

Face Recognition

- DeepFace
- FaceNet
- SphereFace
- ArcFace

실험은 즐겁게, 리액션은 풍부하게

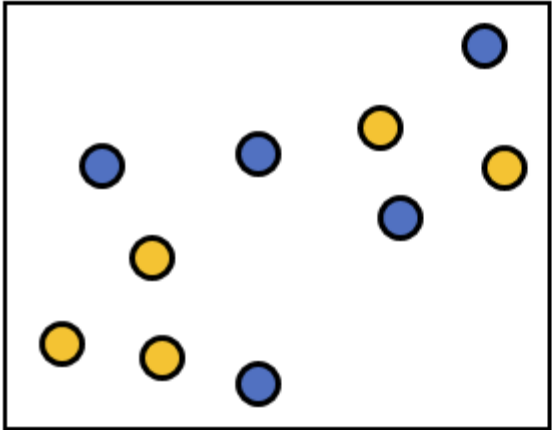
김지성 KIM JI SEONG

얼굴인식은 어떻게 동작 하는걸까?



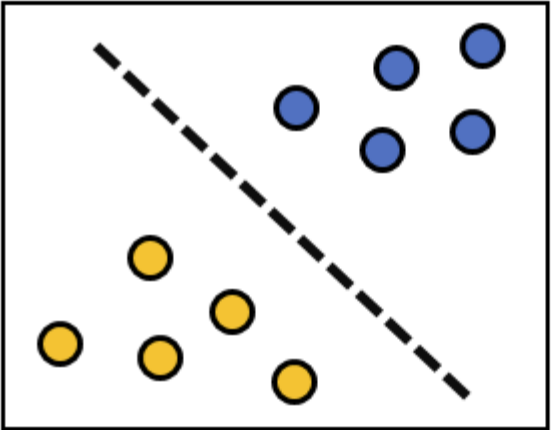
스마트폰은 어떻게 내 얼굴을 몇 초 만에 분석하고 기억할까?

Metric Learning





Original feature space

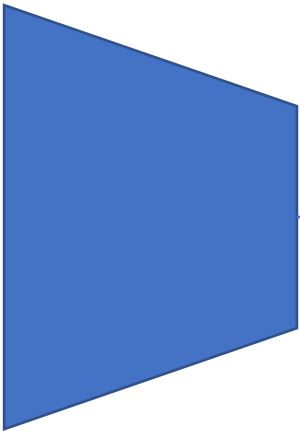
Metric learning



Embedding space

-  Class 1
-  Class 2

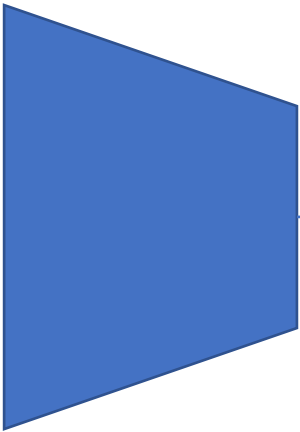
Metric Learning



CNN backbone



FC Layer1
(Embedding layer)



CNN backbone



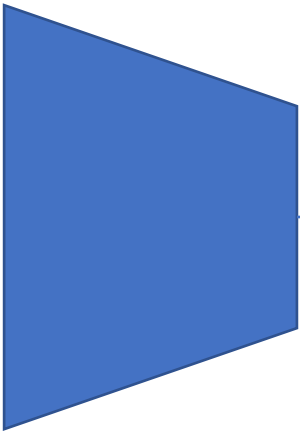
FC Layer1
(Embedding layer)

Vector Distance Calculate

가까운 Distance = 서로 동일 인물
거리가 먼 Distance = 서로 다른 인물

Inference step

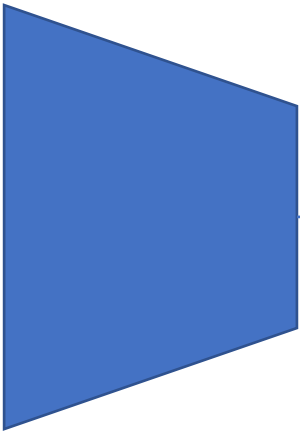
Metric Learning



CNN backbone



FC Layer1
(Embedding layer)



CNN backbone



FC Layer1
(Embedding layer)

일반적인 CNN 구조를 가지지만,
"어떻게" 학습하느냐가 중요한 Task

Vector Distance Calculate

가까운 Distance = 서로 동일 인물
거리가 먼 Distance = 서로 다른 인물

Inference step

Dataset

Angelina Jolie



Robert John Downey Jr



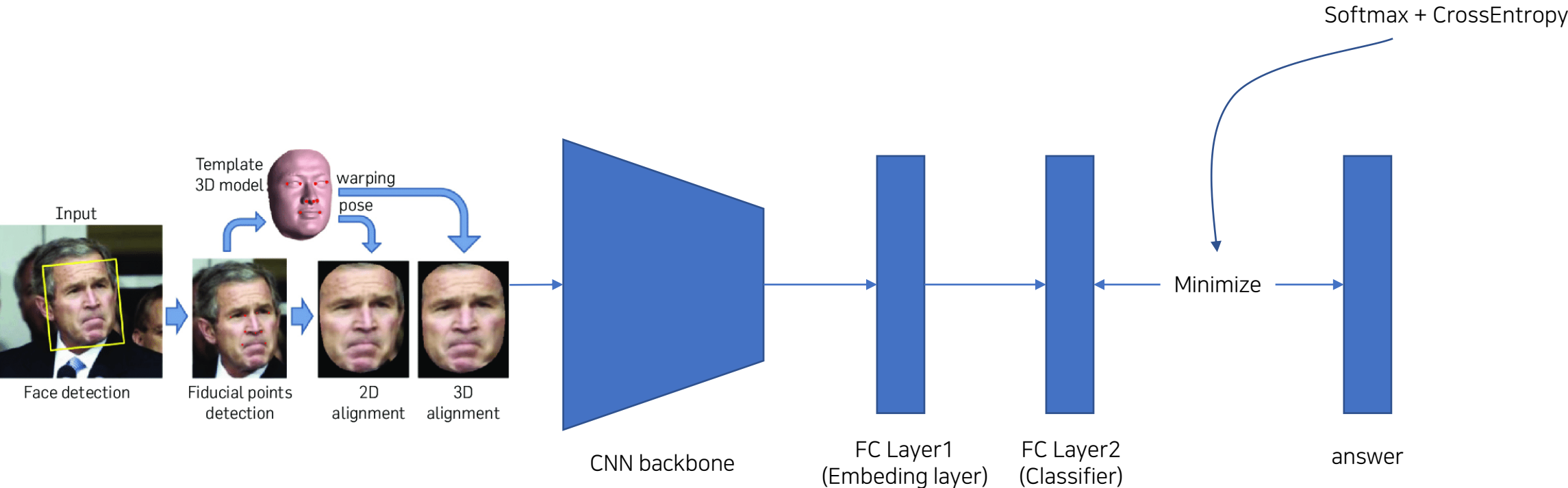
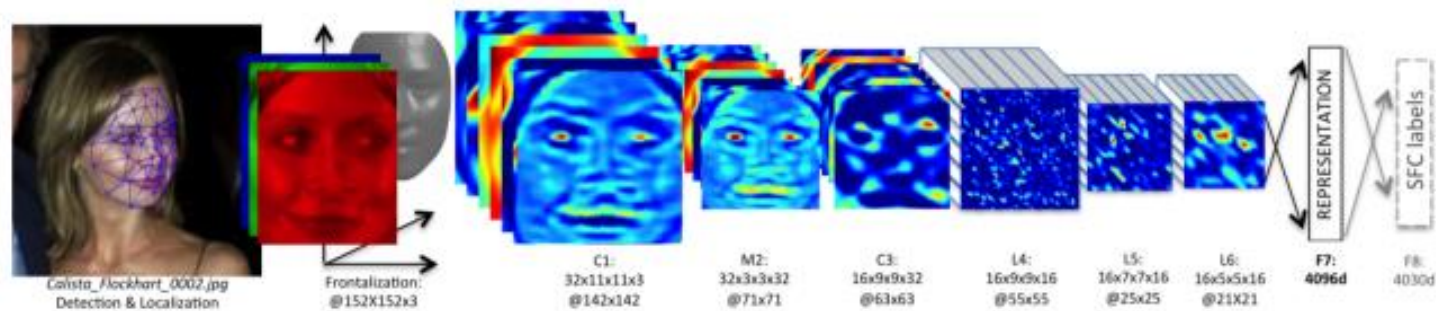
Amanda Seyfried



...

(위 세 인물을 포함해 매우 많은 인물에 대한 사진들)

DeepFace, Taigman et al., 2014

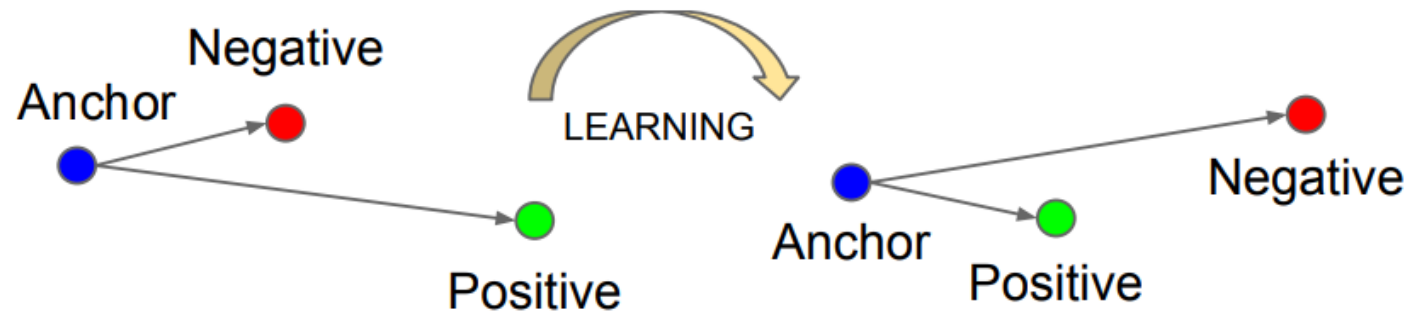


Training step

이 지금까지의 방법론은 얼굴 분류를 학습해서 얼굴 Embedding을 잘 하도록 “유도” 하는데,
“유도”한 것 말고 “직접적으로” 얼굴 Embedding을 잘 하게 만들면 성능이 더 올라가지 않을까?

Triplet : 세 개의 데이터

- Anchor (x_i^a) : 기준 얼굴의 벡터
- Positive (x_i^p) : 기준과 같은 인물의 얼굴의 벡터
- Negative (x_i^n) : 기준과 다른 인물의 얼굴의 벡터

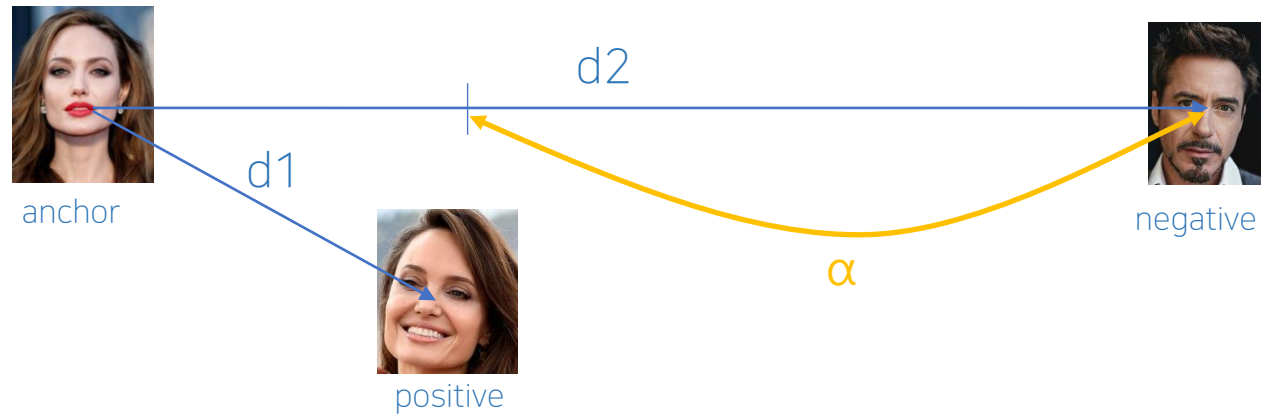


기준 얼굴과 같은 얼굴은 가깝도록, 기준 얼굴과 다른 얼굴은 멀도록

$$\overset{d1}{\|f(x_i^a) - f(x_i^p)\|_2^2} + \alpha < \overset{d2}{\|f(x_i^a) - f(x_i^n)\|_2^2}, \quad (1)$$

- x : 이미지
- $f(x)$: 임베딩 함수
- α : 마진
- x_i^a : 기준 얼굴(anchor) 이미지
- x_i^p : 기준과 같은 인물(positive)의 얼굴 이미지
- x_i^n : 기준과 다른 인물(negative)의 얼굴 이미지

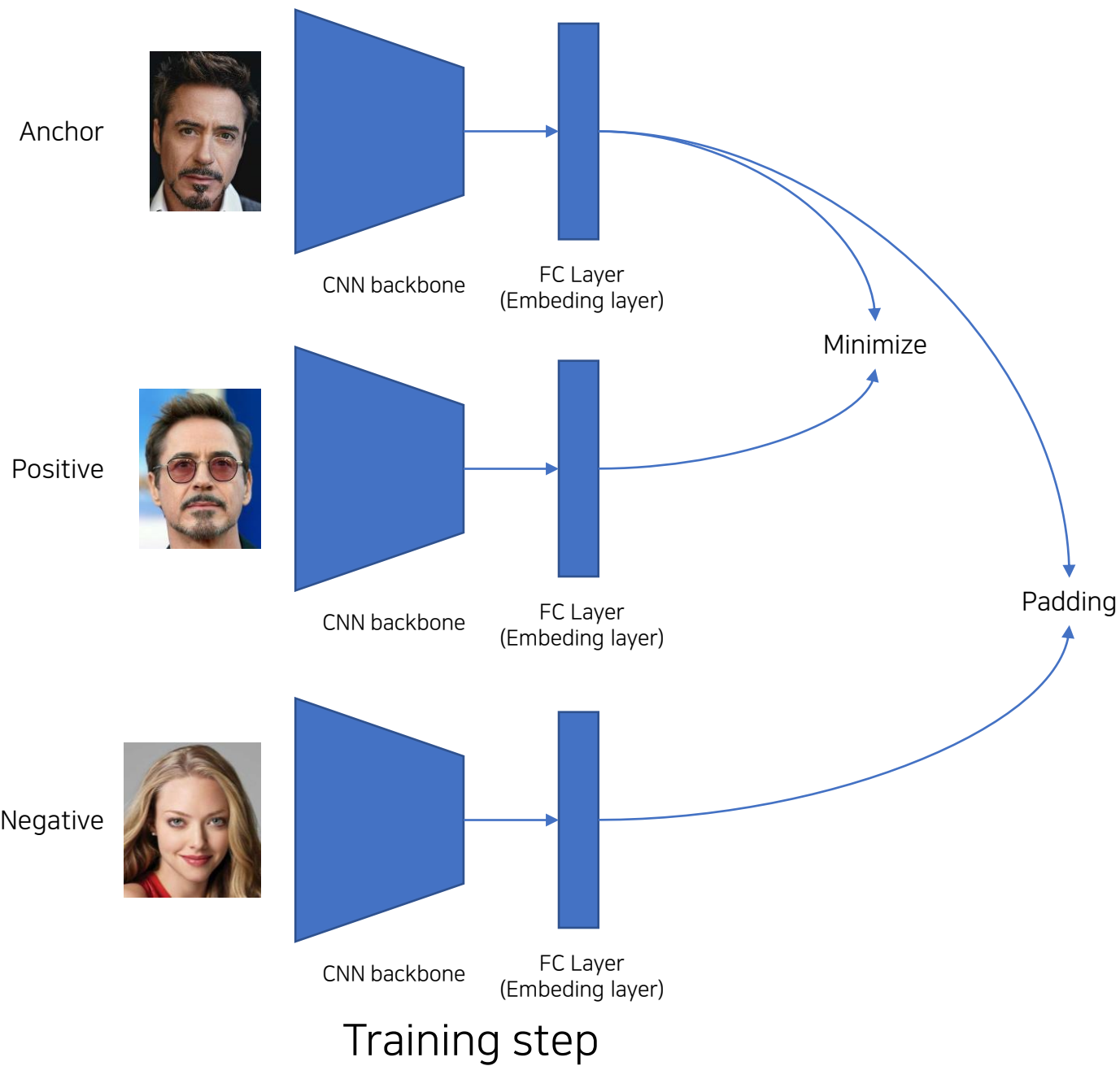
Anchor와 Negative의 거리가 Anchor와 Positive의 거리보다 α 만큼 더 떨어져 있고 싶다!



$$\begin{aligned}\text{LOSS} &= \sum_i^N \left[\|f(x_i^a) - f(x_i^p)\|_2^2 - \|f(x_i^a) - f(x_i^n)\|_2^2 + \alpha \right]_+ . \\ &\hspace{15em} (2) \\ &= \sum_i^N \text{Max}(\|f(x_i^a) - f(x_i^p)\|_2^2 - \|f(x_i^a) - f(x_i^n)\|_2^2 + \alpha, 0)\end{aligned}$$

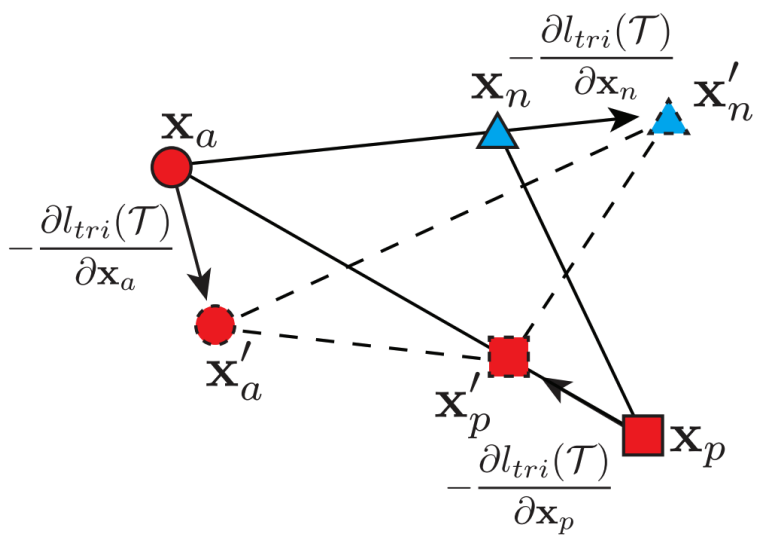
두 명의 사람이 각 10장의 사진을 가질 때 가능한 Triplet의 수 :
Class x Anchor x Positive x Negative = 2 x 10 x 9 x 10 = 1800

FaceNet, Schroff et al., 2014

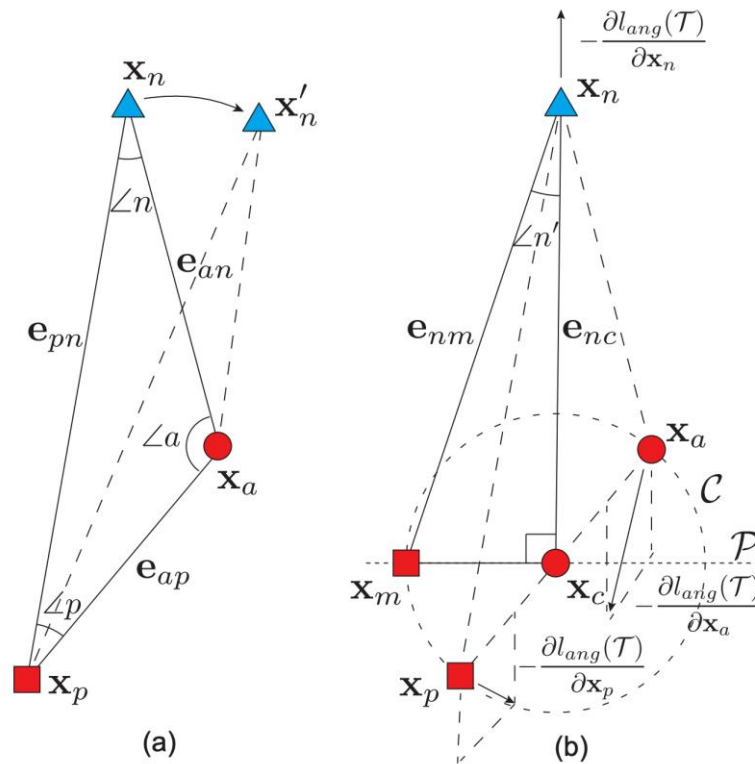


FaceNet, Schroff et al., 2014

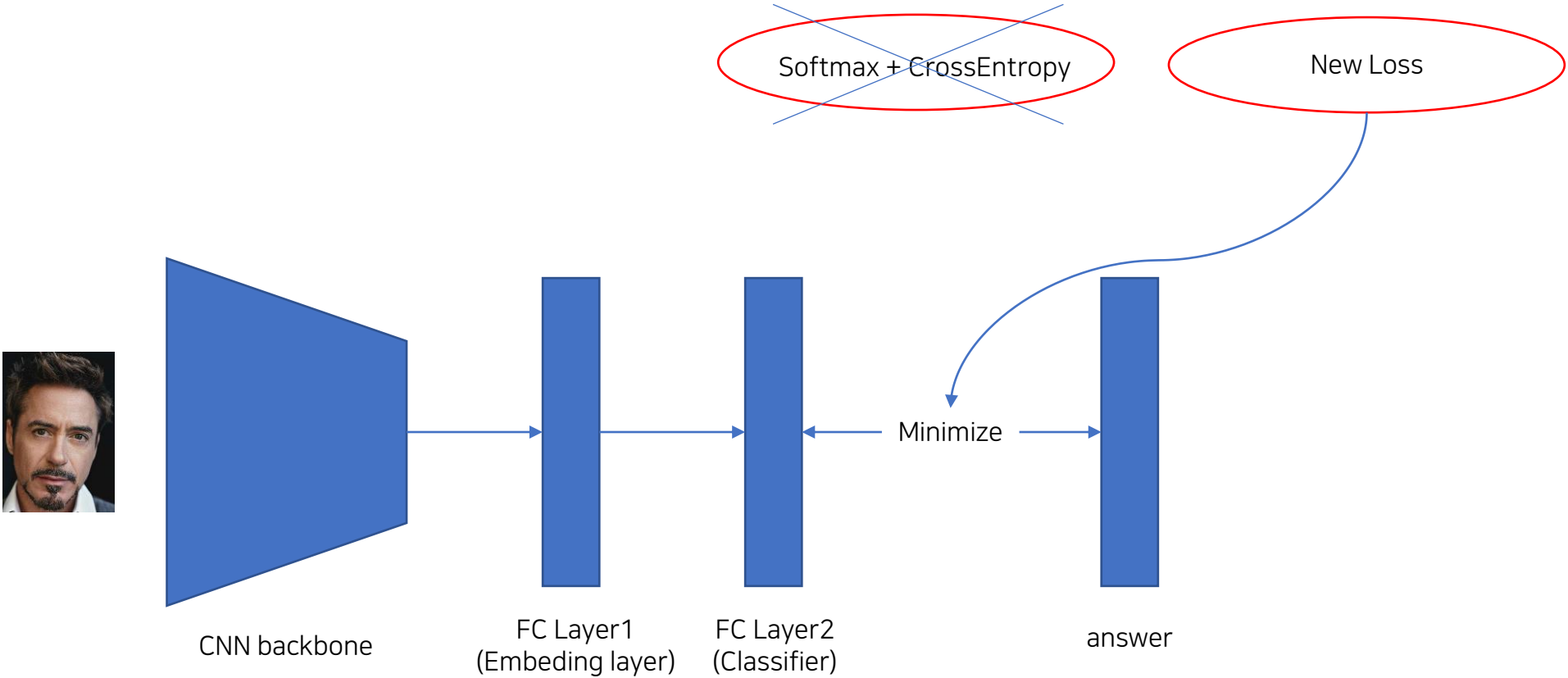




Triplet Loss



Angular Loss



Training step

$$L_i = -\log \left(\frac{e^{\mathbf{W}_{y_i}^T \mathbf{x}_i + \mathbf{b}_{y_i}}}{\sum_j e^{\mathbf{W}_j^T \mathbf{x}_i + \mathbf{b}_j}} \right) = -\log \left(\frac{e^{\|\mathbf{W}_{y_i}\| \|\mathbf{x}_i\| \cos(\theta_{y_i,i}) + \mathbf{b}_{y_i}}}{\sum_j e^{\|\mathbf{W}_j\| \|\mathbf{x}_i\| \cos(\theta_{j,i}) + \mathbf{b}_j}} \right)$$

$$L_{modified} = \frac{1}{N} \sum_i -\log \left(\frac{e^{\|x_i\| \cos \theta_{y_i,i}}}{\sum_j e^{\|x_i\| \cos(\theta_{j,i})}} \right)$$

Margin 추가

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

Margin이 추가된 인덱스에 대해서만 분모에도 margin 추가

벡터의 내적 식으로 변환

$$L_i = -\log \left(\frac{e^{\mathbf{W}_{y_i}^T \mathbf{x}_i + b_{y_i}}}{\sum_j e^{\mathbf{W}_j^T \mathbf{x}_i + b_j}} \right) = -\log \left(\frac{e^{\|\mathbf{W}_{y_i}\| \|\mathbf{x}_i\| \cos(\theta_{y_i,i}) + b_{y_i}}}{\sum_j e^{\|\mathbf{W}_j\| \|\mathbf{x}_i\| \cos(\theta_{j,i}) + b_j}} \right)$$

$$L_{modified} = \frac{1}{N} \sum_i -\log \left(\frac{e^{\|x_i\| \cos \theta_{y_i,i}}}{\sum_j e^{\|x_i\| \cos(\theta_{j,i})}} \right)$$

Margin 추가

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

Margin이 추가된 인덱스에 대해서만 분모에도 margin 추가

L2 Normalization을 사용해
 $\|W\| = 1$ 고정, $b=0$ 고정

벡터의 내적 식으로 변환

$$L_i = -\log \left(\frac{e^{\mathbf{W}_{y_i}^T \mathbf{x}_i + b_{y_i}}}{\sum_j e^{\mathbf{W}_j^T \mathbf{x}_i + b_j}} \right) = -\log \left(\frac{e^{\|W_{y_i}\| \|\mathbf{x}_i\| \cos(\theta_{y_i,i}) + b_{y_i}}}{\sum_j e^{\|W_j\| \|\mathbf{x}_i\| \cos(\theta_{j,i}) + b_j}} \right)$$

$$L_{modified} = \frac{1}{N} \sum_i -\log \left(\frac{e^{\|x_i\| \cos \theta_{y_i,i}}}{\sum_j e^{\|x_i\| \cos(\theta_{j,i})}} \right)$$

Margin 추가

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

Margin이 추가된 인덱스에 대해서만 분모에도 margin 추가

L2 Normalization을 사용해
 $\|W\| = 1$ 고정, $b=0$ 고정

벡터의 내적 식으로 변환

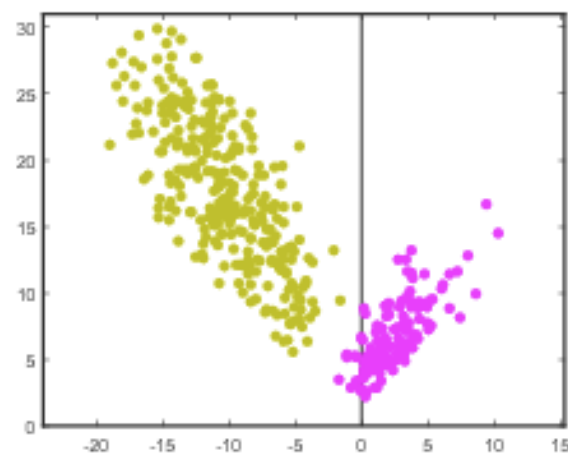
$$L_i = -\log \left(\frac{e^{\mathbf{W}_{y_i}^T \mathbf{x}_i + b_{y_i}}}{\sum_j e^{\mathbf{W}_j^T \mathbf{x}_i + b_j}} \right) = -\log \left(\frac{e^{\|W_{y_i}\| \|\mathbf{x}_i\| \cos(\theta_{y_i,i}) + b_{y_i}}}{\sum_j e^{\|W_j\| \|\mathbf{x}_i\| \cos(\theta_{j,i}) + b_j}} \right)$$

$$L_{modified} = \frac{1}{N} \sum_i -\log \left(\frac{e^{\|x_i\| \cos \theta_{y_i,i}}}{\sum_j e^{\|x_i\| \cos(\theta_{j,i})}} \right)$$

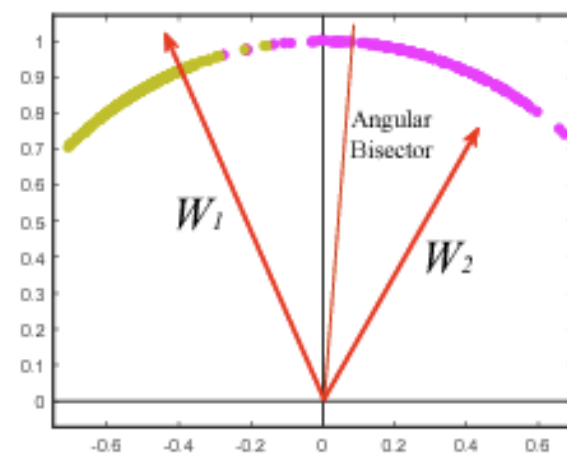
Margin 추가

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

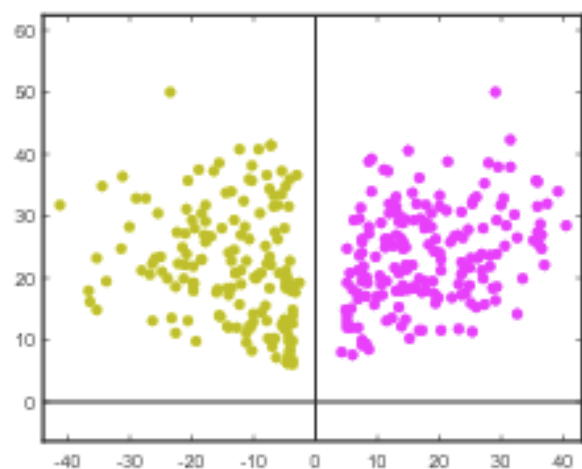
Margin이 추가된 인덱스에 대해서만 분모에도 margin 추가



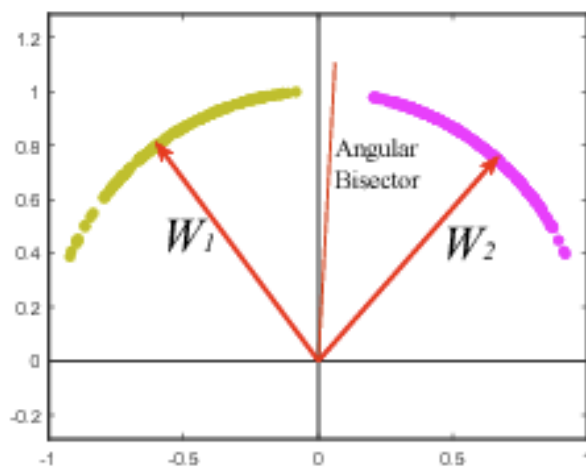
(a) Original Softmax Loss



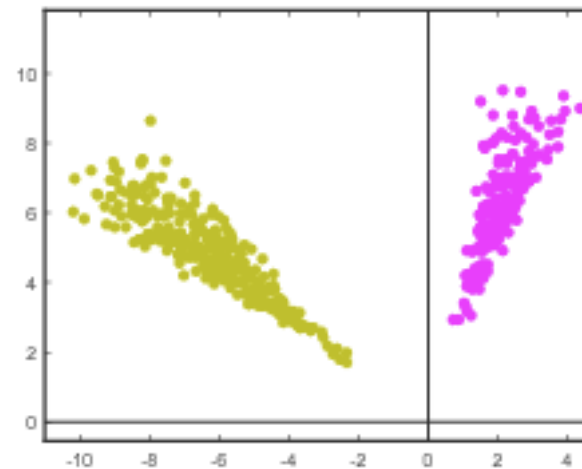
(b) Original Softmax Loss



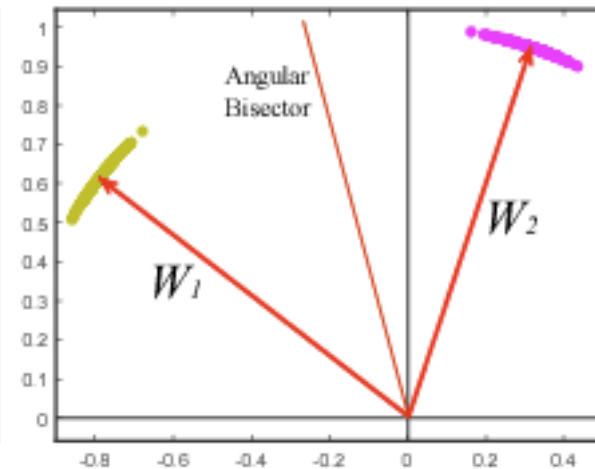
(c) Modified Softmax Loss



(d) Modified Softmax Loss



(e) A-Softmax Loss



(f) A-Softmax Loss

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

SphereFace – Angular loss

Margin
곱->합 으로 변경

$$L = -\frac{1}{N} \sum_{i=1}^N \log \left(\frac{e^{s(\cos(\theta_{y_i} + m))}}{e^{s(\cos(\theta_{y_i} + m))} + \sum_{j=1, j \neq y+i}^n e^{s \cos(\theta_j)}} \right)$$

ArcFace – ArcFace loss

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

SphereFace – Angular loss

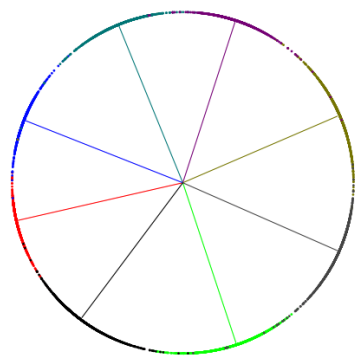
L2 Normalization을 사용해
 $\|x\| = 1$ 고정

Margin
곱->합 으로 변경

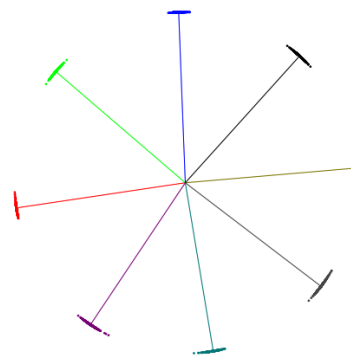
$$L = -\frac{1}{N} \sum_{i=1}^N \log \left(\frac{e^{s(\cos(\theta_{y_i} + m))}}{e^{s(\cos(\theta_{y_i} + m))} + \sum_{j=1, j \neq y+i}^n e^{s \cos(\theta_j)}} \right)$$

ArcFace – ArcFace loss

scaling factor 추가



(a) Softmax



(b) ArcFace

Figure 3. Toy examples under the softmax and ArcFace loss on 8 identities with 2D features. Dots indicate samples and lines refer to the centre direction of each identity. Based on the feature normalisation, all face features are pushed to the arc space with a fixed radius. The geodesic distance gap between closest classes becomes evident as the additive angular margin penalty is incorporated.

Method	#Image	LFW	YTF
DeepID [32]	0.2M	99.47	93.20
Deep Face [33]	4.4M	97.35	91.4
VGG Face [24]	2.6M	98.95	97.30
FaceNet [29]	200M	99.63	95.10
Baidu [16]	1.3M	99.13	-
Center Loss [38]	0.7M	99.28	94.9
Range Loss [46]	5M	99.52	93.70
Marginal Loss [9]	3.8M	99.48	95.98
SphereFace [18]	0.5M	99.42	95.0
SphereFace+ [17]	0.5M	99.47	-
CosFace [37]	5M	99.73	97.6
MS1MV2, R100, ArcFace	5.8M	99.83	98.02

Table 4. Verification performance (%) of different methods on LFW and YTF.



이 사람들 서로 페이스아이디 해제
되는지 궁금하다 누가 자리 좀
마련해주세요



실험 데이터

강형욱



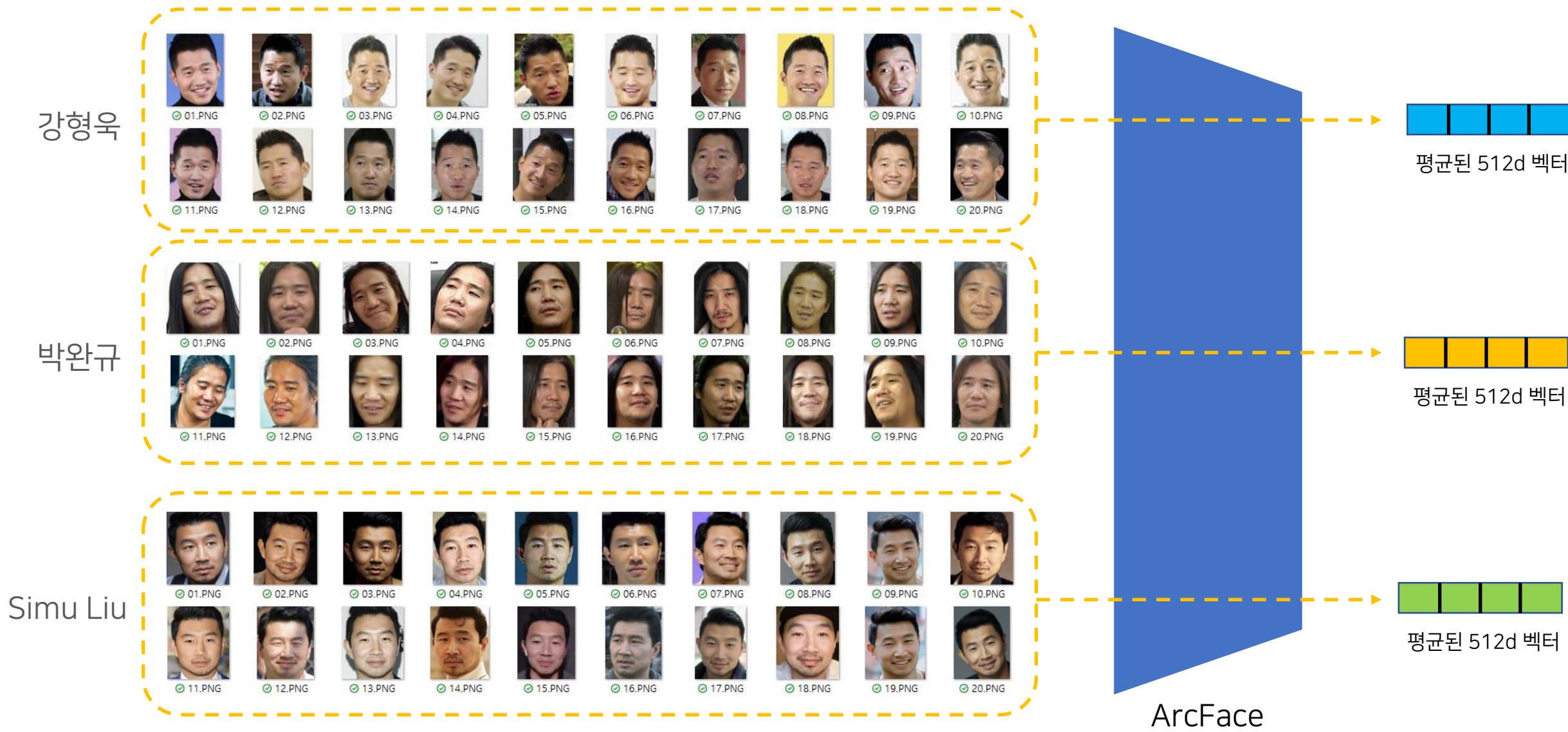
박완규

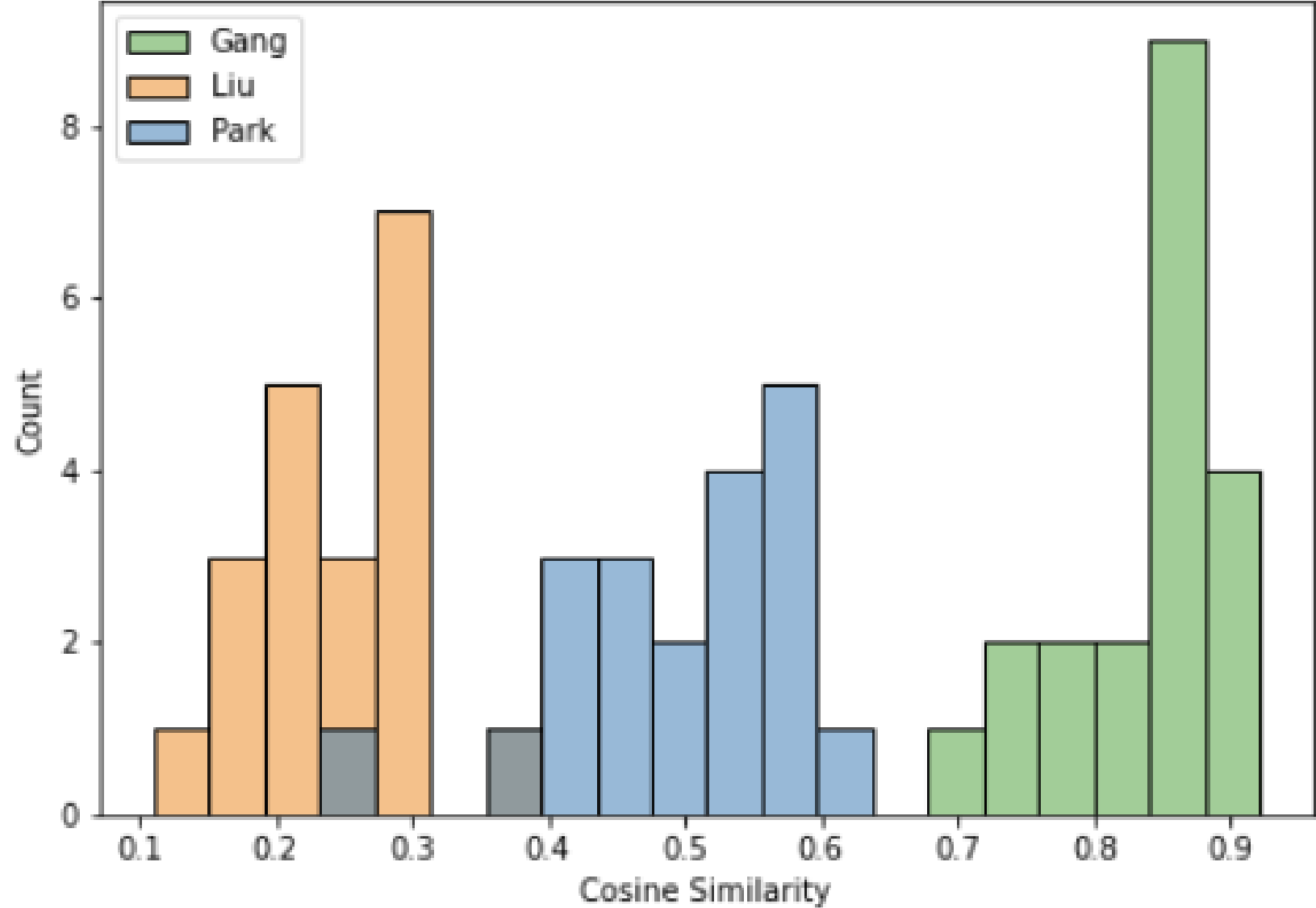


Simu Liu

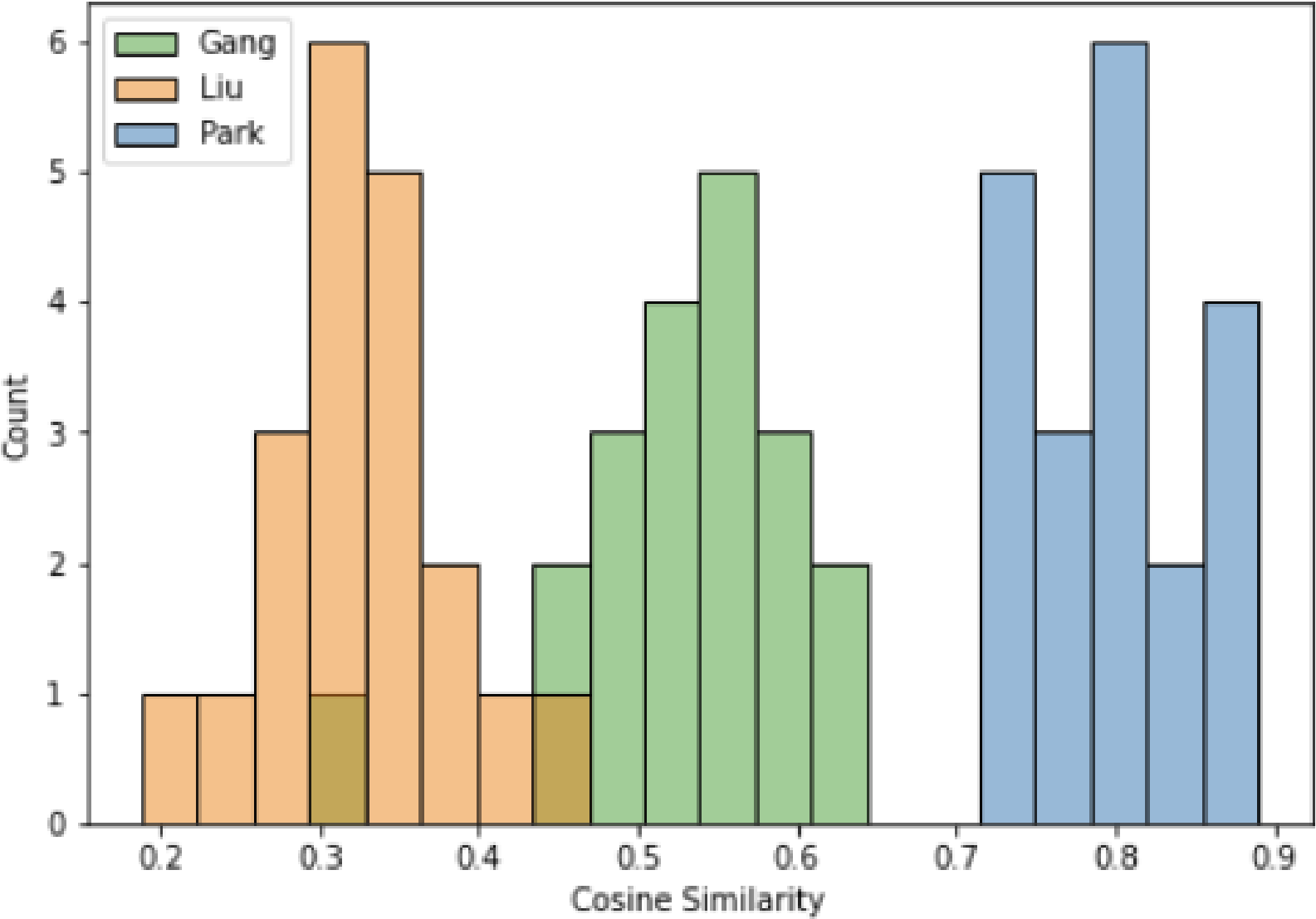


실험

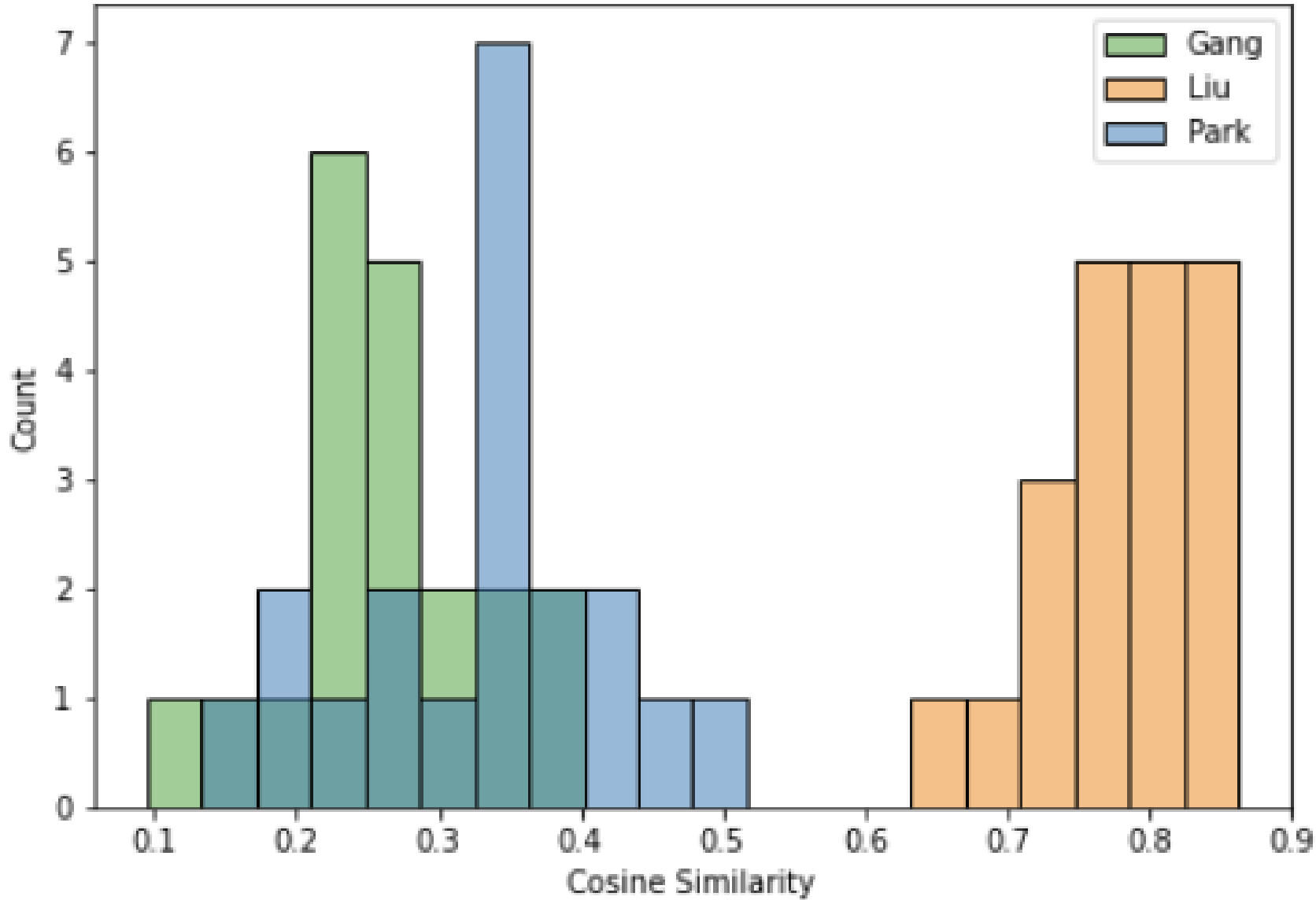




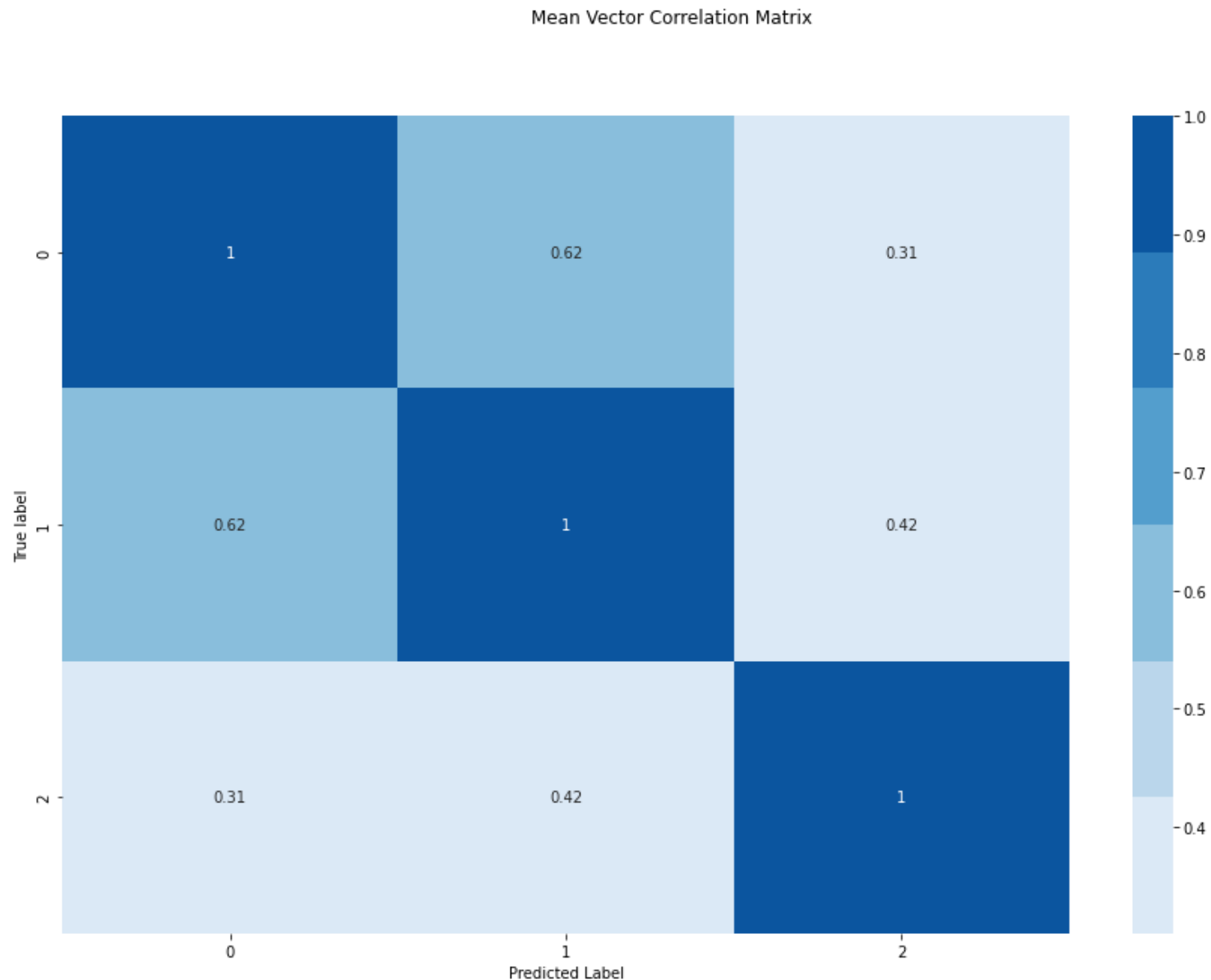
강형욱 평균벡터 기준으로 이미지 전체에 대해 거리측정한 결과의 히스토그램



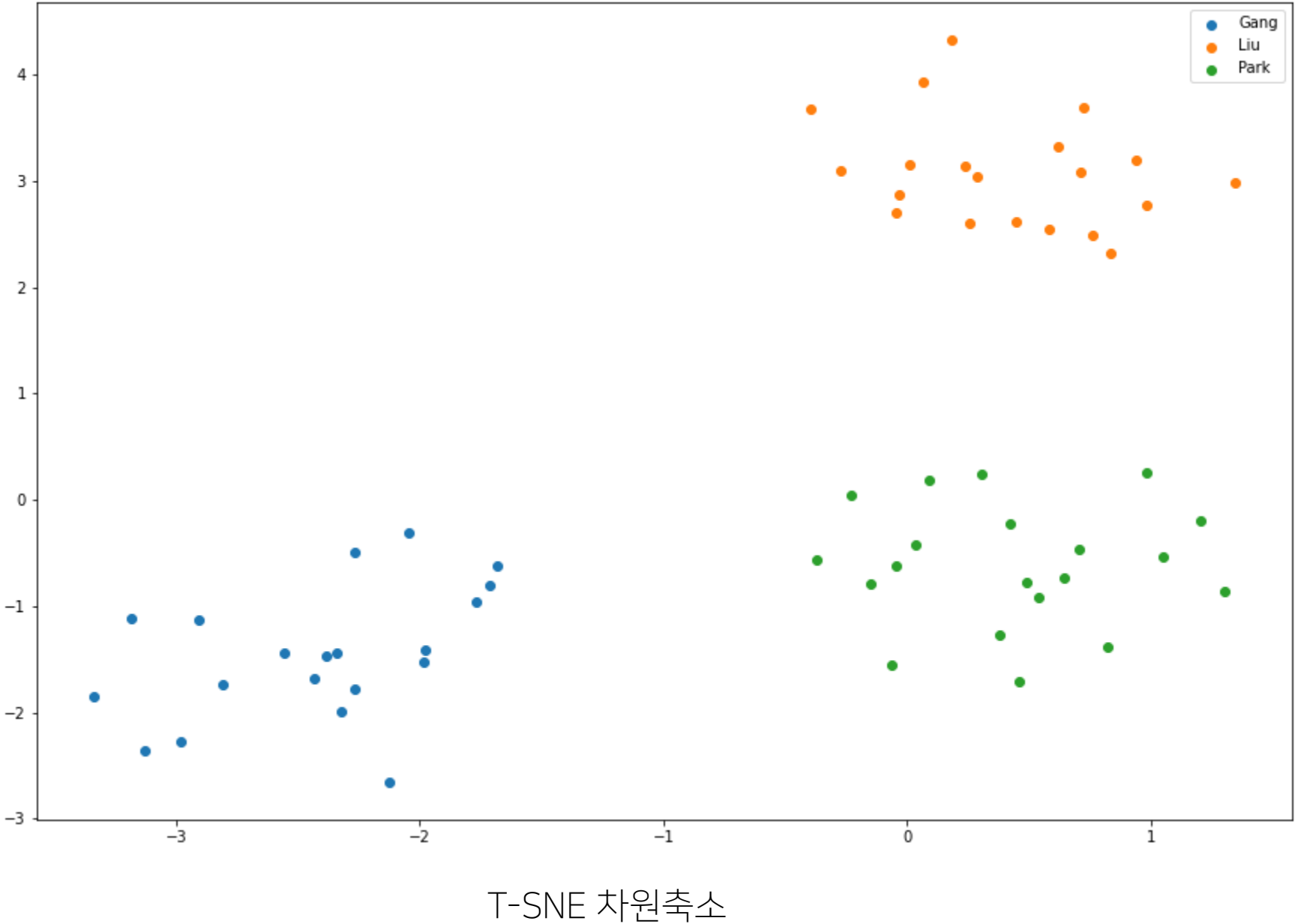
박완규 평균벡터 기준으로 이미지 전체에 대해 거리측정한 결과의 히스토그램



Liu 평균벡터 기준으로 이미지 전체에 대해 거리측정한 결과의 히스토그램



평균벡터간 Cosine 거리 상관관계



이 사람들 서로 페이스아이디 해제
되는지 궁금하다 누가 자리 좀
마련해주세요

