Advanced Topics in Deep Learning

Rui Zhao
Executive R&D Director, SenseTime
zhaorui@sensetime.com
May 9, 2019

About SenseTime

Brief Introduction

20 years
Research Exp.

2300
Employees

150+
AI PhDs

Original
Deep Learning Platform

Advanced Technology

AI+Ecnomics

AI+Smart City

AI+Smart Phone

AI+Chip

AI+AutoDrive

AI+Healthcare

Largest AI technology and solution provider in Asia

Business Area

Proprietary technology empowers businesses and services


Smart City


Mobile


General Cultural and Entertainment


Intelligent Automobile


Smart Healthcare


Business Intelligence


Education

Advertising

About SenseTime

Face and Body

Detection TechPublic and Professional Image Detection

High Volume Video Comprehension and Mining

Improving Video And Image Processing

SLAM and 3D Vision

Robot Sensing and Control

Autonomous Driving

Deep Learning Platform

Medical Image Analysis

About SenseTime

We have offices in Hong Kong, Beijing, Shenzhen, Shanghai, Chengdu, Hangzhou, Kyoto, Tokyo and Singapore.

Outline

- Face recognition (60min)
- Large scale clustering (45min)
- Continue Learning (30min)

Face Recognition

Outline——Face Recognition

- Introduction
- Literatures
- Practical problems when widely applied
- Efforts to push the limit

Introduction

- Application Scenarios

Face Verification
(1:1)

Identity check in entrances,
Face unlock on smart phones

Face Identification

1:N

Rank-based

Acquire identity information

1:n

Threshold-based

Blacklist monitoring,
VIP services.

LFW Face Verification Protocol

- Labeled faces in the wild (LFW) dataset is a widely used face verification (1:1) protocol, which contains 6,000 face pairs. In all 6,000 pairs, match and mismatch pairs each account for half.

Match Pairs

Benjamin Netanyahu

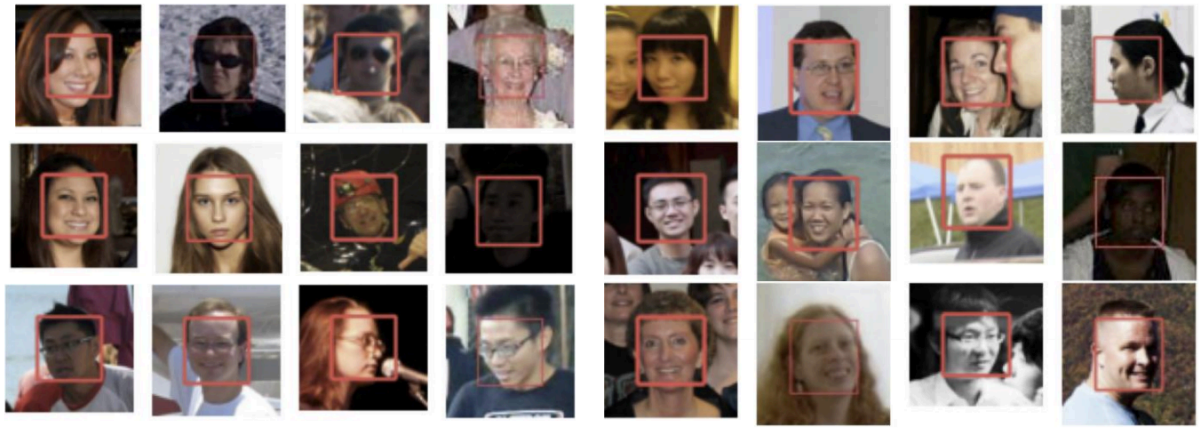
Mismatch Pairs

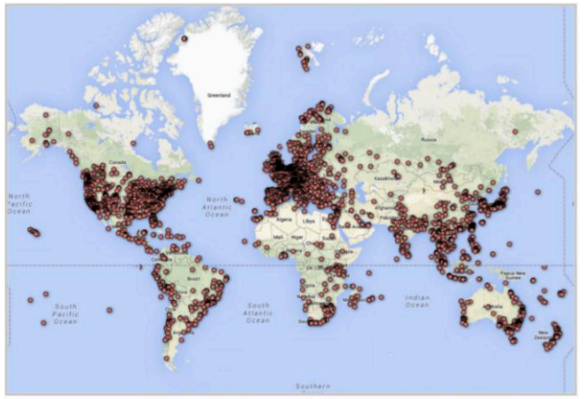
Barbara Felt Miller Leticia Dolera

Huang, Gary B., et al. "Labeled faces in the wild: A database forstudying face recognition in unconstrained environments." *Workshop on faces in'Real-Life'Images: detection, alignment, and recognition*. 2008.

MegaFace 1M Face Identification Protocol

- The MegaFace identification dataset includes 1M images of 690K different individuals (from Flickr) as the gallery set and 100K photos of 530 unique individuals from FaceScrub as the probe set.





GPS locations

Radom sample of MegaFace Photos with provided detections in red

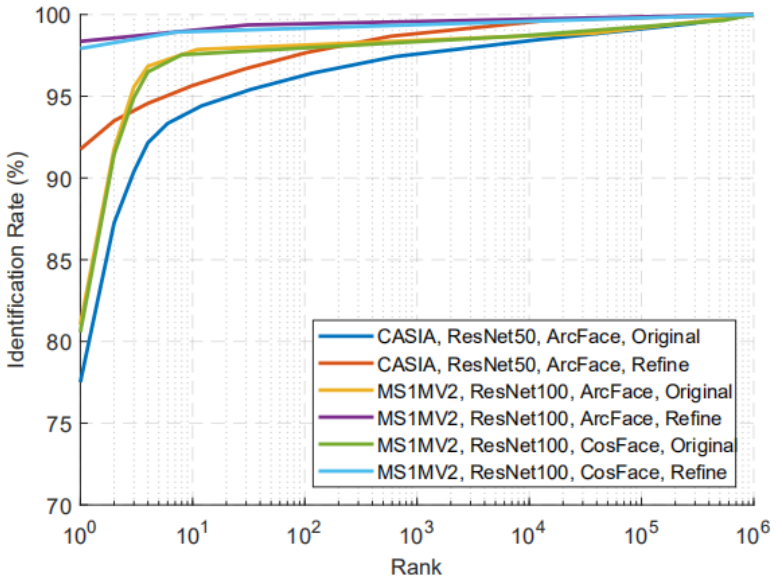
Kemelmacher-Shlizerman, Ira, et al. "The megaface benchmark: 1 million faces for recognition at scale." Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2016.

Ng, Hong-Wei, and Stefan Winkler. "A data-driven approach to cleaning large face datasets." 2014 IEEE International Conference on Image Processing (ICIP). IEEE, 2014.

Introduction

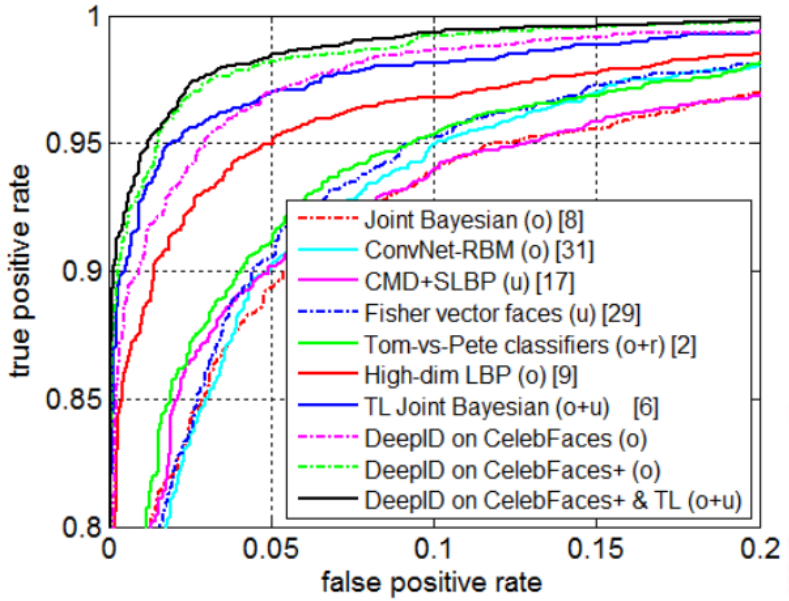
- Evaluation metrics
  - 1:N : top K, the probability of finding the target image within top K in ranking list;
  - 1:1, 1:n : True Positive Rate (TPR) @ False Positive Rate (FPR) (e.g. 99.06% @1e-3).

1:N



CMC Curve

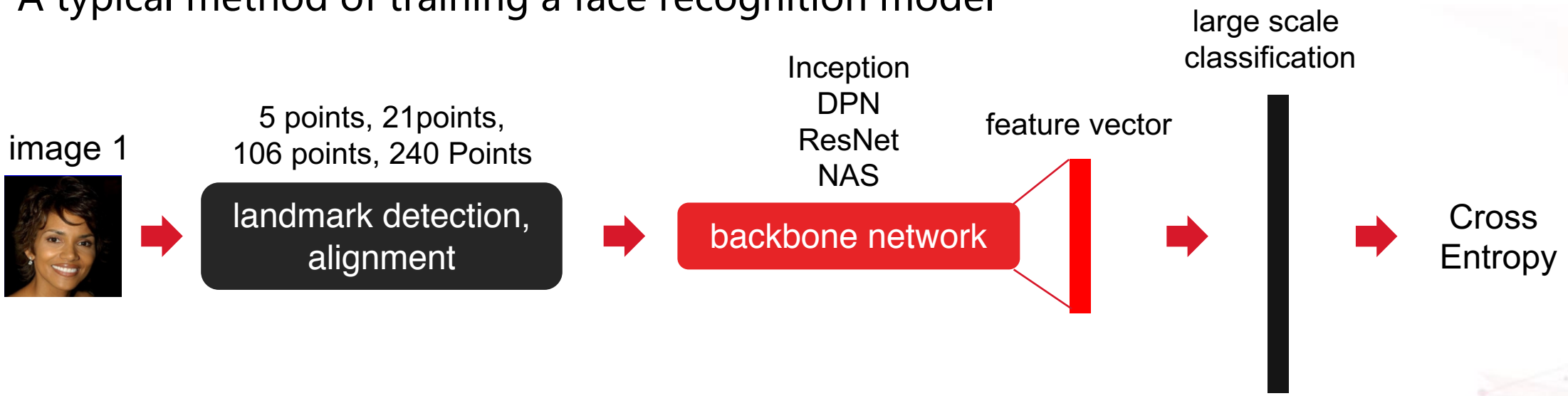
1:1 , 1:n

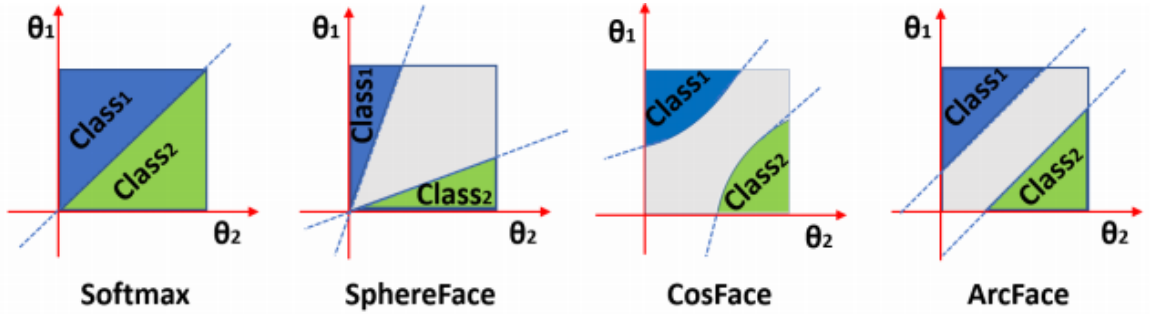


ROC Curve

Introduction

- A typical method of training a face recognition model



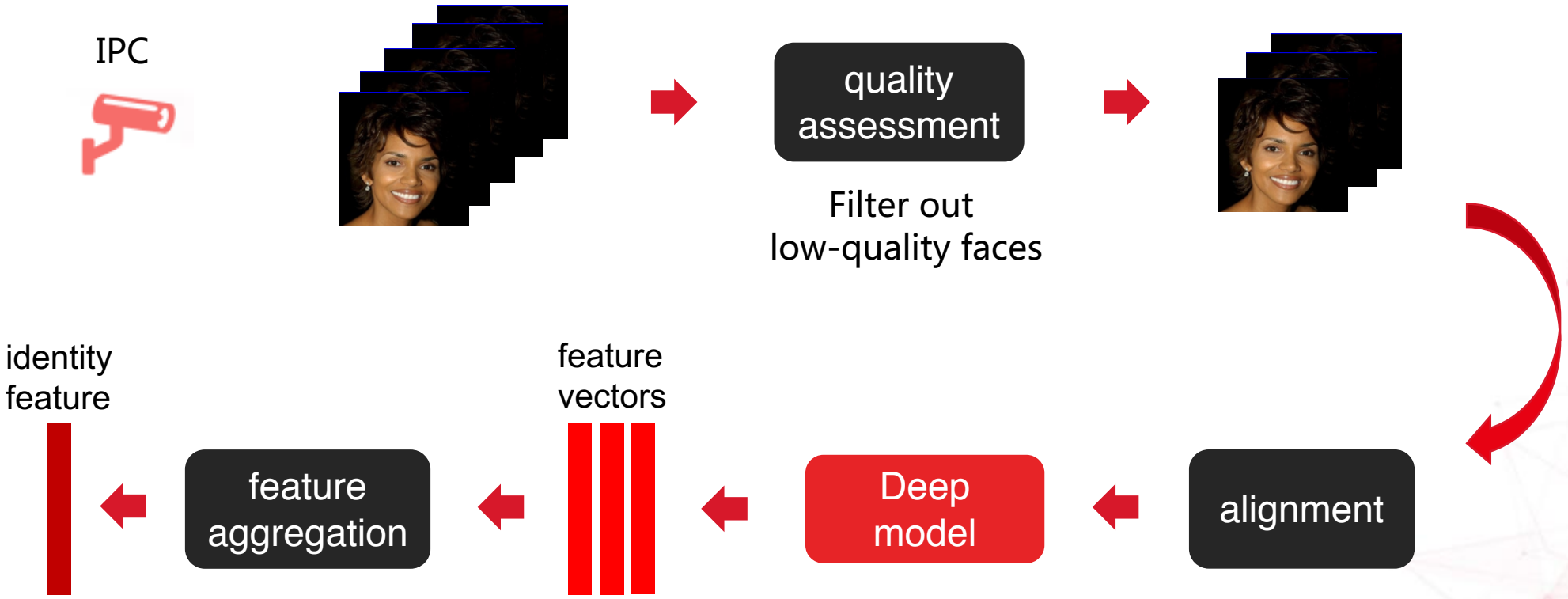


$$L_4 = -\frac{1}{N}\sum_{i=1}^{N}\log\frac{e^{s(\cos(m_1\theta_{y_i}+m_2)-m_3)}}{e^{s(\cos(m_1\theta_{y_i}+m_2)-m_3)}+\sum_{j=1,j\neq y_i}^{n}e^{s\cos\theta_j}}$$

[1] W. Liu, Y. Wen, Z. Yu, M. Li, B. Raj, and L. Song. Sphereface: Deep hypersphere embedding for face recognition. In CVPR, 2017.
[2] H. Wang, Y. Wang, Z. Zhou, X. Ji, Z. Li, D. Gong, J. Zhou, and W. Liu. Cosface: Large margin cosine loss for deep face recognition. In CVPR, 2018.
[3] J. Deng, J. Guo, N. Xue, ArcFace: Additive Angular Margin Loss for Deep Face Recognition. arXiv:1801.07698v2, 2018.

Introduction

- Basic pipeline of face recognition

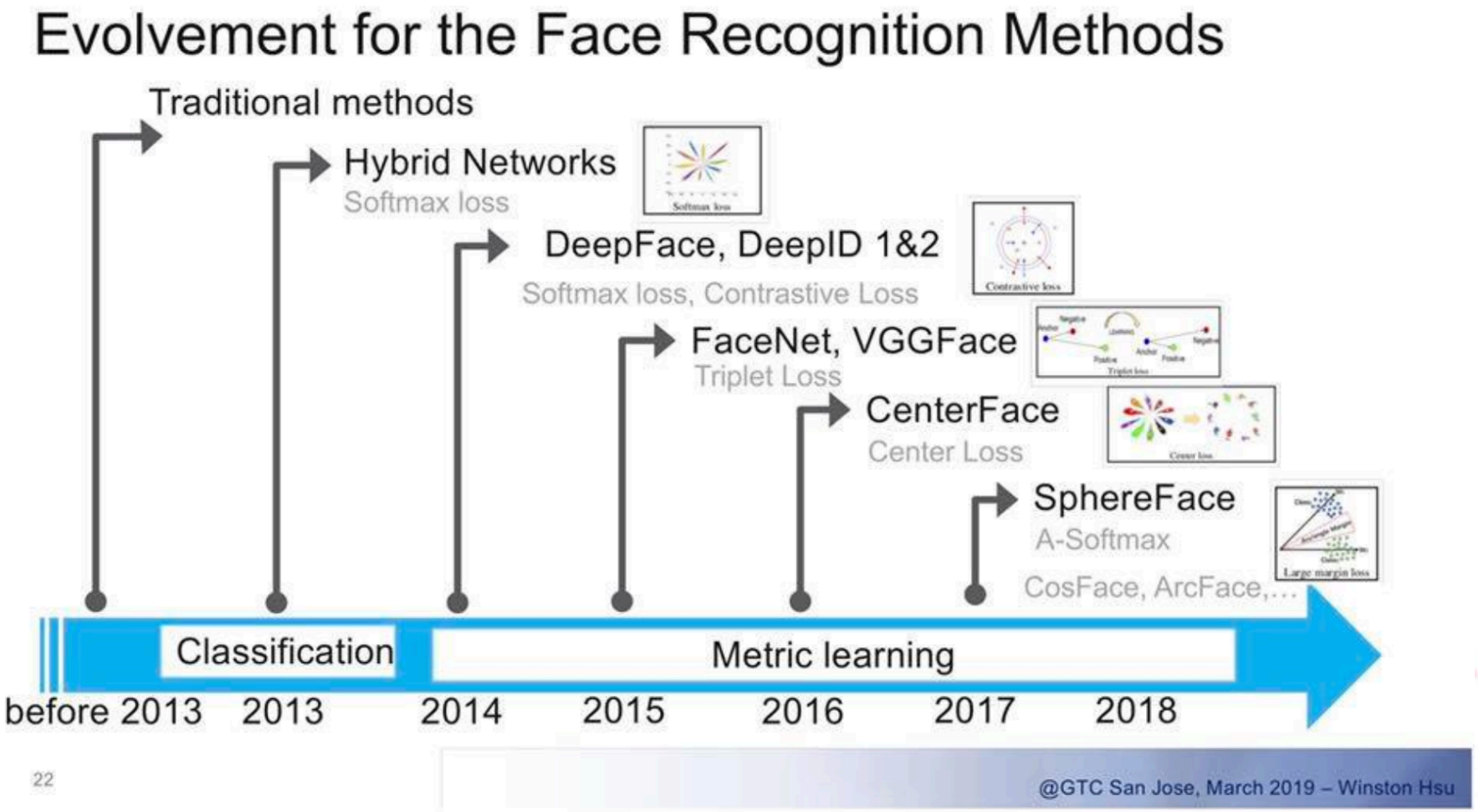


Outline——Face Recognition

- Introduction
- Literatures
- Practical problems when widely applied
- Efforts to push the limit

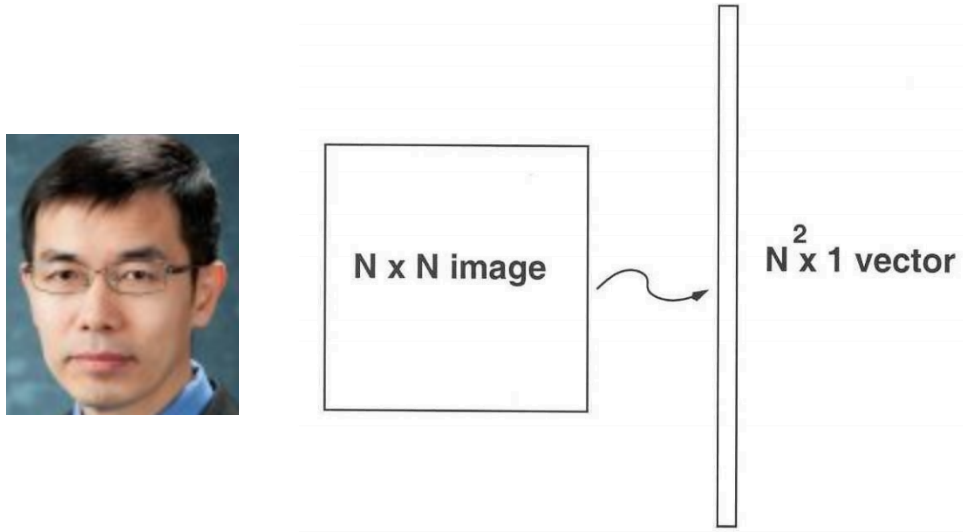
Literatures

Traditional methods before DL era (EigenFace, LBP, Sparse coding)



Literatures

Traditional methods before DL era



How to extract feature in low-dimensional space?

Subspace Learning

Manually design

Sparse coding

Literatures

Subspace Learning

M. Turk et al., 1991

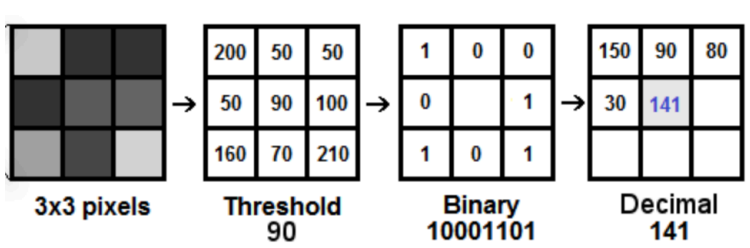
Suppose Γ is an N^2x1 vector, corresponding to an NxN face image I.

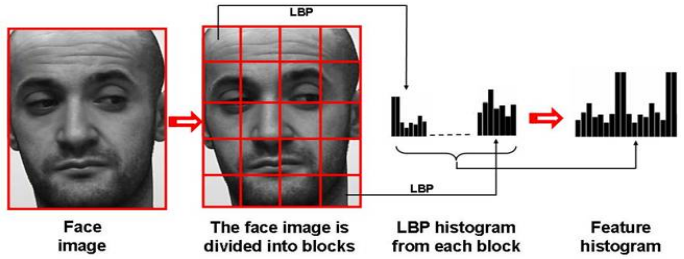
The idea is to represent Γ (Φ=Γ - mean face) into a low-dimensional space:

$$\Phi = w_1 u_1 + w_2 u_2 + \cdots w_K u_K \ (K << N^2)$$

Manually design

Ahonen et al., 2006





Sparse coding

Honglak et al, 2006

$$\mathcal{L}_{sc} = \underbrace{||WH - X||_2^2}_{\text{reconstruction term}} + \underbrace{\lambda||H||_1}_{\text{sparsity term}}$$

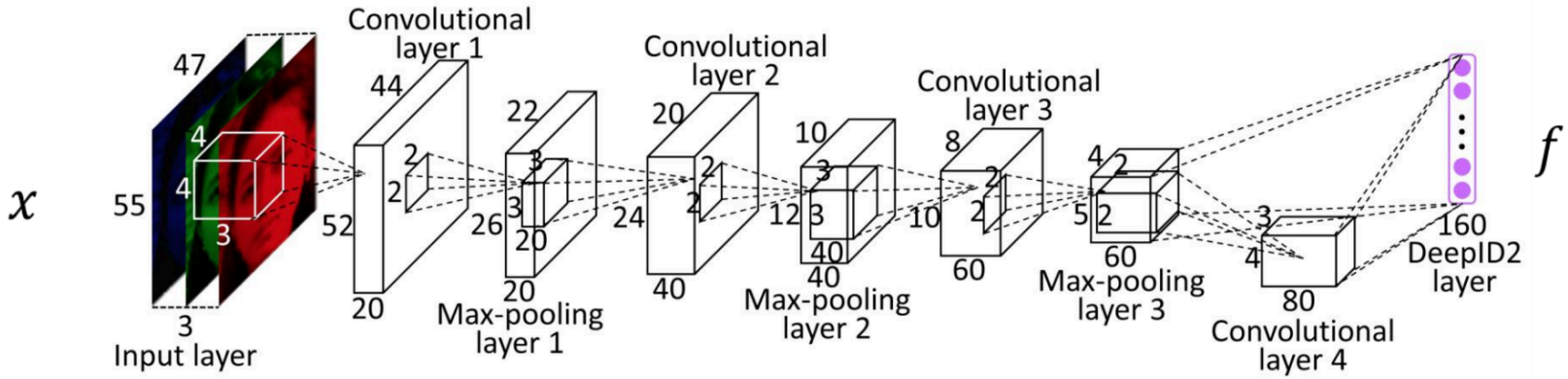
X is an input image
W is a over-complete dictionary
H is sparse codes

Some Euclidean metric based losses

- Deep learning face representation by joint identification-verification (NIPS 2014)
- FaceNet: A unified embedding for face recognition and clustering (CVPR 2015)
- A discriminative feature learning approach for deep face recognition（ECCV 2016）

DeepID2

- The key challenge of face recognition is to develop effective feature representations for reducing intra-personal variations while enlarging inter-personal differences. DeepID2 uses both face identification and verification signals as loss functions.



Here x is the input face image and f is the extracted DeepID2 vector (feature vector). Loss function utilize f to compute the cost.

Sun, Yi, Xiaogang Wang, and Xiaoou Tang. "Deep learning face representation from predicting 10,000 classes." CVPR. 2014.
Sun, Yi, et al. "Deep learning face representation by joint identification-verification." NIPS. 2014.

DeepID2

- DeepID2 feature vectors are learned under two supervisory signals.
- The identification loss classifies each face image into one of n different identities. Identification is achieved by an n-way softmax layer, which outputs a probability distribution over the n classes. Then the distribution is inputted to cross-entropy loss.
- The verification loss encourages features from faces of the same identity to be similar. The verification signal directly regularizes features and can effectively reduce the intra-personal variations.

Identification Loss:

$$\text{Ident}(f, t, \theta_{id}) = -\sum_{i=1}^{n} -p_i \log \hat{p}_i = -\log \hat{p}_t$$

Softmax classifier

$$\hat{p}_t = \frac{e^{W_t^T f_i + b_t}}{\sum_{j=1}^{N} e^{W_j^T f_i + b_j}}$$

Verification Loss:

$$\text{Verif}(f_i, f_j, y_{ij}, \theta_{ve}) = \begin{cases} \frac{1}{2}\|f_i - f_j\|_2^2 & \text{if } y_{ij} = 1 \\ \frac{1}{2}\max\left(0, m - \|f_i - f_j\|_2\right)^2 & \text{if } y_{ij} = -1 \end{cases}$$

same person

different person

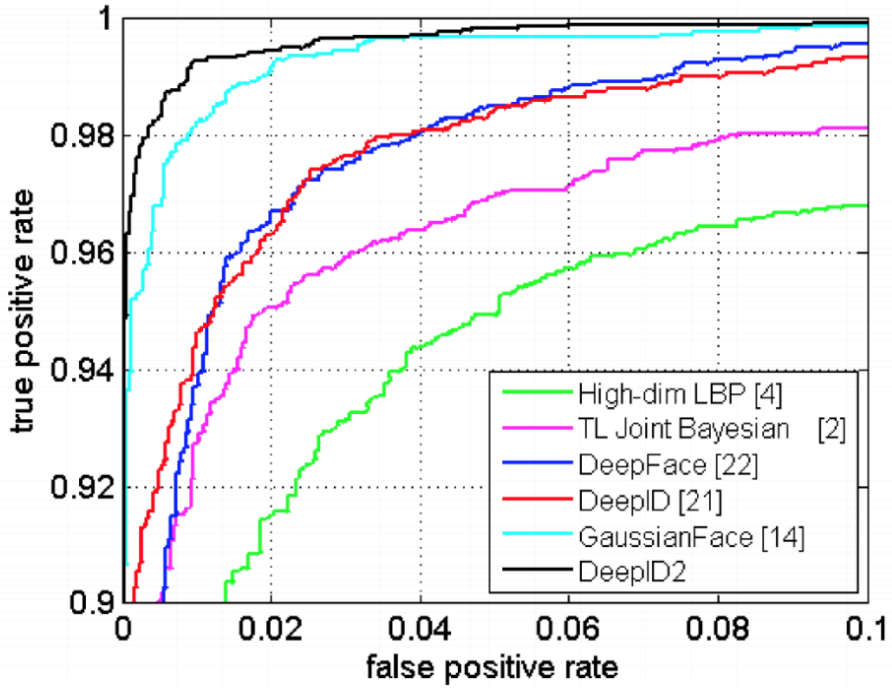
Sun, Yi, Xiaogang Wang, and Xiaoou Tang. "Deep learning face representation from predicting 10,000 classes." CVPR. 2014.
Sun, Yi, et al. "Deep learning face representation by joint identification-verification." NIPS. 2014.

DeepID2

- Some results of DeepID2

method	accuracy (%)
high-dim LBP [4]	95.17 ± 1.13
TL Joint Bayesian [2]	96.33 ± 1.08
DeepFace [22]	97.35 ± 0.25
DeepID [21]	97.45 ± 0.26
GaussianFace [14]	98.52 ± 0.66
DeepID2	99.15 ± 0.13

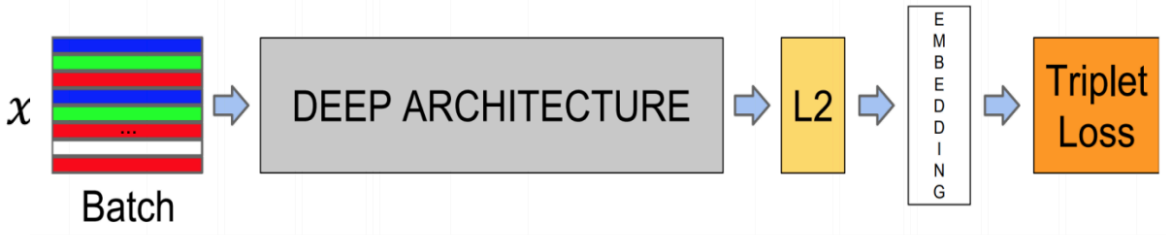
Accuracy comparison with the previous best results on LFW at that time.

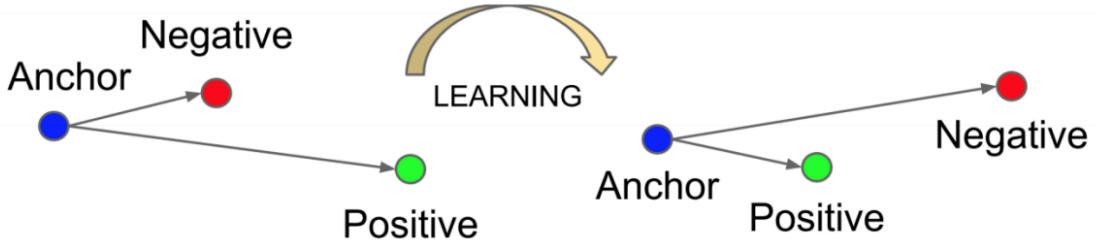


Sun, Yi, Xiaogang Wang, and Xiaoou Tang. "Deep learning face representation from predicting 10,000 classes." CVPR. 2014.
Sun, Yi, et al. "Deep learning face representation by joint identification-verification." NIPS. 2014.

Triplet Loss (Google FaceNet)

- The network consists of a batch input layer and a deep CNN followed by L2 normalization, which results in the face embedding.
- 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.





Schroff, Florian, Dmitry Kalenichenko, and James Philbin. "Facenet: A unified embedding for face recognition and clustering." Proceedings of the IEEE conference on computer vision and pattern recognition. 2015.

Triplet Loss (Google FaceNet)

- The embedding is represented by $f(x)$. It embeds an image x into a d-dimensional Euclidean space. Here we want to ensure that an image x_i^a (anchor) of a specific person is closer to all other images x_i^p (positive) of the same person than it is to any image x_i^n (negative) of any other person. The loss that is being minimized is then

$$L = \sum_i^N \max(0, \|f(x_i^a) - f(x_i^p)\|_2^2 - \|f(x_i^a) - f(x_i^n)\|_2^2 + \alpha)$$

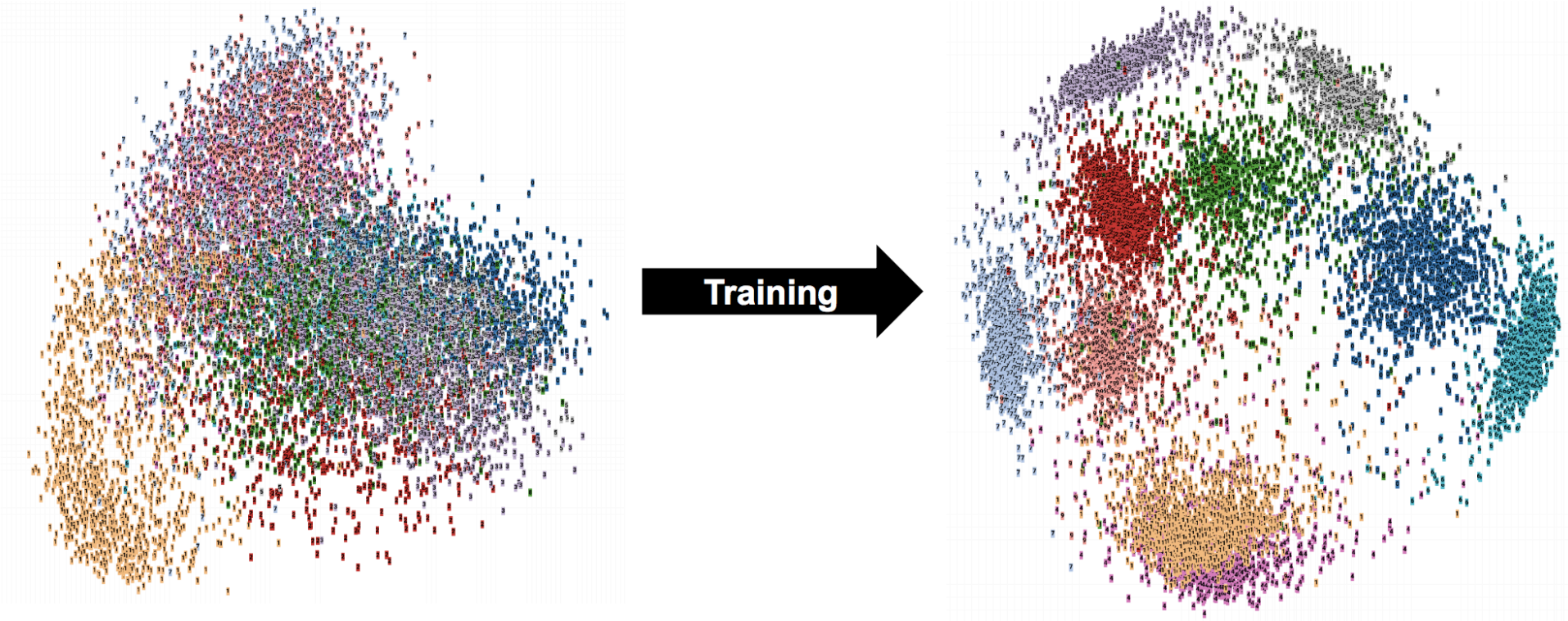
- In order to ensure fast convergence, the following formulation helps to select x_i^n such that

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

Schroff, Florian, Dmitry Kalenichenko, and James Philbin. "FaceNet: A unified embedding for face recognition and clustering." Proceedings of the IEEE conference on computer vision and pattern recognition. 2015.

Triplet Loss (Google FaceNet)

- MNIST feature distribution of training with triplet loss



Schroff, Florian, Dmitry Kalenichenko, and James Philbin. "Facenet: A unified embedding for face recognition and clustering." Proceedings of the IEEE conference on computer vision and pattern recognition. 2015.

Center Loss

- As training approaches, DeepID2 and Triplet Loss respectively construct loss functions for image pairs and triplet. However, compared to the image samples, the number of training pairs or triplets dramatically grows. It inevitably results in slow convergence and instability.
- Center loss also uses identification loss (softmax cross-entropy loss) as one of training supervisory signals:

$$\mathcal{L}_S = -\sum_{i=1}^{m} \log \frac{e^{W_{y_i}^T \boldsymbol{x}_i + b_{y_i}}}{\sum_{j=1}^{n} e^{W_j^T \boldsymbol{x}_i + b_j}}$$

- Besides softmax loss, an auxiliary loss item is added to gather features in their corresponding centers:

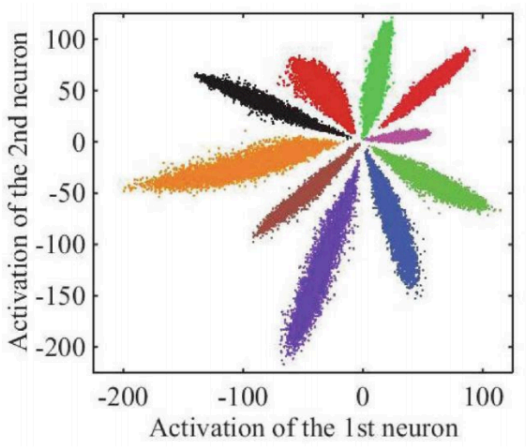
$$\mathcal{L} = \mathcal{L}_S + \lambda \mathcal{L}_C$$

$$= -\sum_{i=1}^{m} \log \frac{e^{W_{y_i}^T \boldsymbol{x}_i + b_{y_i}}}{\sum_{j=1}^{n} e^{W_j^T \boldsymbol{x}_i + b_j}} + \frac{\lambda}{2} \sum_{i=1}^{m} \|\boldsymbol{x}_i - \boldsymbol{c}_{y_i}\|_2^2$$

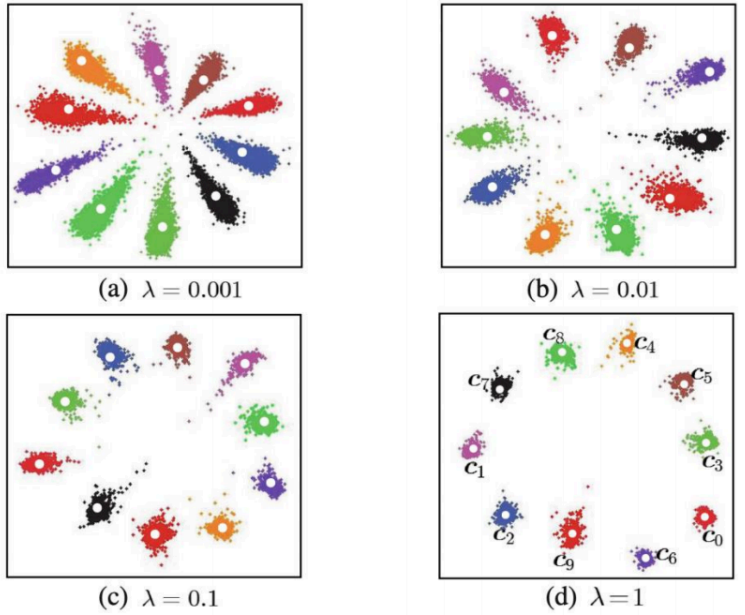
Wen, Yandong, et al. "A discriminative feature learning approach for deep face recognition." European conference on computer vision. Springer, Cham, 2016.

Center Loss

- Feature visualization of softmax and center loss



Softmax Loss only



Softmax combined with Center Loss

Wen, Yandong, et al. "A discriminative feature learning approach for deep face recognition." European conference on computer vision. Springer, Cham, 2016.

Center Loss

- Some results:

Method	Images	Networks	Acc. on LFW	Acc. on YTF
DeepFace [34]	4M	3	97.35 %	91.4 %
DeepID-2+ [32]	-	1	98.70 %	-
DeepID-2+ [32]	-	25	99.47 %	93.2 %
FaceNet [27]	200M	1	99.63 %	95.1 %
Deep FR [25]	2.6M	1	98.95 %	97.3 %
Baidu [21]	1.3M	1	99.13 %	-
model A	0.7M	**1**	97.37 %	91.1 %
model B	0.7M	**1**	99.10 %	93.8 %
model C (Proposed)	**0.7M**	**1**	**99.28 %**	**94.9 %**

Verification performance of different methods on LFW and YTF datasets

Barebones_FR - cnn	Small	59.363 %
NTechLAB - facenx_small	Small	58.218 %
3DiVi Company - tdvm6	Small	33.705 %
Model A-	Small	41.863 %
Model B-	Small	57.175 %
Model C- (Proposed)	**Small**	**65.234 %**

Identification rates of different methods on MegaFace with 1M distractors

Wen, Yandong, et al. "A discriminative feature learning approach for deep face recognition." European conference on computer vision. Springer, Cham, 2016.

Some Cosine metric based losses

- Large-Margin Softmax Loss for Convolutional Neural Networks (ICML 2016)
- SphereFace: Deep Hypersphere Embedding for Face Recognition （CVPR 2017）
- NormFace: L2 Hypersphere Embedding for Face Verification (ACM-MM 2017)
- CosFace: Large Margin Cosine Loss for Deep Face Recognition (CVPR 2018)
- ArcFace: Additive Angular Margin Loss for Deep Face Recognition (CVPR 2019)
- AdaCos: Adaptively Scaling Cosine Logit for Learning Deep Face Representation (CVPR 2019)

A-Softmax (SphereFace)

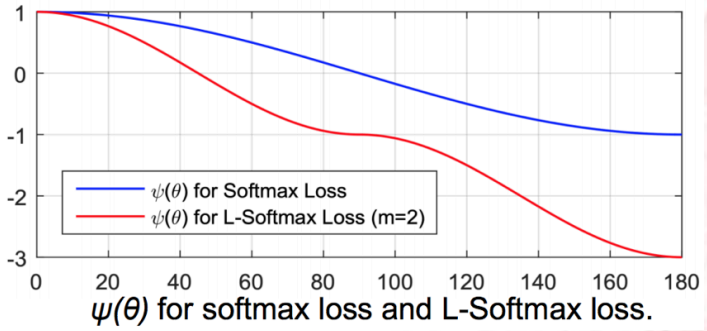
$$\mathcal{L}_S = -\sum_{i=1}^{m} \log \frac{e^{W_{y_i}^T \boldsymbol{x}_i + b_{y_i}}}{\sum_{j=1}^{n} e^{W_j^T \boldsymbol{x}_i + b_j}}$$

- A-Softmax normalized all class weights so that maps them into a hypersphere. This can make cosine metric more nature.

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

$$\psi(\theta) = \begin{cases} \cos(m\theta), & 0 \le \theta \le \frac{\pi}{m} \\ \mathcal{D}(\theta), & \frac{\pi}{m} < \theta \le \pi \end{cases}$$

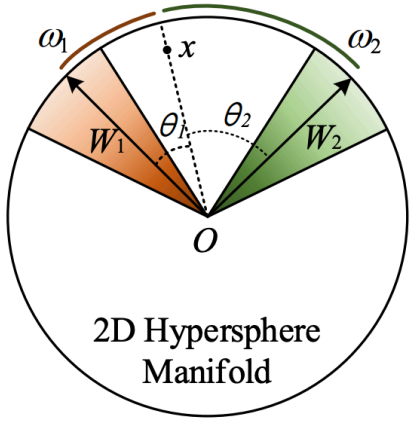
$$L_i = -\log\left(\frac{e^{\|\boldsymbol{x}_i\|\psi(\theta_{y_i,i})}}{e^{\|\boldsymbol{x}_i\|\psi(\theta_{y_i,i})} + \sum_{j\neq y_i} e^{\|\boldsymbol{x}_i\|\cos(\theta_{j,i})}}\right)$$

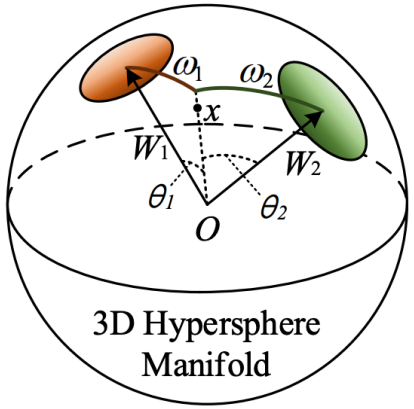


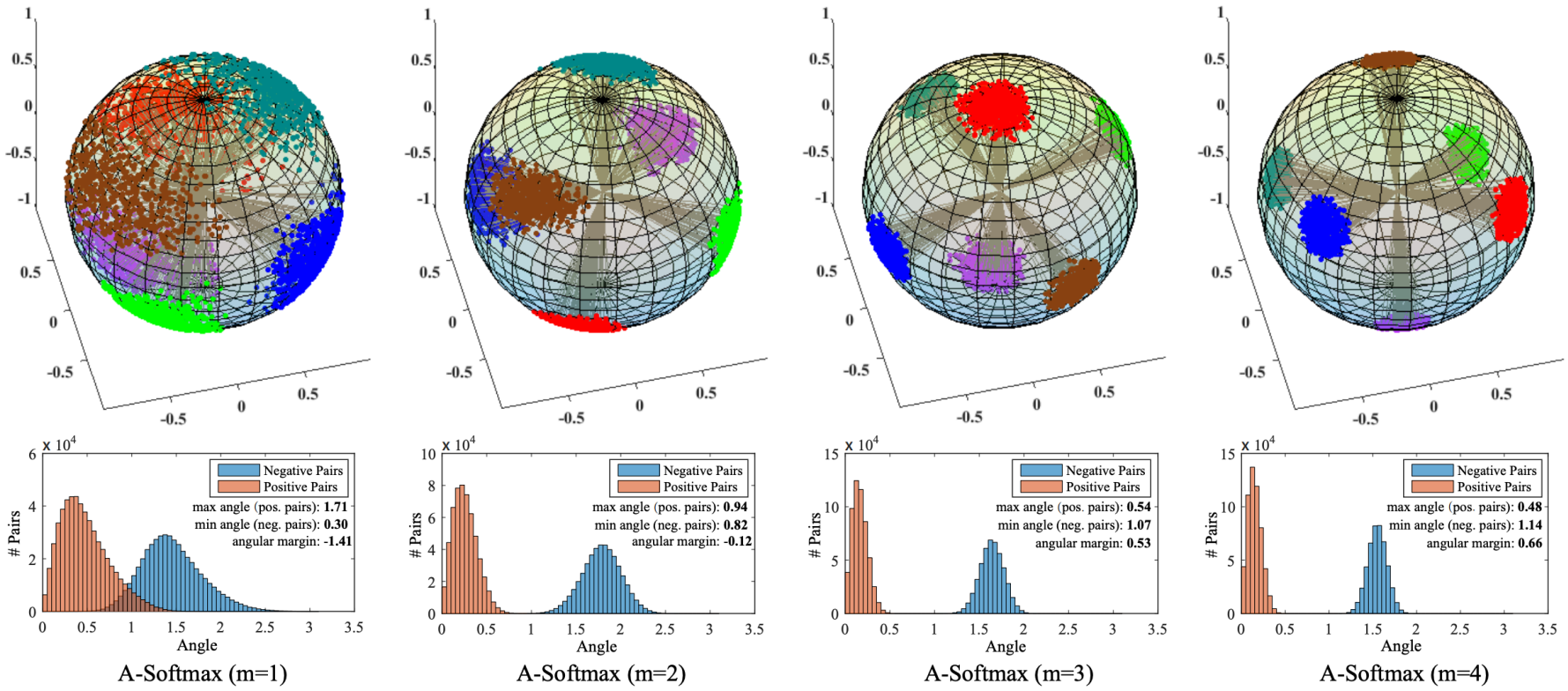
ψ(θ) for softmax loss and L-Softmax loss.

Liu, Weiyang, et al. "Sphereface: Deep hypersphere embedding for face recognition." Proceedings of the IEEE conference on computer vision and pattern recognition. 2017.

A-Softmax (SphereFace)







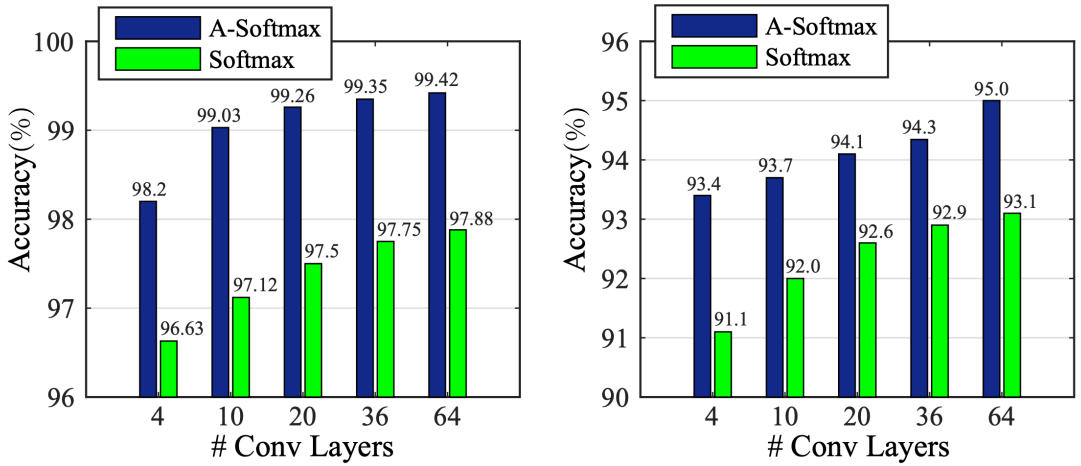
- Visualization of features learned with different m by using a 6-class subset of the CASIA-WebFace dataset.

Liu, Weiyang, et al. "Sphereface: Deep hypersphere embedding for face recognition." Proceedings of the IEEE conference on computer vision and pattern recognition. 2017.

A-Softmax (SphereFace)

Softmax Loss	Small	54.855	65.925
Softmax+Contrastive Loss [26]	Small	65.219	78.865
Triplet Loss [22]	Small	64.797	78.322
L-Softmax Loss [16]	Small	67.128	80.423
Softmax+Center Loss [34]	Small	65.494	80.146
SphereFace (single model)	Small	**72.729**	**85.561**
SphereFace (3-patch ensemble)	Small	**75.766**	**89.142**

Performance (%) on MegaFace challenge.



Accuracy (%) on LFW and YTF with different number of convolutional layers. Left side is for LFW, while right side is for YTF.

Method	Models	Data	LFW	YTF
DeepFace [30]	3	4M*	97.35	91.4
FaceNet [22]	1	200M*	**99.65**	95.1
Deep FR [20]	1	2.6M	98.95	**97.3**
DeepID2+ [27]	1	300K*	98.70	N/A
DeepID2+ [27]	25	300K*	99.47	93.2
Baidu [15]	1	1.3M*	99.13	N/A
Center Face [34]	1	0.7M*	99.28	94.9
Yi et al. [37]	1	WebFace	97.73	92.2
Ding et al. [2]	1	WebFace	98.43	N/A
Liu et al. [16]	1	WebFace	98.71	N/A
Softmax Loss	1	WebFace	97.88	93.1
Softmax+Contrastive [26]	1	WebFace	98.78	93.5
Triplet Loss [22]	1	WebFace	98.70	93.4
L-Softmax Loss [16]	1	WebFace	99.10	94.0
Softmax+Center Loss [34]	1	WebFace	99.05	94.4
SphereFace	1	WebFace	**99.42**	**95.0**

Accuracy (%) on LFW and YTF dataset.

Liu, Weiyang, et al. "Sphereface: Deep hypersphere embedding for face recognition." Proceedings of the IEEE conference on computer vision and pattern recognition. 2017.

NormFace

$$\mathcal{L}_S = -\sum_{i=1}^{m} \log \frac{e^{W_{y_i}^T \boldsymbol{x}_i + b_{y_i}}}{\sum_{j=1}^{n} e^{W_j^T \boldsymbol{x}_i + b_j}}$$

- NormFace normalized both features and class weights in softmax cross-entropy loss so that all logits in softmax become cosine metirc.

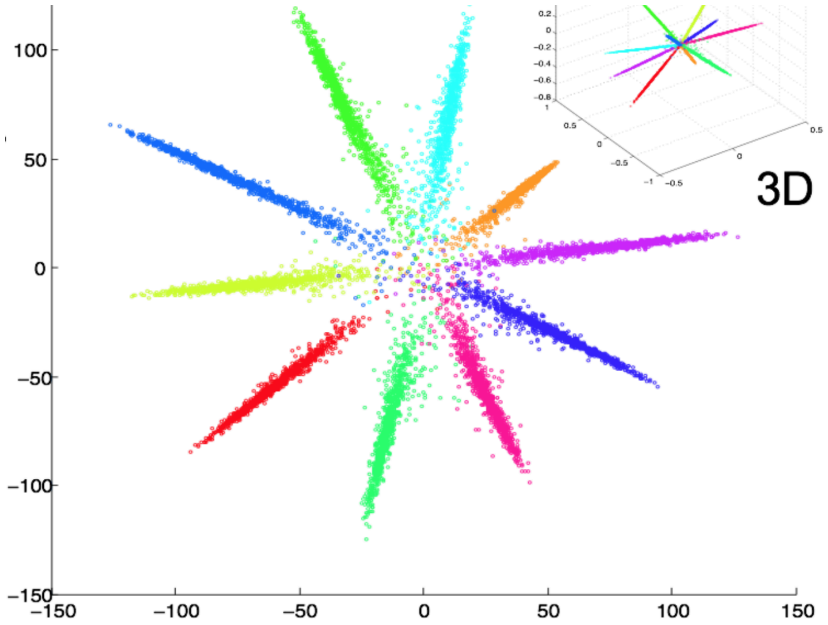
$$f_{i,j} = s \cdot \cos\theta_{i,j} \qquad P_{i,j} = \frac{e^{f_{i,j}}}{\sum_{k=1}^{C} e^{f_{i,k}}}$$

$$\mathcal{L}_{\text{CE}} = -\frac{1}{N}\sum_{i=1}^{N} \log P_{i,y_i} = -\frac{1}{N}\sum_{i=1}^{N} \log \frac{e^{f_{i,y_i}}}{\sum_{k=1}^{C} e^{f_{i,k}}}$$

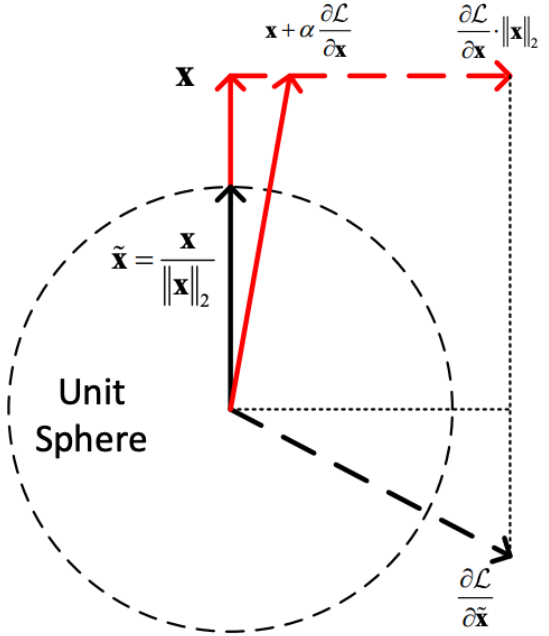
where hyper parameter s is the scaling parameter that enlarges the range of cosine logits

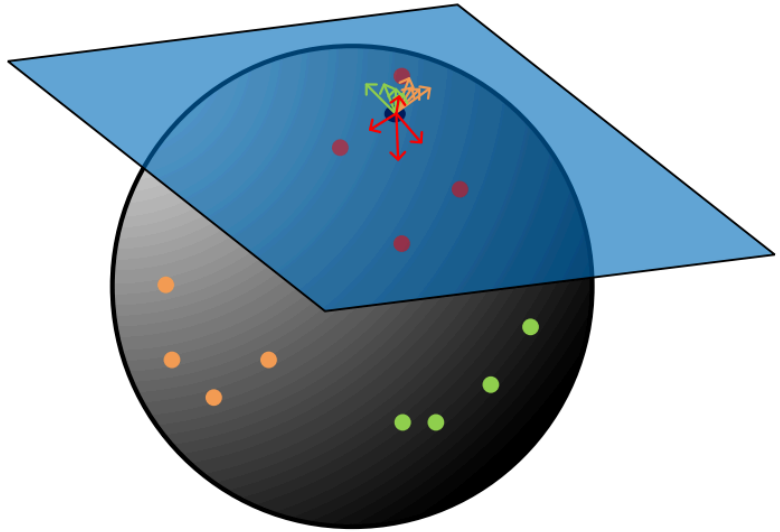
Wang, Feng, et al. "NormFace: l 2 hypersphere embedding for face verification." Proceedings of the 25th ACM international conference on Multimedia. ACM, 2017.

NormFace



MNIST 2-D feature visualization.





The normalization operation and its gradient in 2-dimensional space.

Wang, Feng, et al. "Normface: l 2 hypersphere embedding for face verification." Proceedings of the 25th ACM international conference on Multimedia. ACM, 2017.

NormFace

loss function	Normalization	Accuracy
softmax	No	98.28%
softmax + dropout	No	98.35%
softmax + center[36]	No	99.03%
softmax	feature only	98.72%
softmax	weight only	98.95%
softmax	Yes	99.16% ± 0.025%
softmax + center	Yes	99.17% ± 0.017%
C-contrasitve	Yes	99.15% ± 0.017%
C-triplet	Yes	99.11% ± 0.008%
C-triplet + center	Yes	99.13% ± 0.017%
softmax + C-contrastive	Yes	99.19% ± 0.008%

LFW resutls. Here normalization indicate w/o NormFace operation.

loss function	Normalization	Accuracy
softmax + center[36]	No	93.74%
softmax	Yes	94.24%
softmax + HIK-SVM	Yes	94.56%
C-triplet + center	Yes	94.3%
C-triplet + center + HIK-SVM	Yes	94.58%
softmax + C-contrastive	Yes	94.34%
softmax + C-contrastive + HIK-SVM	Yes	94.72%

YTF resutls. Here normalization indicate w/o NormFace operation.

Wang, Feng, et al. "Normface: l 2 hypersphere embedding for face verification." Proceedings of the 25th ACM international conference on Multimedia. ACM, 2017.

CosFace & ArcFace

- Both CosFace and ArcFace add a margin hyperparameter in NormFace. Their difference is the location where margins are added. For CosFace, margin is added on cosine metrics:

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

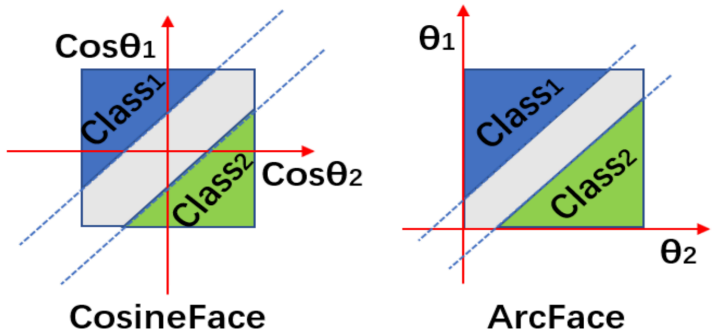
- For ArcFace, margin is added on angular:

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

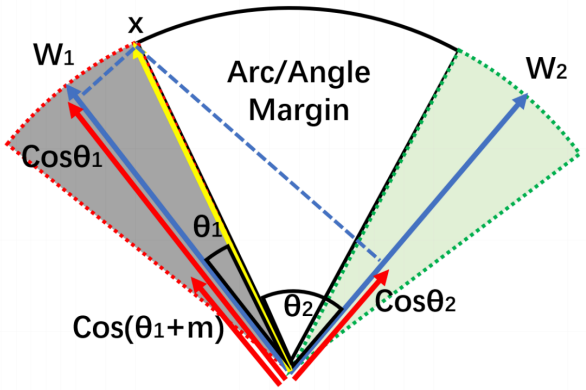
- Although their formulations are different, their main ideas are same.

Wang, Hao, et al. "Cosface: Large margin cosine loss for deep face recognition." Proceedings of the IEEE CVPR. 2018.
Deng, Jiankang, et al. "Arcface: Additive angular margin loss for deep face recognition." Proceedings of the IEEE CVPR. 2019.

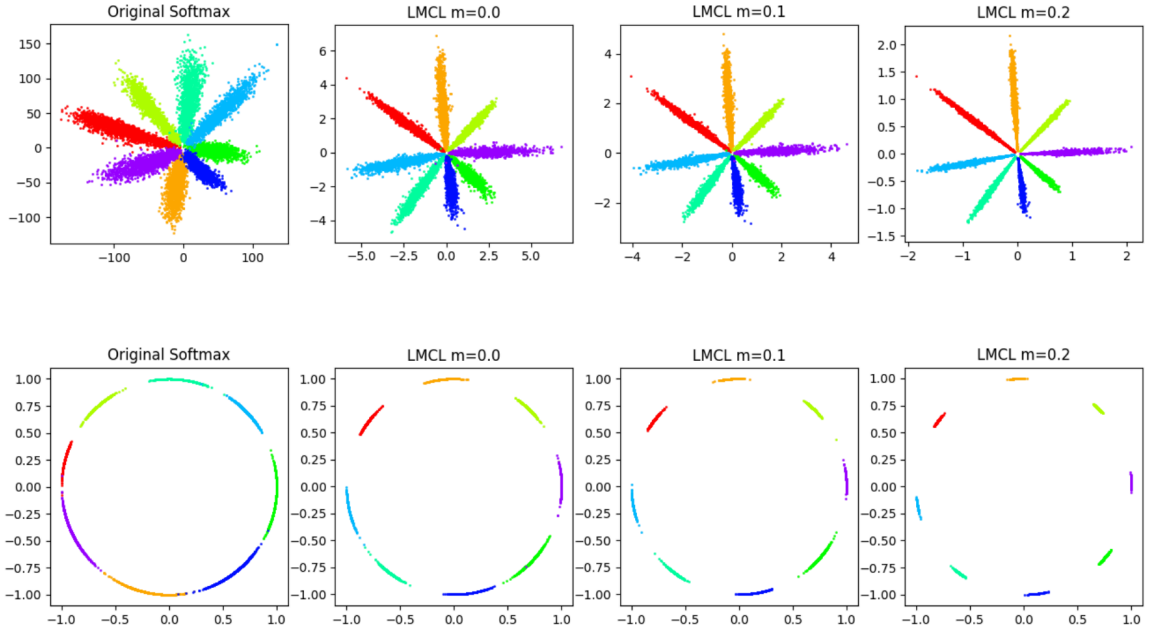
CosFace & ArcFace



Decision margins of CosFace and ArcFace under binary classification case.



Geometrical interpretation of ArcFace.



A toy experiment of CosFace with different margin on 8 identities with 2D features. The first row maps the 2D features onto the Euclidean space, while the second row projects the 2D features onto the angular space. The gap becomes evident as the margin term margin increases

Wang, Hao, et al. "Cosface: Large margin cosine loss for deep face recognition." Proceedings of the IEEE CVPR. 2018.
Deng, Jiankang, et al. "Arcface: Additive angular margin loss for deep face recognition." Proceedings of the IEEE CVPR. 2019.

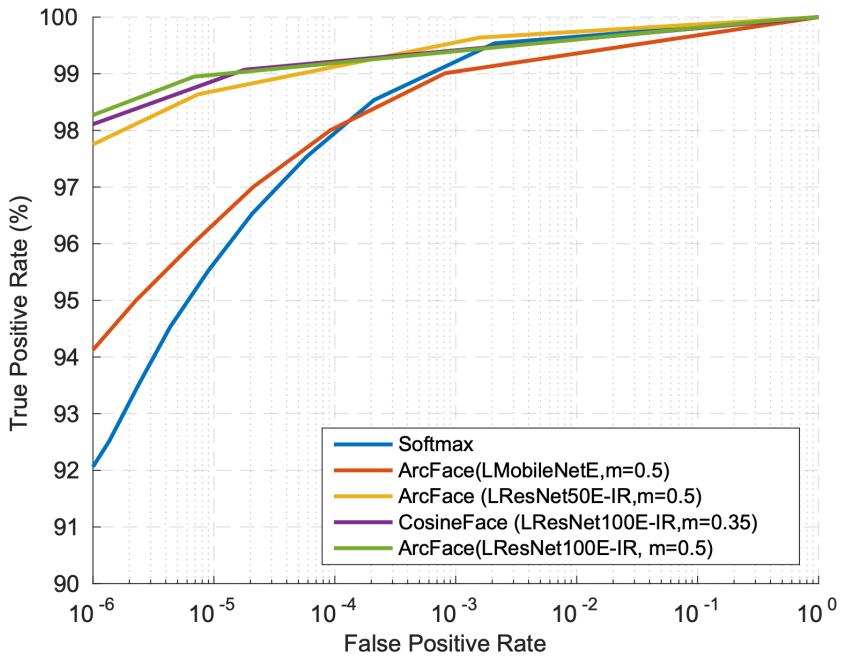
CosFace & ArcFace

Method	LFW	YTF	MF1 Rank1	MF1 Veri.
Softmax Loss [23]	97.88	93.1	54.85	65.92
Softmax+Contrastive [30]	98.78	93.5	65.21	78.86
Triplet Loss [29]	98.70	93.4	64.79	78.32
L-Softmax Loss [24]	99.10	94.0	67.12	80.42
Softmax+Center Loss [42]	99.05	94.4	65.49	80.14
A-Softmax [23]	**99.42**	95.0	72.72	85.56
A-Softmax-NormFea	99.32	95.4	75.42	88.82
LMCL	99.33	**96.1**	**77.11**	**89.88**

Comparison of the proposed CosFace with state-of-the-art loss functions in face recognition community.

Method	Protocol	MF2 Rank1	MF2 Veri.
3DiVi	Large	57.04	66.45
Team 2009	Large	58.93	71.12
NEC	Large	62.12	66.84
GRCCV	Large	75.77	74.84
SphereFace	Large	71.17	84.22
CosFace (Single-patch)	Large	74.11	86.77
CosFace(3-patch ensemble)	Large	**77.06**	**90.30**

CosFace identification and verification evaluation on MegaFace C2.



Methods	Rank1@10^6	VR@FAR10^{-6}
Softmax	78.89	94.95
Softmax-pretrain,Triplet-finetune	80.6	94.65
Softmax-pretrain@VGG2, Triplet-finetune	78.87	95.43
SphereFace(m=4, λ=5)	82.95	97.66
CosineFace(m=0.35)	82.75	98.41
ArcFace(m=0.4)	82.29	98.20
ArcFace(m=0.5)	**83.27**	**98.48**

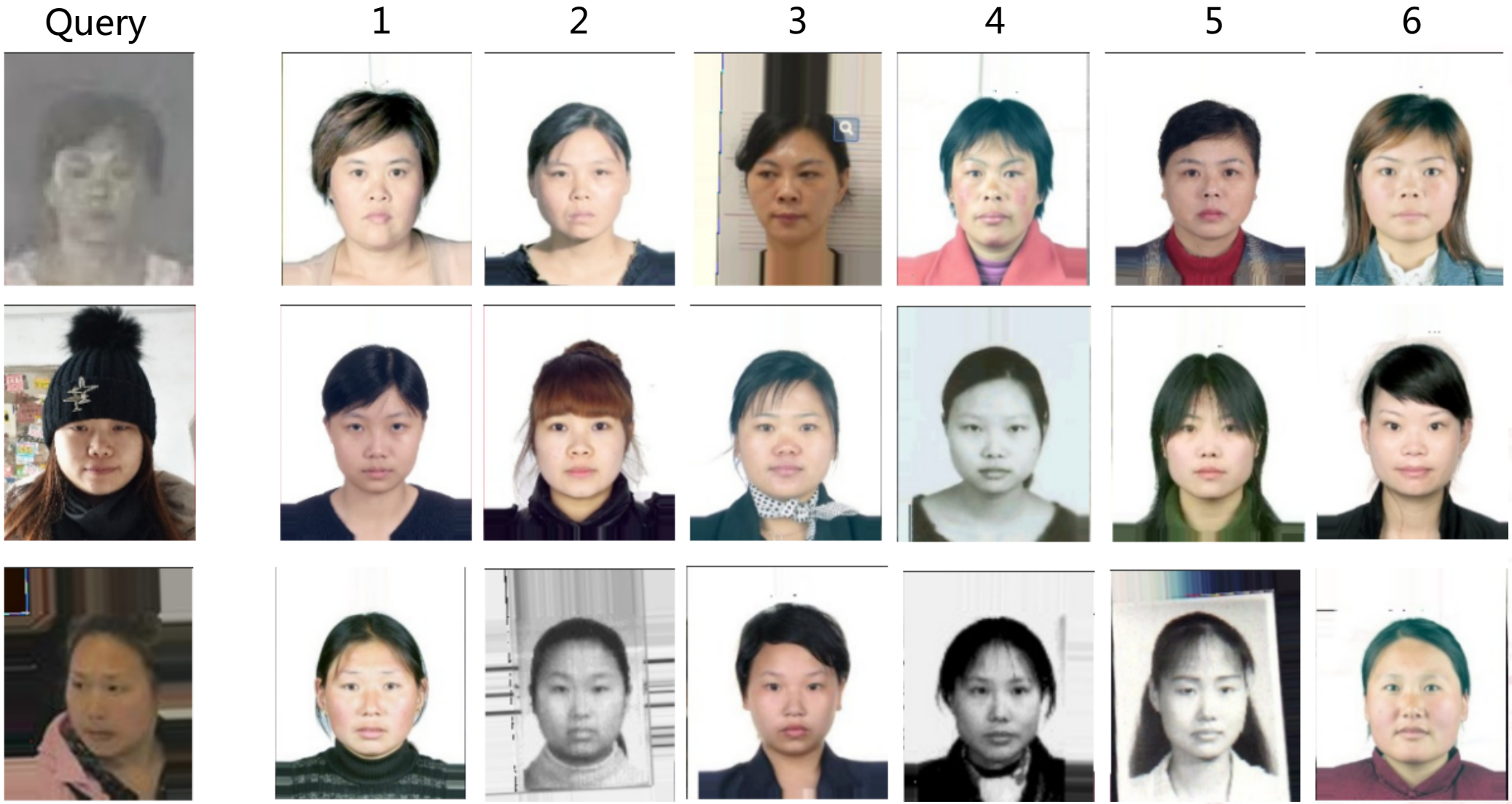
Identification and verification results of different methods on MegaFace C1.

Wang, Hao, et al. "Cosface: Large margin cosine loss for deep face recognition." Proceedings of the IEEE CVPR. 2018.
Deng, Jiankang, et al. "Arcface: Additive angular margin loss for deep face recognition." Proceedings of the IEEE CVPR. 2019.

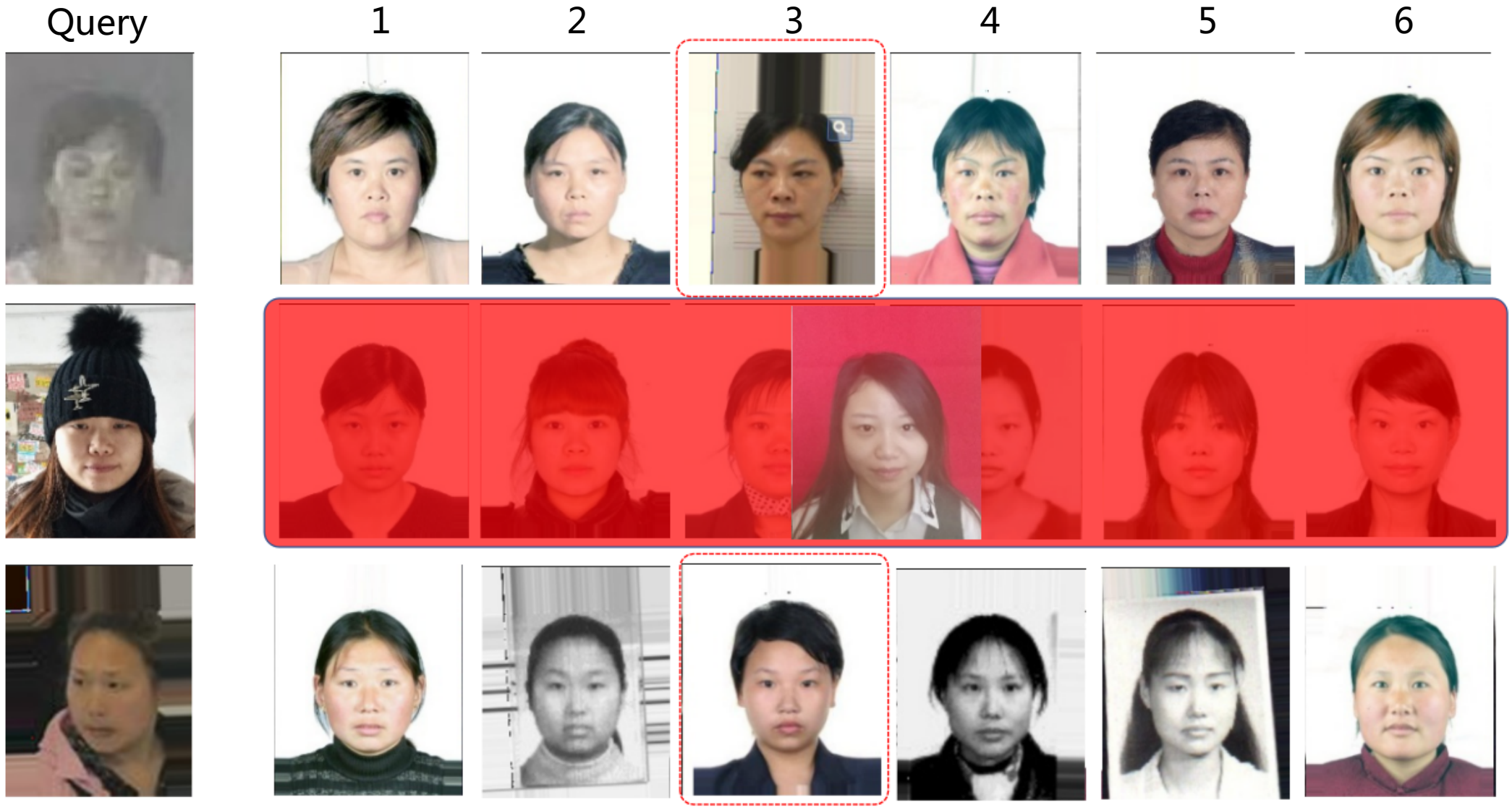
Outline——Face Recognition

- Introduction
- Literatures
- Practical problems when widely applied
- Efforts to push the limit

Practical problems

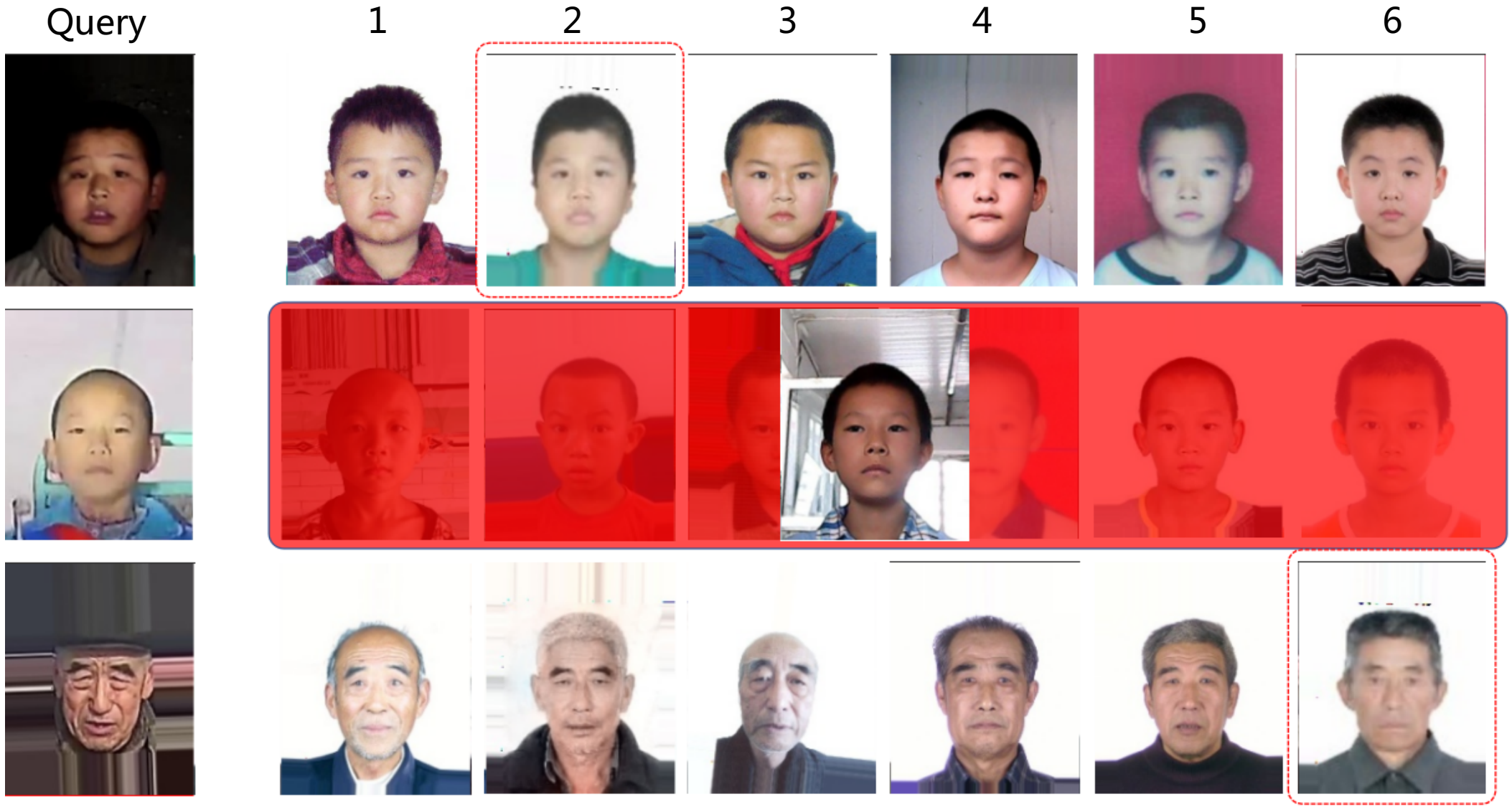


Practical problems



Practical problems

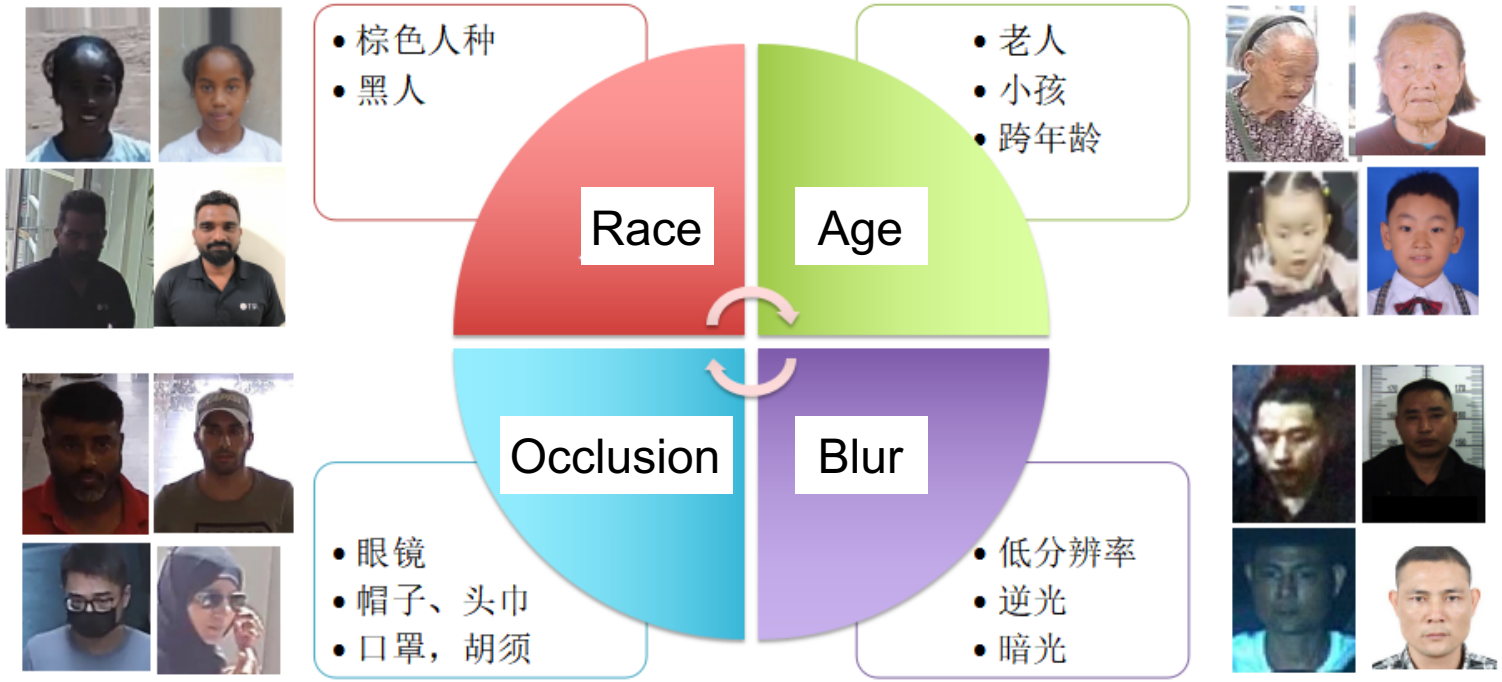
Practical problems



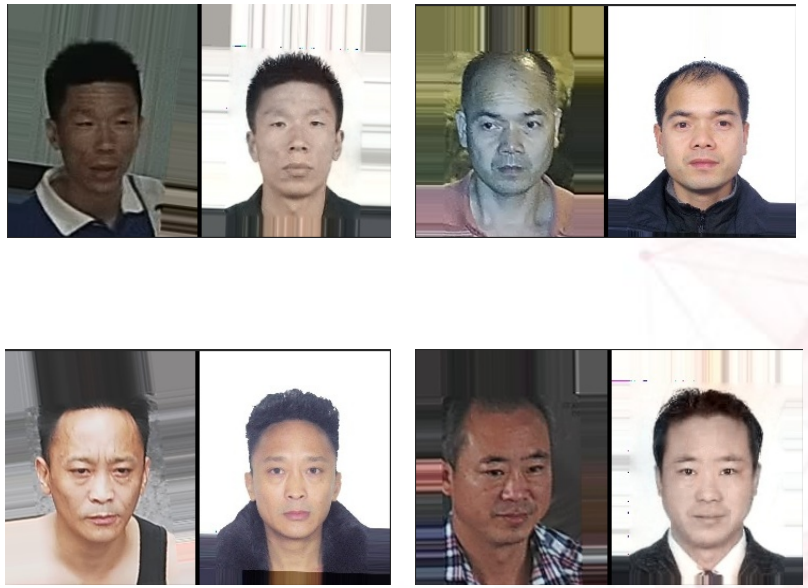
Practical problems

- Distance between academic and industry settings.

Low Intra-Class Similarity



High Inter-Class Similarity

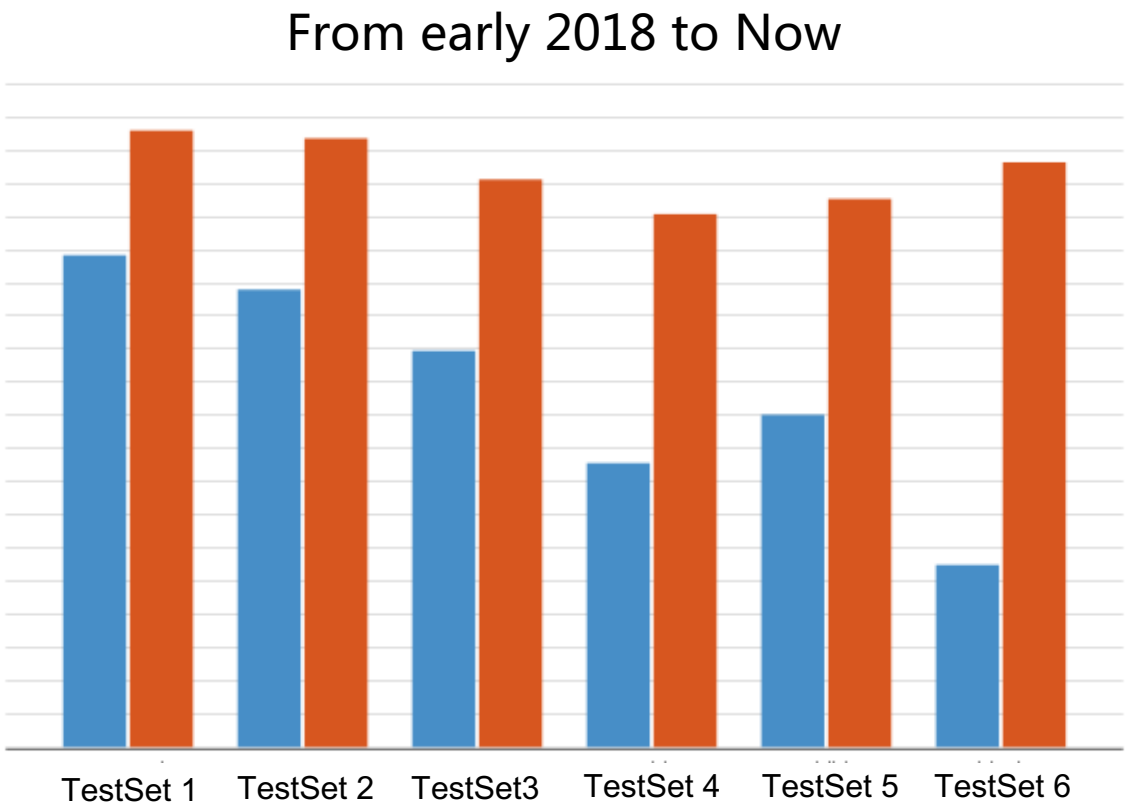


Outline——Face Recognition

- Introduction
- Literatures
- Practical problems when widely applied
- Efforts to push the limit

Efforts to push the limit

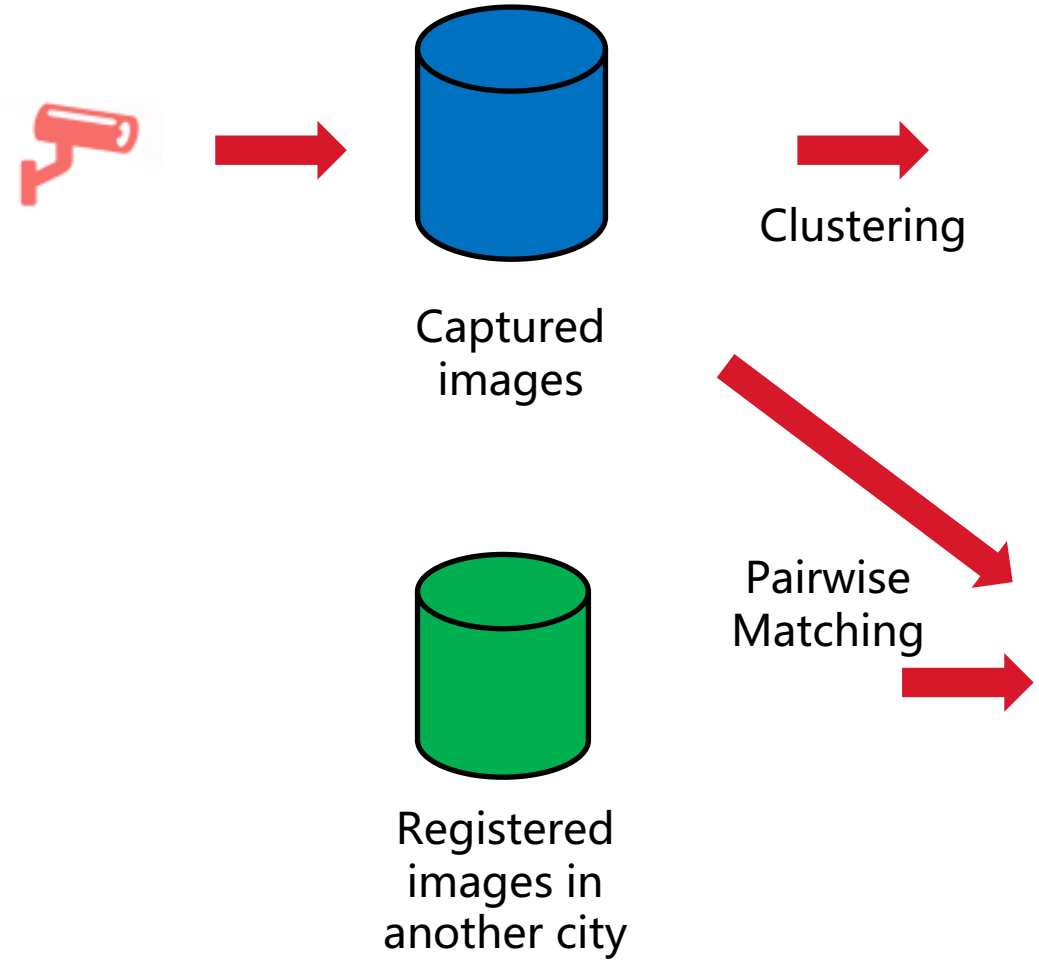
- Accuracy improvement in last year.

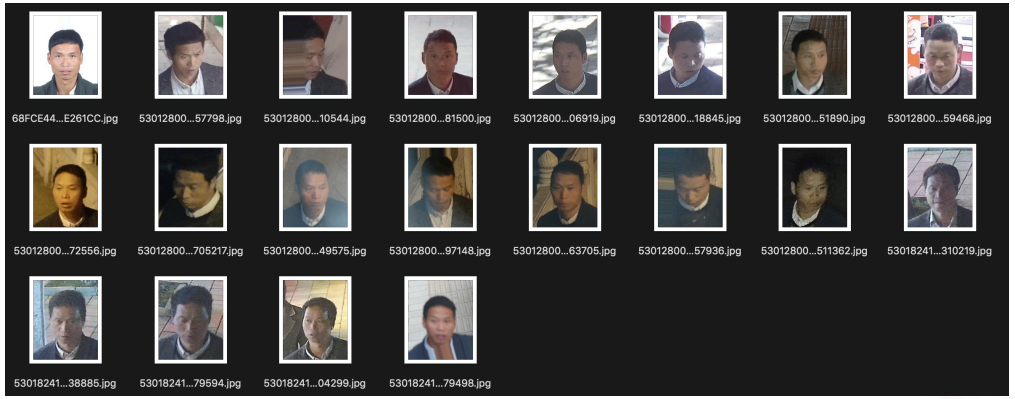


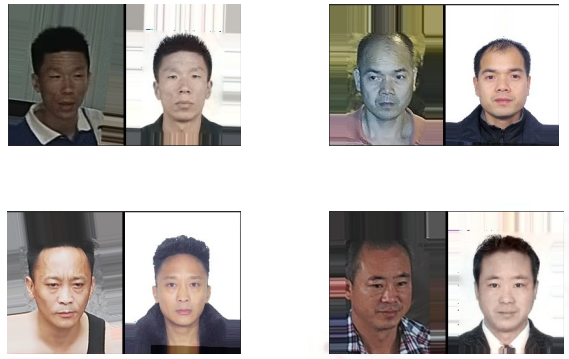
- Data Analysis
- Network architecture & loss design
- Engineering

Efforts to push the limit

- Unlabeled data







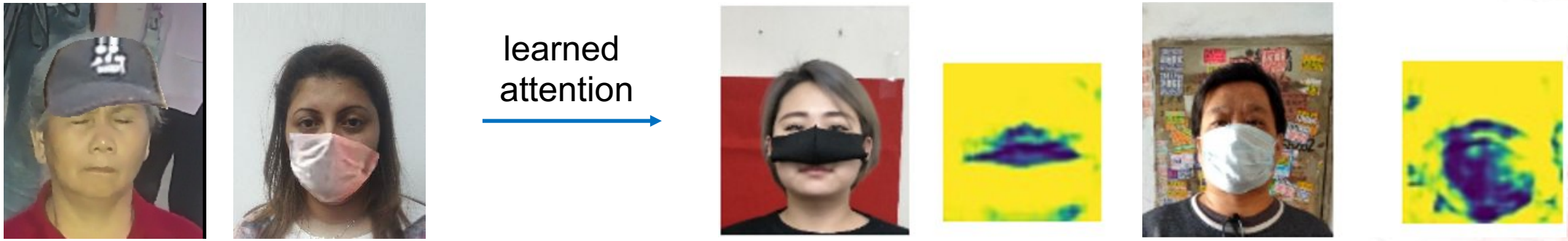
Efforts to push the limit

- Data Augmentation

cutout [1]

Random Mask

Template Mask

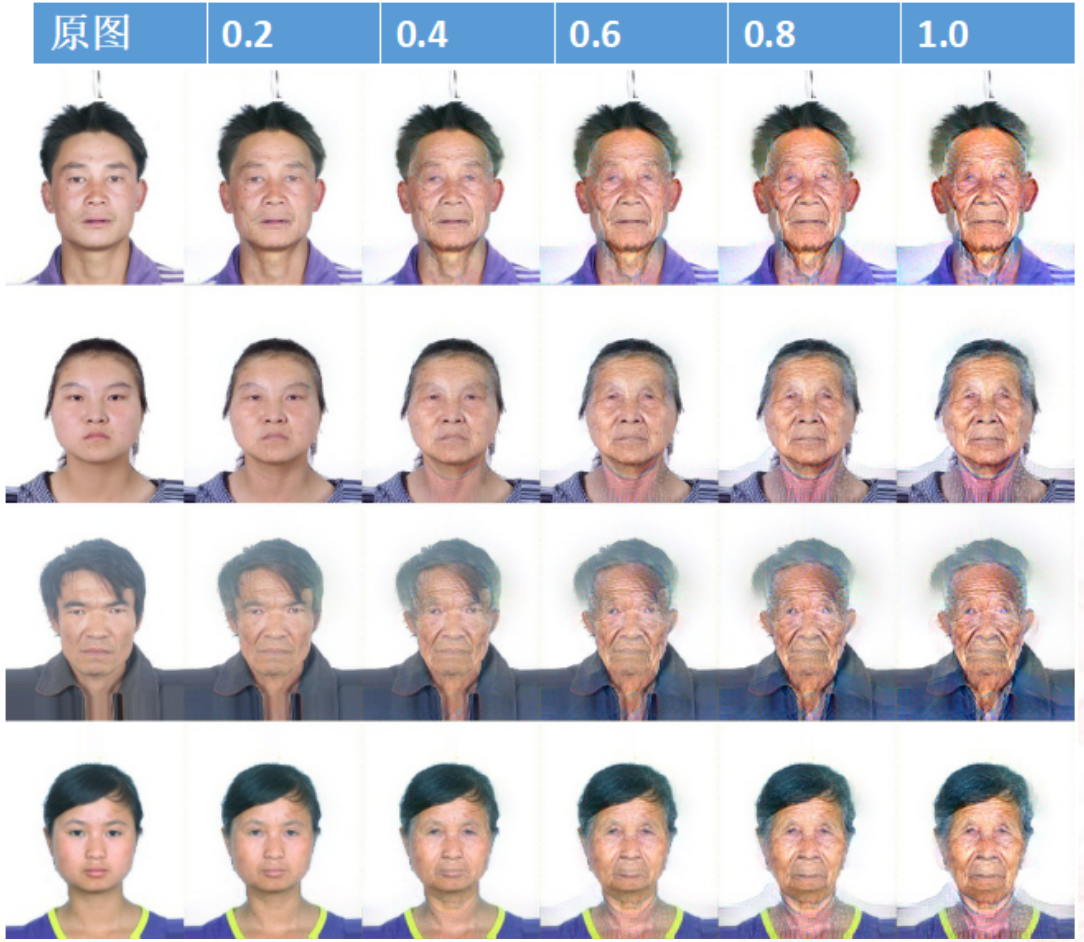


[1] Terrance DeVries and Graham W. Taylor, Improved Regularization of Convolutional Neural Networks with Cutout, arXiv:1708.04552v2, 2017.

Efforts to push the limit

- Data Generation

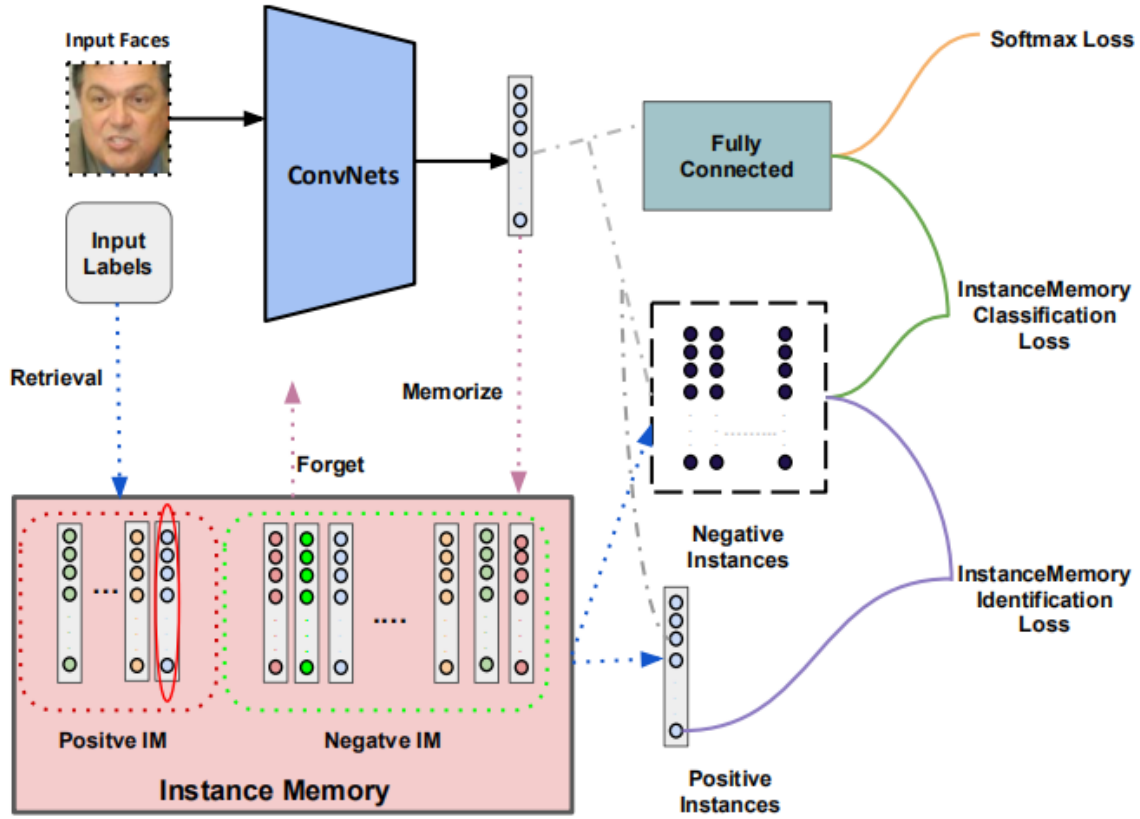
Real Image GAN Image



[1] Tero Karras, Samuli Laine, Timo Aila, “A Style-Based Generator Architecture for Generative Adversarial Networks”. CVPR 2019.
[2] Paul Upchurch1,* Jacob Gardner1,* Geoff Pleiss1 Robert Pless2 Noah Snavely1 Kavita Bala1, “Deep Feature Interpolation for Image Content Changes”, CVPR 2017.

Efforts to push the limit

- Optimization Objective Design

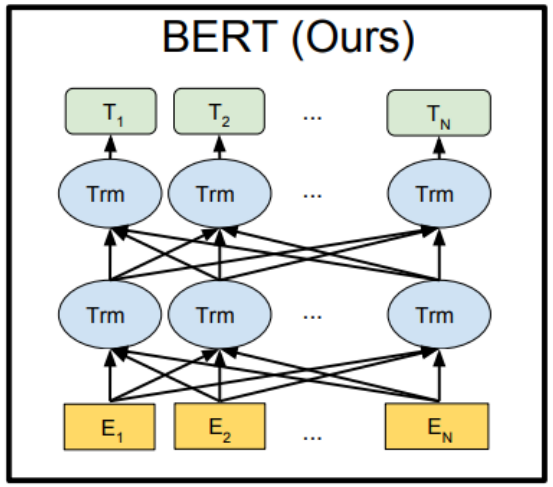


Method	LFW	CPLFW	CALFW
Human	97.27	81.21	82.32
CosFace	99.75	92.10	95.98
ArcFace	99.80	92.50	95.98
IM_IDT(Cos)	99.78	92.28	96.03
IM_IDT(Arc)	99.80	92.42	96.10
IM_CLS(Arc)	**99.83**	**93.00**	**96.15**

Efforts to push the limit

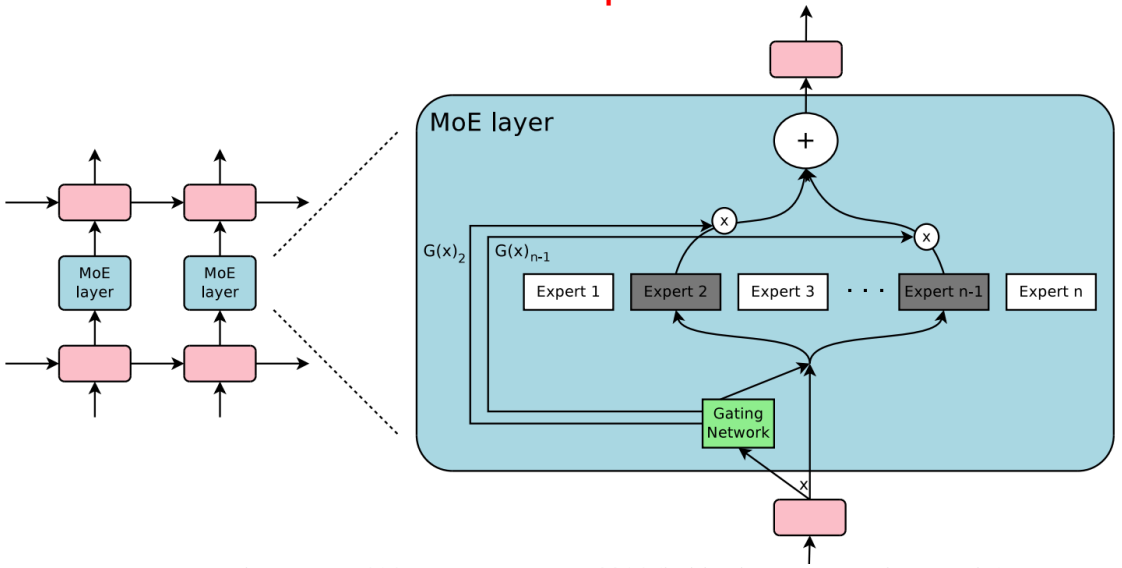
- Train outrageously large model?

BERT: 340M Params



System	MNLI-(m/mm) 392k	QQP 363k	QNLI 108k	SST-2 67k	CoLA 8.5k	STS-B 5.7k	MRPC 3.5k	RTE 2.5k	Average -
Pre-OpenAI SOTA	80.6/80.1	66.1	82.3	93.2	35.0	81.0	86.0	61.7	74.0
BiLSTM+ELMo+Attn	76.4/76.1	64.8	79.9	90.4	36.0	73.3	84.9	56.8	71.0
OpenAI GPT	82.1/81.4	70.3	88.1	91.3	45.4	80.0	82.3	56.0	75.2
$BERT_{BASE}$	84.6/83.4	71.2	90.1	93.5	52.1	85.8	88.9	66.4	79.6
$BERT_{LARGE}$	**86.7/85.9**	**72.1**	**91.1**	**94.9**	**60.5**	**86.5**	**89.3**	**70.1**	**81.9**

MoE with 2048 Experts: 8.7B Params



Results on WMT'14 En → De newstest2014 (bold values represent best results).

Model	Test Perplexity	Test BLEU	ops/timestep	Total #Parameters	Training Time
MoE with 2048 Experts	**4.64**	**26.03**	85M	8.7B	1 day/64 k40s
GNMT (Wu et al., 2016)	5.25	24.91	214M	278M	1 day/96 k80s
GNMT +RL (Wu et al., 2016)	8.08	24.66	214M	278M	1 day/96 k80s
PBMT (Durrani et al., 2014)		20.7			
DeepAtt (Zhou et al., 2016)		20.6			

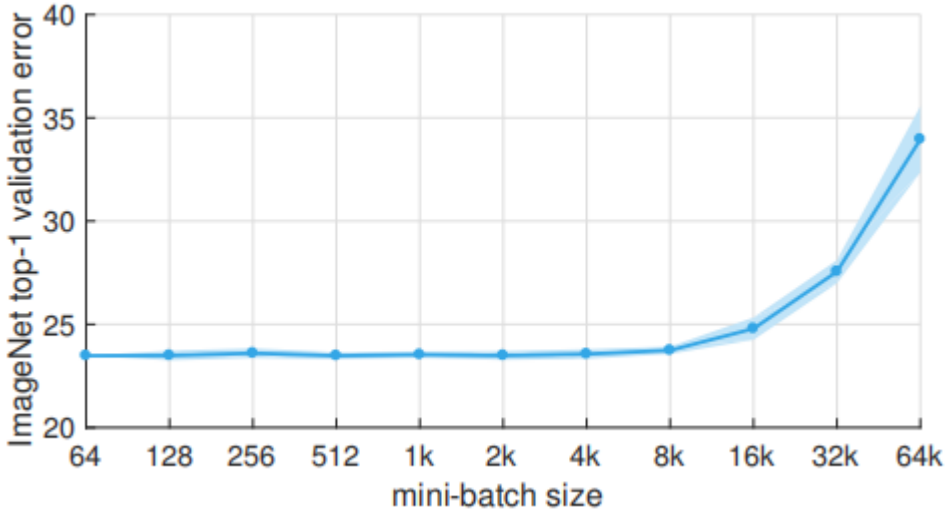
[1] Jacob Devlin Ming-Wei Chang Kenton Lee Kristina Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding", arXiv:1810.04805v1, 2018.
[2] N. Shazeer, et al, OUTRAGEOUSLY LARGE NEURAL NETWORKS: THE SPARSELY-GATED MIXTURE-OF-EXPERTS LAYER, arXiv: 1701.06538, 2017.

Efforts to push the limit

14000 GPUs in SenseTime

model	TestSet1	TestSet2
Baseline	85.99%	76.59%
Baseline with 2x params	88.01%	79.40%

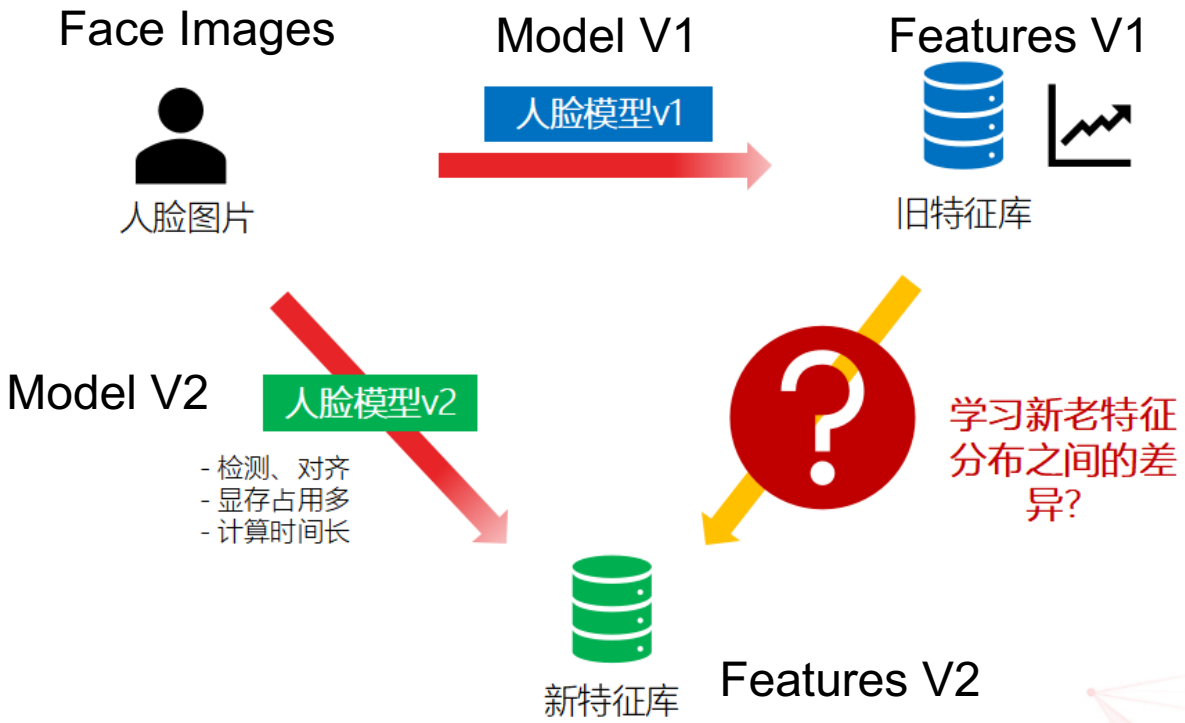
- How to train outrageously large model ?
  - Data parallelism: gradient compression, asynchronous BP
  - Model parallelism: distributed layer params
  - Limited-precision training (fp16 and int8)
  - Optimization issue: convergence with large batch size, massive number of classes, etc.

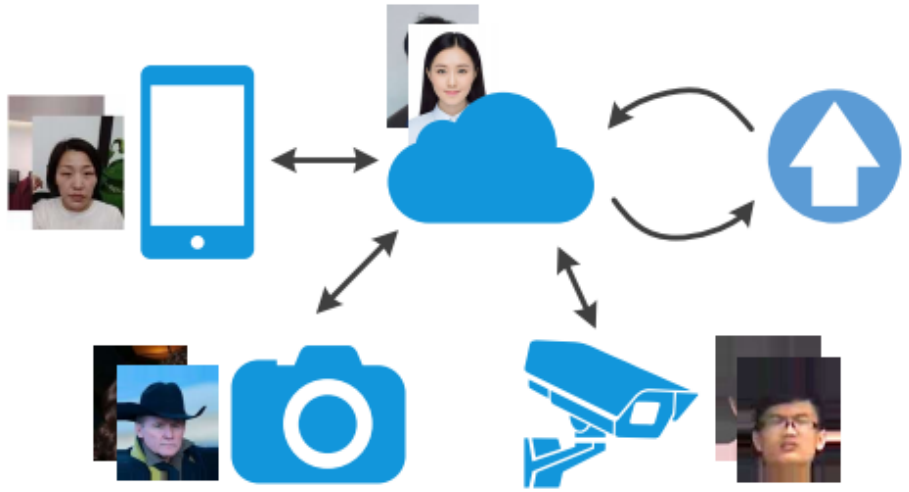


[1] Priya Goyal et. al. Accurate, Large Minibatch SGD: Training ImageNet in 1 Hour, arXiv:1706.02677v2, 2017.

Version Upgrade for Massive Features

- Problem definition



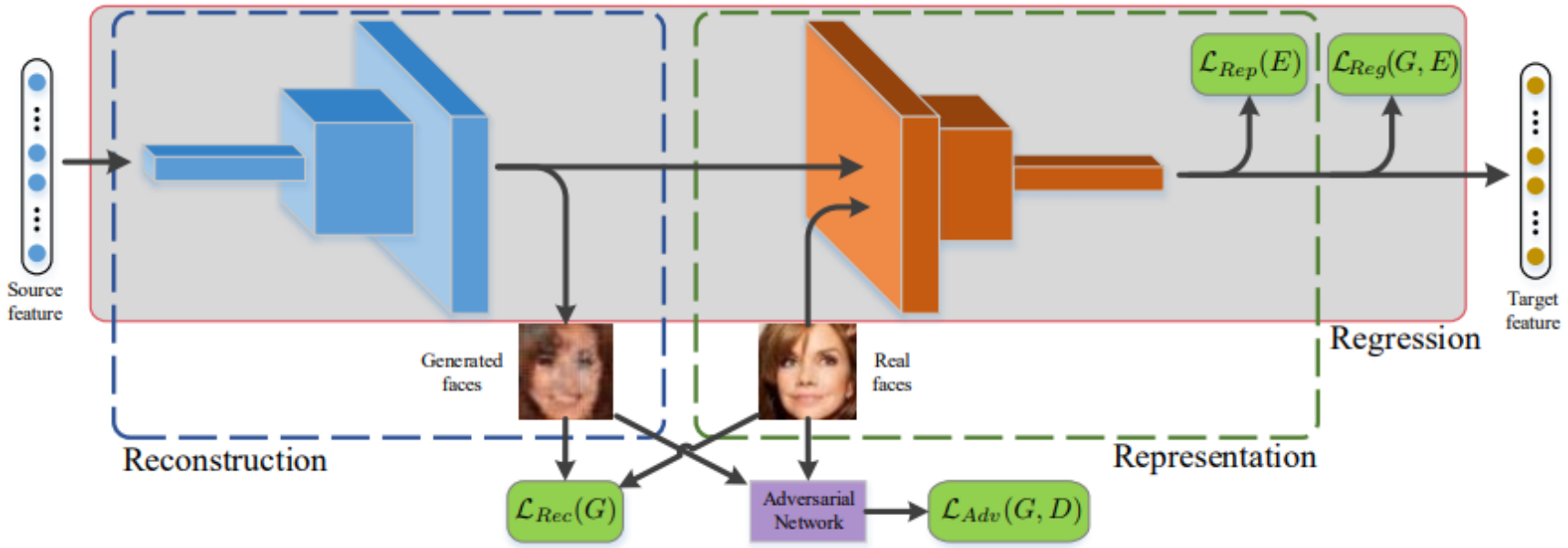


For example in a city with **100 thousand** cameras, about **200 billion** face images are captured within six months. If we re-extract features for these images with a single 8-P4 GPU server, it would cost **6.4 years**.

How to calculate similarity score between features from different devices / models?

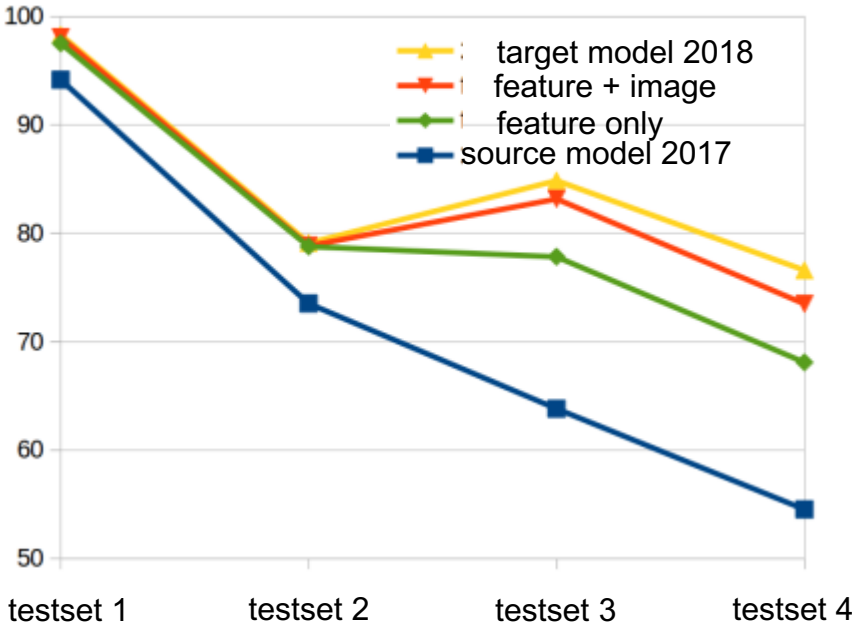
Version Upgrade for Massive Features

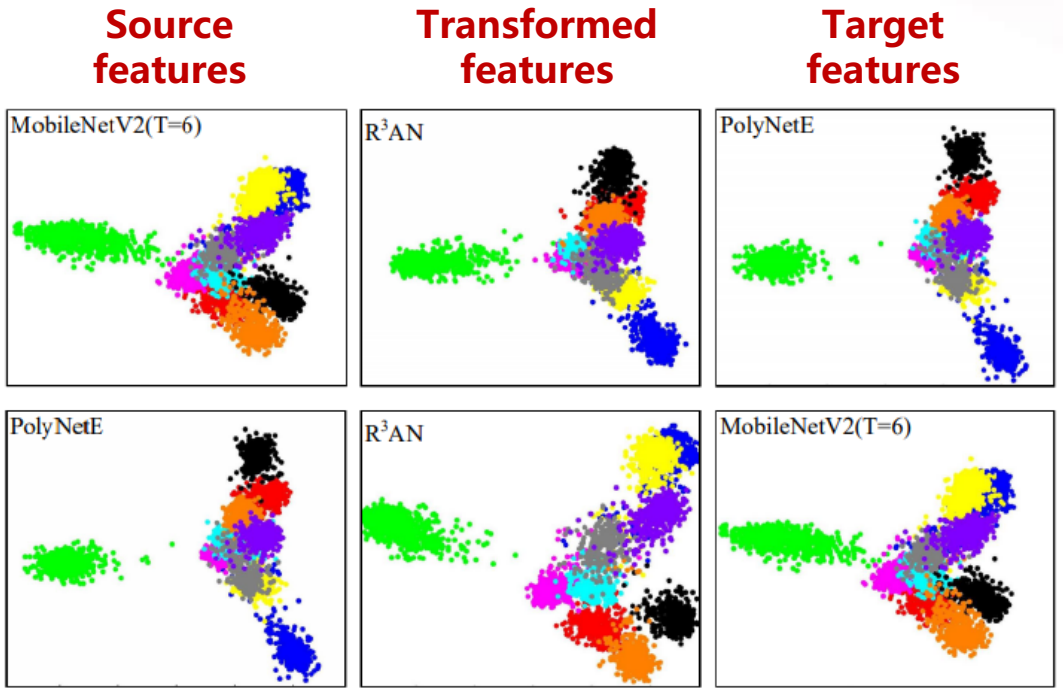
- Methodology
  - Adversarial loss for face generation
  - Reconstruction loss
  - Representation learning
  - Feature regression



Version Upgrade for Massive Features

- Experimental results





Example scenarios	#features	Extract feature directly	Upgrade Time
Images per individual	1.4 billion	16 days	23 min
100 thousand cameras in six months	200 billion	6.4 years	2.5 days

* Time cost is calculated based on using a 8xP4 GPU server.

Version Upgrade for Massive Features

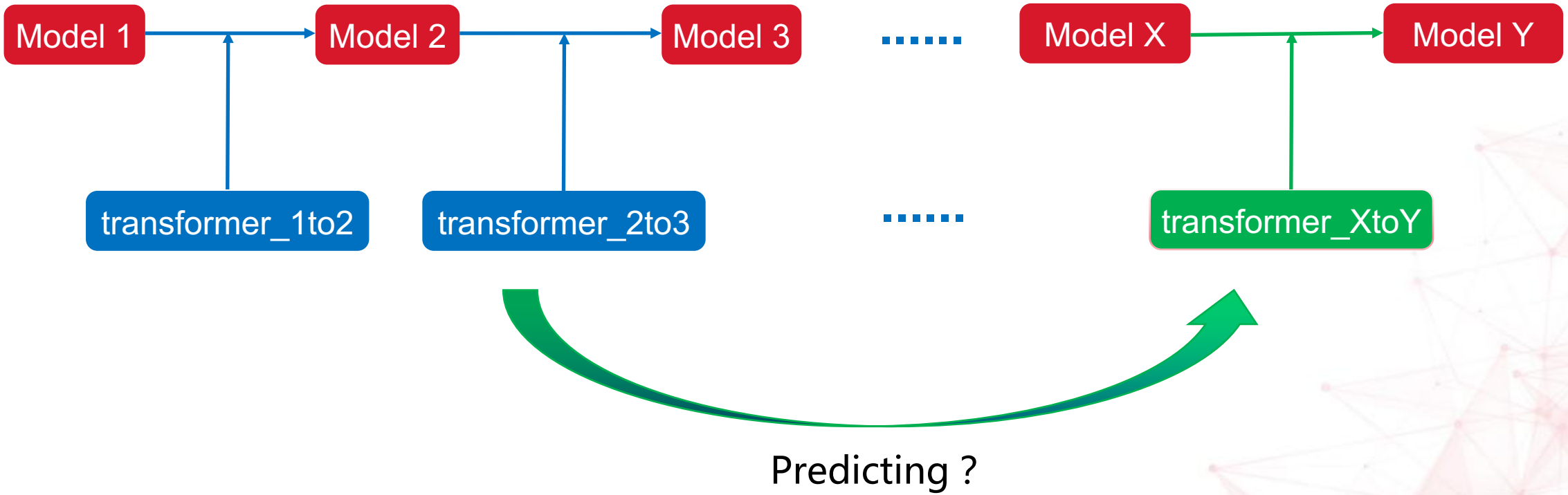
- Upgrade for multiple versions

Exp No.		TestSet 1	TestSet 2	TestSet 3	Sim avg
	model A	*0.9757*	*78.45*	*68.82*	
0	*A =>* B *(B')*	*0.9780*	*82.95*	*73.83*	*0.955715*
	model B	*0.9791*	*84.19*	*76.36*	
1	*B => C*	*0.9864*	*87.96*	*80.30*	*0.959841*
2	*B' => C (direct)*	*0.9843*	*86.21*	*77.43*	*0.943097*
3	*B' => C (train)*	*0.9861*	*86.61*	*78.50*	*0.947483*
4	A => C	0.9848	86.83	78.39	0.947001
	Model C	*0.9877*	*89.59*	*82.72*	

- *0 & 2 & model B : the trained B-C transformer works well directly for B' features*
- *2 & 3 : If we train a B'-C transformer, better C' features can be obtained.*

Version Upgrade for Massive Features

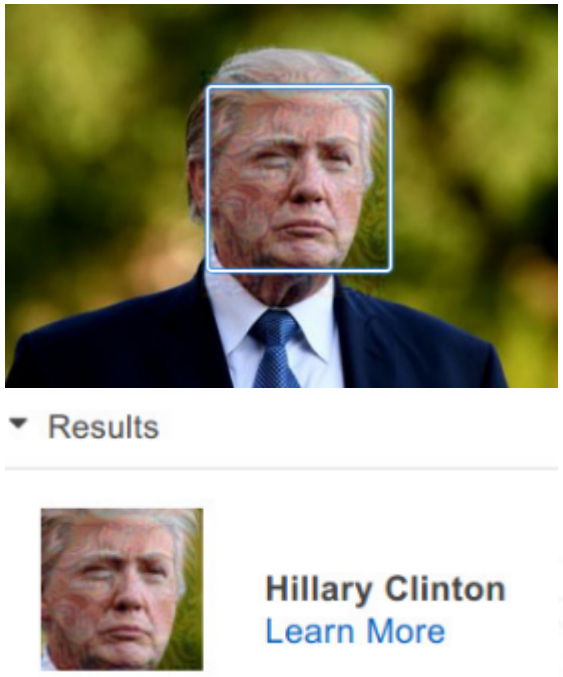
- Follow-ups
  - Further minimize the performance gap between B and B’ features.
  - Next-version Model prediction?



Future work

- Adversarial attack and defense?

CAAD 2018

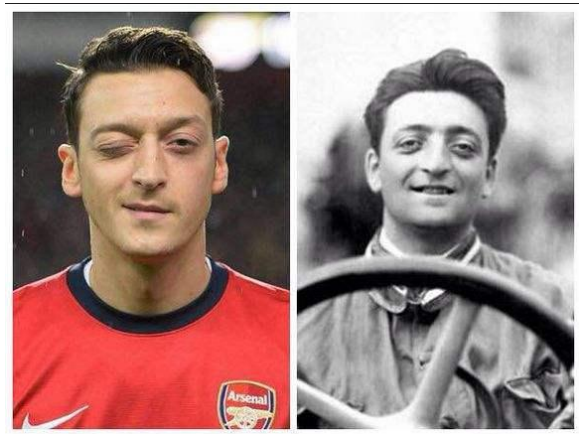


[1] Song, Qing, Yingqi Wu, and Lu Yang. “Attacks on State-of-the-Art Face Recognition using Attentional Adversarial Attack Generative Network.” In *arXiv preprint* 2018
[2] http://ppwwyyxx.com/2018/Geekpwn-CAAD-CTF/.

Future work

- Face recognition for familiar persons?

Arthur Curry



Future work

- Achieve the goal of Skyeye, technically ?

Retrieved from Google image search: https://www.google.com/search?q=person+of+interest&source=lnms&tbm=isch&sa=X&ved=0ahUKEwirwqaWjYTiAhVXUd4KHd1YBrcQ_AUIDygC&cshid=1557049551037590&biw=1398&bih=716

Recommended reading

Sun, Yi, et al. “Deep learning face representation by joint identification-verification.” *Advances in neural information processing systems*. 2014. (**DeepID**)

Schroff, Florian, Dmitry Kalenichenko, and James Philbin. “FaceNet: A unified embedding for face recognition and clustering.” *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015. (**FaceNet**)

Wen, Yandong, et al. “A discriminative feature learning approach for deep face recognition.” *European conference on computer vision*. Springer, Cham, 2016. (**Center Loss**)

Deng, Jiankang, et al. “Arcface: Additive angular margin loss for deep face recognition.” Proceedings of the IEEE CVPR. 2019. (**ArcFace Loss**)

Large-scale Clustering

Outline——Large-scale Clustering

- Introduction
- Baseline approaches
- Recent approaches
- Heterogeneous data grouping

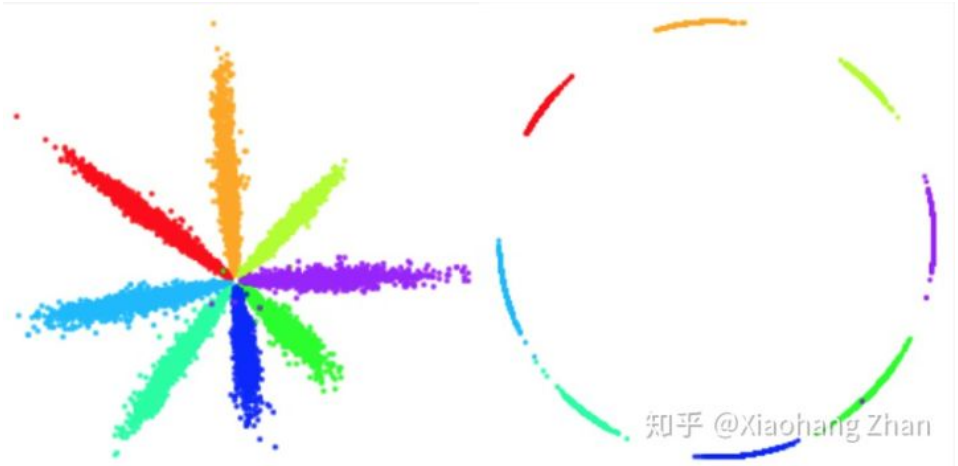
Introduction

Face verification (is this the same person), recognition (who is this person) and clustering (find common people among these faces).

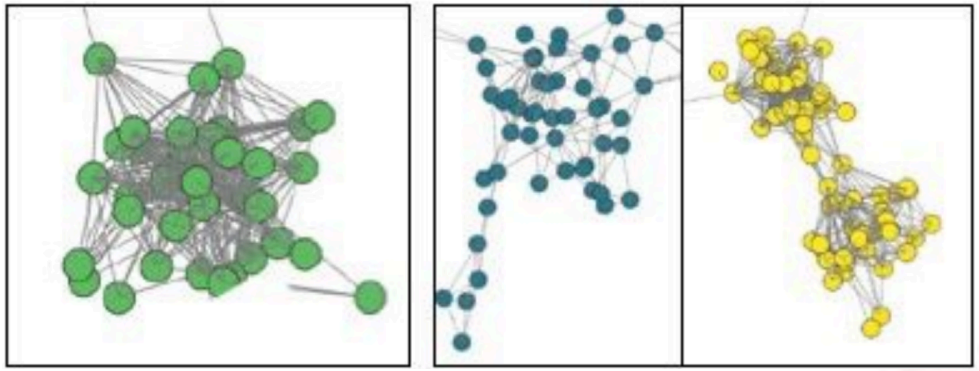
Introduction

Feature representations: LBP, HOG, CNN features

CNN Feature distribution L2 normalized



Expected In practice



Classic approaches:
K-means, Spectral clustering, DBSCAN, Hierarchical Agglomerative Clustering,
Affinity Propagation, Graph Agglomerative Clustering, etc.

[1] A. K. Jain. Data clustering: 50 years beyond k-means. *Pattern recognition letters*, 31(8):651–666, 2010.

[2] Otto, C., Wang, D., & Jain, A. K. (2018). Clustering millions of faces by identity. IEEE transactions on pattern analysis and machine intelligence (PAMI), 40(2), 289-303.

Baseline approach

K-means clustering

Assumption: the clusters are spherical

k-means clustering aims to partition the *n* observations into *k* (≤ *n*) sets **S** = {S_1, S_2, …, S_k} so as to minimize the within-cluster variance.

$$\arg\min_{\mathbf{S}} \sum_{i=1}^{k} \sum_{\mathbf{x} \in S_i} \left\| \mathbf{x} - \boldsymbol{\mu}_i \right\|^2 = \arg\min_{\mathbf{S}} \sum_{i=1}^{k} |S_i| \operatorname{Var} S_i$$

Another perspective of view:
A simplified version of Expectation-Maximization algorithm for optimize the model

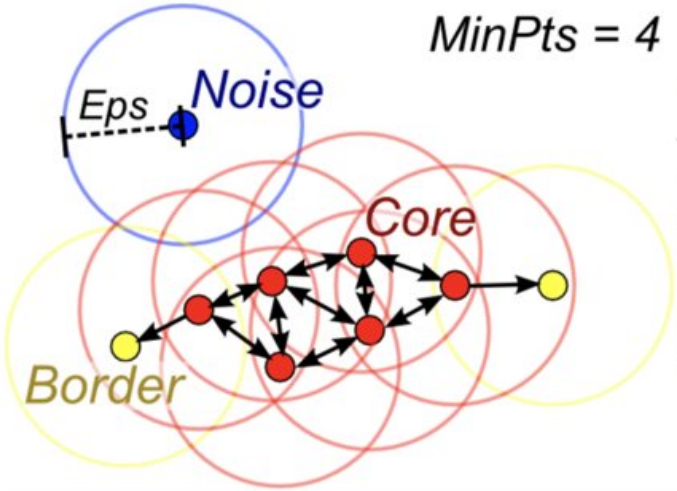
$$P(f_i = x | C = c) \sim \mathcal{N}(\mu_{c,i}, 1)$$

E-step: assign data points to the closest cluster center, and calculate the

M-step: compute the centroid of each cluster

Baseline approach

DBSCAN: Density-based spatial clustering of applications with noise



Red: Core Points

Yellow: Border points. Still part of the cluster because it's within epsilon of a core point, but not does not meet the min_points criteria

Blue: Noise point. Not assigned to a cluster

1. No limit on the number of clusters
2. No assumption on prior distribution
3. Noise awareness

***Core**:* A point *p* is a *core point* if at least minPts points are within distance *ε* of it (including *p*).

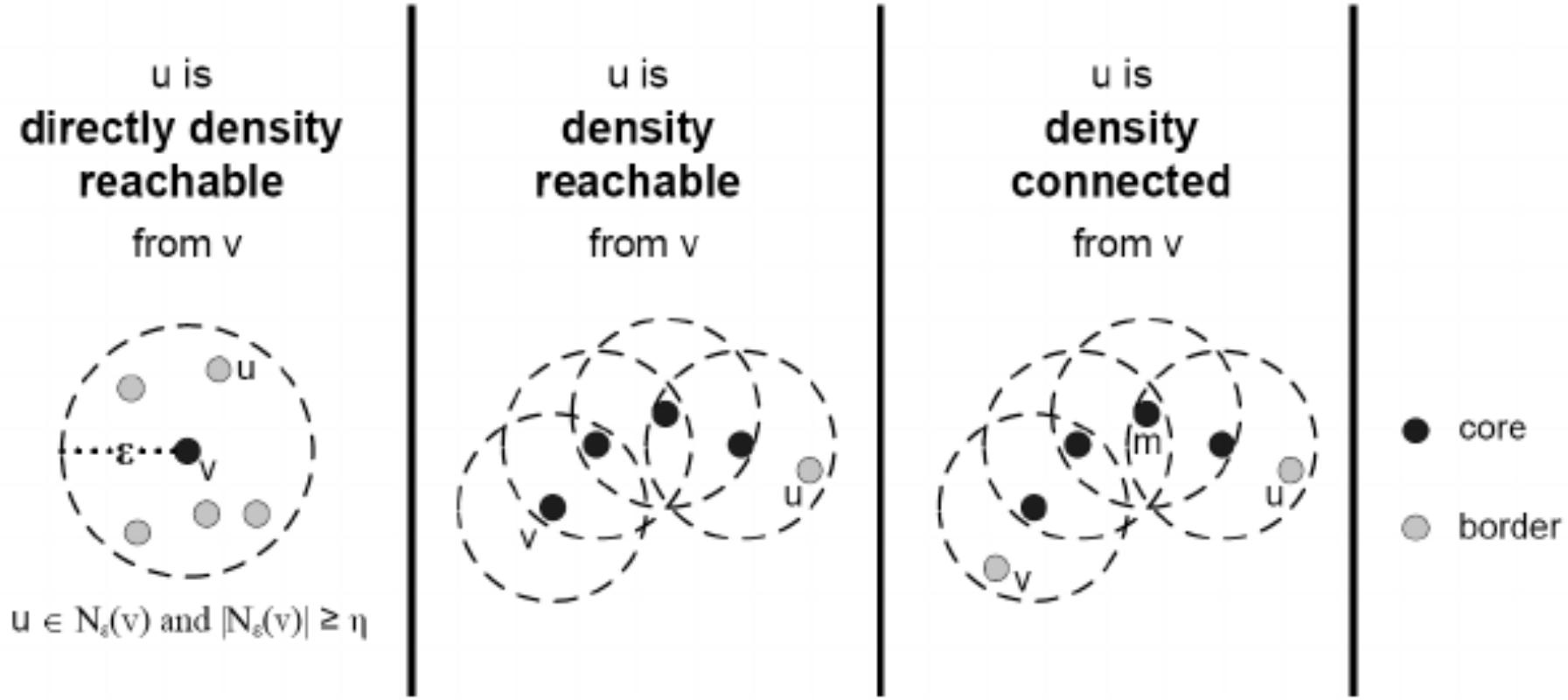
Density-reachable: A point *q* is *directly reachable* from *p* if point *q* is within distance *ε* from core point *p*.

Outliers: All points not reachable from any other point are *outliers* or *noise points*

Baseline approach

```
DBSCAN(DB, distFunc, eps, minPts) {
    C = 0                                                   /* Cluster counter */
    for each point P in database DB {
        if label(P) ≠ undefined then continue               /* Previously processed in inner loop */
        Neighbors N = RangeQuery(DB, distFunc, P, eps)      /* Find neighbors */
        if |N| < minPts then {                              /* Density check */
            label(P) = Noise                                /* Label as Noise */
            continue
        }
        C = C + 1                                           /* next cluster label */
        label(P) = C                                        /* Label initial point */
        Seed set S = N \ {P}                                /* Neighbors to expand */
        for each point Q in S {                             /* Process every seed point */
            if label(Q) = Noise then label(Q) = C           /* Change Noise to border point */
            if label(Q) ≠ undefined then continue           /* Previously processed */
            label(Q) = C                                    /* Label neighbor */
            Neighbors N = RangeQuery(DB, distFunc, Q, eps)  /* Find neighbors */
            if |N| ≥ minPts then {                          /* Density check */
                S = S ∪ N                                   /* Add new neighbors to seed set */
            }
        }
    }
}
```

Baseline approach



A cluster then satisfies two properties:

1.All points within the cluster are mutually density-connected.

2.If a point is density-reachable from any point of the cluster, it is part of the cluster as well.

Faster DBSCAN

Weakness in DBSCAN:

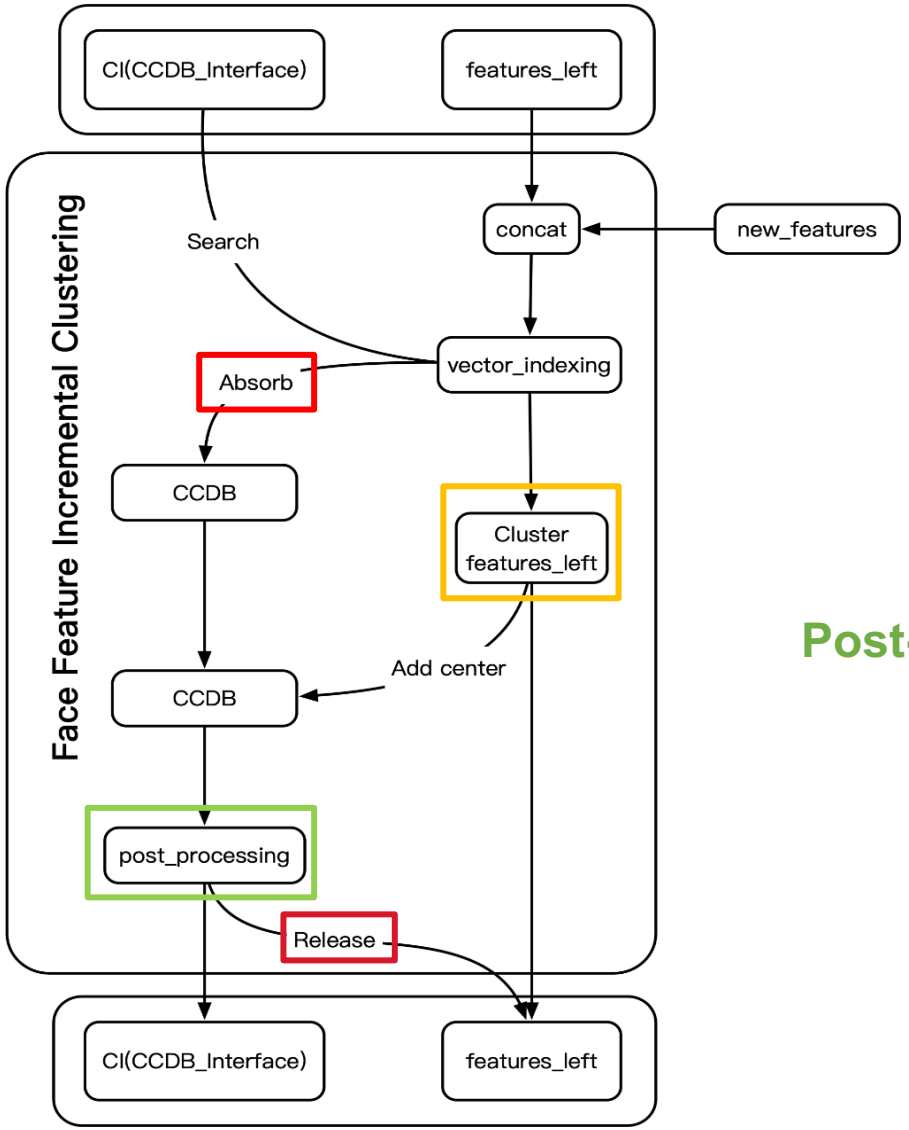
- Finding in large scale data the ϵ - nearest neighbors is costly
- Use a global threshold for all clusters is problematic

Our modification:

- Change ϵ -NN to k-NN, and we use FAISS[1] implementation.
- Choose an adaptive threshold for clusters of different variance.
- Implement the incremental DBSCAN clustering.

[1] https://github.com/facebookresearch/faiss.wiki.git, BSD-licensed

Solution for incremental clustering



Absorb： 将现有的图片以及之前的类(档案)进行撞库，通过比对特征和聚类中心，把特征加入已有的类别中。

Clustering： 要是对Absorb操作剩下的feature进行聚类，生成新的cluster；并加入到类中心库中

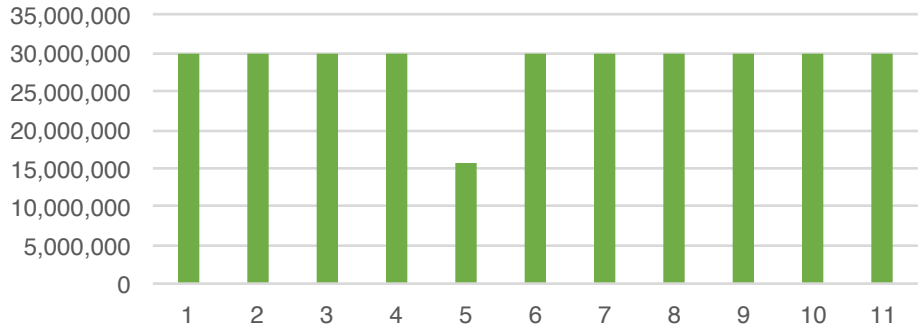
Post-processing: 考虑到同一个人的feature可能被聚到不同的cl类中，所以考虑对已有的类进行merge。

Release: 对于明显的不正常，即size特别大的类别进行释放，留作下一个阶段进行重新聚类

Experimental results for demonstration

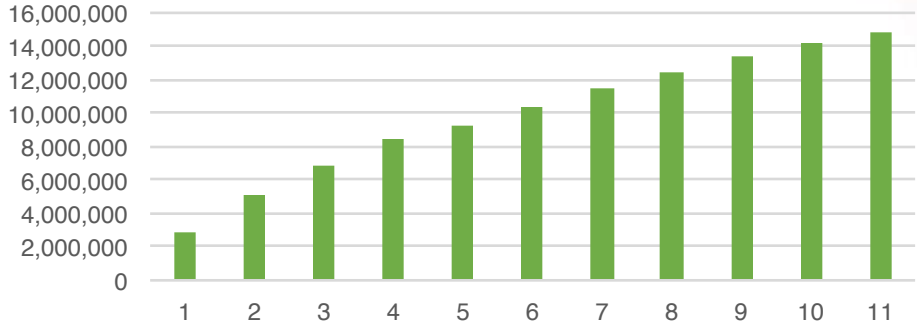
Analysis on real-world incremental clustering results of totally 0.3 billion images

#incremental images



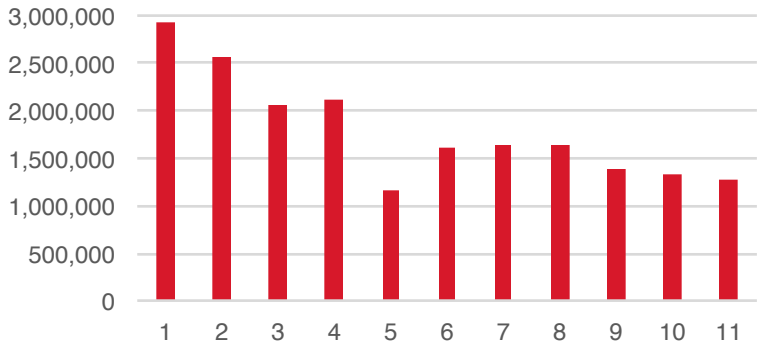
Incremental clustering

#total clusters

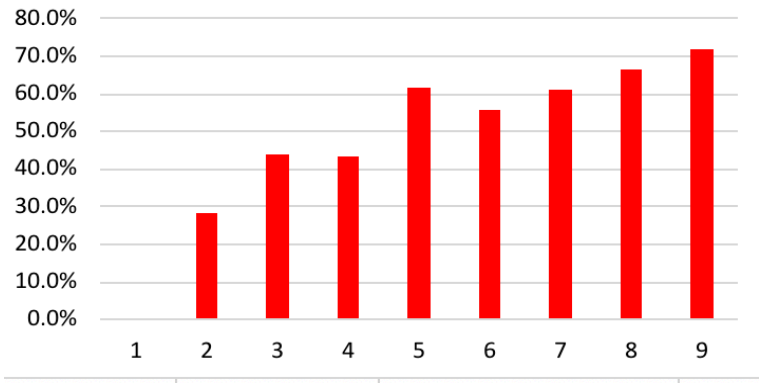


Statistical Results

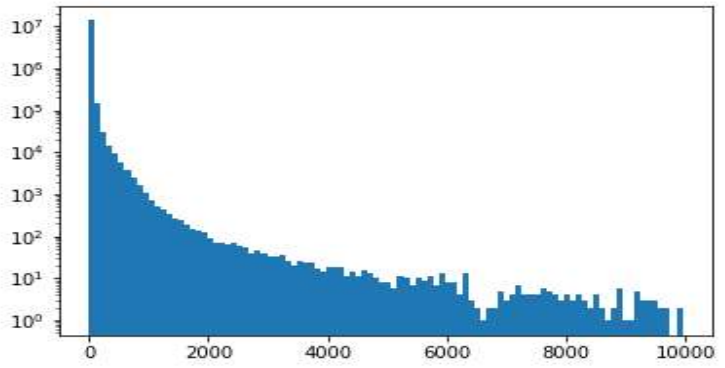
#classes incrementally added



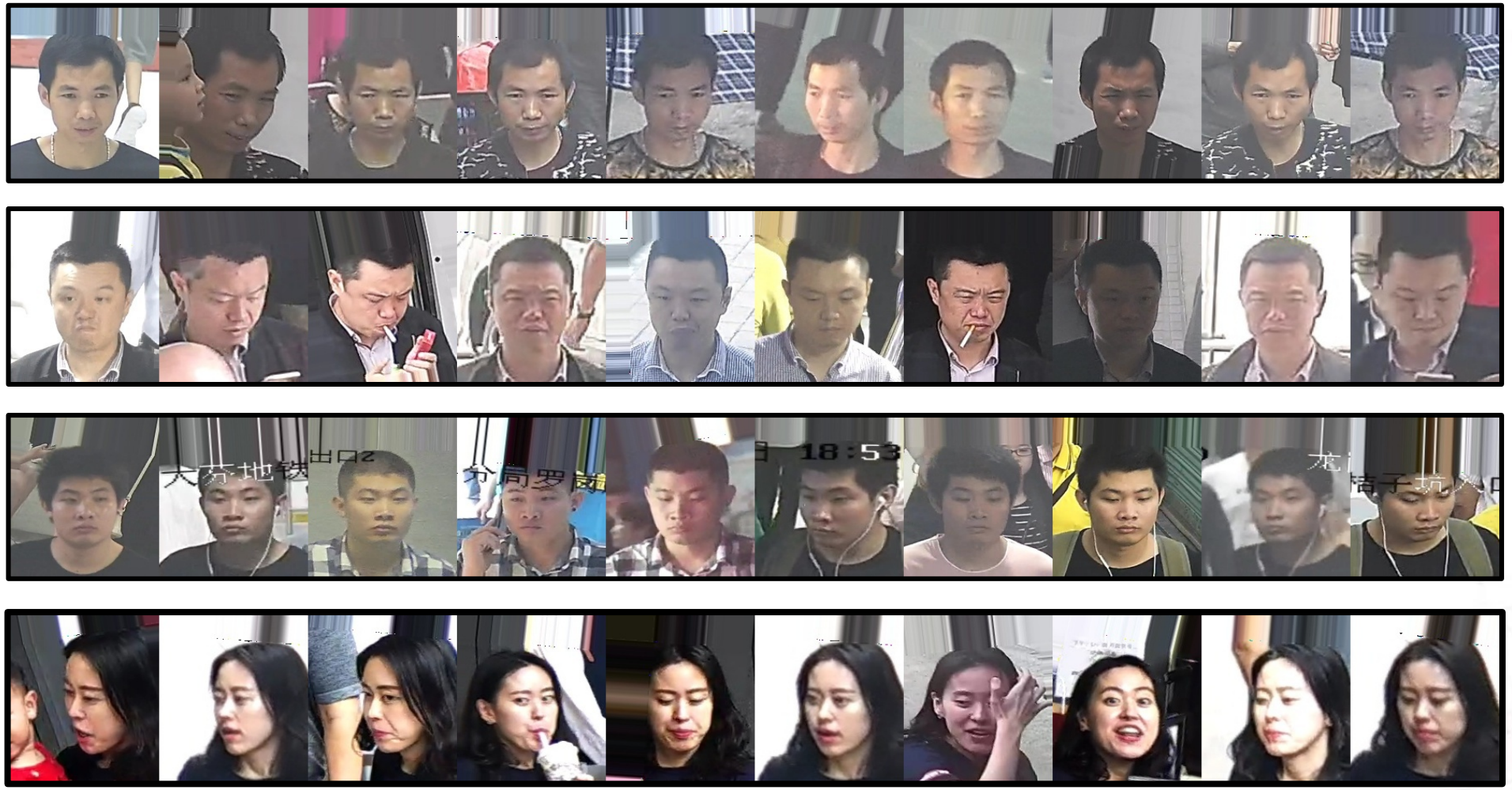
Percentage of absorbed images incrementally



#images for all clusters



Qualitative results



Failure case for example

Remaining challenges

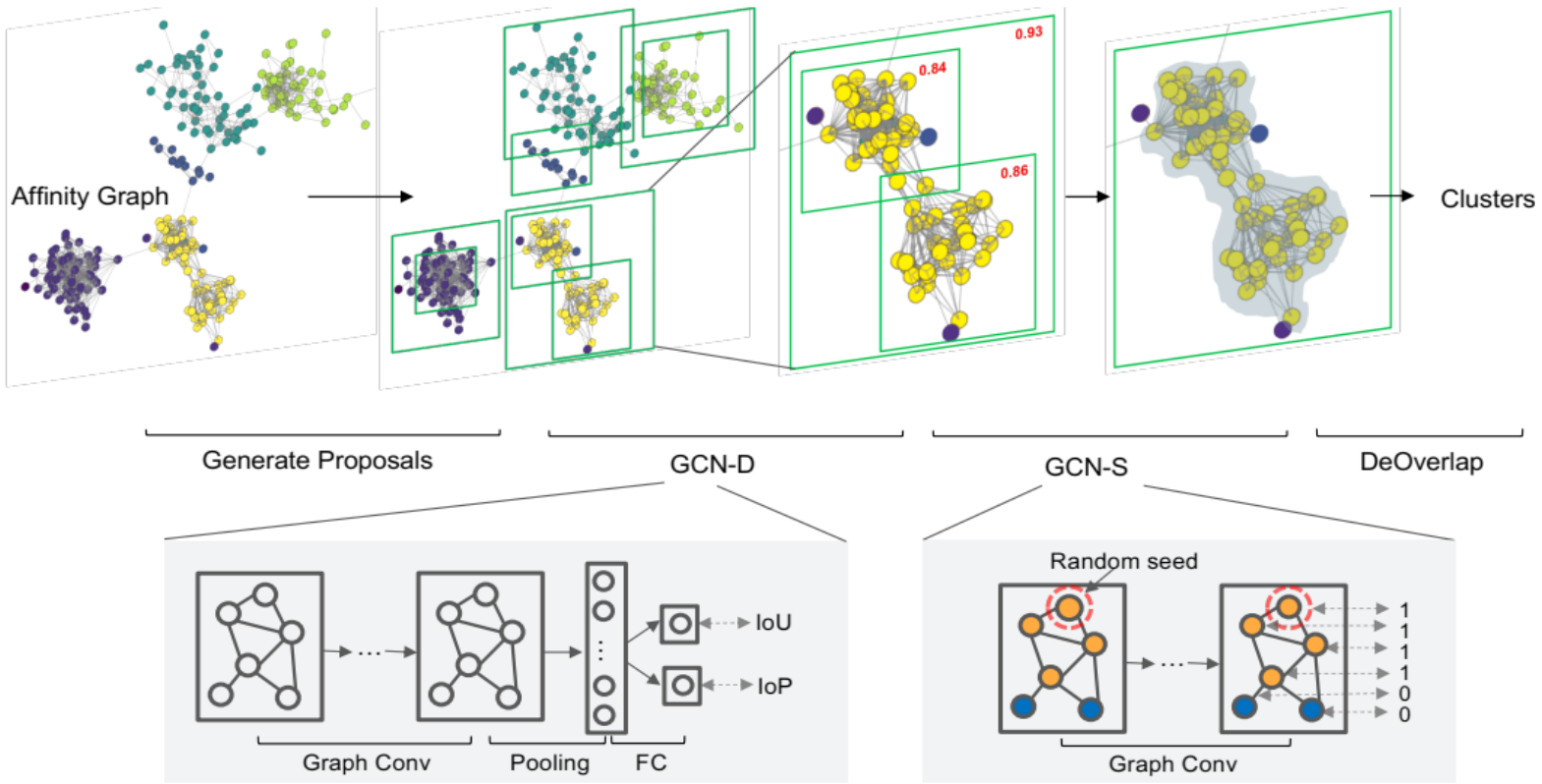
- **DBSCAN:** connect two different clusters through weak linkage in hyperspace

- **Features**: it is hard to differentiate the old, the child, and occluded faces in feature space.

- **Test benchmark**: large-scale testing data with clean groundtruth label is difficult to obtain.

Currently only small-scale benchmark of less than 1 million images is available.

Proposed method——GCN-based Clustering



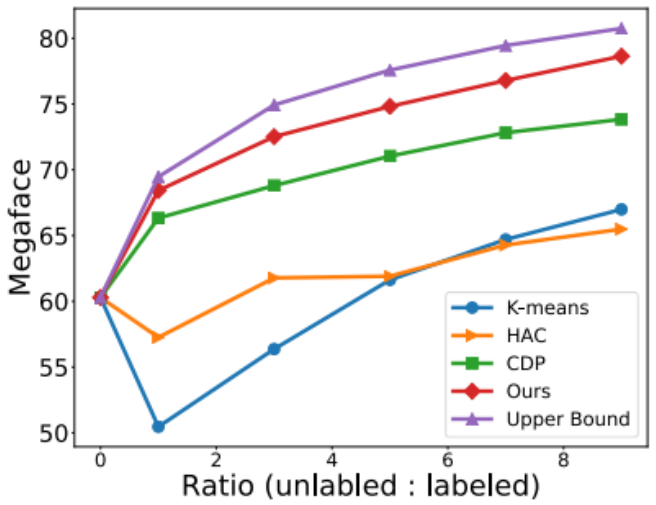
1. GCN-D: use GCN to generate proposal cluster

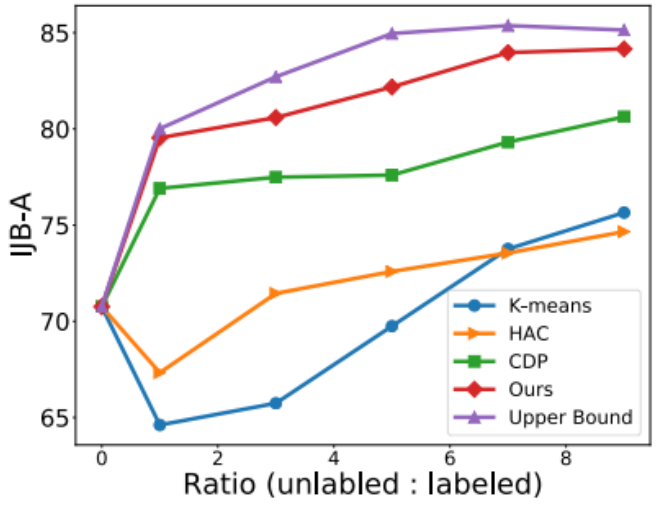
2. GCN-S: Binary classify each node in the proposal graph to refine clustering result.

Learning to Cluster Faces on an Affinity Graph. L. Yang, X. Zhan, **D. Chen**, J. Yan, C. Loy, D. Lin CVPR19(Oral)

Proposed method——GCN-based Clustering

Methods	#clusters	Precision	Recall	F-score	Time
K-Means	5000	52.52	70.45	60.18	13h
DBSCAN	7206	0.14	71.83	0.271	89h
HAC	117392	66.84	70.01	68.39	18h
Approximate Rank Order	307265	81.1	7.3	13.34	**250s**
CDP	29658	80.19	70.47	75.01	350s
GCN-D	19879	95.72	76.42	84.99	2000s
GCN-D + GCN-S	19879	98.24	75.93	**85.66**	2200s



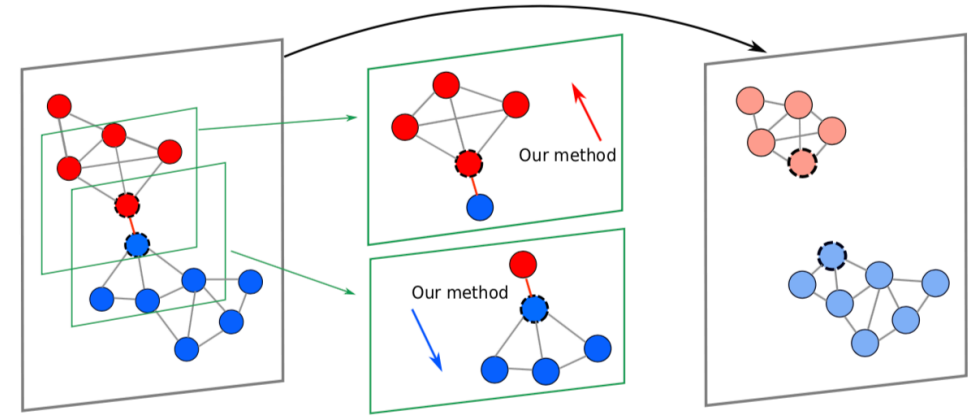


1. GCN can improve F-score

2. Face recognition model benefit from better clustering results.

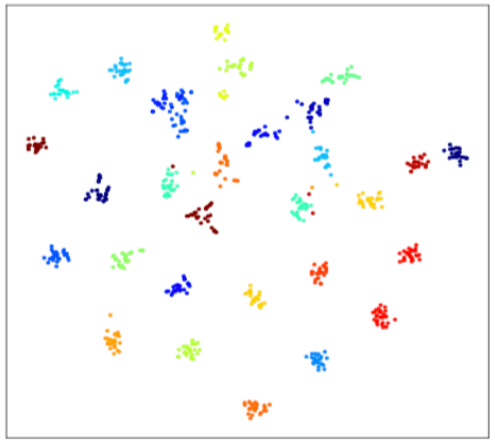
Proposed method——Feature aggregation

Features of the same class forms a cluster, and neighboring context is helpful to distinguish different clusters.

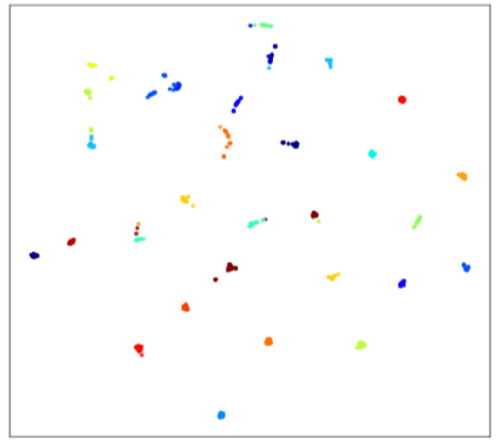


(a) The original feature embeddings (b) The proposed method over the neighbourhood graph (c) The obtained graph embeddings

Feature aggregation makes features in a cluster distribute more compact.



(a) embeddings of the baseline



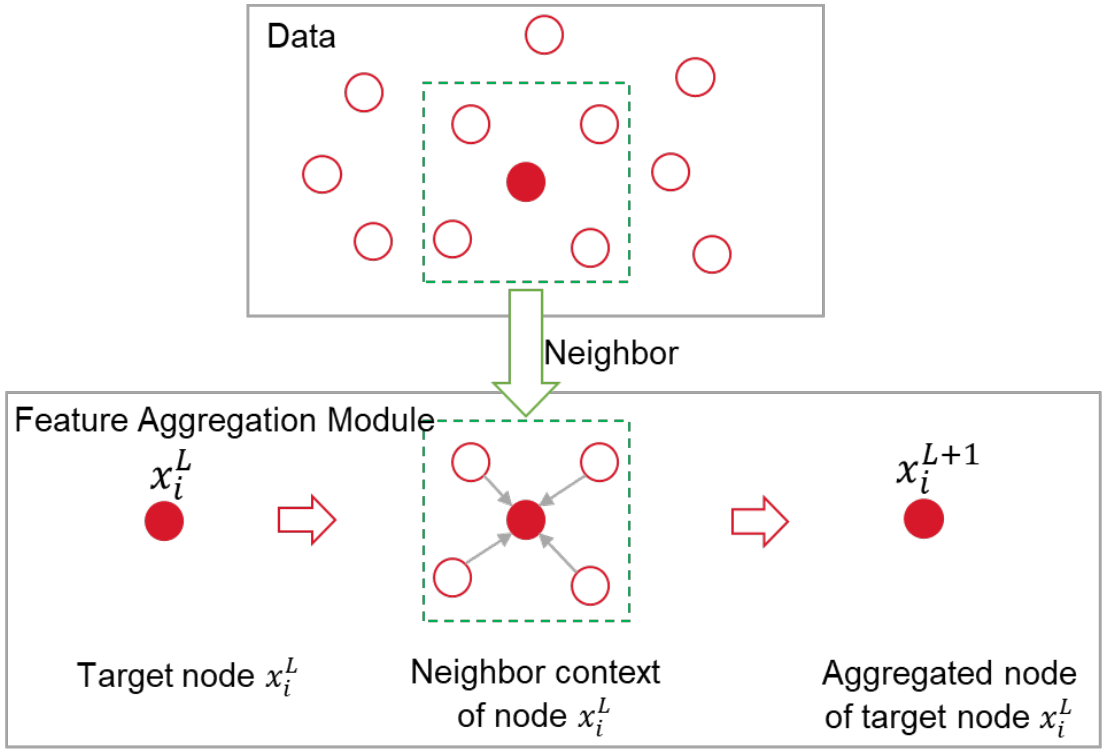
(b) embeddings of our MNE

$$H^{l+1} = f\left(H^{(l)}, A\right) = \sigma\left(AH^{(l)}W^{(l)}\right)$$

New feature Affinity matrix Original feature

Proposed method——Feature aggregation

Update every feature by its neighboring context



9

Formulation

$$\mathbf{x}_i^{L+1} = \alpha * \mathbf{x}_i^L + (1-\alpha) * \sum_{j \in N(i)} \left(w_{i,j} * \mathbf{x}_j^L \right)$$

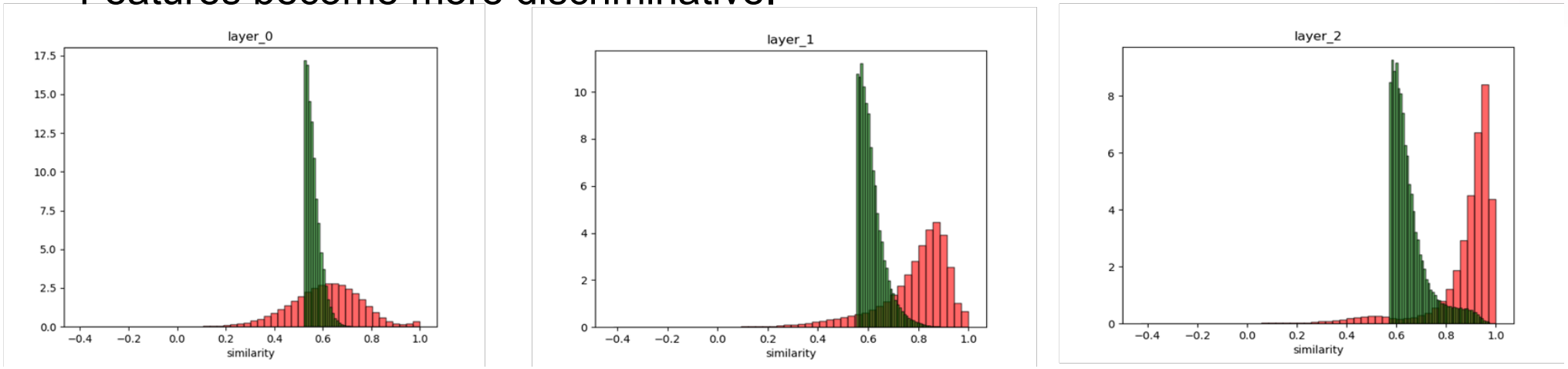
where the weights are defined as below

$$w'_{i,j} = \frac{\cos(\mathbf{x}_i, \mathbf{x}_j)}{\sum_{k \in N(i)} \cos(\mathbf{x}_i, \mathbf{x}_k)}$$

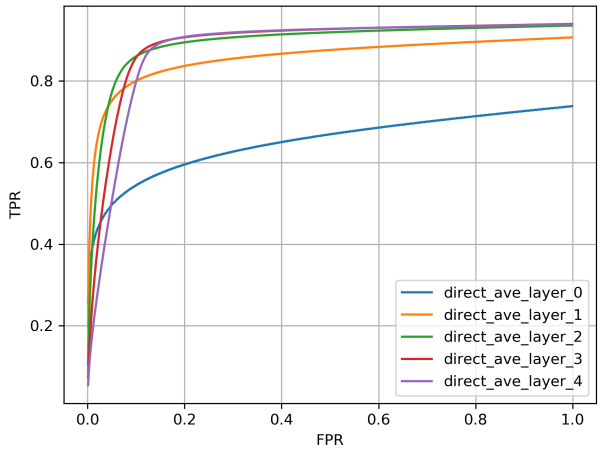
$$w_{i,j} = \begin{cases} w'_{i,j}, & \text{if } w'_{i,j} > \theta \\ 0, & \text{otherwise} \end{cases}$$

Proposed method——Feature aggregation

True-match pairs versus false-match pairs along with feature aggregation.
Features become more discriminative.

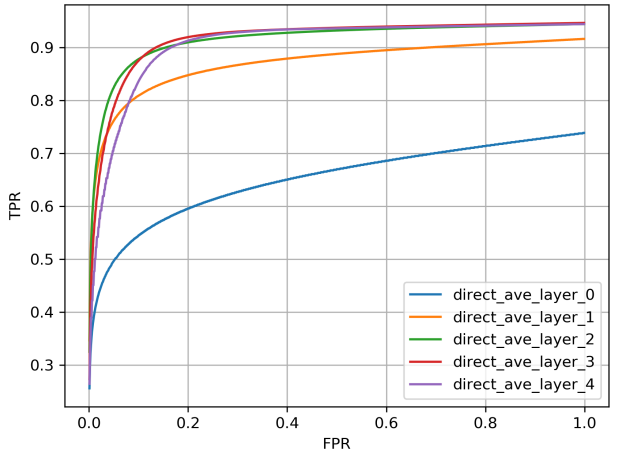


Precision-Recall Curve



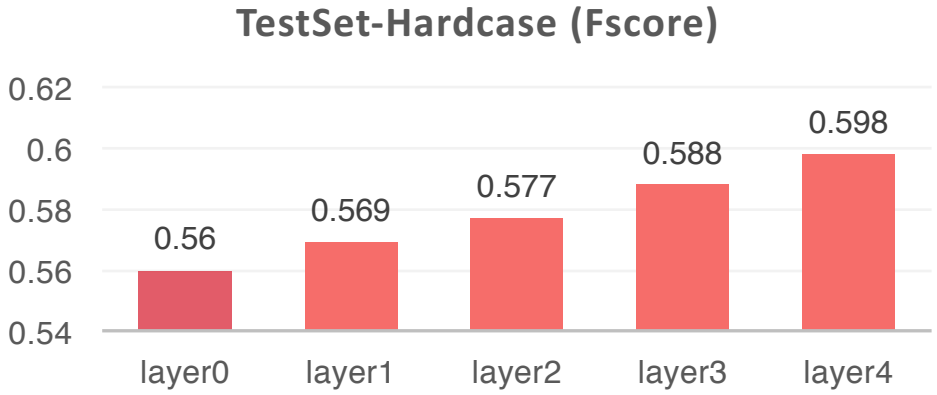
FA with context neighborhood K=8

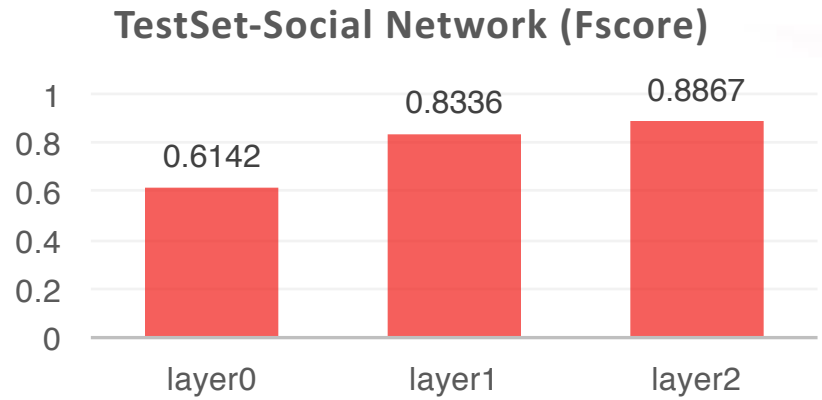
Precision-Recall Curve

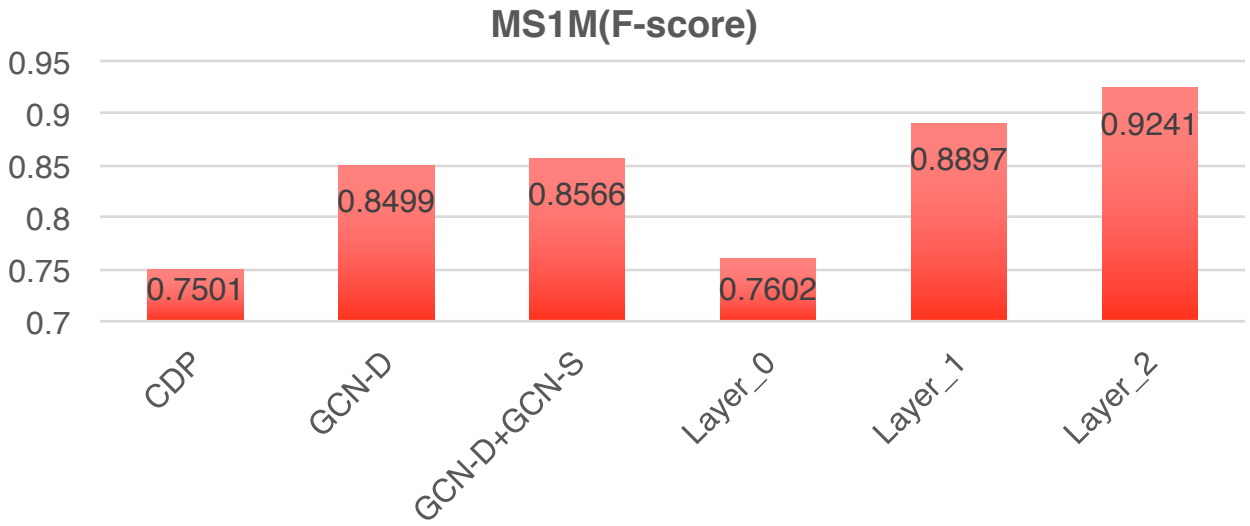


FA with context neighborhood K=32

Proposed method——Feature aggregation





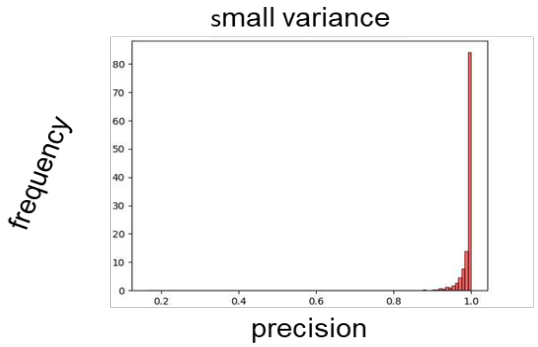


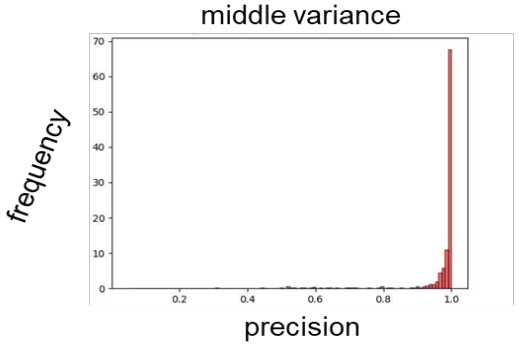
[1] X. Zhan, Z. Liu, J. Yan, D. Lin, and C. C. Loy. Consensus-driven propagation in massive unlabeled data for face recognition. In *European Conference on Computer Vision (ECCV)*, September 2018.

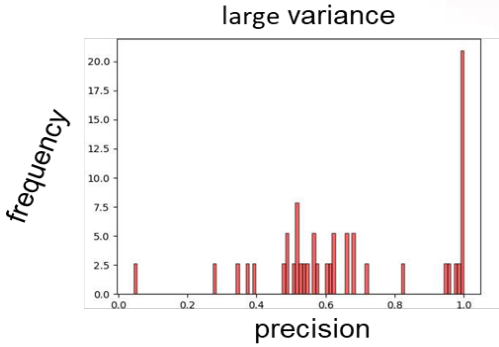
[2] L. Yang, X. Zhan, D. Chen, J. Yan, C. C. Loy, D. Lin. Learning to Cluster Faces on an Affinity Graph.

Proposed method——Handling huge clusters

- **Relationship between precision and variance for each cluster**

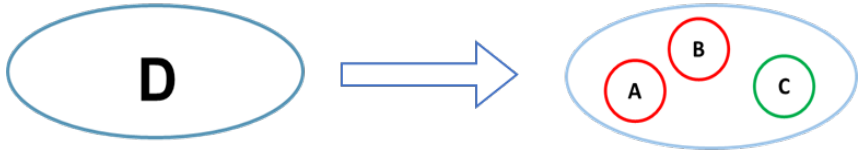






- **Divide large-variance clusters.**

Sample-wise similarity Step1: divide large cluster



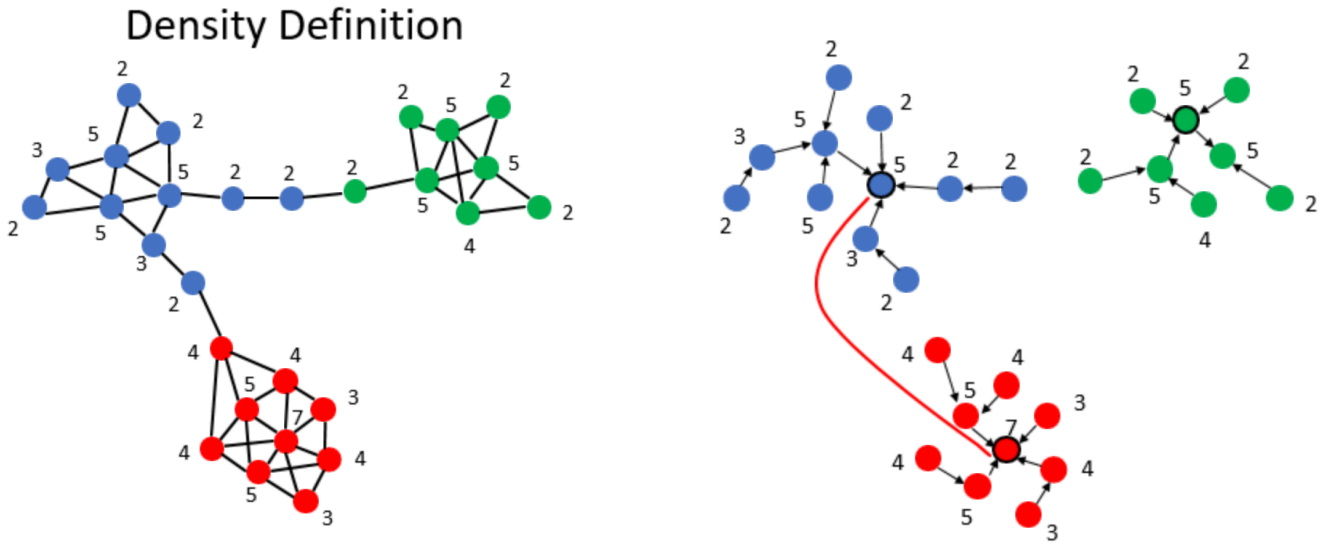
Set-wise similarity Step2: re-organize small clusters

- **Preliminary results:**

  - For the 78 large-variance clusters in MS1M dataset, average f-score can be improved from 73.2 to 80.5
  - For the 500 large-variance clusters in Hardcase dataset, average f-score can be improved from 72.8 to 80.2

Proposed method——Handling huge clusters

- From DBSCAN to DensityPeak



- Basic Idea
  - Cluster centers are surrounded by neighbors with lower local density
  - Cluster centers are far away from other points with a higher local density
- Advantages
  - Based only on distance between data points
  - Can produce nonspherical clusters

	original feature								original feature						
	th		precision	recall	fscore	pairwise	expansion		th		precision	recall	fscore	pairwise	expansion
dbscan	0.8		0.9995	0.2797	0.437	0.4379	33.53	density peak	0.72		0.9906	0.5977	0.7455	0.7625	12.89
	0.78		0.999	0.3647	0.5344	0.5429	28.28		0.7		0.9789	0.6798	0.8024	0.8139	10.08
	0.76		0.9966	0.4522	0.6221	0.6355	23.43		0.68		0.9623	0.7305	0.8305	0.8339	8.1
	0.74		0.9883	0.5397	0.6982	0.7088	19.13		0.66		0.9334	0.7698	0.8437	0.8284	6.82
	0.72		0.9569	0.6216	0.7537	0.7008	15.38		0.65		0.9131	0.7884	0.8462	0.8155	6.275
	0.7		0.8883	0.694	0.7792	0.4124	12.29		0.64		0.8763	0.8056	0.8394	0.7795	5.7864
	0.68		0.7167	0.7557	0.7357	0.001	9.78		0.62		0.7863	0.8344	0.8096	0.6786	4.9865
	0.66		0.4668	0.8104	0.5924	0.001	7.7		0.6		0.6584	0.8564	0.7444	0.5066	4.3759

A. Rodriguez and A. Laio. Clustering by fast search and find of density peaks. In Science 2014.

Experimental Results

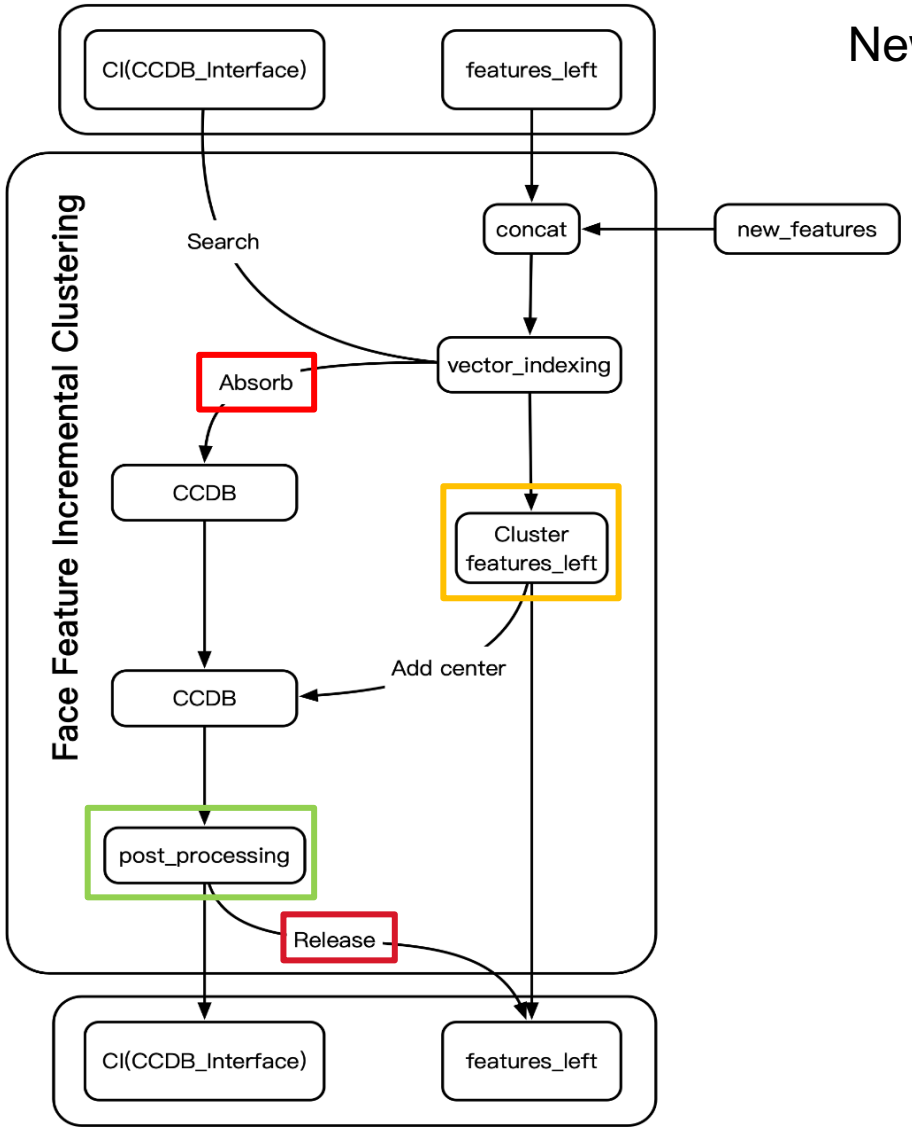
Feature Aggregation + DBSCAN

Experimental results							
Settings					Pairwise Evaluation		
Tag	aggr topk	layer	AUC	clustering threshold	precision	recall	F-score
DBSCAN					0.9204	0.5765	0.709
CDP					0.8019	0.7047	0.7501
GCN-D					0.9572	0.7642	0.8499
GCN-D + GCN-S					0.9824	0.7593	**0.8566**
Agg	32	layer_2	0.9087	0.92	0.9577	0.8927	**0.9241**
Agg	64	layer_2	0.9139	0.94	0.9152	0.901	0.908

1, GCN-based approach achieved better performance.

2, Combining feature aggregation and DBSCAN achieved 7% higher f-score than GCN-D+GCN-S

Proposed method——Incremental solution



New framework: no absorb, clustering directly with incremental data

origin feature + full							
threshold	prec	recall	F1	expand			
0.73	0.976	0.5945	0.7389	12.8655			
origin feature+increment + old framework							
th_high	th_merge	th_sub	th_release	prec	recall	F1	expand
0.77	0.725	0.65,0.6	1000	0.7384	0.7055	0.7216	6.5732
0.75	0.75	0.65,0.6	1000	0.748	0.6985	0.7224	6.6217
0.73	0.75	0.65,0.6	1000	0.7463	0.6967	0.7207	6.613
origin feature+increment + new framework							
th_high	th_merge	th_sub	th_release	prec	recall	F1	expand
0.77	0.75	0.65	1000	0.8742	0.7594	0.8127	6.728
0.77	0.7	0.65	1000	0.8241	0.7672	0.7946	6.59
0.73	0.75	0.65	1000	0.8879	0.774	0.8271	6.4238
0.7	0.75	0.65	1000	0.8321	0.7782	0.8042	6.0261
0.73	0.77	0.65	1000	0.8323	0.7784	0.8044	6.0209
aggregation feature+increment + new framework							
th_high	th_merge	th_sub	th_release	prec	recall	F1	expand
0.88	0.7	0.8	1000	0.8173	0.8845	0.8496	3.9378
0.88	0.78	0.8	1000	0.8391	0.871	0.8548	4.1948
0.9	0.8	0.8	1000	0.8682	0.8714	0.8698	4.3445
0.92	0.8	0.8	1000	0.8871	0.8727	0.8799	4.3501
0.94	0.8	0.8	1000	0.8981	0.868	0.8828	4.4275

Outline——Large-scale Clustering

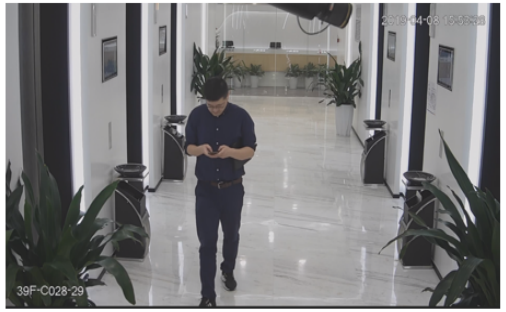
- Introduction
- Baseline approaches
- Recent approaches
- Heterogeneous data grouping

Heterogeneous data grouping

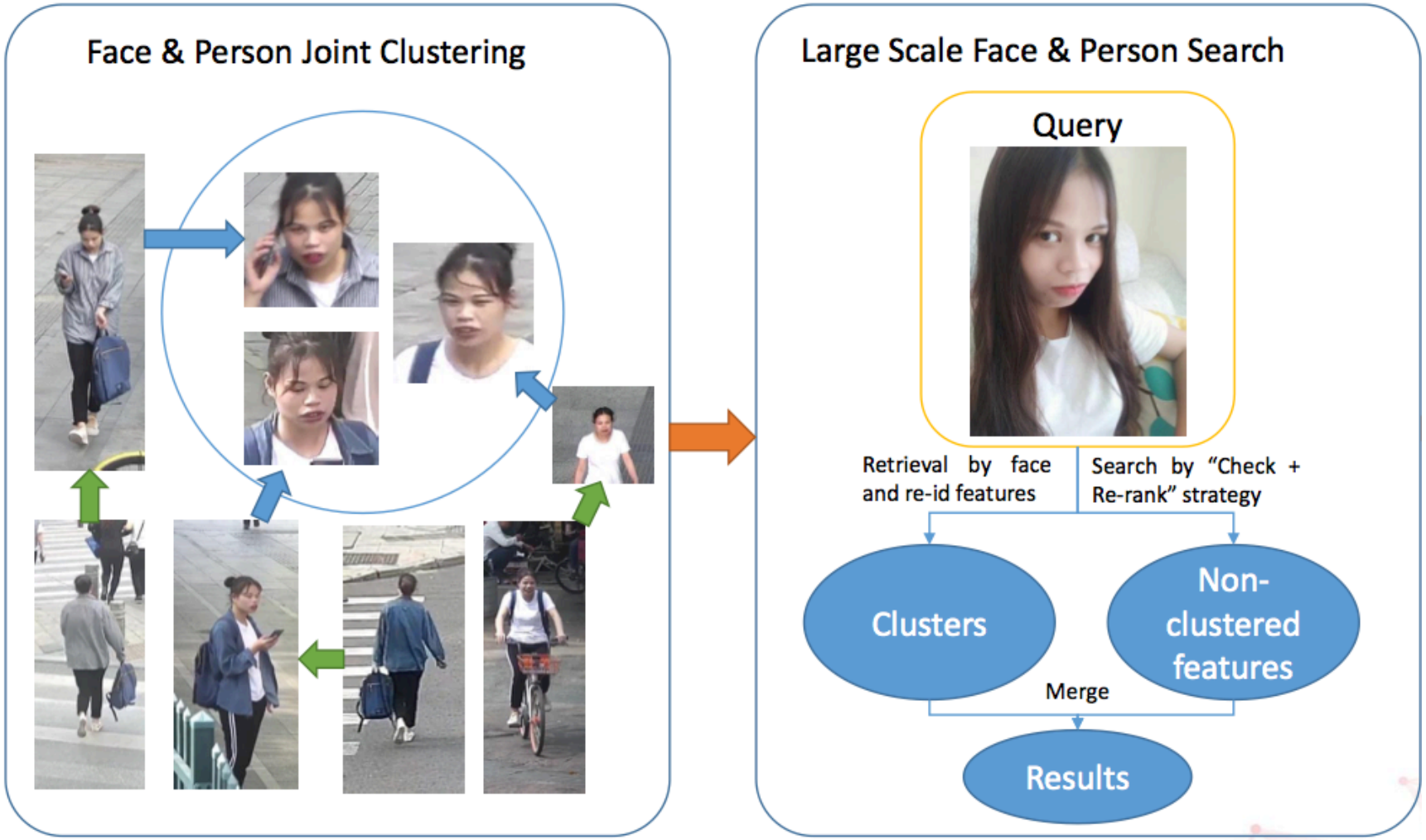
Face & body joint search

Wider Challenge (Person search)

In video surveillance, we need to track suspects even without face and cloth changed.



Heterogeneous data grouping



Recommended reading

A. K. Jain. Data clustering: 50 years beyond k-means. *Pattern recognition letters*, 31(8):651–666, 2010.

Otto, C., Wang, D., & Jain, A. K. (2018). Clustering millions of faces by identity. *IEEE transactions on pattern analysis and machine intelligence (PAMI)*, *40*(2), 289-303.

X. Zhan, Z. Liu, J. Yan, D. Lin, and C. C. Loy. Consensus-driven propagation in massive unlabeled data for face recognition. In *European Conference on Computer Vision (ECCV)*, September 2018.

L. Yang, X. Zhan, D. Chen, J. Yan, C. C. Loy, D. Lin. Learning to Cluster Faces on an Affinity Graph. In arXiv:1904.02749, 2019

DBSCAN in Wikipedia: https://en.wikipedia.org/wiki/DBSCAN

Continual Learning

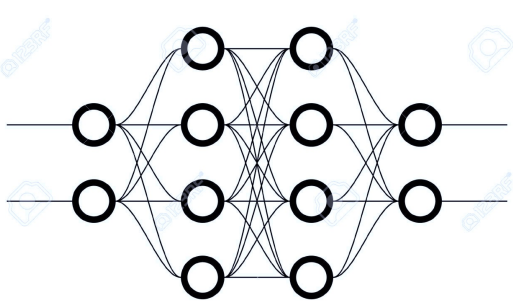
Outline——Continual Learning

- Introduction
- Literatures
- Settings and Designs
- How does it work in real system

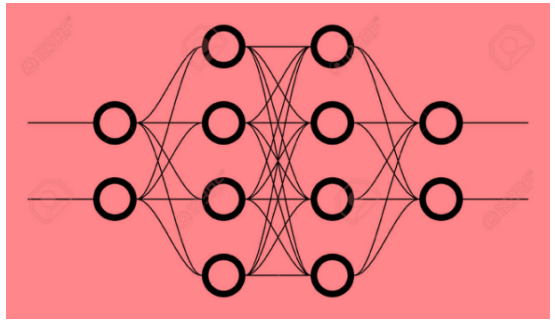
Introduction

Problem of catastrophic forgetting:

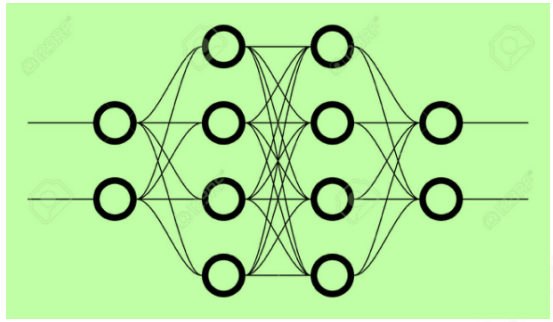
Training a model (pre-trained on Dataset A) with data from Dataset B will deteriorate its performance on A.



1. Train on ImageNet

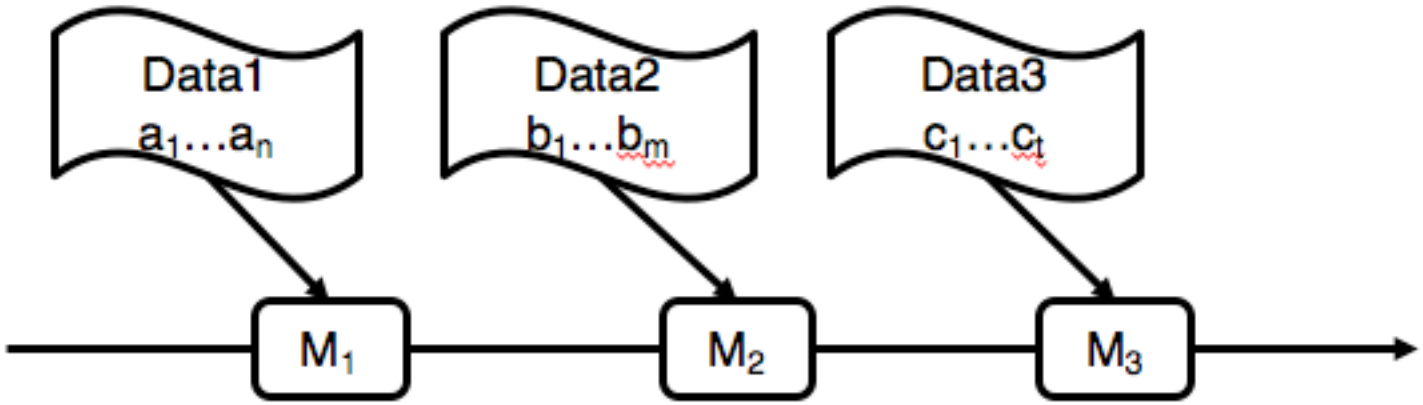


2. Train on COCO Dataset



3. Forget ImageNet

Introduction



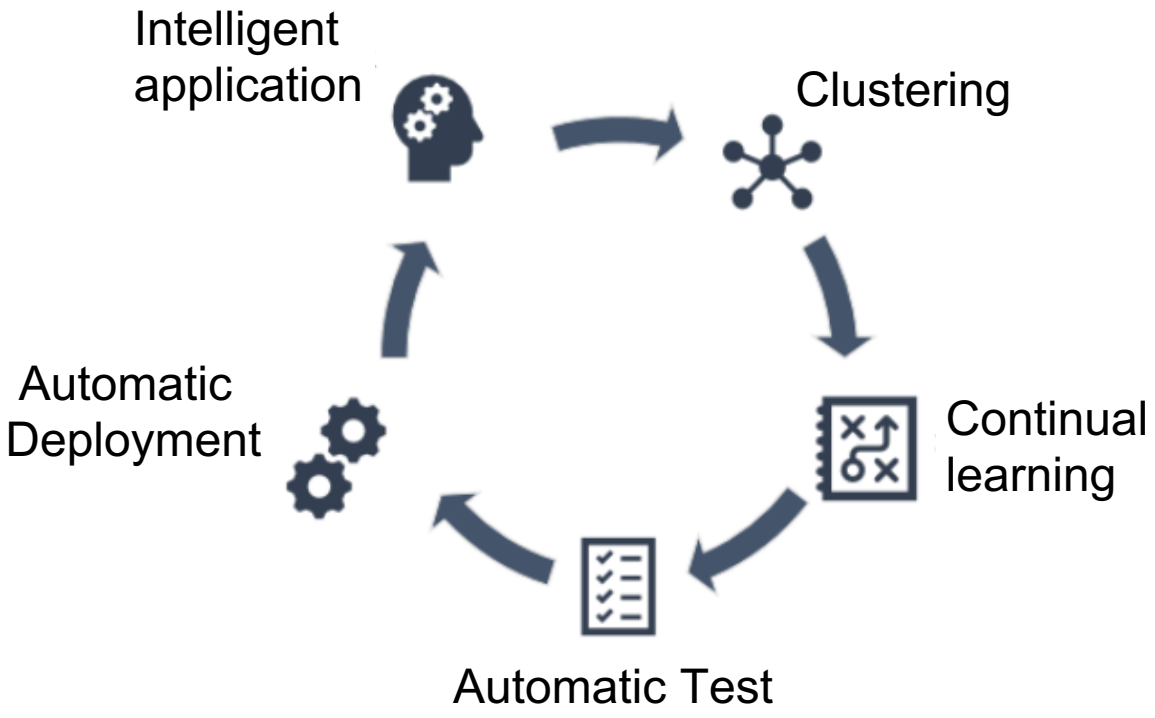
Continual learning (CL) is the ability to learn continually from a stream of experiential data, building on what was learnt previously, while being able to reapply, adapt and generalize it to new situations.

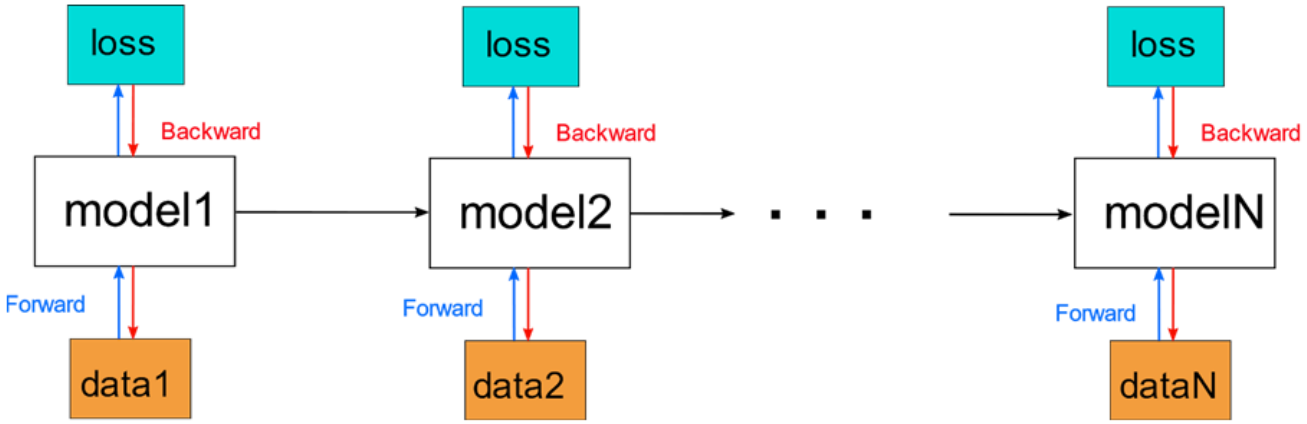
--- NIPS Workshops 2018

Introduction

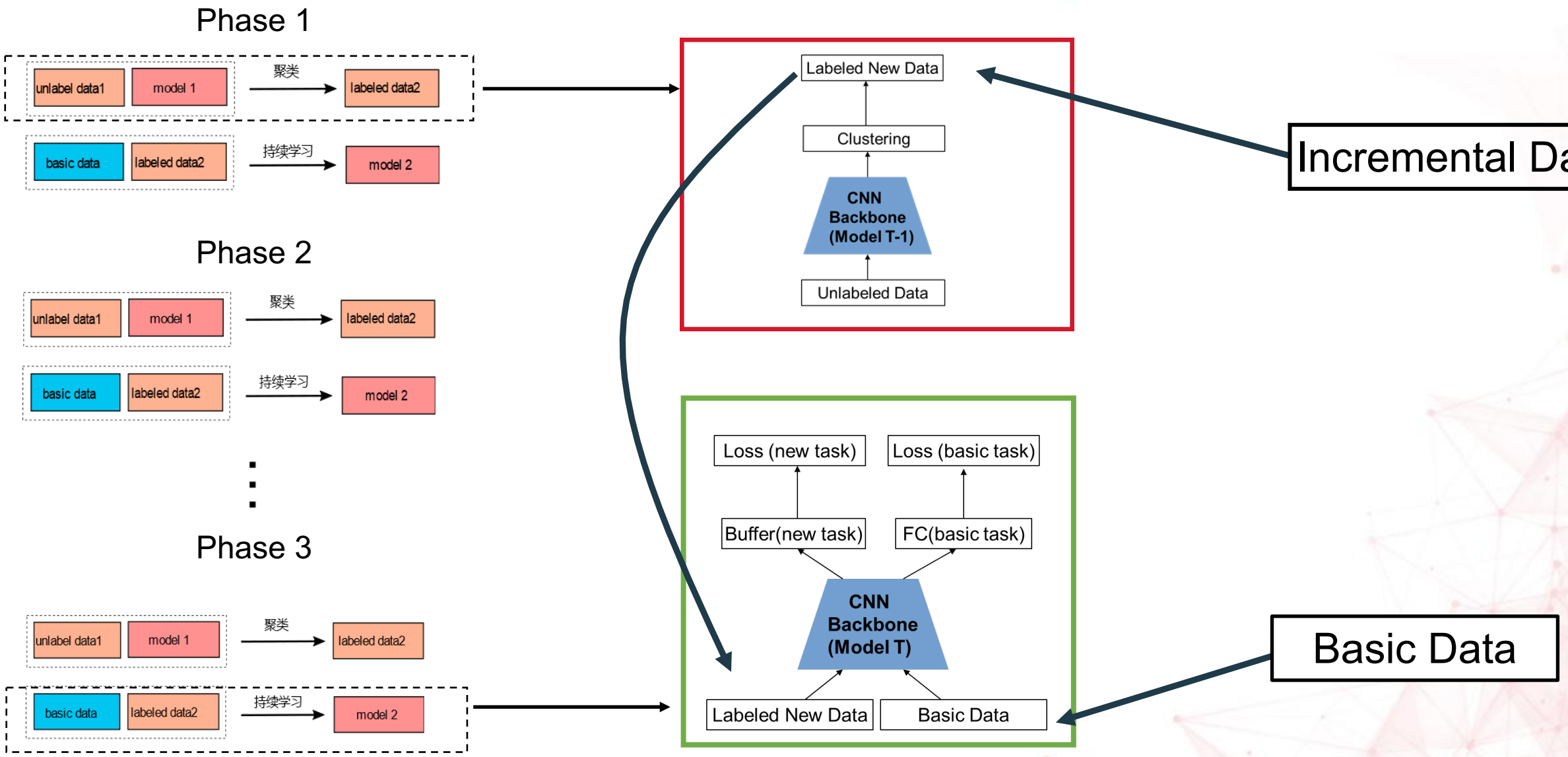
Continual learning system aims to better utilization of data in client-side system, which is

- of limited storage budget that cannot keep all sequential data
- privacy-sensitive. The data cannot be accessed outside.
- self-contained.





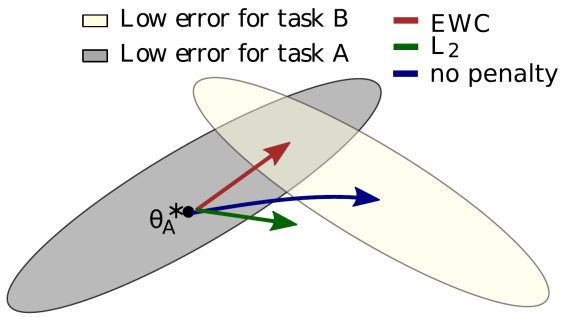
Introduction



Outline——Continual Learning

- Introduction
- Literatures
- Settings and Designs
- How does it work in real system

Literatures

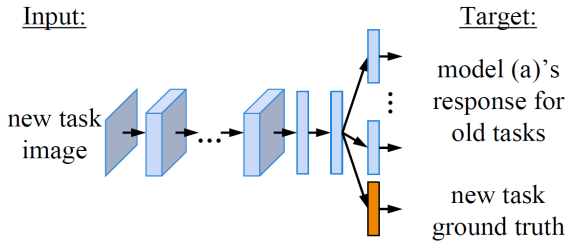


Regularize model parameters

[2017 PNAS Kirkpatrick] Overcoming catastrophic forgetting in neural networks

[2017 ICML Zenke] Continual Learning Through Synaptic Intelligence

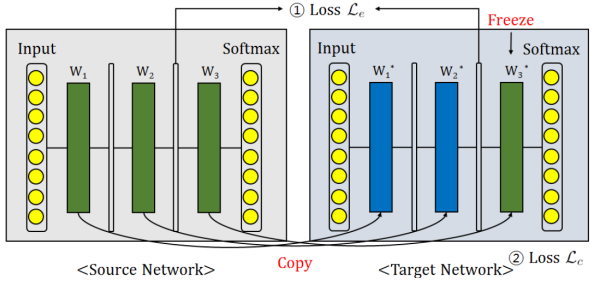
[2018 ECCV Aljundi] Memory Aware Synapses: Learning what (not) to forget



Regularize prediction responses

[2018 PAMI Li] Learning without Forgetting

[2018 ECCV Castro] End-to-End Incremental Learning



Regularize feature representations

[2016 arXiv Jung] Less-forgetting Learning in Deep Neural Networks

Outline——Continual Learning

- Introduction
- Literatures
- Settings and Designs
- How does it work in real system

Settings and Designs

Generally, continual learning or life-long learning has the following settings:

- Continual learning with incremental domain
- Continual learning with incremental class
- Continual learning with incremental task

Currently, continual learning with **incremental class** is the focus in academic research. .

For open-set problem like face recognition and person re-identification, we focus on feature learning, and the continual learning is based on **data of both incremental domains and incremental identities / classes**.

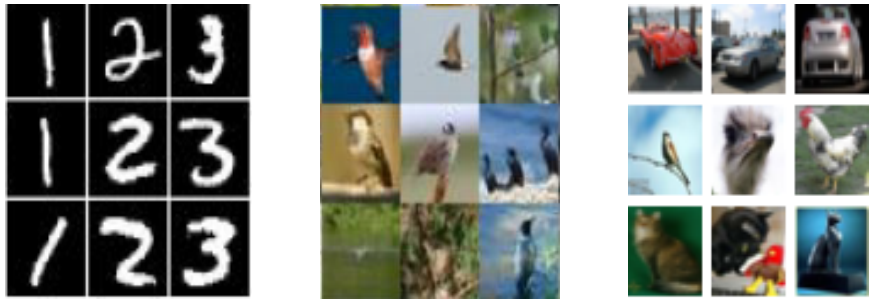
Another difference is that we allow usage of limited amount of historical data in continual learning. Academic setting has strict limit on access to old data.

Settings and Designs

Academic setting

SenseTime

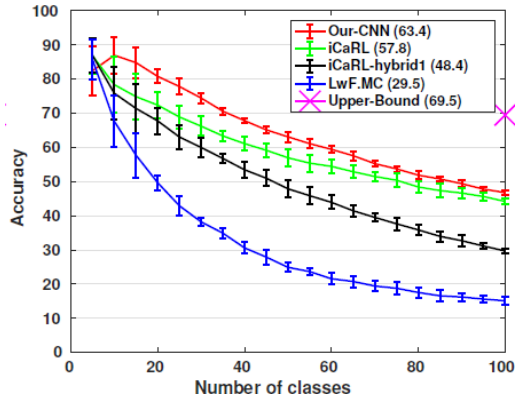
- Dataset



MNIST, CIFAR, CUB, ImageNet

Face, Person Re-Identification

- Evaluation



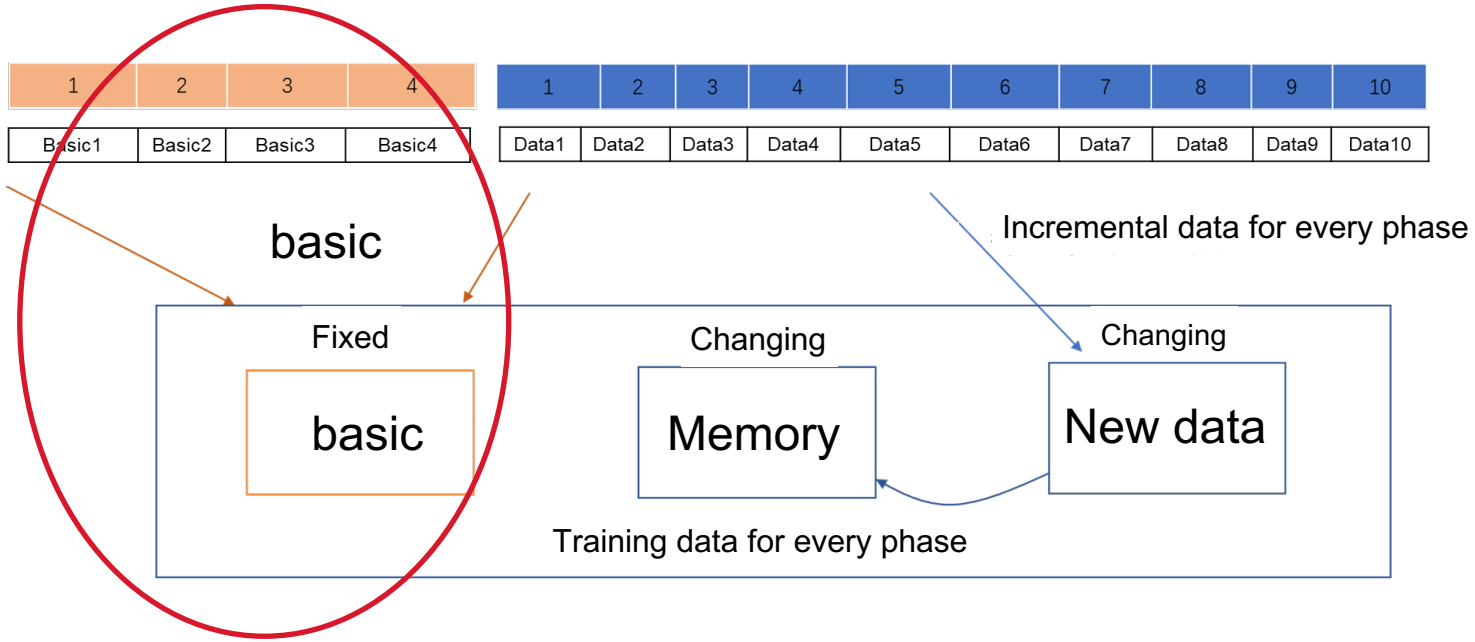
Accuracy of observed classes

Phase	0	1	2	3	4	5	6	7
Common	88.07	91.7	90.56	93.61	95.24	95.24	96.05	94.69
1:N avg	65.30	68.78	69.39	69.46	69.33	73.28	74.34	71.38
1:n 200W	47.72	54.31	48.85	59.09	56.84	62.92	66.24	57.79

Accuracy of continual learning for face recognition

Settings and Designs

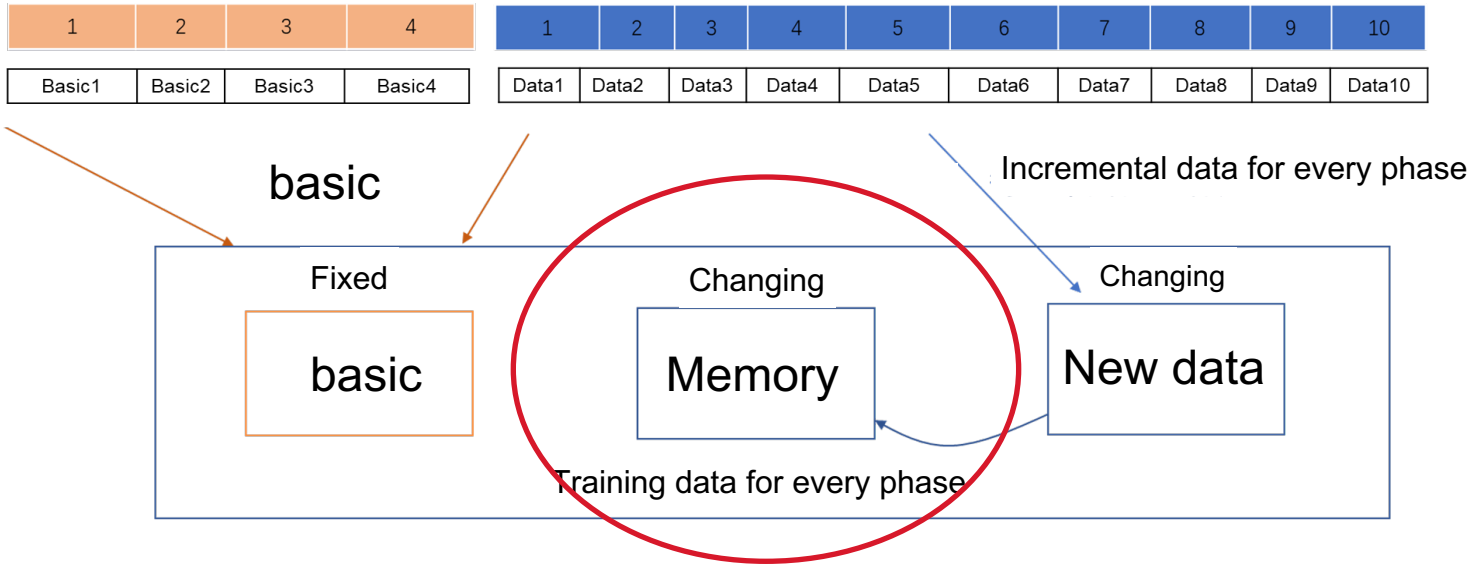
For a well trained model, we add training data for baseline model (basic) to maintain the baseline performance in continual learning.



Question: Increasing training data occupies more computation resource, which is limited.
How to reduce the dependency of old training data?

Settings and Designs

We use memory to consolidate old data and incremental data.

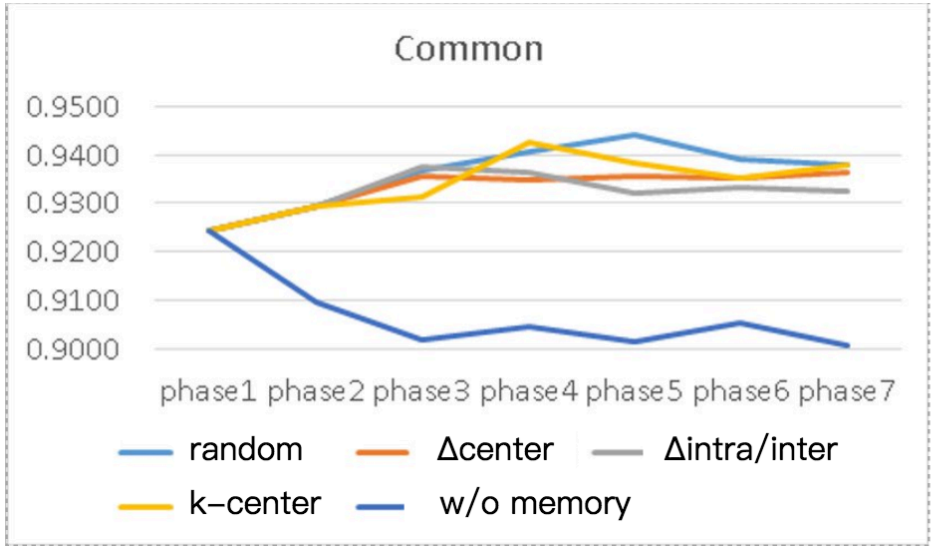


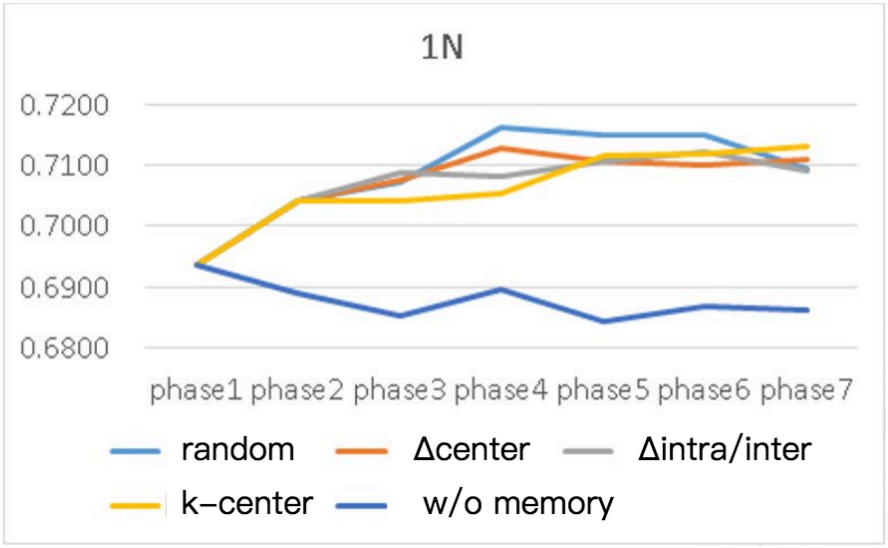
How to select to preserve a limited amount of data in memory?

Settings and Designs

Spare 25% of memory for new data every training phase, and shrink the original memory to 75% by the following strategies:

- Randomly select
- Change of cluster center after training
- Change of intra-class similarity / inter-class similarity after training
- Greedy k-center selection [1]





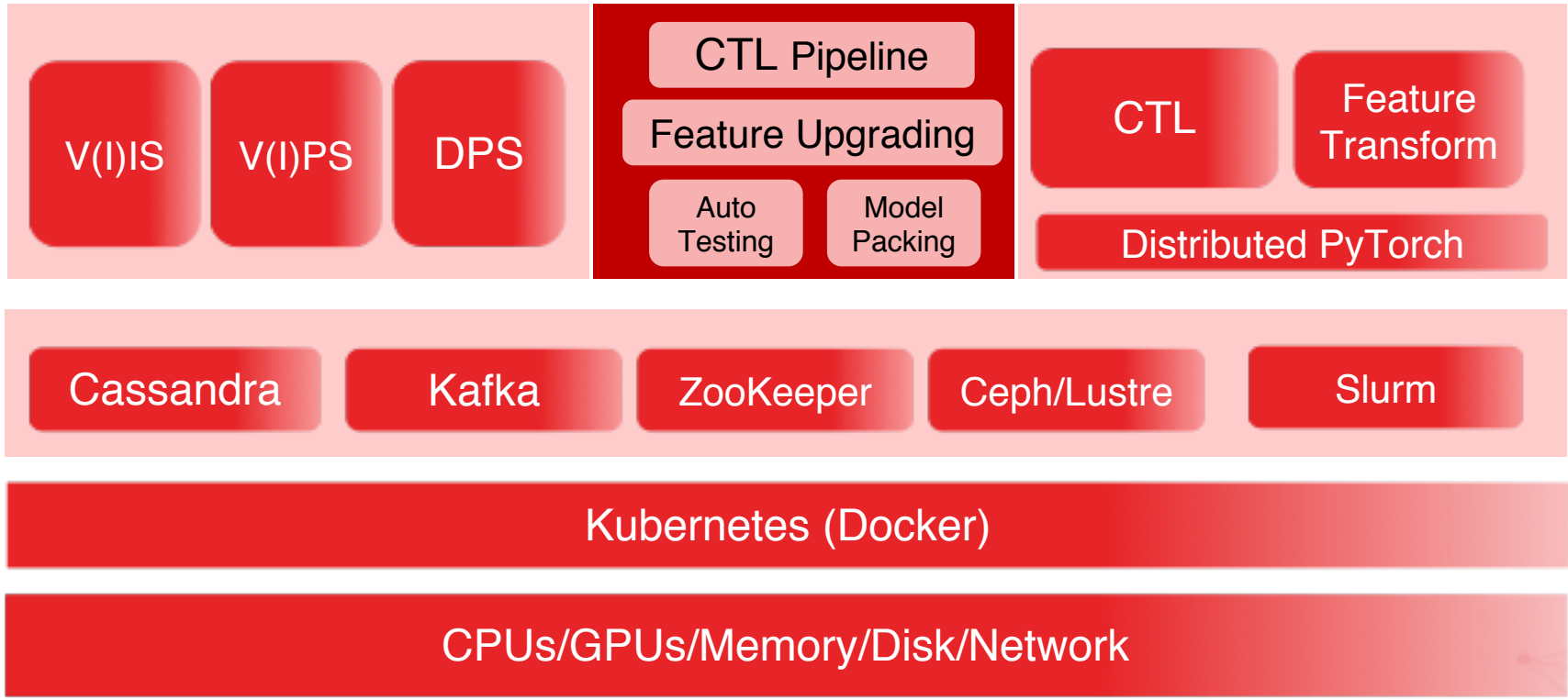
[1] O. Sener, S. Savarese. ACTIVE LEARNING FOR CONVOLUTIONAL NEURAL NETWORKS: A CORE-SET APPROACH. In arXiv: 1708.00489v4, 2018.

Outline——Continual Learning

- Introduction
- Literatures
- Settings and Designs
- How does it work in real system

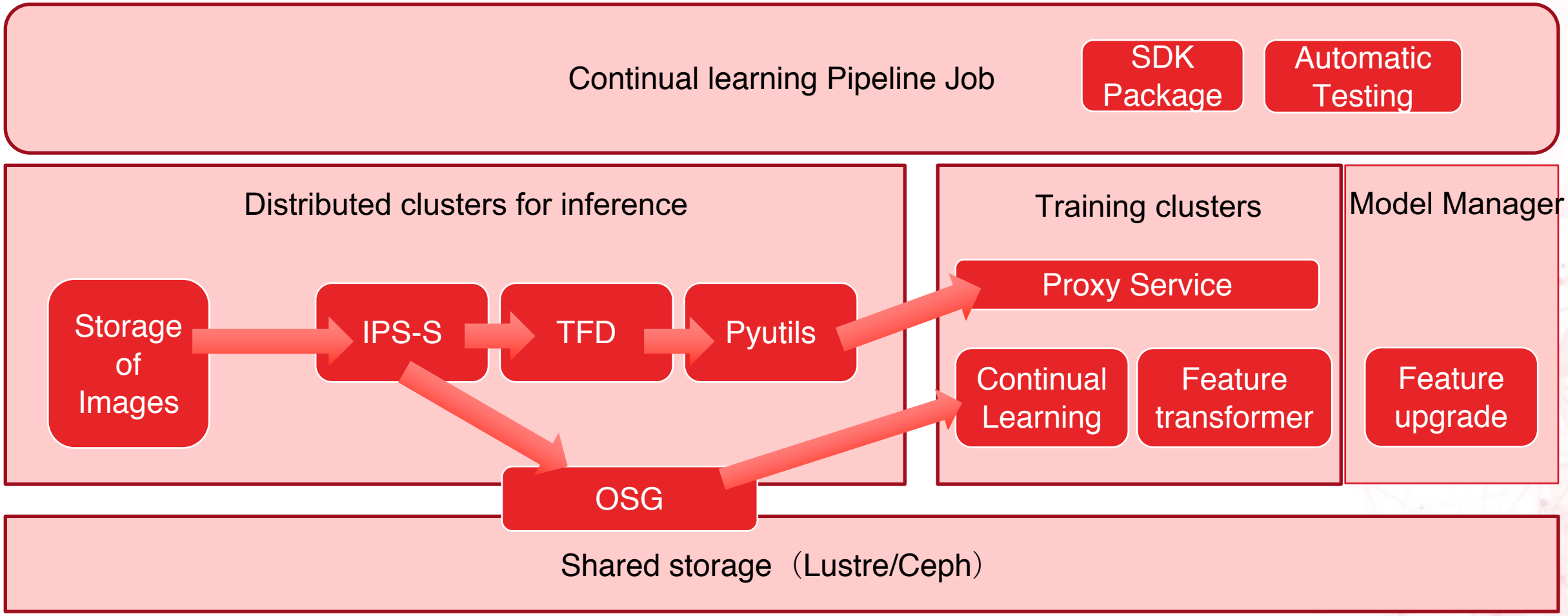
How does it work in real system

System architecture



How does it work in real system

Pipeline



IPS-S: Image processing service
TFD: time-space feature database
Pyutils: spark job service for incremental clustering
OSG: object-based storage

Recommended readings

Kirkpatrick, James, et al. “Overcoming catastrophic forgetting in neural networks.” *Proceedings of the national academy of sciences* 114.13 (2017): 3521-3526. (PNAS 2017)

Li, Zhizhong, and Derek Hoiem. “Learning without forgetting.” IEEE transactions on pattern analysis and machine intelligence 40.12 (2018): 2935-2947. (PAMI 2018)

Hou, Saihui, et al. "Lifelong learning via progressive distillation and retrospection." *Proceedings of the European Conference on Computer Vision (ECCV)*. 2018.

Continual learning workshop in NIPS 2018: https://sites.google.com/view/continual2018

Thank you, and Q&A?

Lead AI Innovation to Power the Future!

If you interested in full-time or internship working opportunities in SenseTime, don't hesitate to contact me by zhaorui@sensetime.com or add WeChat through the above QR code.