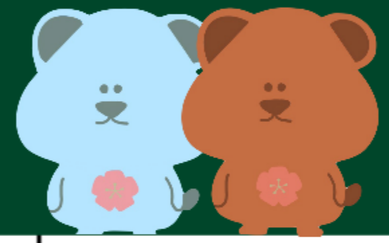
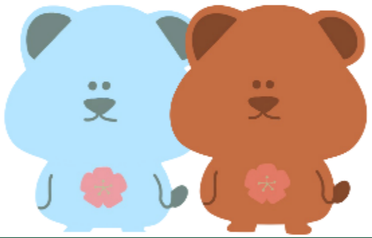


ArcFace: Additive Angular Margin Loss for Deep Face Recognition

고급심화 차수빈

EWHA EURON

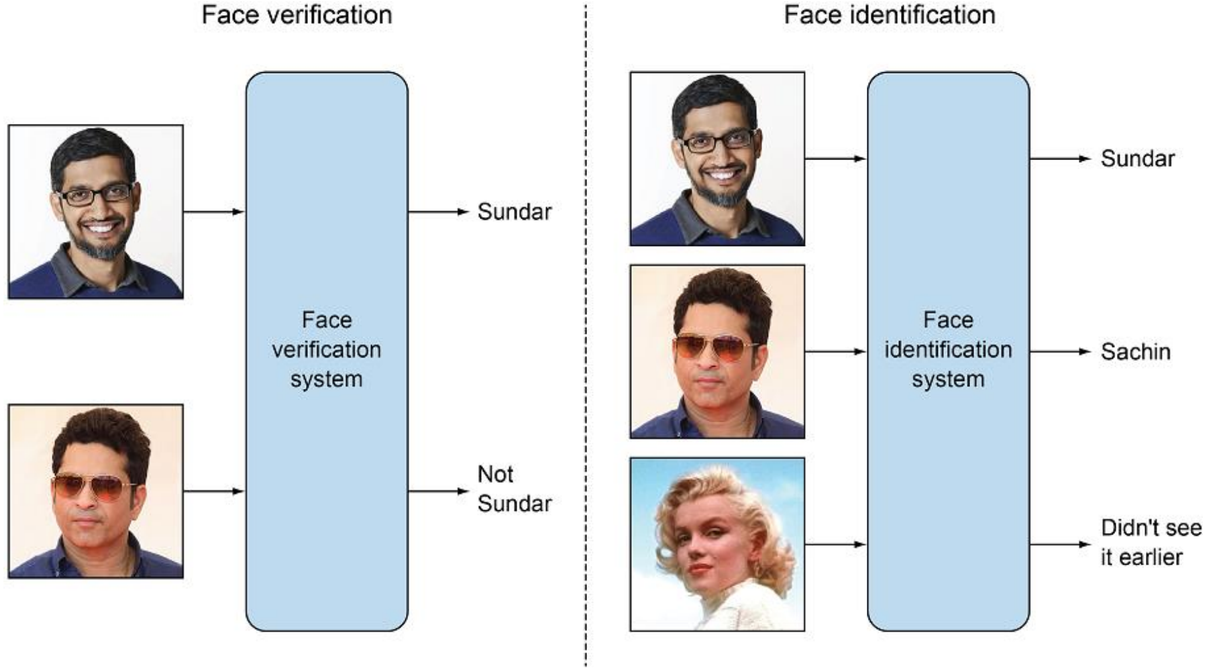
0. Background



EWHA EURON

#0. Background

- Face Recognition
  - 얼굴을 포함하는 입력 정지 영상 또는 비디오에 대해 얼굴 영역의 자동적인 검출 및 분석을 통해 해당 얼굴이 어떤 인물인지 판별해 내는 기술
  - **얼굴검증**과 **얼굴식별** 기술로 구분할 수 있음
    - 얼굴검증(Face Verification): 입력으로 들어오는 두 개의 얼굴 영상이 동일 인물인지 여부를 판단하는 1:1 검증 문제(-> binary classification)
    - 얼굴식별(Face Identification): 입력으로 들어오는 하나의 얼굴 영상이 사전에 등록된 N명의 인물 중 어떤 인물에 해당하는지 판단하는 1:N 검증 문제



#0. Background

- **Face Recognition**
  - Image Classification 방식으로 Face recognition을 하려면 엄청난 수의 데이터를 수집해야 함
  - 또한 기존 학습 데이터에 없는 새로운 사람이 추가될 경우 네트워크를 다시 학습해야 하는 문제가 발생

  => **Similarity Learning**을 활용

DB에 보유한 class의 대표 이미지와 입력된 input 이미지 간의
distance를 구하고 이 **distance**를 가지고 어떤 사람인지를 판단

- 전처리를 거친 이미지를 CNN 네트워크에 통과시켜 embedding을 구하고, 출력된 feature vector들 간의 거리(margin)를 구함
- 같은 얼굴이면 distance가 작게, 다른 얼굴이면 distance를 크게 나오도록 네트워크를 학습시키는 것이 중요

#0. Background

- **Face Recognition**
  🤔 서로 다른 얼굴 이미지를 잘 구분하기 위해 feature representation extraction을 어떻게 수행해야 할까?
  - 단순 softmax loss

$$L_1 = -\frac{1}{N}\sum_{i=1}^{N}\log\frac{e^{W_{y_i}^T x_i + b_{y_i}}}{\sum_{j=1}^{n} e^{W_j^T x_i + b_j}}$$

- $X_i \in \mathbb{R}^d$: feature of the ith sample(y_i 클래스에 속하는 샘플)
- d: 임베딩 차원(여기서는 512로 고정)
- $W_j \in \mathbb{R}^d$: 가중치(weight)
- $b_j \in \mathbb{R}^n$: 편향(bias)

- 해당 softmax loss는 몇 가지 문제점이 존재
  - 클래스 간 representation vector의 margin이 작음
    ⇒ 클래스 내 샘플 간의 높은 유사성을 강제
  - 특징 임베딩을 직접적으로 최적화하지는 않음
    ⇒ 대규모 데이터셋에 대해 잘 대응하지 못함

- 단순 softmax loss의 단점을 해결하고자 여러 가지 loss들이 제안됨
  ⇒ Triplet Loss: 거리 개념 도입
  ⇒ Angular Margin Loss: 각도 개념 도입

#0. Background

참고논문) [FaceNet: A Unified Embedding for Face Recognition and Clustering]

- **Face Recognition**

  🤔 서로 다른 얼굴 이미지를 잘 구분하기 위해 feature representation extraction을 어떻게 수행해야 할까?

  - Triplet Loss

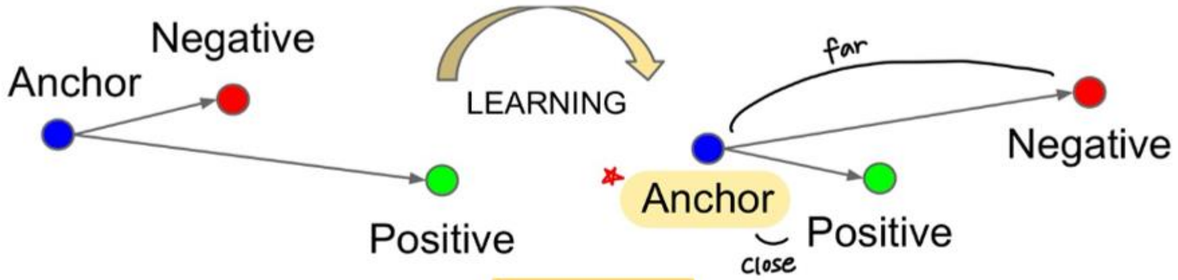
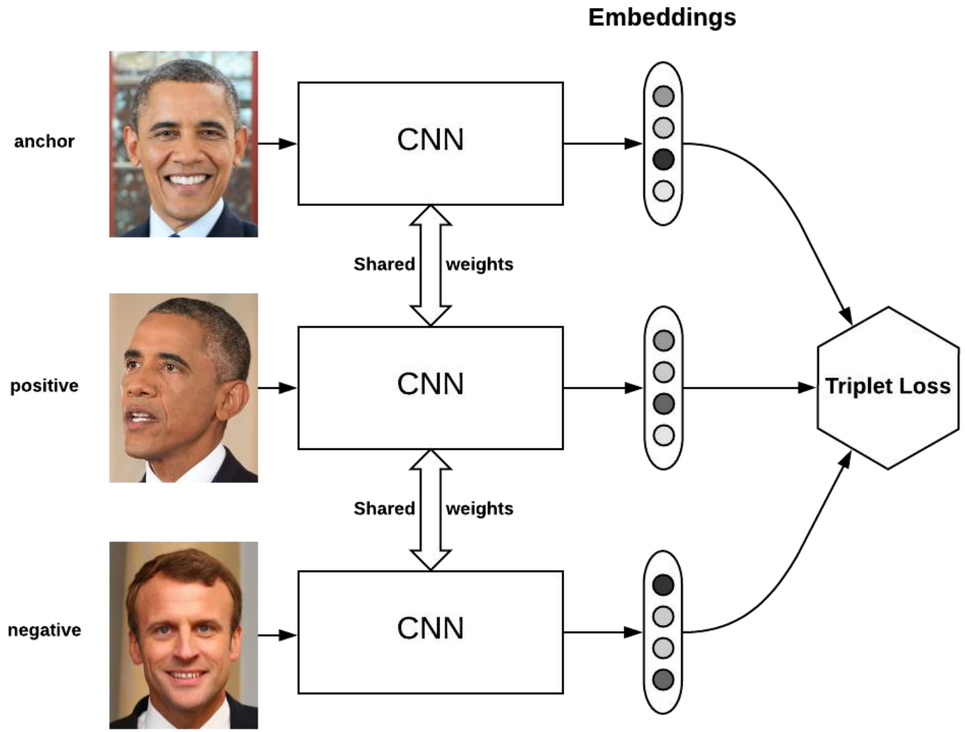


Figure 3. The **Triplet Loss** minimizes the distance between an *anchor* and a *positive*, both of which have the same identity, and maximizes the distance between the *anchor* and a *negative* of a different identity.



#0. Background

참고논문) [FaceNet: A Unified Embedding for Face Recognition and Clustering]

- **Face Recognition**

  🤔 서로 다른 얼굴 이미지를 잘 구분하기 위해 feature representation extraction을 어떻게 수행해야 할까?

  - Triplet Loss

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

anchor positive margin anchor negative

가까워져도록!

$$\forall_{\text{all}} (f(x_i^a), f(x_i^p), f(x_i^n)) \in \mathcal{T} . \quad (2)$$

triplet loss의 main idea

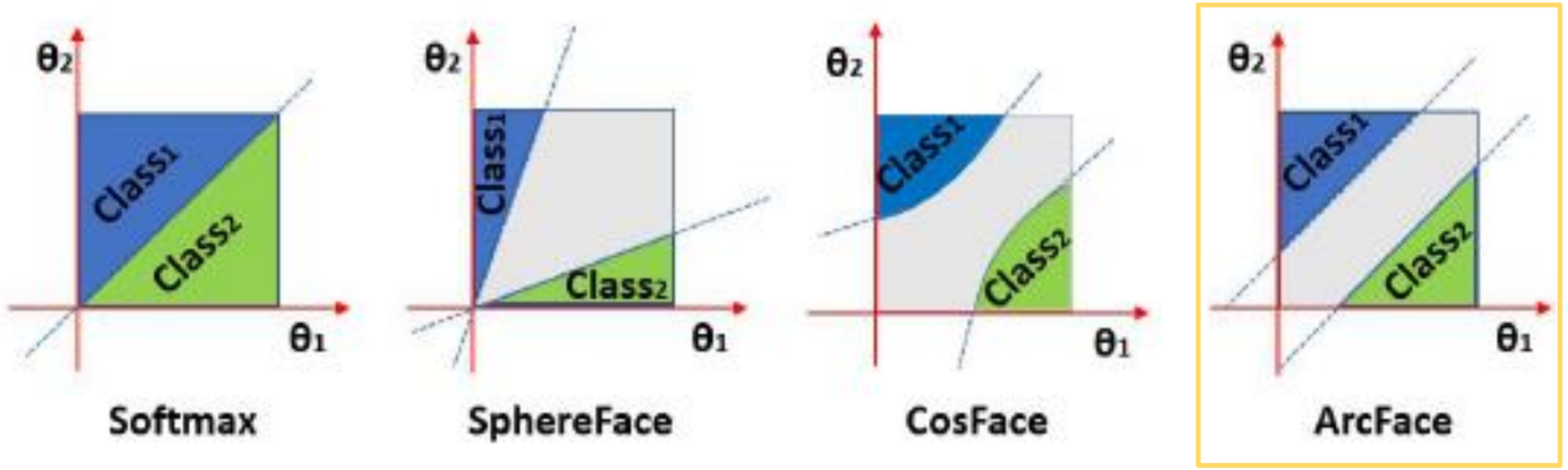
$$\sum_i^N \left[\|f(x_i^a) - f(x_i^p)\|_2^2 - \|f(x_i^a) - f(x_i^n)\|_2^2 + \alpha \right]_+ .$$

triplet loss 계산식

EWHA EURON

#0. Background

- **Face Recognition**
  🤔 서로 다른 얼굴 이미지를 잘 구분하기 위해 feature representation extraction을 어떻게 수행해야 할까?
  - Angular Margin Loss
    - Triplet Loss 또한 Triplet selection의 문제가 남아 있음
      ⇒ 모델의 학습에 확실하게 영향을 주는 hard triplet을 어떻게 설정할까
    - 각도 개념으로 각 representation vector 간의 margin을 정의해 보자!
      ⇒ pair 없이 학습 진행 가능
      ⇒ 클래스 내(intra)에서는 좀 더 멀어지게끔, 클래스 간(inter)에는 좀 더 밀집하도록



#0. Background

참고논문) SphereFace: Deep Hypersphere Embedding for Face Recognition

- **Face Recognition**

  🤔 서로 다른 얼굴 이미지를 잘 구분하기 위해 feature representation extraction을 어떻게 수행해야 할까?

  - SphereFace
    - multiplicative angular margin penalty를 통해 클래스 내 조밀성과 클래스 간 다양성을 달성하고자 함

      ⇒ FC Layer의 weight(W)과 hidden layer의 vector(x) 간의 각도 차이를 증가시킴
    - 갑.분 각도?

      ⇒ softmax loss에서 W_j가 정규화 되어있고 bias가 0이라고 가정해보자!

      ㄴ 둘 간의 각도로써 decision boundary를 표현할 수 있음

      ⇒ 해당 각도를 더욱 크게 벌려주도록 penalty를 주어 학습시켜보자

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

Table 1: Comparison of decision boundaries in binary case. Note that, θ_i is the angle between $\boldsymbol{W}_i$ and $\boldsymbol{x}$.

#0. Background

참고논문) SphereFace: Deep Hypersphere Embedding for Face Recognition

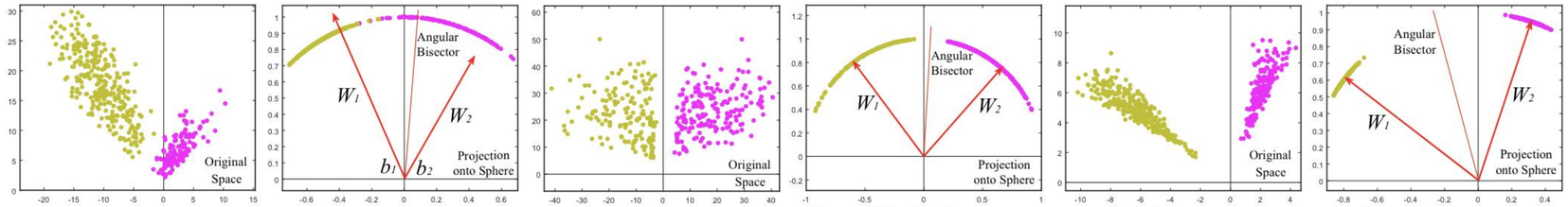
- **Face Recognition**

  🤔 서로 다른 얼굴 이미지를 잘 구분하기 위해 feature representation extraction을 어떻게 수행해야 할까?

  - SphereFace
    - softmax loss에서 W_j가 정규화 되어있고 bias가 0이라고 가정해보자!
      ⇒ 둘 간의 각도로써 decision boundary를 표현할 수 있음

$$L_{\text{ang}} = \frac{1}{N}\sum_i -\log\left(\frac{e^{\|\boldsymbol{x}_i\|\cos(m\theta_{y_i,i})}}{e^{\|\boldsymbol{x}_i\|\cos(m\theta_{y_i,i})} + \sum_{j\neq y_i} e^{\|\boldsymbol{x}_i\|\cos(\theta_{j,i})}}\right) \tag{6}$$

⇒ 해당 각도를 더욱 크게 벌려주도록 학습시켜 discriminative power를 증대



(a) Original Softmax Loss (b) Original Softmax Loss (c) Modified Softmax Loss (d) Modified Softmax Loss (e) A-Softmax Loss (f) A-Softmax Loss

Figure 2: Comparison among softmax loss, modified softmax loss and A-Softmax loss. In this toy experiment, we construct a CNN to learn 2-D features on a subset of the CASIA face dataset. In specific, we set the output dimension of FC1 layer as 2 and visualize the learned features. Yellow dots represent the first class face features, while purple dots represent the second class face features. One can see that features learned by the original softmax loss can not be classified simply via angles, while modified softmax loss can. Our A-Softmax loss can further increase the angular margin of learned features.

#0. Background

참고논문) [CosFace: Large Margin Cosine Loss for Deep Face Recognition]

- **Face Recognition**
  🤔 서로 다른 얼굴 이미지를 잘 구분하기 위해 feature representation extraction을 어떻게 수행해야 할까?
  - CosFace
    - SphereFace보다 좀 더 엄격한 penalty를 부여하여 angular margin을 확보
      ⇒ SphereFace는 각도에 곱하기 연산, CosFace는 뺄셈 연산

$$L_{lmc} = \frac{1}{N}\sum_{i} -\log \frac{e^{s(\cos(\theta_{y_i,i})-m)}}{e^{s(\cos(\theta_{y_i,i})-m)} + \sum_{j\neq y_i} e^{s\cos(\theta_{j,i})}},$$

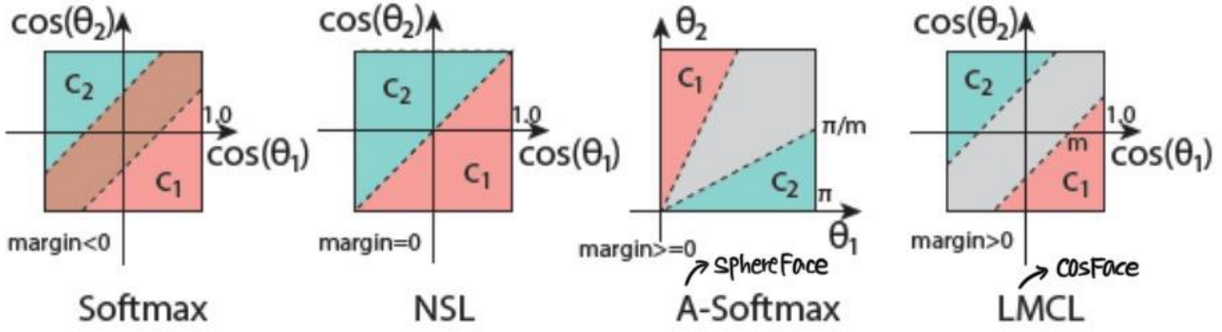
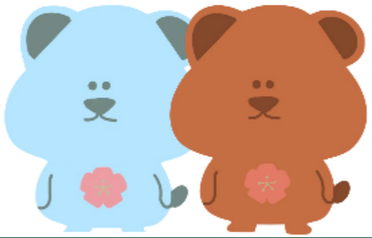


Figure 2. The comparison of decision margins for different loss functions the binary-classes scenarios. Dashed line represents decision boundary, and gray areas are decision margins.

1. Overview



EWHA EURON

#1. Overview

- **ArcFace: Additive Angular Margin Loss for Deep Face Recognition**
  - DCNN은 일반적으로 자세 정규화(pose normalization) 단계 이후 얼굴 이미지를 작은 내부 클래스와 큰 내부 클래스 간의 거리(→ margin)를 가지는 특징으로 매핑
  - SphereFace나 CosFace와 다르게 각도에 직접적으로 penalty를 주는 방식

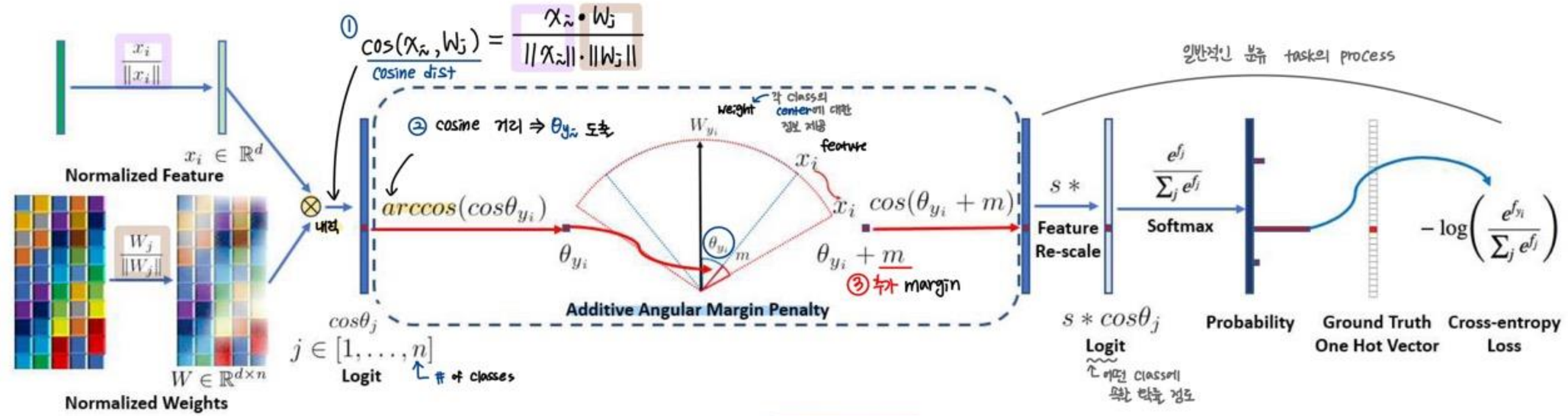


Figure 2. Training a DCNN for face recognition supervised by the ArcFace loss. Based on the feature x_i and weight W normalisation, we get the $\cos\theta_j$ (logit) for each class as $W_j^T x_i$. We calculate the $arccos\theta_{y_i}$ and get the angle between the feature x_i and the ground truth weight W_{y_i}. In fact, W_j provides a kind of centre for each class. Then, we add an angular margin penalty m on the target (ground truth) angle θ_{y_i}. After that, we calculate $\cos(\theta_{y_i} + m)$ and multiply all logits by the feature scale s. The logits then go through the softmax function and contribute to the cross entropy loss.

#1. Overview

- **Proposed Approach**
  Loss 관점에서..

  ○ Softmax Loss

$$L_1 = -\frac{1}{N}\sum_{i=1}^{N}\log\frac{e^{W_{y_i}^T x_i + b_{y_i}}}{\sum_{j=1}^{n} e^{W_j^T x_i + b_j}}$$

  ○ Angular Margin Loss
  ⇒ 가중치(W_j)와 feature(x_i) 간의 내적을 각도의 표현으로 대체

$$L_3 = -\frac{1}{N}\sum_{i=1}^{N}\log\frac{e^{s(\cos(\theta_{y_i}+m))}}{e^{s(\cos(\theta_{y_i}+m))} + \sum_{j=1,j\neq y_i}^{n} e^{s\cos\theta_j}}. \quad (3)$$

margin

"angle"

  - weight를 rescale → s로 표현
  - feature(x_i)와 가중치(W_{yi}) 사이에 penalty 부여 ⇒ margin

#1. Overview

- **Proposed Approach**
  - SphereFace, CosFace
    - 3가지 마진 패널티 제안 ⇒ 곱셈 각도 마진(m1), 추가 각도 마진(m2), 추가 코사인 마진(m3)
    - 모두 대상 logit을 패널티로 주어 클래스 내 조밀성과 클래스 간 다양성을 강제

$$L_4 = -\frac{1}{N}\sum_{i=1}^{N}\log\frac{e^{s(\cos(m_1\theta_{y_i}+m_2)-m_3)}}{e^{s(\cos(m_1\theta_{y_i}+m_2)-m_3)}+\sum_{j=1,j\neq y_i}^{n}e^{s\cos\theta_j}}. \quad (4)$$

margin penalty 추가

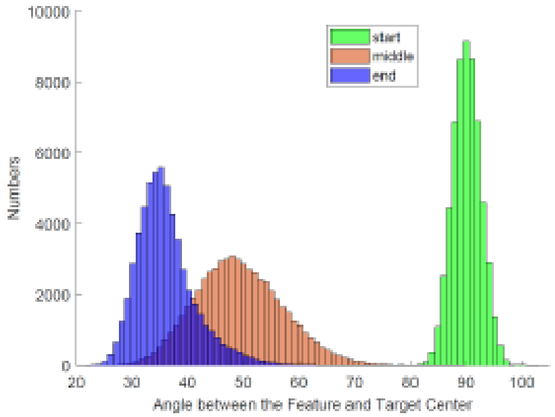
    - Margin 적용 방식
      - SphereFace: m1이 m2보다 작기만 하면 됨 → θ_2가 작기만 하다면 해당 클래스로 분류될 수 있음
      - CosFace: cos 값에 margin 적용 → non-linear
      - ArcFace: 각도에 "직접적"으로 margin(penalty) 적용 ⇒ completely linear

#1. Overview

- **Target Logit Analysis**

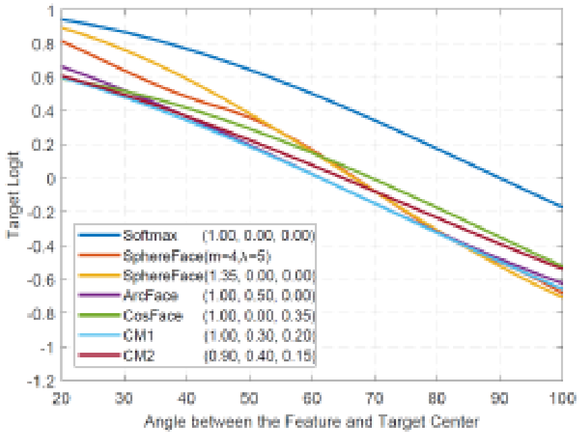
  🤔 ArcFace에서 제안한 loss가 과연 효과적인가

  - Θ_j distribution
    ArcFace 훈련 동안의 start-middle-end의 분포
    → 학습이 진행됨에 따라 분포가 점점 더 작아지고 있음



(a) θ_j Distributions

  - Target Logit Curves
    → 각도의 경사가 완만 ⇒ 안정적인 수렴



(b) Target Logits Curves

#1. Overview

- **Comparison with Other Losses**

  🤔 ArcFace 구현 시 고려된 loss들

  - Intra-Loss
    - 샘플과 실제 센터 간의 각도/아크를 줄이는 것
    - 클래스 내 조밀성을 향상시키기 위해 설계되었음

$$L_5 = L_2 + \frac{1}{\pi N}\sum_{i=1}^{N}\theta_{y_i}.$$

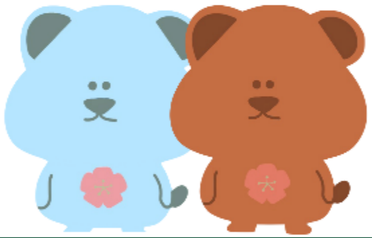
  - Inter-Loss
    - 서로 다른 센터 간의 각도/아크를 증가시킴으로써 클래스 간 차이를 강화하기 위해 설계되었음

$$L_6 = L_2 - \frac{1}{\pi N\,(n-1)}\sum_{i=1}^{N}\sum_{j=1,j\neq y_i}^{n}\arccos(W_{y_i}^{T}W_j).$$

  - Triplet loss
    - 삼중 샘플 사이의 각도/아크 마진을 확장하는 것을 목표
    - FaceNet에서는 정규화된 특징에 유클리드 마진을 적용
    - ArcFace에서는 특징의 **각도** 표현을 사용하여 삼중 손실을 적용

$$\arccos(x_i^{pos}x_i) + m \leq \arccos(x_i^{neg}x_i).$$

2. Experiments



EWHA EURON

#2. Experiments

- **Implementation Details**
  - 데이터셋

Datasets	#Identity	#Image/Video	
CASIA [41]	10K	0.5M	Train
VGGFace2 [3]	9.1K	3.3M	
MS1MV2	85K	5.8M	
MS1M-DeepGlint [1]	87K	3.9M	
Asian-DeepGlint [1]	94 K	2.83M	
LFW [10]	5,749	13,233	Validation
CFP-FP [28]	500	7,000	
AgeDB-30 [19]	568	16,488	
CPLFW [44]	5,749	11,652	
CALFW [45]	5,749	12,174	
YTF [38]	1,595	3,425	
MegaFace [12]	530 (P)	1M (G)	Test
IJB-B [37]	1,845	76.8K	
IJB-C [18]	3,531	148.8K	
Trillion-Pairs [1]	5,749 (P)	1.58M (G)	
iQIYI-VID [17]	4,934	172,835	

Table 1. Face datasets for training and testing. "(P)" and "(G)" refer to the probe and gallery set, respectively.

  - 실험 설정
    - 임베딩 네트워크: ResNet50 및 ResNet100
    - 특징 스케일: 64
    - 각도 마진: 0.5
    - 모든 실험은 MXNet으로 구현되며, 네 개의 NVIDIA Tesla P40 (24GB) GPU에서 학습
    - 훈련 및 테스트 중에는 각각 512의 배치 크기를 사용하며, 테스트 시에는 임베딩 네트워크만을 유지하여 512 차원의 특징을 추출

#2. Experiments

- **Ablation Study on Losses**
  - Loss function
    - CASIA 데이터셋을 ResNet50 모델에 활용하여 학습시킴
    - ArcFace의 성능이 전반적으로 높은 성능을 달성
    - margin에 따른 성능 차이는 두드러지지 x(최적: 0.5)
    - 다른 loss들을 혼합하여 사용한 것보다

      ArcFace Loss function을 단독으로 사용한 경우 성능이 더 좋았음

Loss Functions	LFW	CFP-FP	AgeDB-30
ArcFace (0.4)	99.53	95.41	94.98
ArcFace (0.45)	99.46	95.47	94.93
ArcFace (0.5)	**99.53**	**95.56**	**95.15**
ArcFace (0.55)	99.41	95.32	95.05
SphereFace [15]	99.42	-	-
SphereFace (1.35)	99.11	94.38	91.70
CosFace [35]	99.33	-	-
CosFace (0.35)	99.51	95.44	94.56
CM1 (1, 0.3, 0.2)	99.48	95.12	94.38
CM2 (0.9, 0.4, 0.15)	99.50	95.24	94.86
Softmax	99.08	94.39	92.33
Norm-Softmax (NS)	98.56	89.79	88.72
NS+Intra	98.75	93.81	90.92
NS+Inter	98.68	90.67	89.50
NS+Intra+Inter	98.73	94.00	91.41
Triplet (0.35)	98.98	91.90	89.98
ArcFace+Intra	99.45	95.37	94.73
ArcFace+Inter	99.43	95.25	94.55
ArcFace+Intra+Inter	99.43	95.42	95.10
ArcFace+Triplet	99.50	95.51	94.40

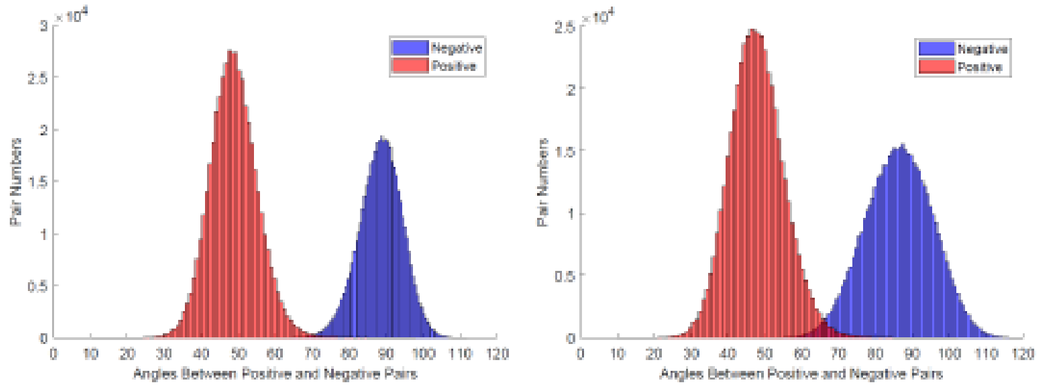
Table 2. Verification results (%) of different loss functions ([CASIA, ResNet50, loss*]).

#2. Experiments

- **Ablation Study on Losses**
  - 실제 각도에 대한 통계량
    - ArcFace와 클래스 내 조밀성(Intra) 및 클래스 간 차이(Inter)를 강화하는 다른 손실 간의 추가 비교 수행
    - ArcFace는 이미 매우 좋은 클래스 내 조밀성과 클래스 간 차이를 갖고 있음
    - 또한, 테스트 세트에서 Triplet-Loss보다 ArcFace가 더 명확한 마진을 갖고 있음

      ⇒ ArcFace의 장점이 더욱 부각됨

	NS	ArcFace	IntraL	InterL	TripletL
W-EC	44.26	14.29	8.83	46.85	-
W-Inter	69.66	71.61	31.34	75.66	-
Intra1	50.50	38.45	17.50	52.74	41.19
Inter1	59.23	65.83	24.07	62.40	50.23
Intra2	33.97	28.05	12.94	35.38	27.42
Inter2	65.60	66.55	26.28	67.90	55.94



(a) ArcFace (b) Triplet-Loss

#2. Experiments

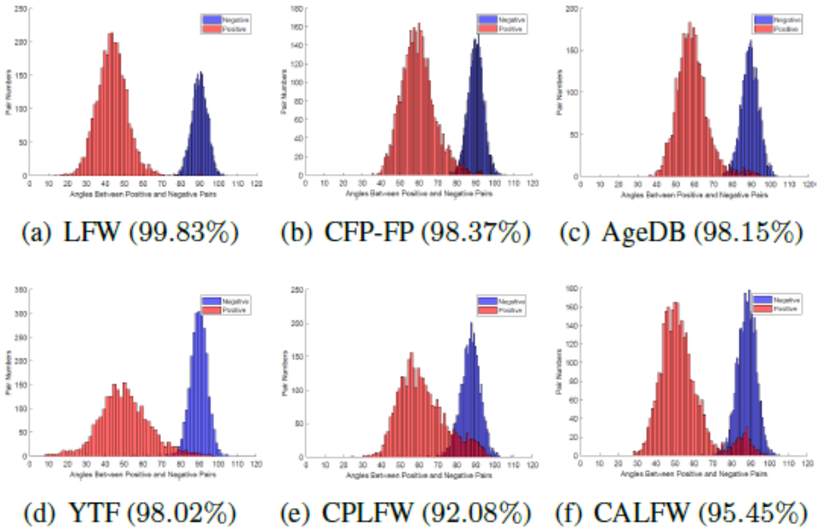
- **Evaluation Results**
  - LFW, YTF, CALFW and CPLFW
    - ArcFace는 LFW와 YTF 데이터셋에서 기존 방법들(SphereFace와 CosFace)을 큰 폭으로 능가하며, CPLFW와 CALFW와 같은 얼굴 데이터셋에서도 우수한 성능을 보임
    - 성능 뿐만 아니라 positive & negative pair에 대한 angle distribution 확인 결과 ArcFace가 매우 높은 분별력을 보임을 확인

Method	#Image	LFW	YTF
DeepID [30]	0.2M	99.47	93.20
Deep Face [31]	4.4M	97.35	91.4
VGG Face [22]	2.6M	98.95	97.30
FaceNet [27]	200M	99.63	95.10
Baidu [13]	1.3M	99.13	-
Center Loss [36]	0.7M	99.28	94.9
Range Loss [43]	5M	99.52	93.70
Marginal Loss [6]	3.8M	99.48	95.98
SphereFace [15]	0.5M	99.42	95.0
SphereFace+ [14]	0.5M	99.47	-
CosFace [35]	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.

Method	LFW	CALFW	CPLFW
HUMAN-Individual	97.27	82.32	81.21
HUMAN-Fusion	99.85	86.50	85.24
Center Loss [36]	98.75	85.48	77.48
SphereFace [15]	99.27	90.30	81.40
VGGFace2 [3]	99.43	90.57	84.00
MS1MV2, R100, ArcFace	**99.82**	**95.45**	**92.08**

Table 5. Verification performance (%) of open-sourced face recognition models on LFW, CALFW and CPLFW.



(a) LFW (99.83%) (b) CFP-FP (98.37%) (c) AgeDB (98.15%)

(d) YTF (98.02%) (e) CPLFW (92.08%) (f) CALFW (95.45%)

#2. Experiments

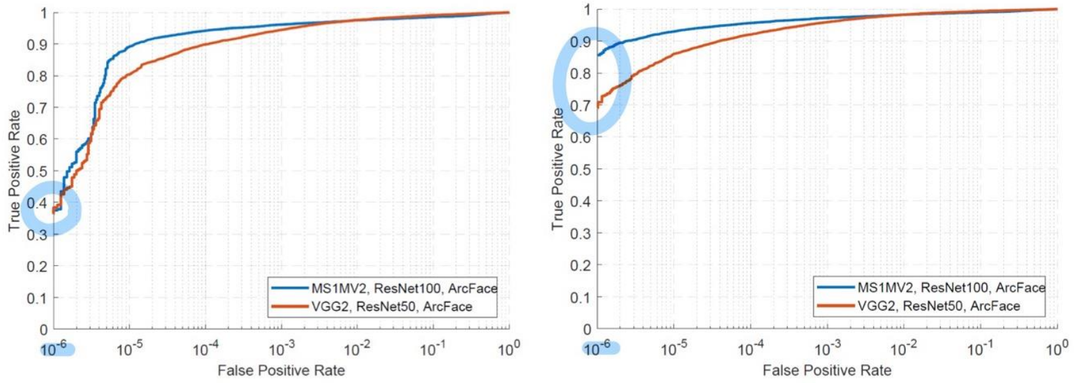
- **Evaluation Results**
  - Face Identification
    - MegaFace 데이터셋에 대해 ArcFace의 성능이 가장 우수하였음
    - 본 논문에서 제안한 학습 데이터셋(R, 라벨이 잘못 지정된 이미지를 수동으로 정제) 활용 시 ArcFace의 성능이 가장 높았음

Methods	Id (%)	Ver (%)
Softmax [15]	54.85	65.92
Contrastive Loss[15, 30]	65.21	78.86
Triplet [15, 27]	64.79	78.32
Center Loss[36]	65.49	80.14
SphereFace [15]	72.729	85.561
CosFace [35]	77.11	89.88
AM-Softmax [33]	72.47	84.44
SphereFace+ [14]	73.03	-
CASIA, R50, ArcFace	77.50	92.34
CASIA, R50, ArcFace, R	91.75	93.69
FaceNet [27]	70.49	86.47
CosFace [35]	82.72	96.65
MS1MV2, R100, ArcFace	81.03	96.98
MS1MV2, R100, CosFace	80.56	96.56
MS1MV2, R100, ArcFace, R	98.35	98.48
MS1MV2, R100, CosFace, R	97.91	97.91

#2. Experiments

- **Evaluation Results**
  - IJB-B and IJB-C
    - IJB-B 및 IJB-C 데이터셋에서 ArcFace는 VGG2 데이터셋을 훈련 데이터로 사용하고 ResNet50를 임베딩 네트워크로 사용하여 최근 방법들과 비교
    - ArcFace는 이전의 최첨단 모델보다 약 3~5%의 성능 향상을 보이며, TAR(@FAR=1e-4)를 IJB-B에서 94.2%, IJB-C에서 95.6%로 높일 수 있었음
    - 또한, ArcFace는 IJB-B와 IJB-C에서 심지어 FAR=1e-6 설정에서도 우수한 성능을 보여주었음

Method	IJB-B	IJB-C
ResNet50 [3]	0.784	0.825
SENet50 [3]	0.800	0.840
ResNet50+SENet50 [3]	0.800	0.841
MN-v [40]	0.818	0.852
MN-vc [40]	0.831	0.862
ResNet50+DCN(Kpts) [39]	0.850	0.867
ResNet50+DCN(Divs) [39]	0.841	0.880
SENet50+DCN(Kpts) [39]	0.846	0.874
SENet50+DCN(Divs) [39]	0.849	0.885
VGG2, R50, ArcFace	0.898	0.921
MS1MV2, R100, ArcFace	**0.942**	**0.956**



(a) ROC for IJB-B (b) ROC for IJB-C

Figure 9. ROC curves of 1:1 verification protocol on the IJB-B and IJB-C dataset.

#2. Experiments

- **Evaluation Results**
  - Trillion Pairs Dataset & iQIYI-VID Dataset
    - Trillion Pairs dataset에 대해 실험을 진행한 결과 제안한 학습 데이터셋을 활용 시 유의미한 성능 향상을 확인할 수 있었음
      ⇒ 당시 가장 최신 학습 데이터셋인 CIGIT-IRSEC 데이터셋과 비교해도 동등한 수준의 성능을 기록
    - Video 데이터셋인 Iqiyi-vid 데이터셋에 대한 실험 결과 매우 높은 성능을 기록하였음

Method	Id (@FPR=1e-3)	Ver(@FPR=1e-9)
CASIA	26.643	21.452
MS1MV2	80.968	78.600
DeepGlint-Face	80.331	78.586
MS1MV2+Asian	**84.840** (1st)	80.540
CIGIT_IRSEC	84.234 (2nd)	**81.558** (1st)

Table 8. Identification and verification results (%) on the Trillion-Pairs dataset. ([Dataset*, ResNet100, ArcFace])

Method	MAP(%)
MS1MV2+Asian, R100, ArcFace	79.80
+ MLP	86.40
+ Ensemble	88.26
+ Context	**88.65 (1st)**
Other Participant	87.66 (2nd)

Table 9. MAP of our method on the iQIYI-VID test set. "MLP" refers to a three-layer fully connected network trained on the iQIYI-VID training data.

#3. Conclusion

○ 본 논문에서는 ArcFace라는 새로운 loss function을 제시

⇒ 각도에 직접적으로 penalty를 적용해 보자!

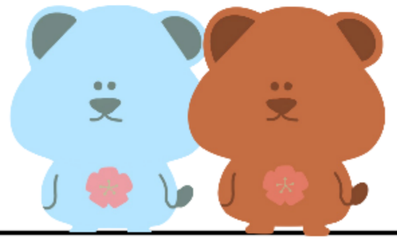
○ 학습 안정성과 성능 측면에서 기존 방법론들에 비해 우수함을 입증함

- Engaging: 구 등의 고차원 공간에서 각도와 arc 간의 distance margin을 직접 최적화
- Effective: 대규모 이미지 및 비디오 데이터셋을 포함한 열 개의 face recognition 벤치마크에서 SOTA 달성
- Easy: 쉬운 구현, 안정적인 수렴
- Efficient: 추가적인 computational cost가 매우 적음

○ 새로운 학습 데이터셋인 MS1MV2 데이터셋을 활용

⇒ face recognition task에 대한 성능 향상

THANK YOU



EWHA EURON