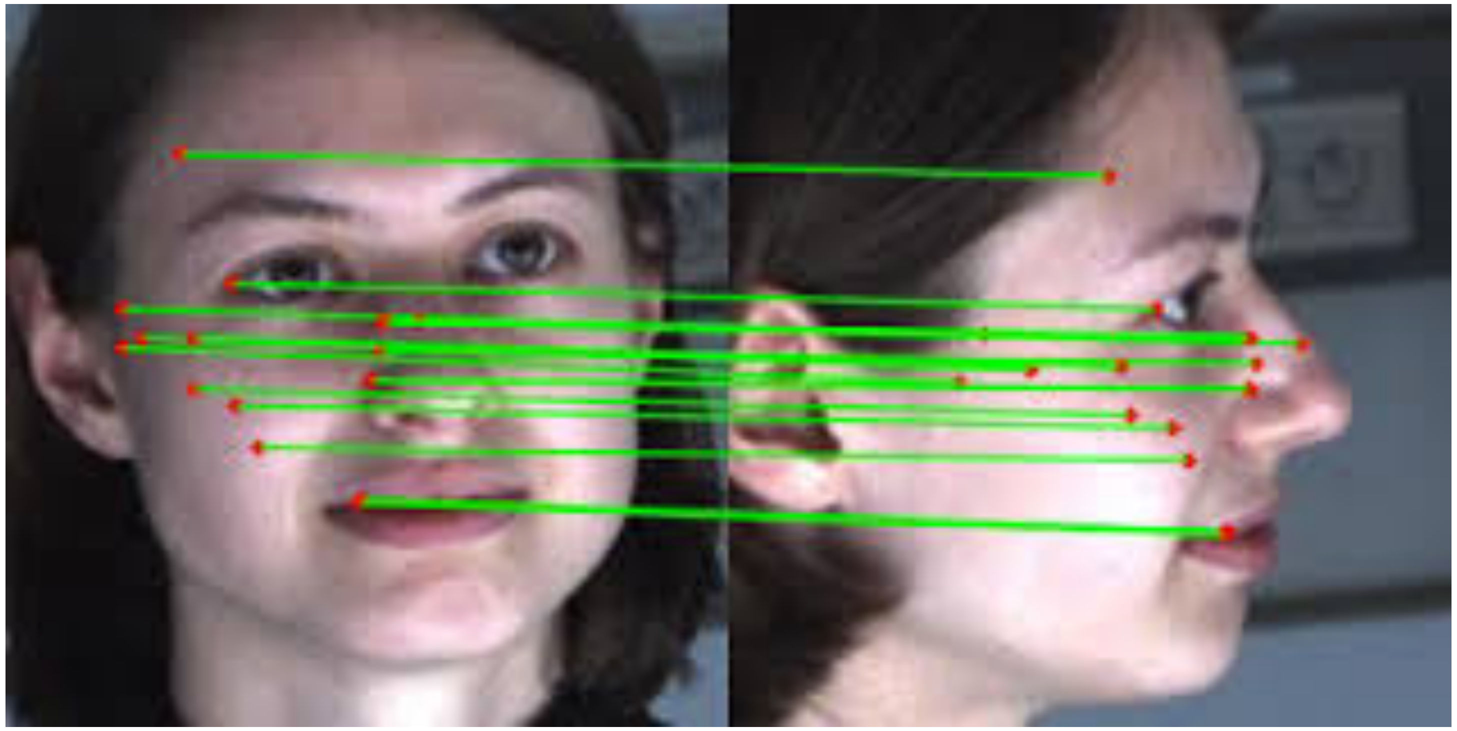
Scale Invariant Feature Transform (SIFT) 2006



ref :https://towardsdatascience.com/sift-scale-invariant-feature-transform-c7233dc60f37

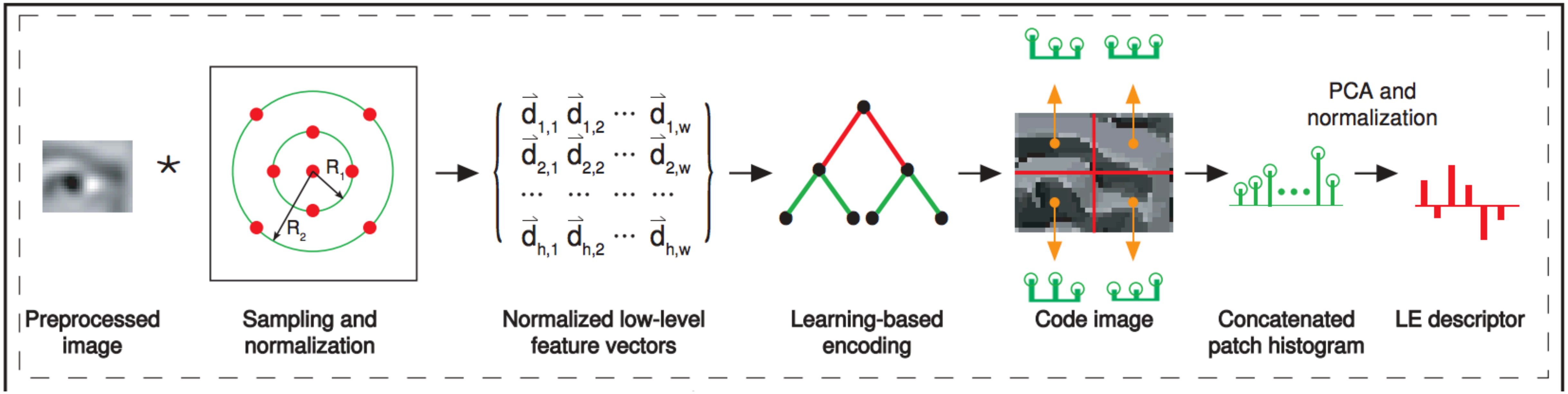
Scale Invariant Feature Transform (SIFT) 2006





ref :https://arxiv.org/pdf/1708.02721.pdf

Learning-based local descriptors



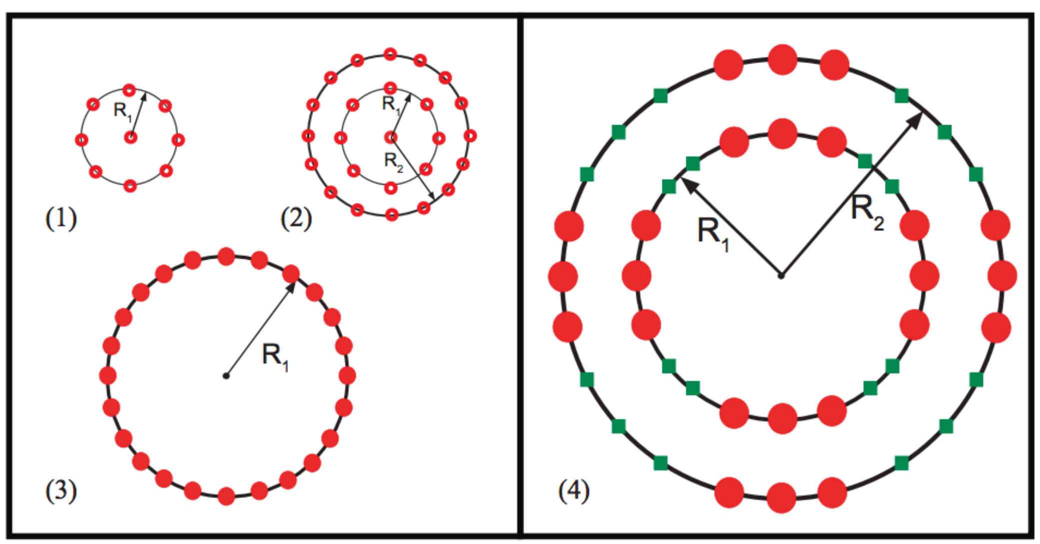
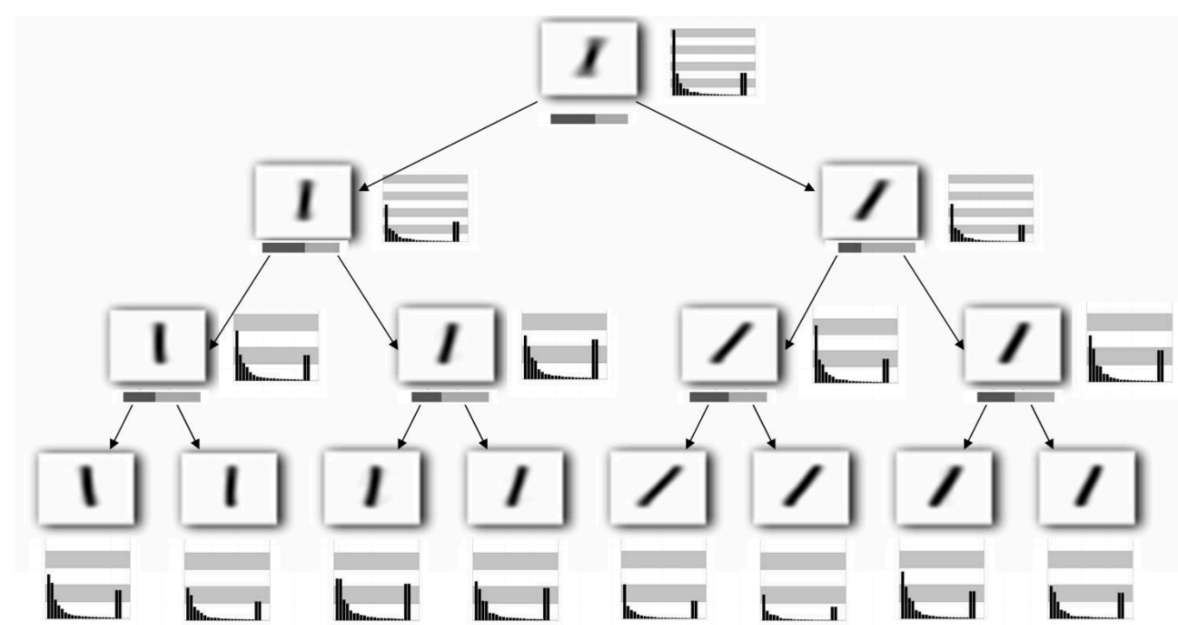


Figure 4. Four typical sampling methods used in our experiments: (1) $R_1 = 1$, with center; (2) $R_1 = 1, R_2 = 2$, with center; (3) $R_1 = 3$, no center; (4) $R_1 = 4, R_2 = 7$, no center. (The sampling dots on the green-square labeled arcs are omitted for better visuality).

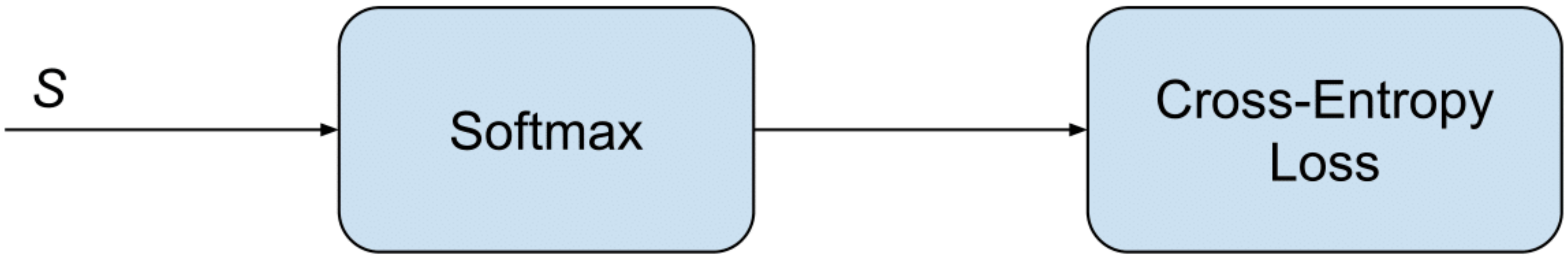


Face verification [1 : 1]

Face identification [1 : N]

ref : https://medium.com/@krsatyam1996/face-recognition-using-openface-92f02045ca2a

Softmax loss (softmax + cross-entropy)



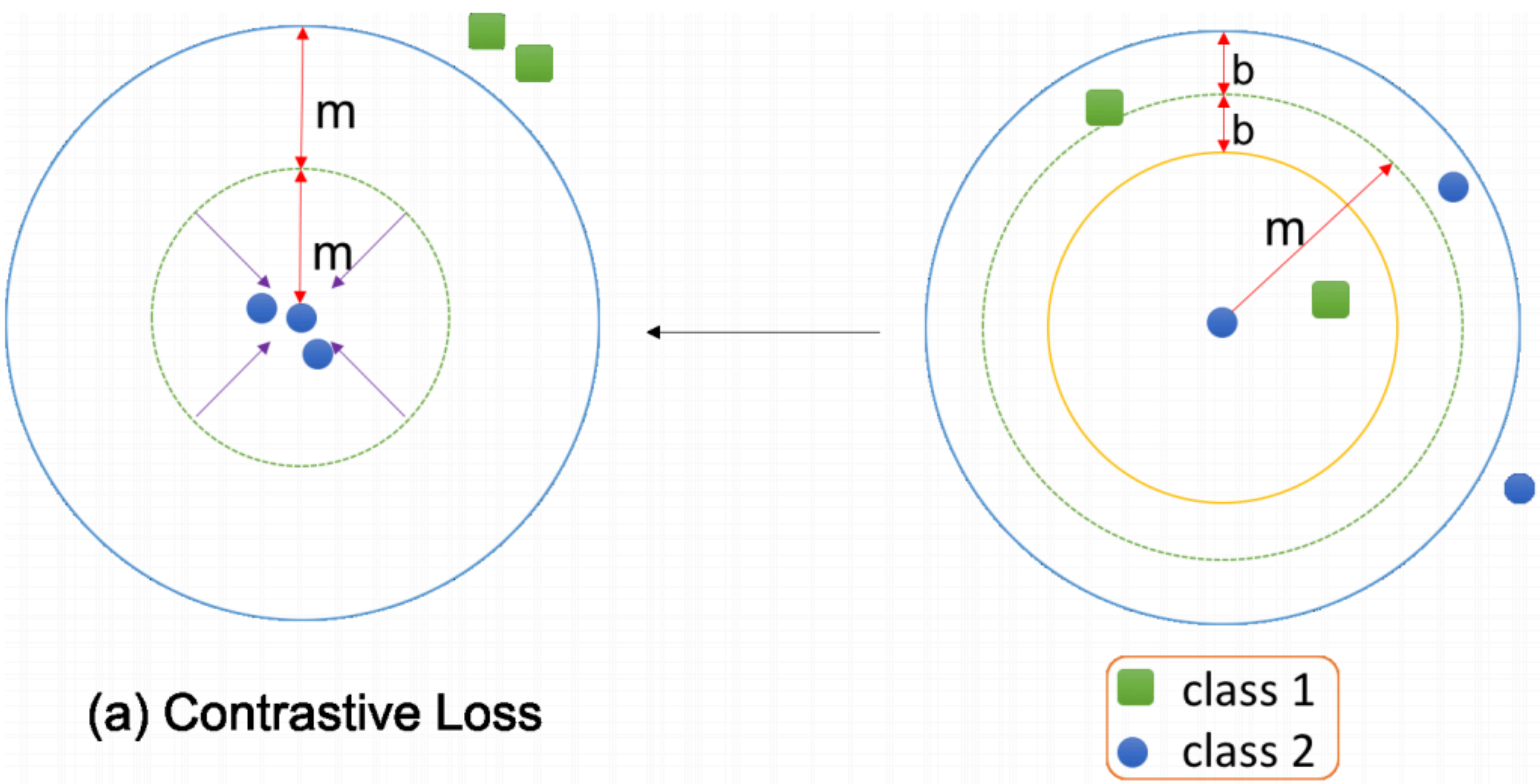
$$f(s)_i = \frac{e^{s_i}}{\sum_j^C e^{s_j}} \qquad CE = -\sum_i^C t_i log(f(s)_i)$$

$$CE = -log\left(\frac{e^{s_p}}{\sum_j^C e^{s_j}}\right)$$

Euclidean-distance-based loss

- **The contrastive loss**

$$\mathcal{L} = y_{ij} max\left(0, \|f(x_i) - f(x_j)\|_2 - \epsilon^+\right) \\ + (1 - y_{ij}) max\left(0, \epsilon^- - \|f(x_i) - f(x_j)\|_2\right)$$



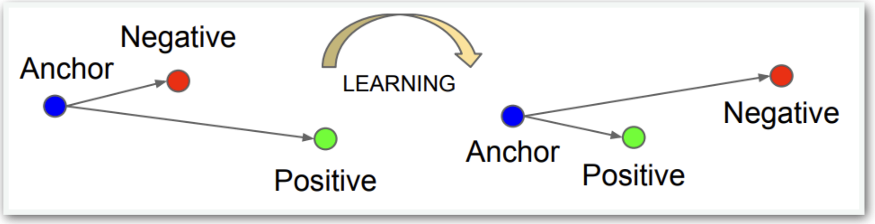
(a) Contrastive Loss

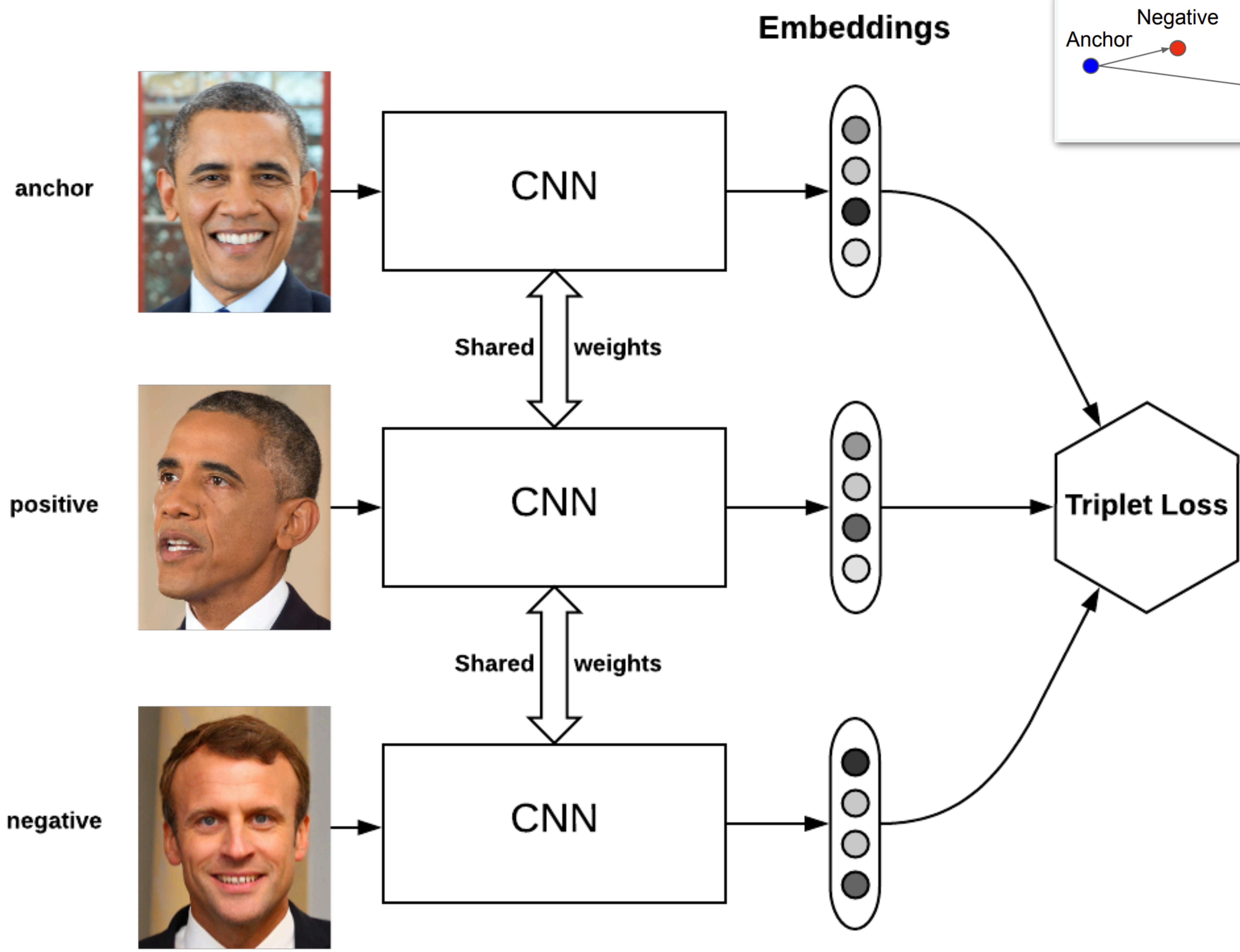
ref : https://www.ijcai.org/proceedings/2017/0308.pdf

Euclidean-distance-based loss

$$\max\left(\|f_a - f_p\|^2 - \|f_a - f_n\|^2 + m, 0\right)$$

- **The triplet loss**



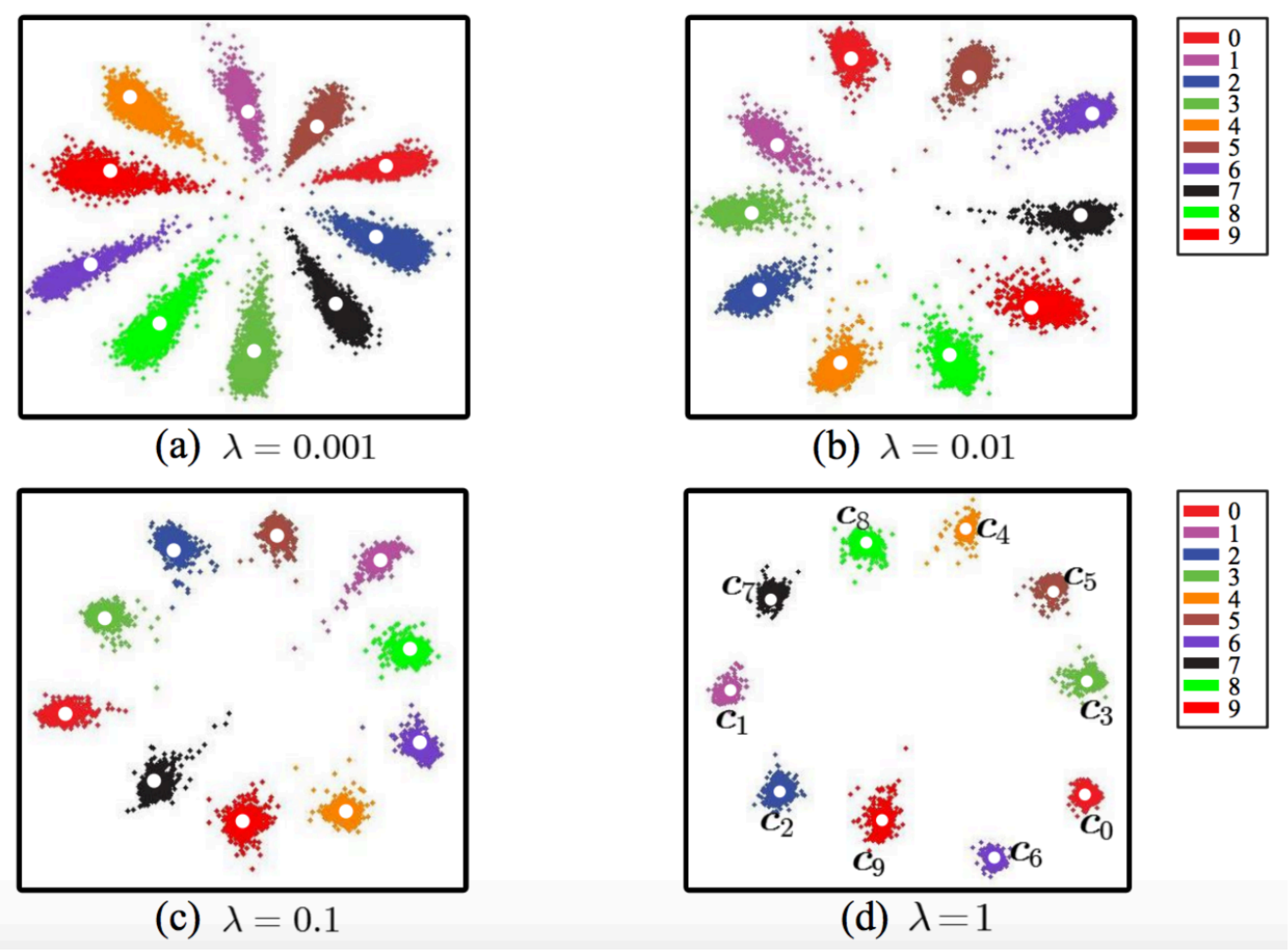


ref : https://omoindrot.github.io/triplet-loss

Euclidean-distance-based loss

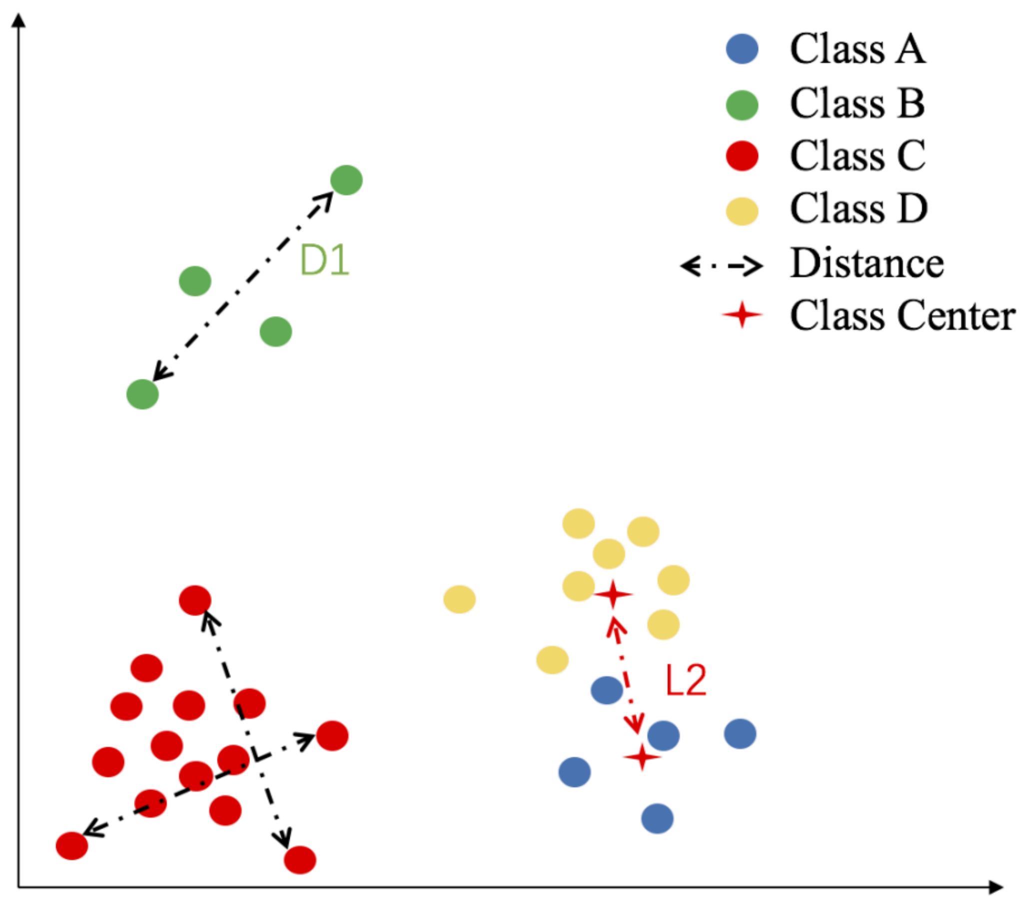
- **The center loss**

$$\mathcal{L}_C = \frac{1}{2}\sum_{i=1}^{m} \|x_i - c_{y_i}\|_2^2$$



Euclidean-distance-based loss

- **The range loss**



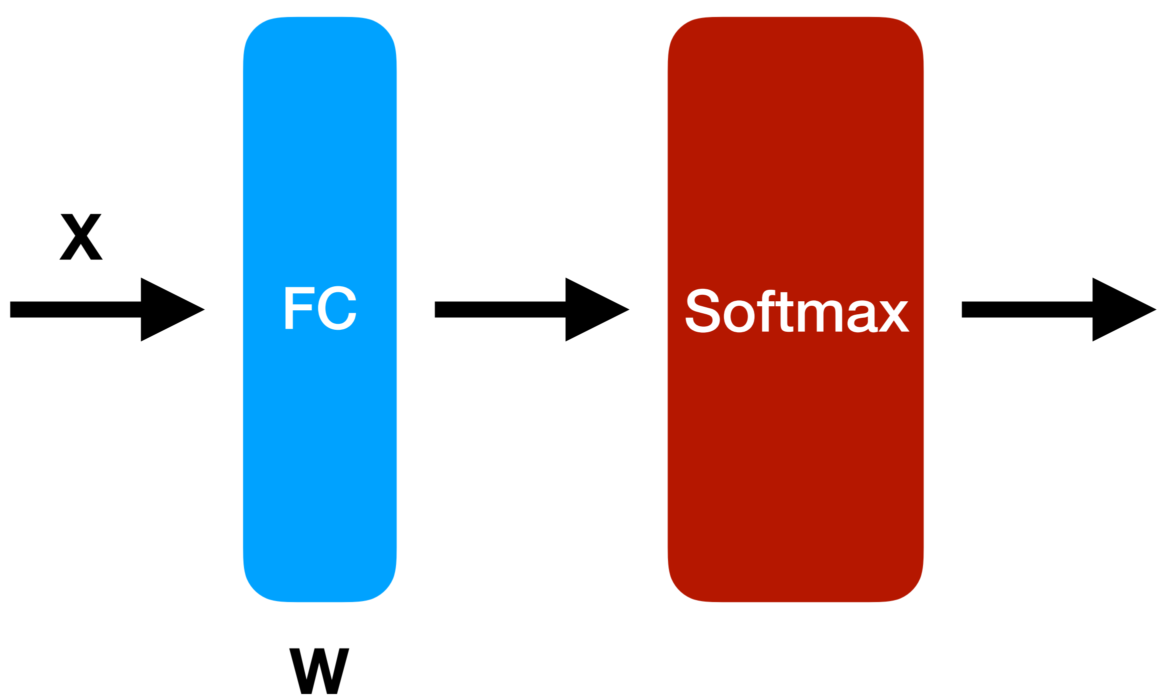
$$\mathcal{L}_R = \alpha \mathcal{L}_{R_{intra}} + \beta \mathcal{L}_{R_{inter}}$$

$$\mathcal{L}_{R_{intra}} = \sum_{i \subseteq I} \mathcal{L}^i_{R_{intra}} = \sum_{i \subseteq I} \frac{k}{\sum_{j=1}^{k} \frac{1}{\mathcal{D}_j}}$$

$$\mathcal{L}_{R_{inter}} = \max(m - \mathcal{D}_{Center}, 0) = \max(m - \|\overline{x}_{\mathcal{Q}} - \overline{x}_{\mathcal{R}}\|_2^2, 0)$$

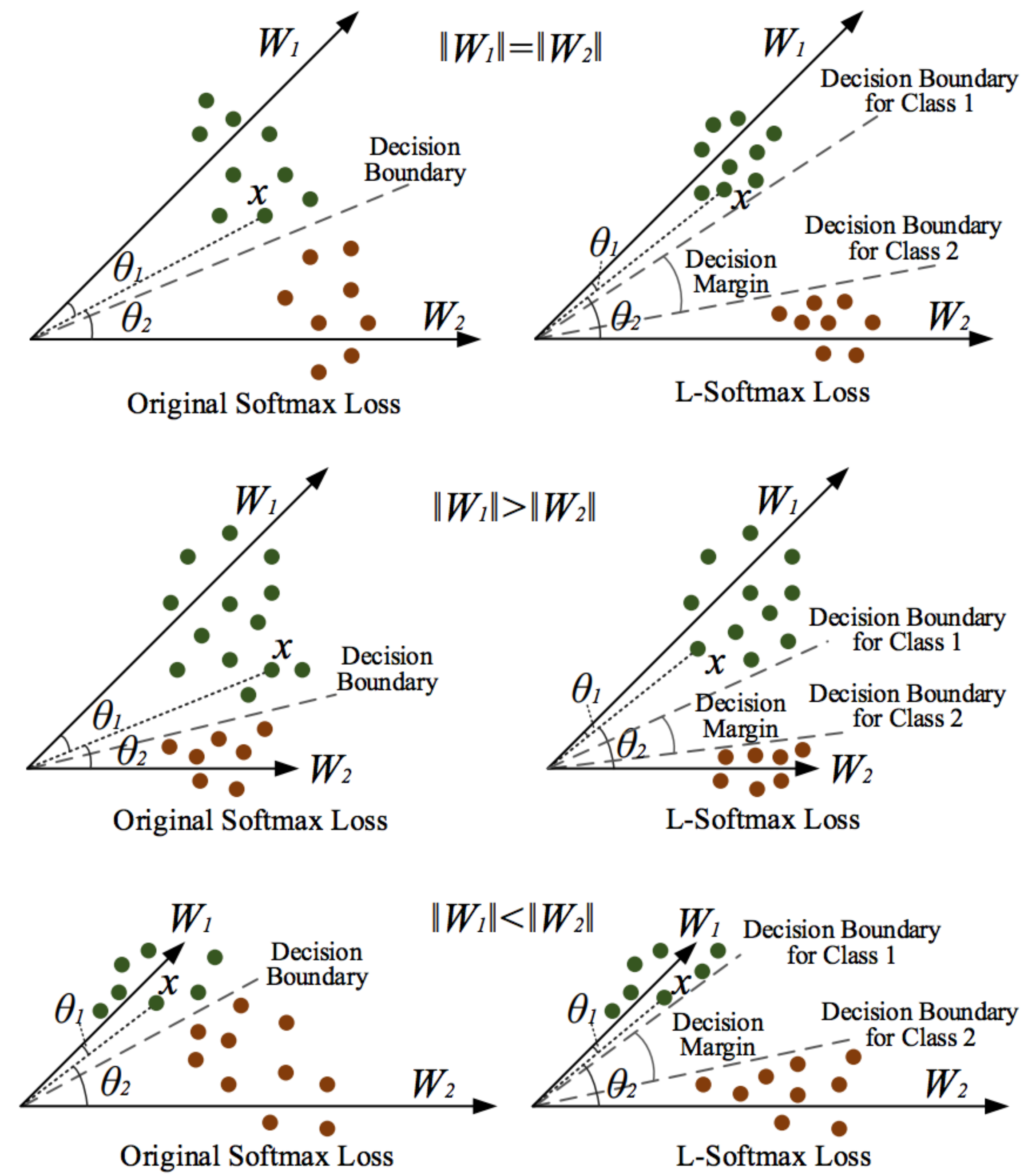
Angular/cosine-margin-based loss

- **L-Softmax**



$$L_i = -\log\left(\frac{e^{\|\boldsymbol{W}_{y_i}\|\|\boldsymbol{x}_i\|\psi(\theta_{y_i})}}{e^{\|\boldsymbol{W}_{y_i}\|\|\boldsymbol{x}_i\|\psi(\theta_{y_i})} + \sum_{j\neq y_i} e^{\|\boldsymbol{W}_j\|\|\boldsymbol{x}_i\|\cos(\theta_j)}}\right)$$

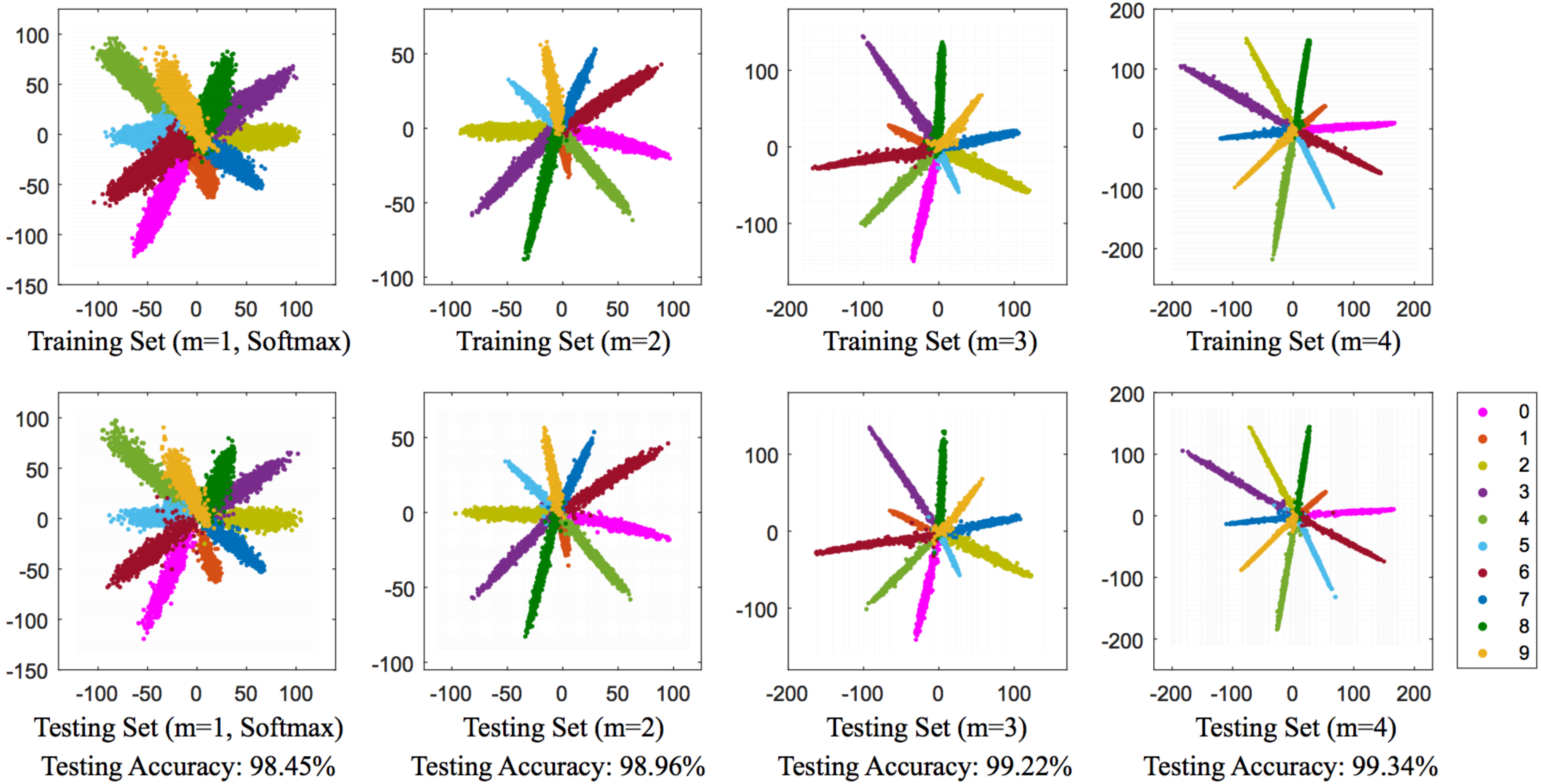
$$\psi(\theta) = (-1)^k \cos(m\theta) - 2k, \quad \theta \in [\frac{k\pi}{m}, \frac{(k+1)\pi}{m}]$$



ref : https://arxiv.org/pdf/1612.02295.pdf

Angular/cosine-margin-based loss

- **L-Softmax**



Angular/cosine-margin-based loss

- **A-Softmax**

$$L_{\text{ang}} = \frac{1}{N}\sum_i -\log\left(\frac{e^{\|\boldsymbol{x}_i\|\psi(\theta_{y_i,i})}}{e^{\|\boldsymbol{x}_i\|\psi(\theta_{y_i,i})} + \sum_{j\neq y_i} e^{\|\boldsymbol{x}_i\|\cos(\theta_{j,i})}}\right)$$

Loss Function	Decision Boundary
Softmax Loss	$(\boldsymbol{W}_1 - \boldsymbol{W}_2)\boldsymbol{x} + b_1 - b_2 = 0$
Modified Softmax Loss	$\|\boldsymbol{x}\|(\cos\theta_1 - \cos\theta_2) = 0$
A-Softmax Loss	$\|\boldsymbol{x}\|(\cos m\theta_1 - \cos\theta_2) = 0$ for class 1 $\|\boldsymbol{x}\|(\cos\theta_1 - \cos m\theta_2) = 0$ for class 2

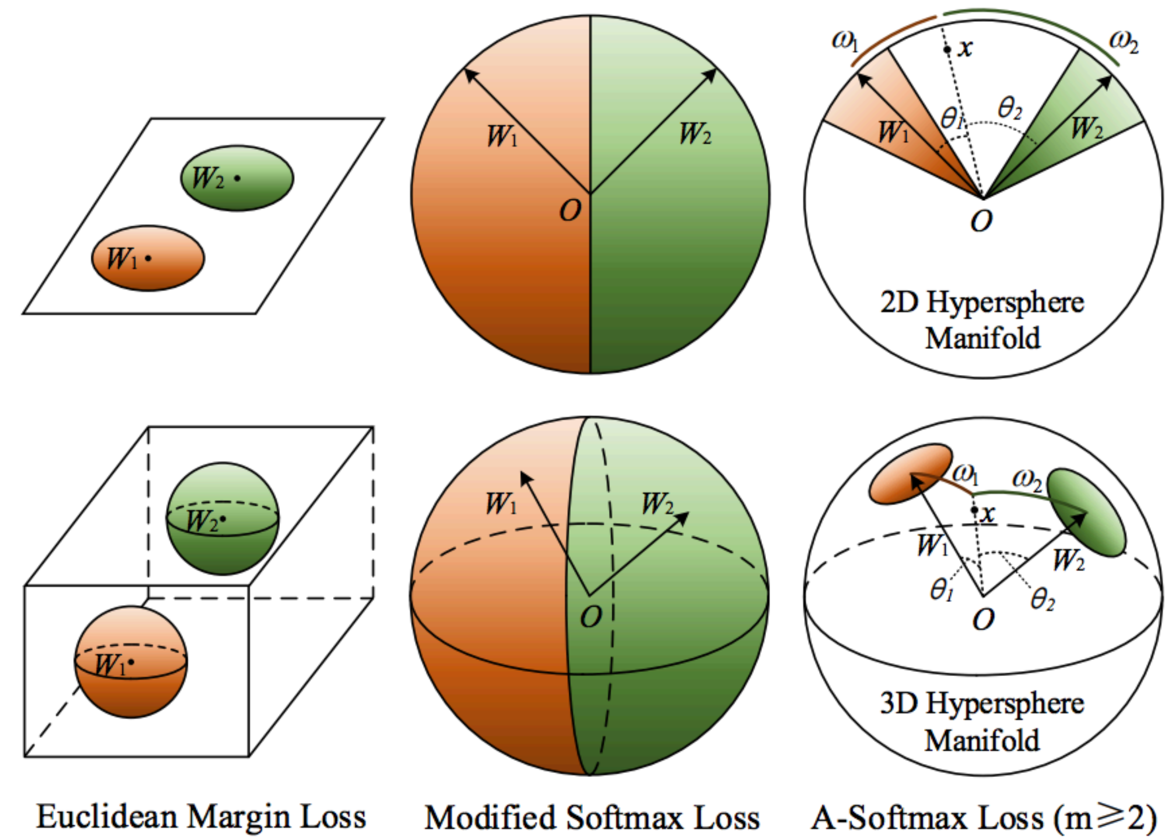
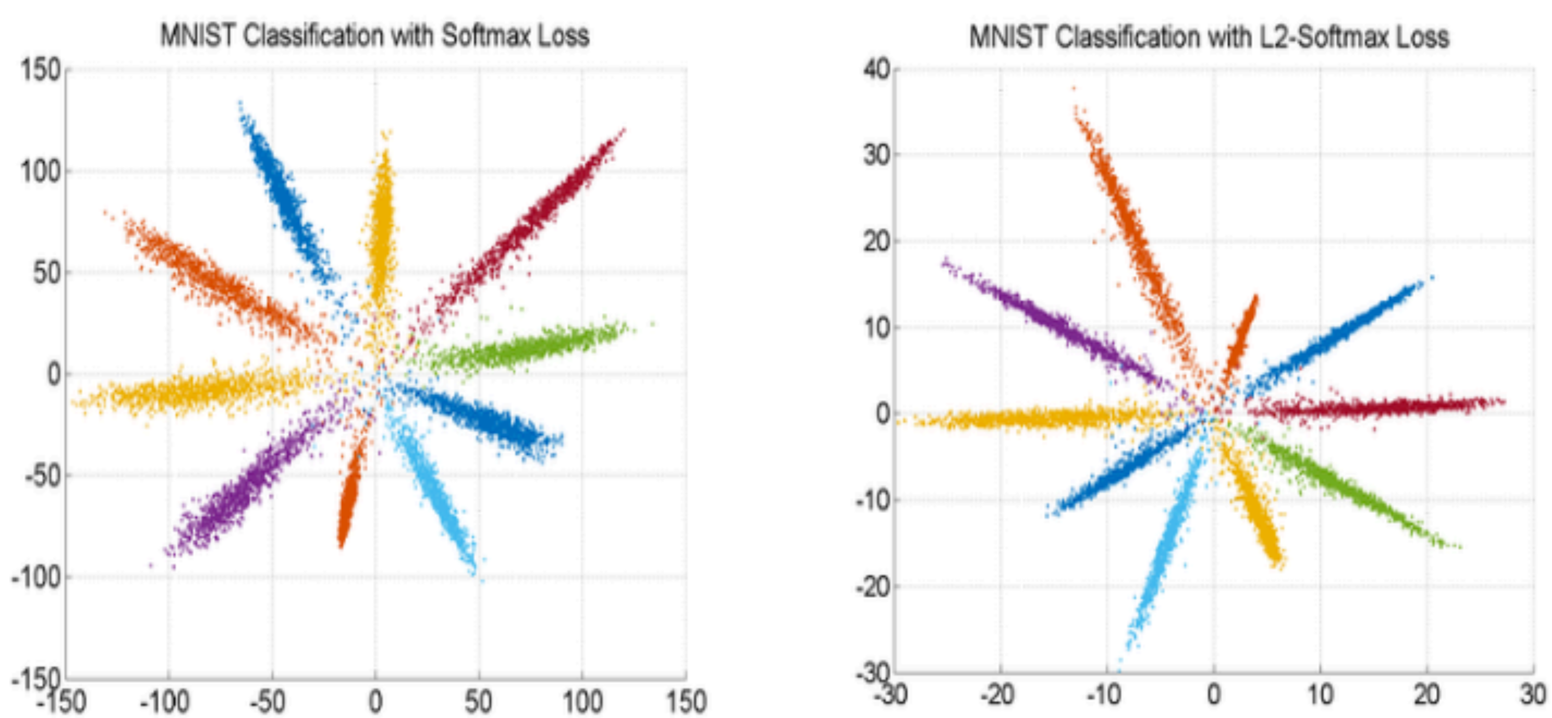


Figure 3: Geometry Interpretation of Euclidean margin loss (e.g. contrastive loss, triplet loss, center loss, etc.), modified softmax loss and A-Softmax loss. The first row is 2D feature constraint, and the second row is 3D feature constraint. The orange region indicates the discriminative constraint for class 1, while the green region is for class 2.

Softmax loss

$$\text{minimize} \quad -\frac{1}{M}\sum_{i=1}^{M}\log\frac{e^{W_{y_i}^T f(\mathbf{x}_i)+b_{y_i}}}{\sum_{j=1}^{C} e^{W_j^T f(\mathbf{x}_i)+b_j}}$$

$$\text{subject to} \quad \|f(\mathbf{x}_i)\|_2 = \alpha, \ \forall i = 1, 2, ...M,$$



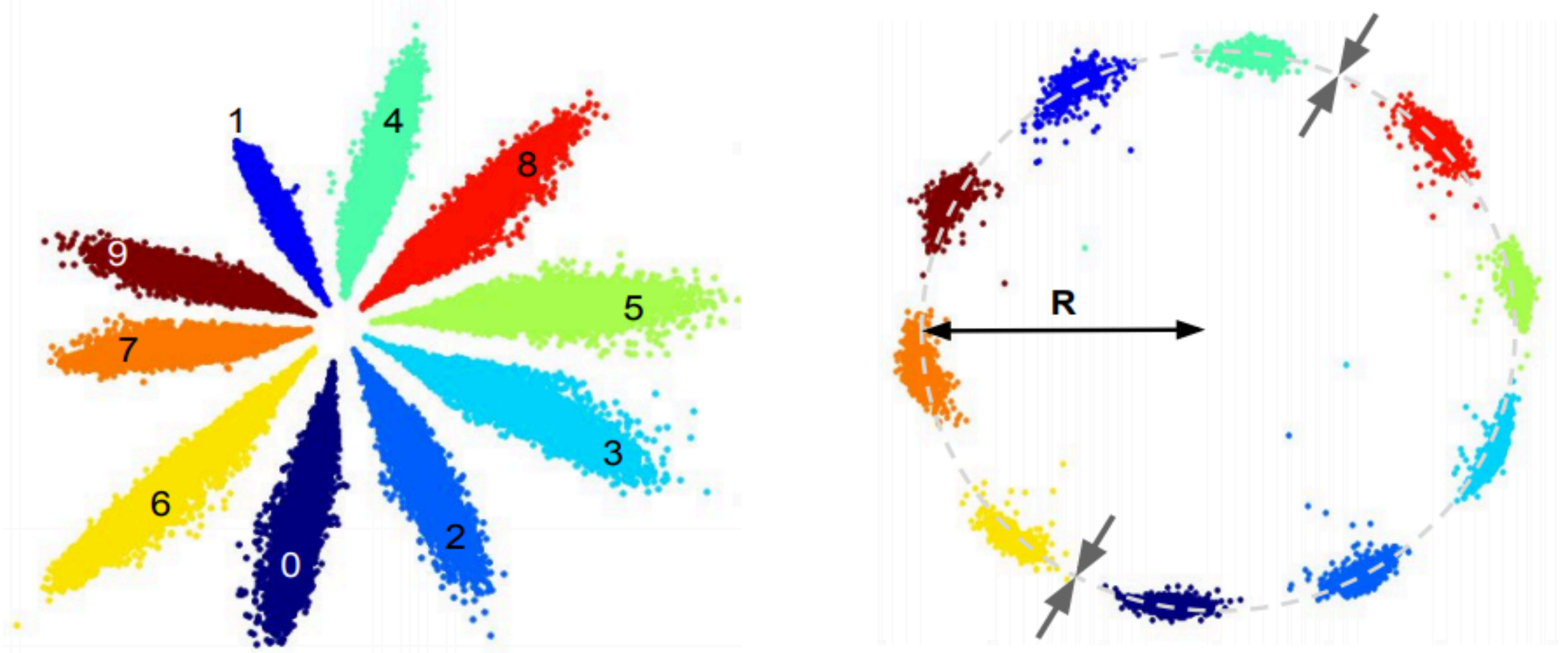
(a) (b)

Figure 3. Vizualization of 2-dimensional features for MNIST digit classification test set using (a) Softmax Loss. (b) L2-Softmax Loss

Softmax loss

- **Ring Softmax**

$$L_R = \frac{\lambda}{2m} \sum_{i=1}^{m} (\|\mathcal{F}(\mathbf{x}_i)\|_2 - R)^2$$



(a) Features trained using Softmax (b) Features trained using Ring loss

Softmax loss

- **Coco (congenerous cosine) loss**

$$\boldsymbol{c}_k = \frac{1}{N_k}\sum_{i\in\mathcal{B}}\delta(l_i,k)\boldsymbol{f}^{(i)}$$

$$\hat{\boldsymbol{c}}_k = \frac{\boldsymbol{c}_k}{\|\boldsymbol{c}_k\|},\ \hat{\boldsymbol{f}}^{(i)} = \frac{\alpha \boldsymbol{f}^{(i)}}{\|\boldsymbol{f}^{(i)}\|},\ p_k^{(i)} = \frac{\exp(\hat{\boldsymbol{c}}_k^T \cdot \hat{\boldsymbol{f}}^{(i)})}{\sum_m \exp(\hat{\boldsymbol{c}}_m^T \cdot \hat{\boldsymbol{f}}^{(i)})},$$

$$\mathcal{L}^{COCO}(\boldsymbol{f}^{(i)},\boldsymbol{c}_k) = -\sum_{i\in\mathcal{B},k} t_k^{(i)}\log p_k^{(i)} = -\sum_{i\in\mathcal{B}}\log p_{l_i}^{(i)},$$