Metric Learning

Oleksandr Obiednikov
Head of Faces Research @ RingUkraine

23 Sep 2017

Agenda

- Face Recognition problem
  - Problem settings
  - Is Face Recognition solved? Challenges.
  - Benchmarks, datasets.
- Why it’s not typical? Metric learning.
  - Siamese models
  - Separating vs discriminative powers? How increase discriminative power of the model?
- Do & Don’ts and where are you likely to stumble doing FaceRecognition?
- Open discussion:
  - Applicability range, Person recognition, Faking FaceRecognition systems with photos?

About me

- ❏ Head of Faces Research @ Ring Ukraine

Previously:

- Lecteur of «ML» course @ LITS Kharkiv
- Data Scientist @ Altexsoft / Sleep.ai
- Data Scientist @ TeamDev
- Software Engineer @ GlobalLogic

Educational background:

- Computer Science, Yandex SHAD
- Applied math, Karazine KhNU

Disclaimer

Sorry,

But all the actual numerical results of RingLabs are under NDA

Please don’t ask about “what mAP has you FaceDetector” or “What is the Face Recognition accuracy on your data”, etc.

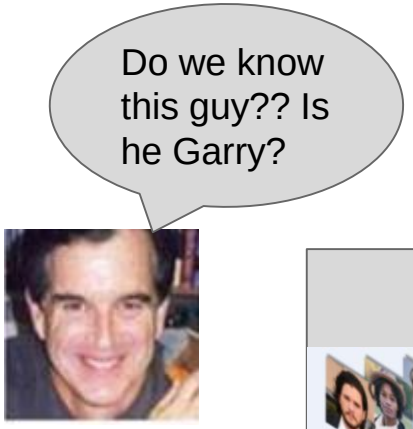
Face Recognition problem

Verification

Input: Two images(videos) with faces
Desired output: Are these faces belong to the same person or not?

Identification

Input: Query image/video with a face
Desired output: Is this guy already in our database? Can we name him/her?

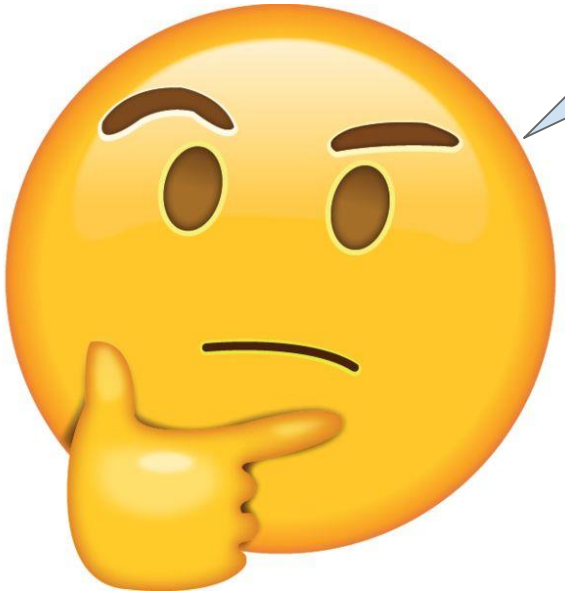


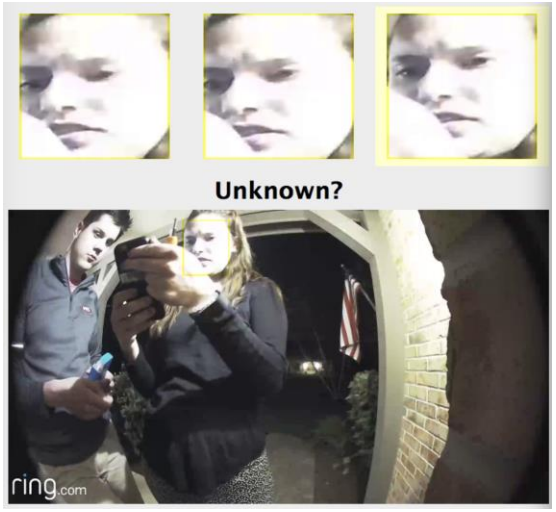
Is it solved?

I’ve heard Face Recognition is solved…
Google and Apple photos, Facebook, even Australia is going to use **Face Recognition** instead of passport..

Is it solved?

Is it really solved?





Challenges

Challenges:

- Difficult initial detection:
  Illuminations, False positives
- High intra-class variability
  Expressions, Aging, Rotations, Apparels
- One-shot learning
  The smaller input from user the better
- Poor input quality of face is too small
  The smaller input from user the better

Face Recognition benchmarks

- **Labeled Faces in the Wild (LFW)**
  + The most popular dataset in FR. Manually created positive and negative pairs.
  - Collected in 2002. Only frontal faces collected with VJ face detector. Relatively small ≈ 6k people
  - *Top verification accuracy > 99%. (in other words fully solved)*
- **IJB-A**
  + Collected in 2015; Images and photos. Manually collected (Amazon Mechanical Turk).
  + *Precision(@Recall = 0.95) = 80%*
  - Not so big amount of training data
- **MS-Celeb-1M**
  + Recall(@ Precision = 0.95) = 73%
  + Huge ≈ 100k unique persons; ≈ 1M images.
  - Collected and labeled automatically. Noisy (≈ 15% noisy images)
- **Others … MegaFace, CA-LFW, Social Face Classification, etc.**

What the problem with these benchmarks?

Face Recognition benchmarks

What the problem with these benchmarks?

1. Validation is done on celebrities => thus it's very likely to have them in training data…
2. It should be relatively big with high variety of poses, illuminations, make-ups, dressing, etc.
3. Quality of the input image may differs from these publicly available datasets

How to solve this?

- Have you own huge database like ***RingLabs*** has :-)

What is not typical in FR?

<Here should be some your ideas, @username>

What is not typical in FR?

From Machine Learning point of view

Typical:

- **Supervised learning:** labels are given in the training set to each example
- Now as usually feature extraction step is done using Deep Learning

Not typical:

- Highly biased to the previous steps like Face Detection
- Not fixed #classes (in this case not fixed # unique users)
- One-shot learning
- Highly unbalanced classes/users in the dataset
- Siamese models

How to solve this?

- Metric learning approach

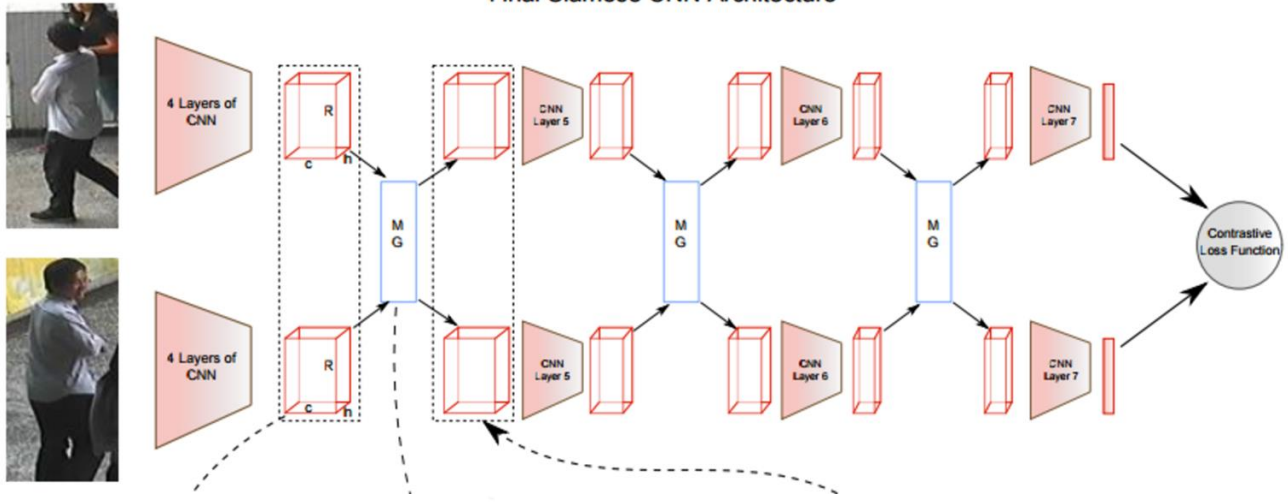
Siamese models

Metric learning via Siamese models

Siamese model = N "same" models (with sharable weights)

N inputs → N models → compare the distances between N outputs.

Idea: Same objects should have close model output. Different objects should have distant model output.



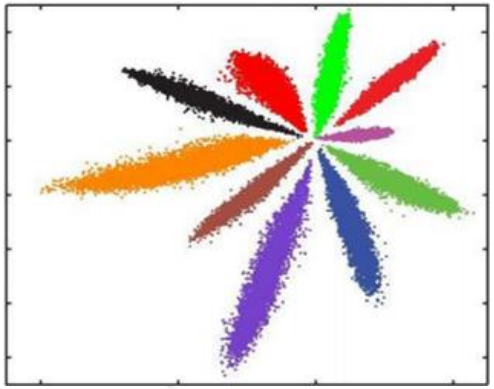
Discriminative Metric Learning

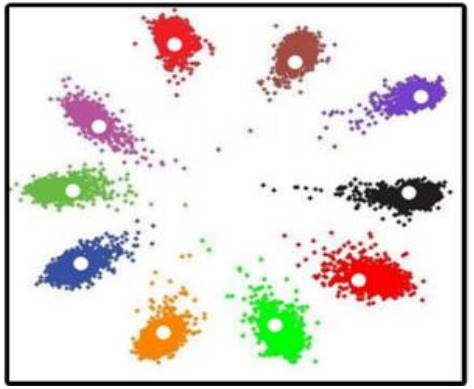
Metric Learning = Discriminative Feature learning = Similarity Learning
Idea: Learn a similarity function that measures how similar or related two objects are.

Applications: Word2Vec & Co., recommender systems, recognition and descriptive problems, etc.

What the difference between separative and discriminative features?

Open question: *How discriminative power correlates with generalization power?*

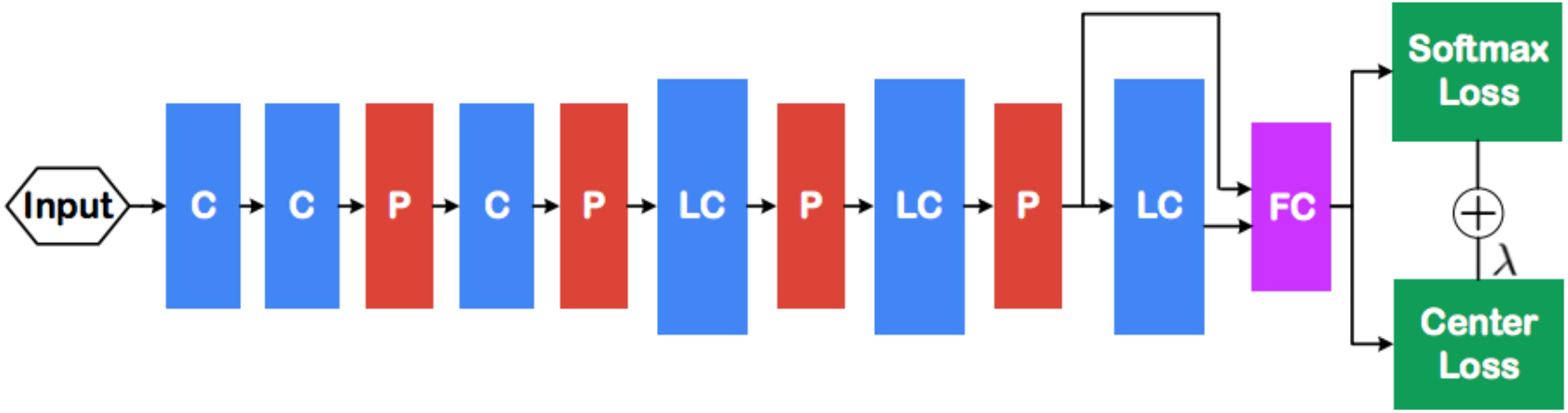




Discriminative Metric Learning

How to learn discriminative features?

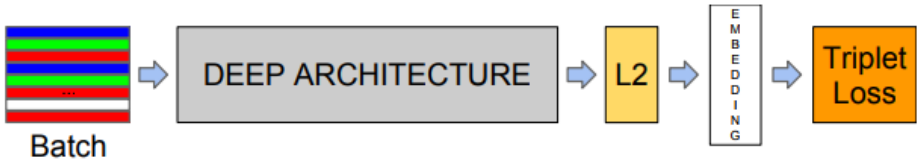
- Use special loss function that learns similarity between objects. *(Triplet-loss, Siamese models)*
- Add special regularization term to enforce discriminative power. (Center-loss)
- Improve softmax (categorical-cross entropy) to learn not only separable, but also discriminative features. *(L-Softmax, A-Softmax, NormFace, etc.)*

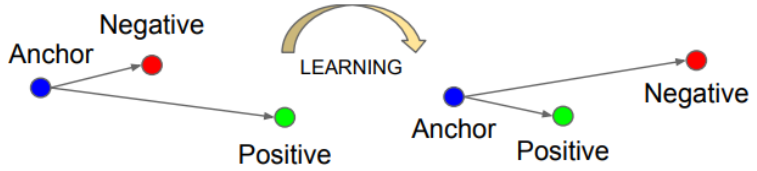


Metric learning via triplet-loss

Triplet loss

- \+ Very intuitive
- \- Triplet mining matters
- \- Triplet mining is super expensive





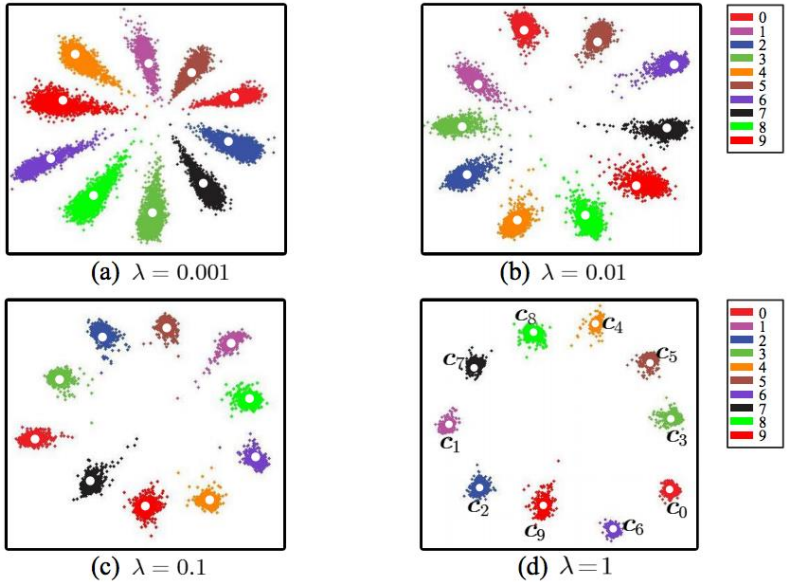
$$\sum_{i}^{N} \left[\|f(x_i^a) - f(x_i^p)\|_2^2 - \|f(x_i^a) - f(x_i^n)\|_2^2 + \alpha \right]_+$$

Metric learning via center-loss

Idea: Let's add a penalty for big inner-class distance.

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

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



(a) $\lambda = 0.001$ (b) $\lambda = 0.01$

(c) $\lambda = 0.1$ (d) $\lambda = 1$

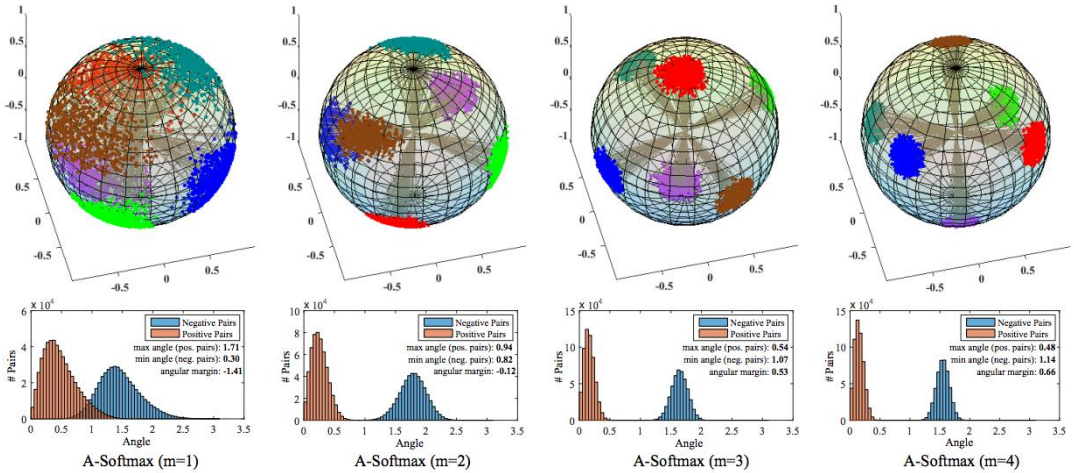
Center-loss

- \+ No triplet mining
- \+ Cheap computationally
- \- No penalty for outer-class distance

How to improve?

- *<Your suggestion>*
- ***NormFace***

SphereFace. A-Softmax



A-Softmax (m=1) A-Softmax (m=2) A-Softmax (m=3) A-Softmax (m=4)

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

Regular softmax on Sphere:

$$L_i = -\log\left(\frac{e^{\boldsymbol{W}_{y_i}^T \boldsymbol{x}_i + b_{y_i}}}{\sum_j e^{\boldsymbol{W}_j^T \boldsymbol{x}_i + b_j}}\right)$$

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

Note: cos($\boldsymbol{\theta}$) = angle between weights W and x

Do & Don'ts of Face Recognition

Dataset (As usually “good dataset” is all you need.)

- **Wider** dataset is **better** than **deeper**
- **The wider** dataset **the better** results
- **The cleaner** dataset **the better** accuracy
- **Mixing** video frame and still images **is better** than using only one of this

Training

- Use **Center-loss + L2 norm** or **A-Softmax**
- **Do not overfeat** on celebrities. Don’t use LFW as a validation set without cleaning you training data from it.
- Augment your data.
- Aligning is important
- Be careful with input (pay attention on your face detector and how it crops faces)

Open discussion and questions

If you don’t know what to ask…

- Applicability range
- Person recognition
- Faking FaceRecognition systems with photos?
- <Your question>

Thank you for your attention!