{ "smartmail_hack": 20.18 }

Face Recognition: From Scratch To Hatch

Eduard Tyantov

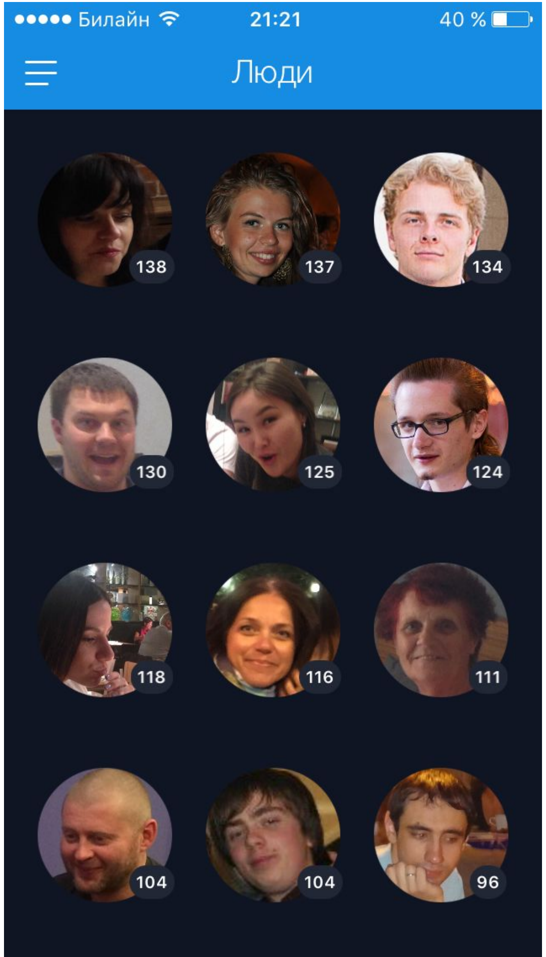
Head of Machine Learning group at Mail.Ru

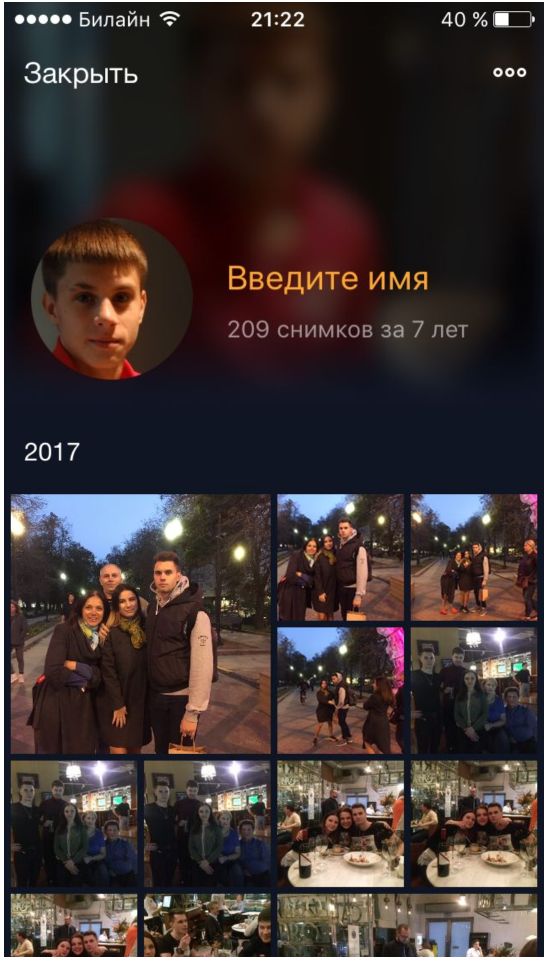
@mail.ru

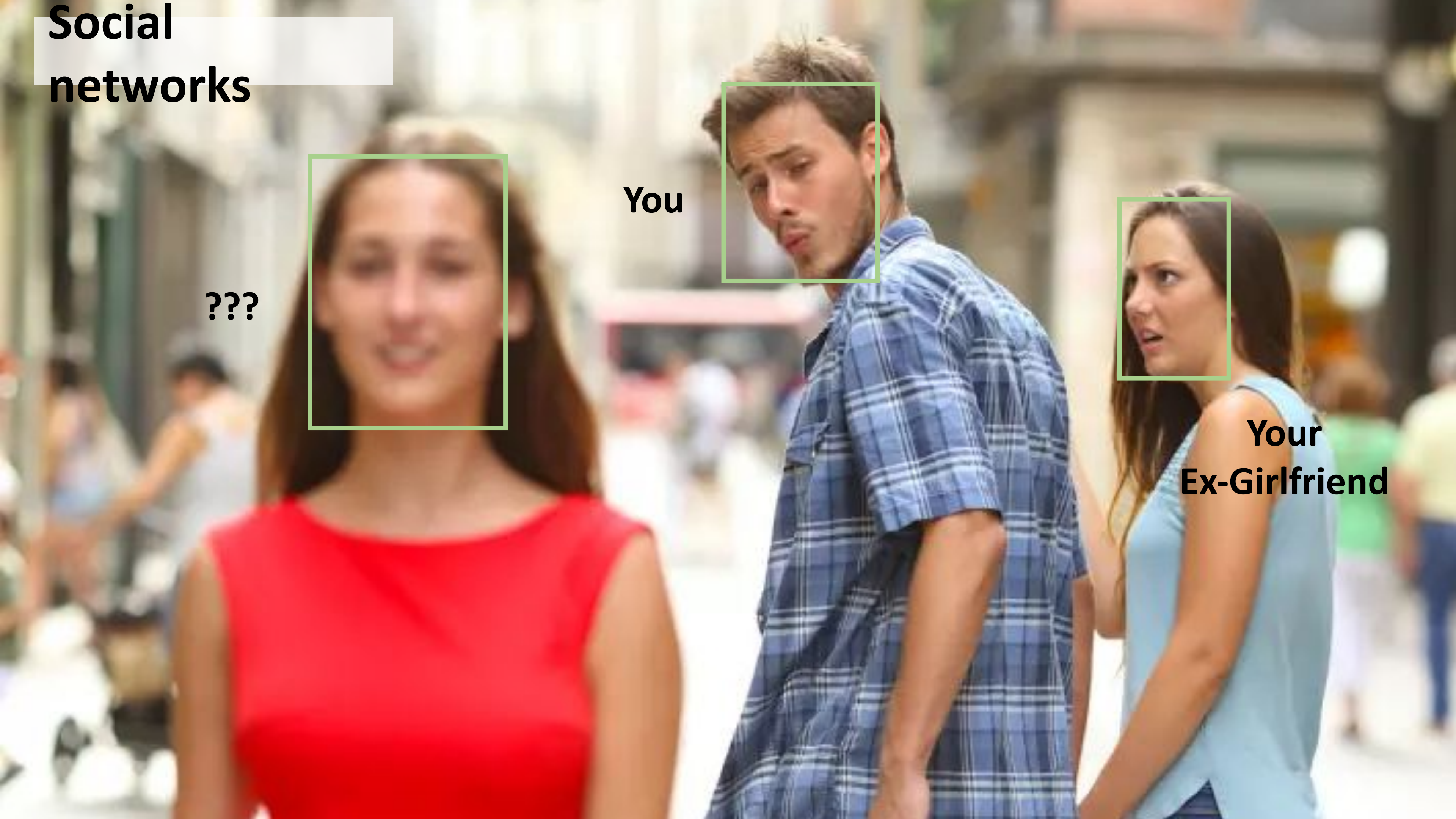
Face Recognition in Cloud@Mail.ru

Users upload photos to Cloud

Backend identifies persons on photos, tags and show clusters



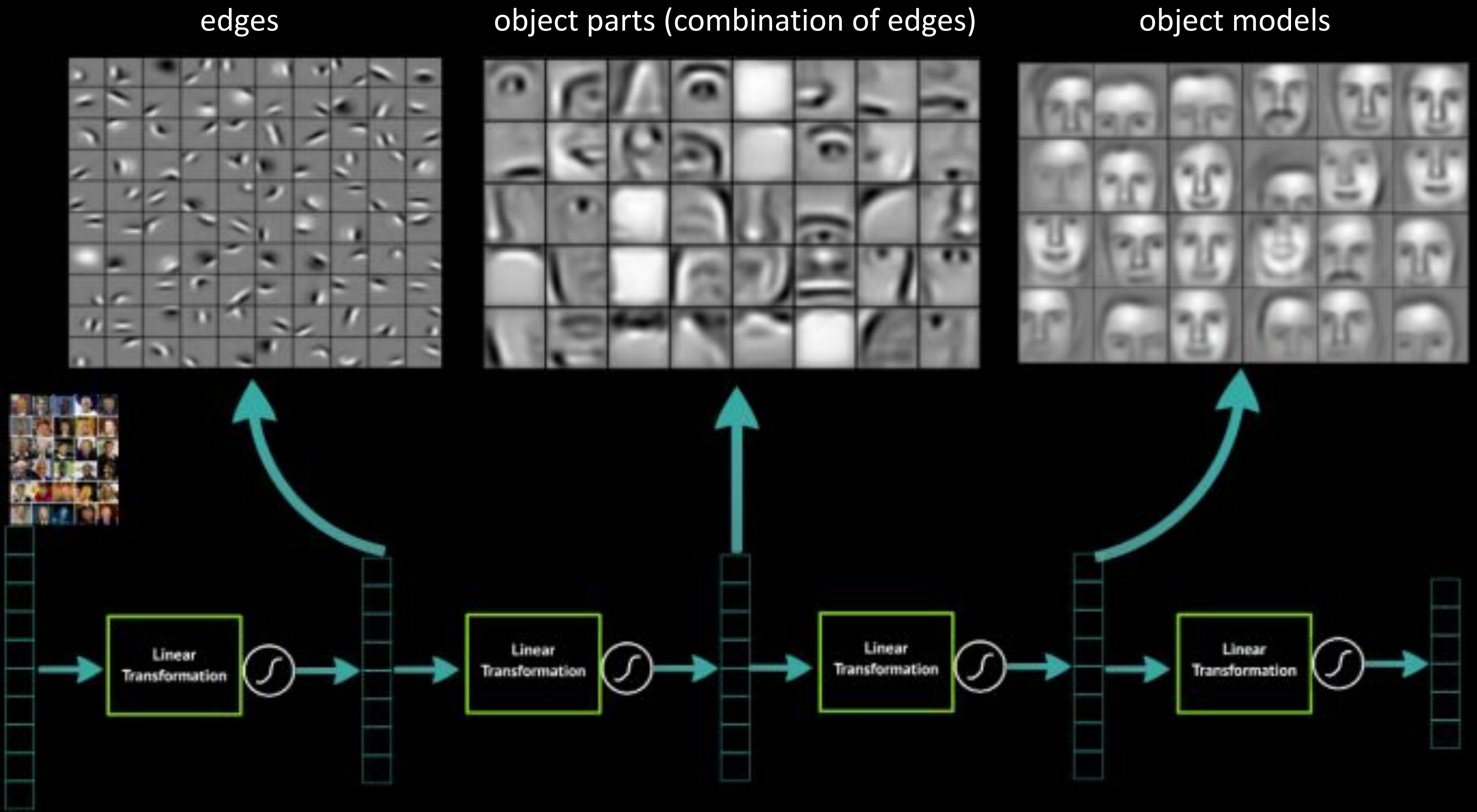



Social networks
???
You
Your Ex-Girlfriend

Convolutional neural networks,

briefly

Deep Learning learns layers of features

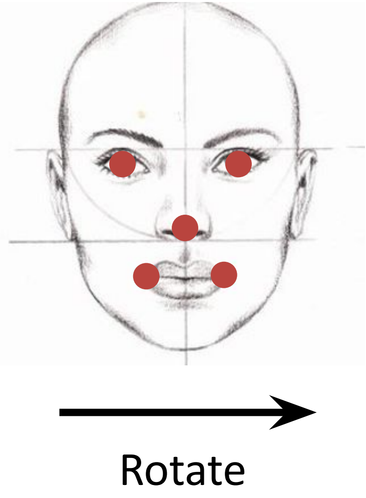


Face Detection

Face detection

Auxiliary task: facial landmarks

- Face alignment: rotation
- Goal: make it easier for Face Recognition

Train Datasets

Wider

– 32k images
– 494k faces

Celeba

– 200k images, 10k persons
– Landmarks, 40 binary attributes

Test Dataset: FDDB

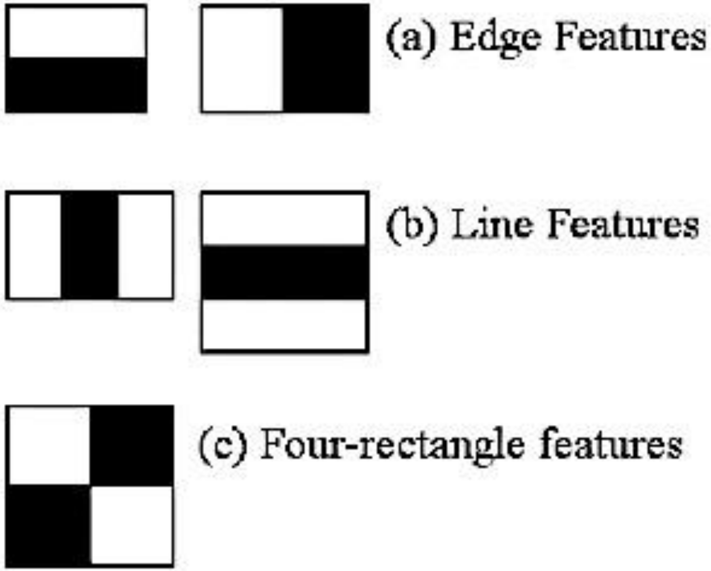
Face Detection Data Set and Benchmark

- 2845 images
- 5171 faces

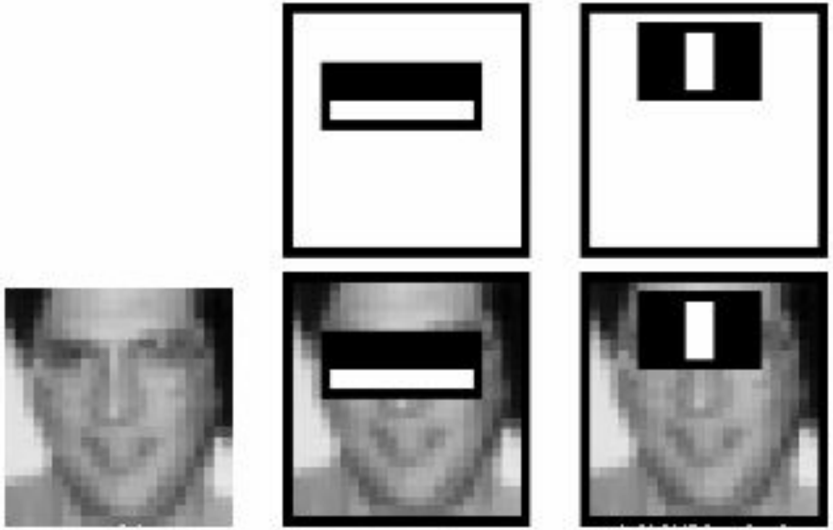
Old school: Viola-Jones

Haar Feature-based Cascade Classifiers

Haar-like features



Examples

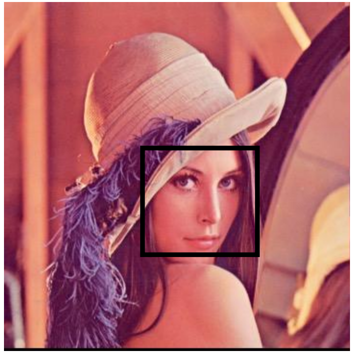


eyes darker

nose lighter

Viola-Jones algorithm: training

for each patch

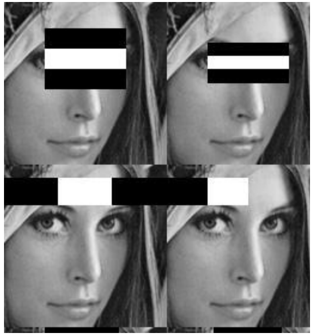


features

160k

AdaBoost ensemble

valuable



6k

weighted sum

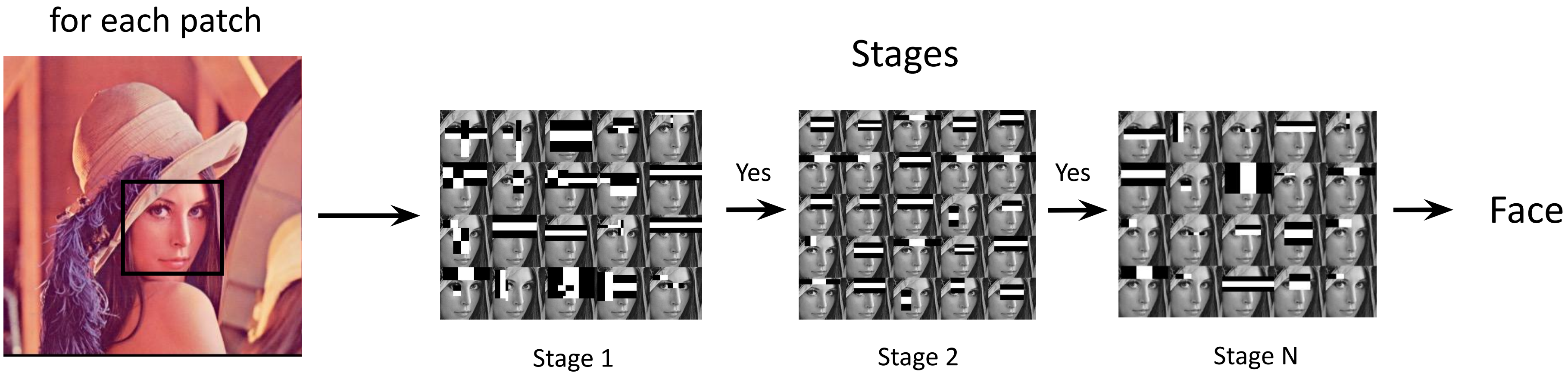
Face or Not

Dataset

Viola-Jones algorithm: inference

Optimization

- Features are grouped into stages
- If a patch fails any stage => discard

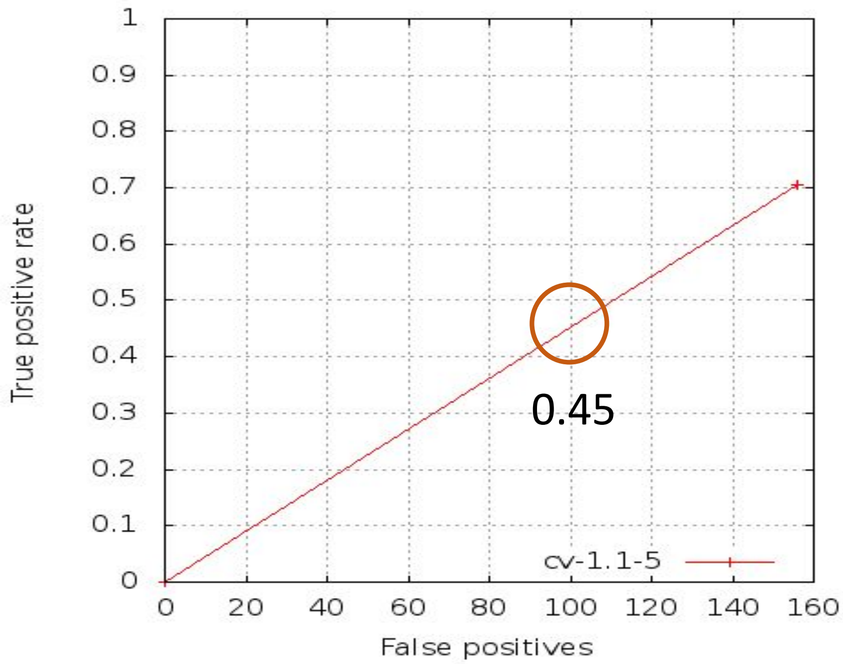


Viola-Jones results

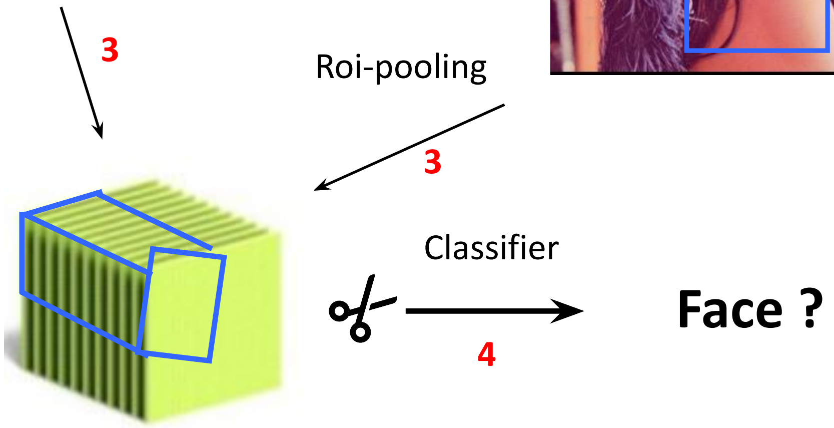
OpenCV implementation

- Fast: ~100ms on CPU
- Not accurate

FDDB results



New school: Region-based Convolutional Networks

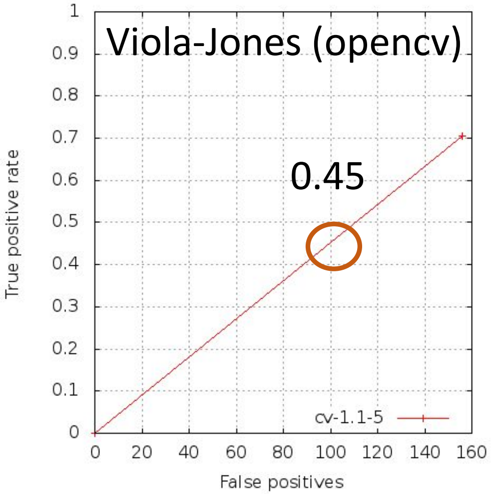


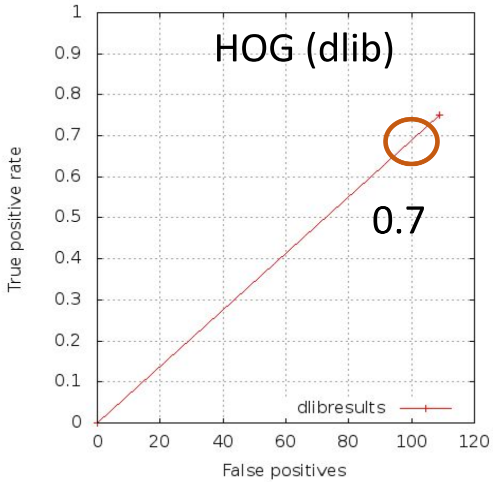
Faster RCNN, algorithm

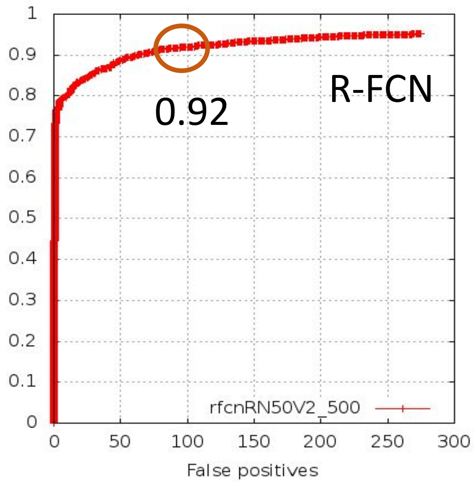
4. Classifier: classes and the bounding box

Comparison: Viola-Jones vs R-FCN

FDDB results







Results

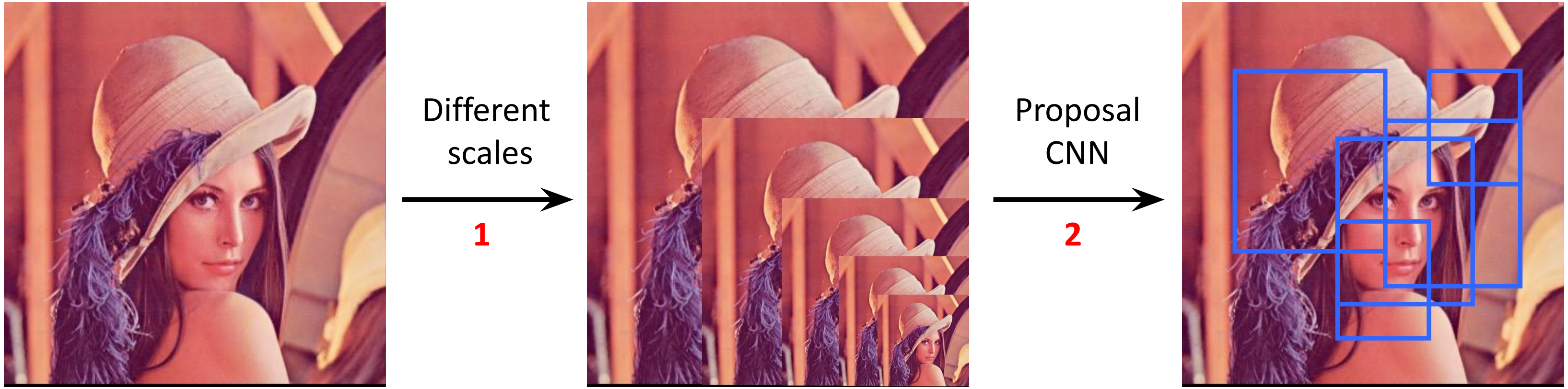
- **92%** accuracy (R-FCN)
- **40ms** on GPU (**slow**)

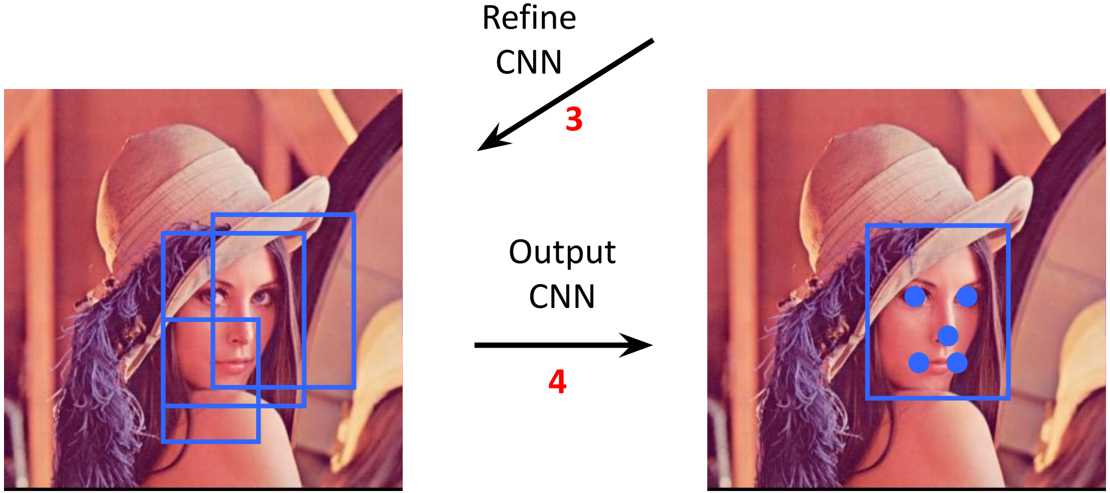
Face detection: how fast

We need faster solution at the same accuracy!

Target: **< 10ms**

Alternative: MTCNN





Cascade of 3 CNN

4. Output -> b-boxes + landmarks

Comparison: **MTCNN vs R-FCN**

MTCNN

- \+ Faster
- \+ Landmarks
- \- Less accurate
- \- No batch processing

Model	GPU Inference	FDDB Precision (100 errors)
R-FCN	40 ms	**92%**
MTCNN	**17 ms**	90%

TensorRT

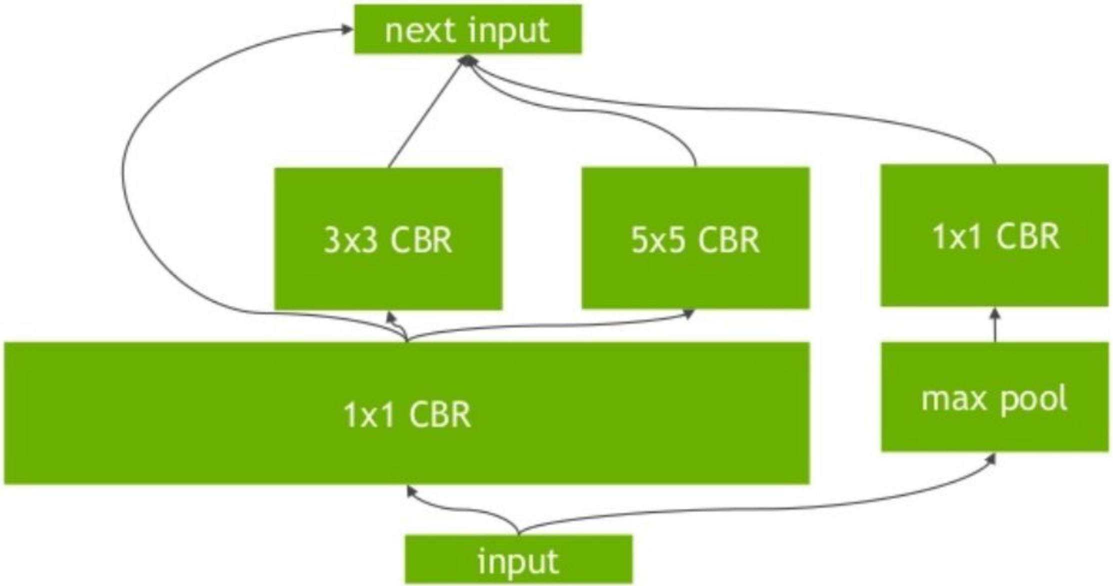
What is TensorRT

NVIDIA TensorRT is a high-performance deep learning inference optimizer

Features

- Improves performance for complex networks
- FP16 & INT8 support
- Effective at small batch-sizes

TensorRT: layer optimizations



1. Vertical layer fusion
2. Horizontal fusion
3. Concat elision

TensorRT: downsides

1. Caffe + TensorFlow supported
2. Fixed input/batch size
3. Basic layers support

Batch processing

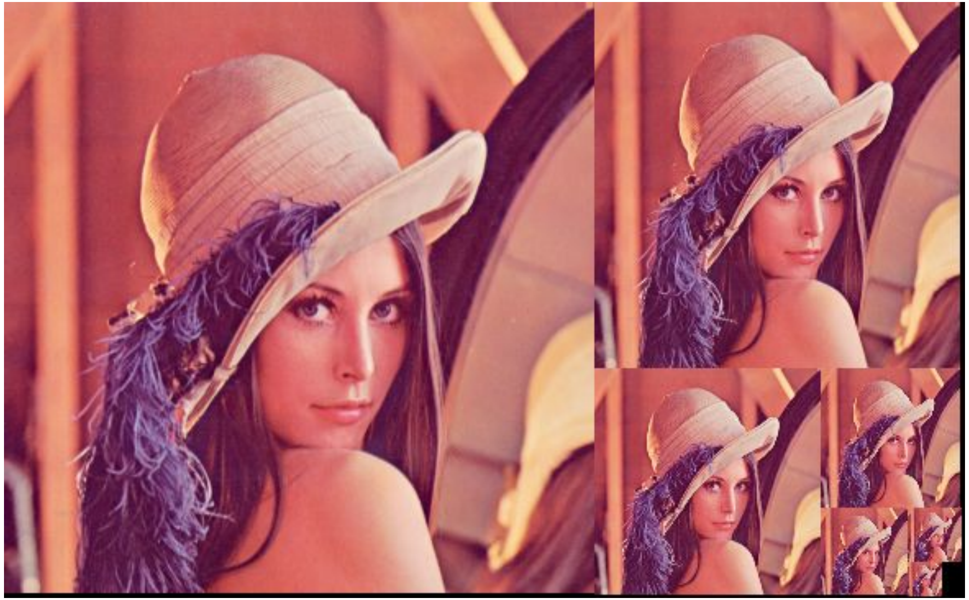
Problem

Image size is fixed, but

MTCNN works at different scales

Solution

Pyramid on a single image



Batch processing

Results

- Single run
- Enables batch processing

Model	Inference ms
MTCNN (Caffe, python)	17
MTCNN (Caffe, C++)	12.7
+ batch	**10.7**

TensorRT: layers

Problem

No PReLU layer => default pre-trained model can't be used

Model	GPU Inference ms	FDDB Precision (100 errors)
MTCNN, batch	10.7	90%
+Tensor RT	**8.8**	**91.2%**

Retrained with ReLU from scratch

-20%

Face detection: inference

Target: **< 10 ms**

Result: 8.8 ms

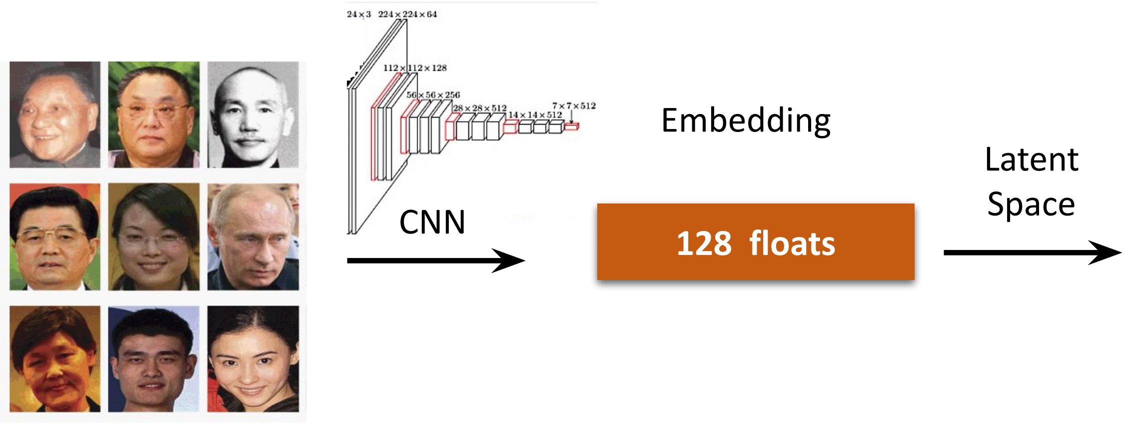
Ingredients

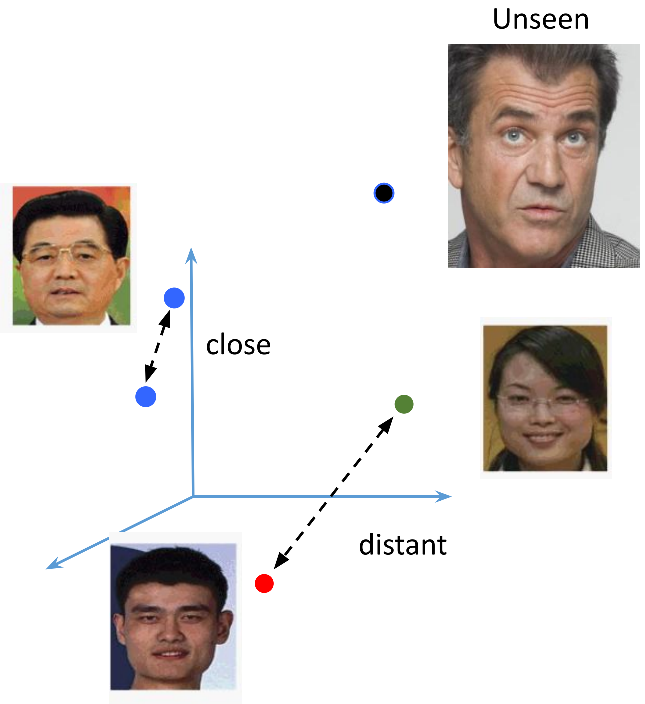
1. MTCNN
2. Batch processing
3. TensorRT

Face Recognition

Face recognition task

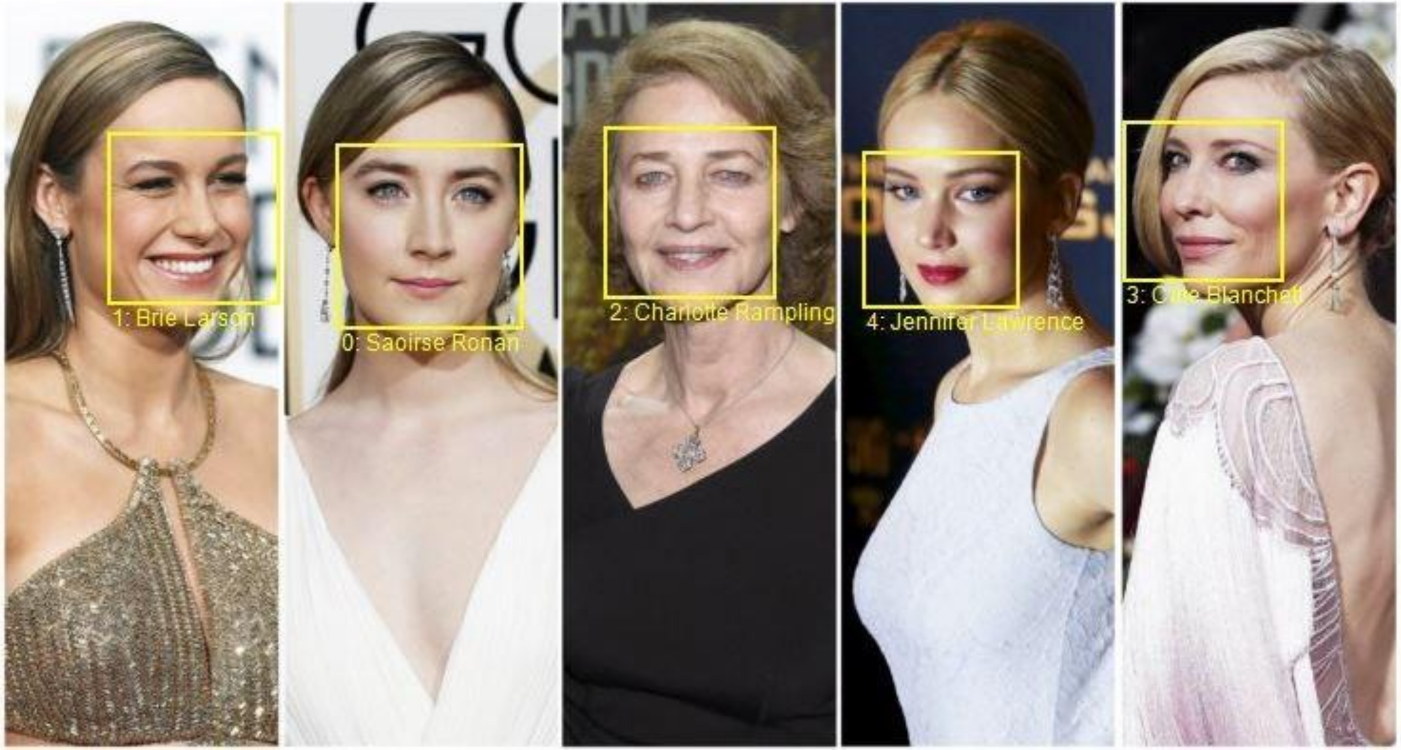
- Goal – to compare faces
- How? To learn metric
- To enable **Zero-shot** learning





Training set: MSCeleb

- Top 100k celebrities
- 10 Million images, 100 per person
- Noisy: constructed by leveraging public search engines



Small test dataset: LFW

Labeled Faces in the Wild Home

- 13k images from the web
- 1680 persons have >= 2 photos

Large test dataset: Megaface

- Identification under up to 1 million “distractors”
- 530 people to find

Explore Most Recent Public Results (last update 3/12/2017) >>

Challenge 1: Train on any dataset, test your method with **1 million** distractors

Participate and download Challenge 1 >>

Challenge 2: Training on **672K** identities (4.7 Million photos), test at Million scale

Participate and download Challenge 2 >>

Megaface leaderboard

Algorithm	Date Submitted	Set 1	Set 2	Set 3	Data Set Size
BingMMLab V1(iBUG cleaned data)	4/10/2018	98.998%	98.998%	98.998%	Large
Orion Star Technology (clean)	3/21/2018	98.355%			Large
iBUG_DeepInsight	2/8/2018	98.063%	98.058%	98.053%	Large
EM-DATA	4/4/2018	96.653%	96.653%	96.653%	Large
SuningUS_AILab	3/21/2018	96.2618947%	96.2618947%	96.2618947%	Large
StartDT-AI	4/16/2018	93.8226%	93.8226%	93.8226%	Large
Intellivision	2/11/2018	93.125%	93.123%	93.136%	Large
ULSee - Face Team	3/27/2018	92.172%			Large
Vocord - deepVo V3	04/27/2017	91.763%	91.711%	91.704%	Large
MTDP_ITC	12/21/2017	87.098%	83.877%	87.184%	Large
TUPUTECH	12/22/2017	86.558%	86.557%	86.579%	Large
Video++	1/5/2018	85.74%	85.737%	85.735%	Large
THU CV-AI Lab	12/12/2017	84.521%	84.513%	84.514%	Large
TencentAILab_FaceCNN_v1	9/21/2017	84.261%	84.255%	84.257%	Large
BingMMLab-v1 (non-cleaned data)	4/10/2018	83.758%	83.758%	83.758%	Large
Orion Star Technology (no clean)	3/21/2018	83.569%			Large
YouTu Lab (Tencent Best-Image)	04/08/2017	83.29%	83.267%	83.295%	Large

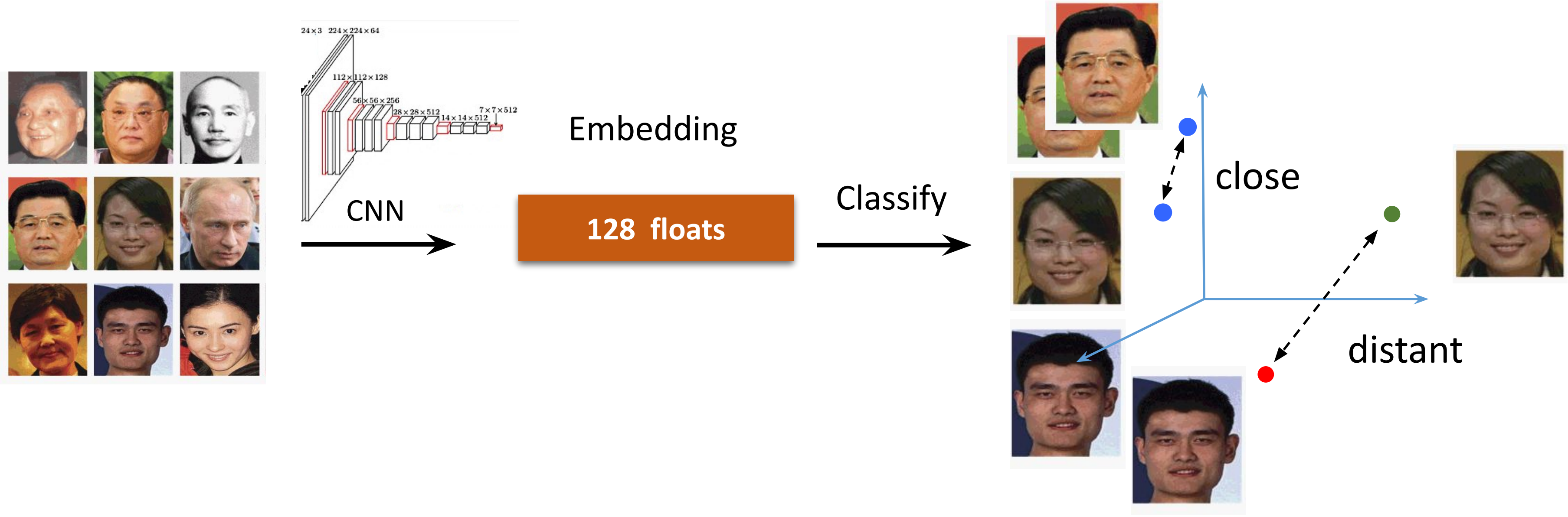
~98%
cleaned

~83%

Metric Learning

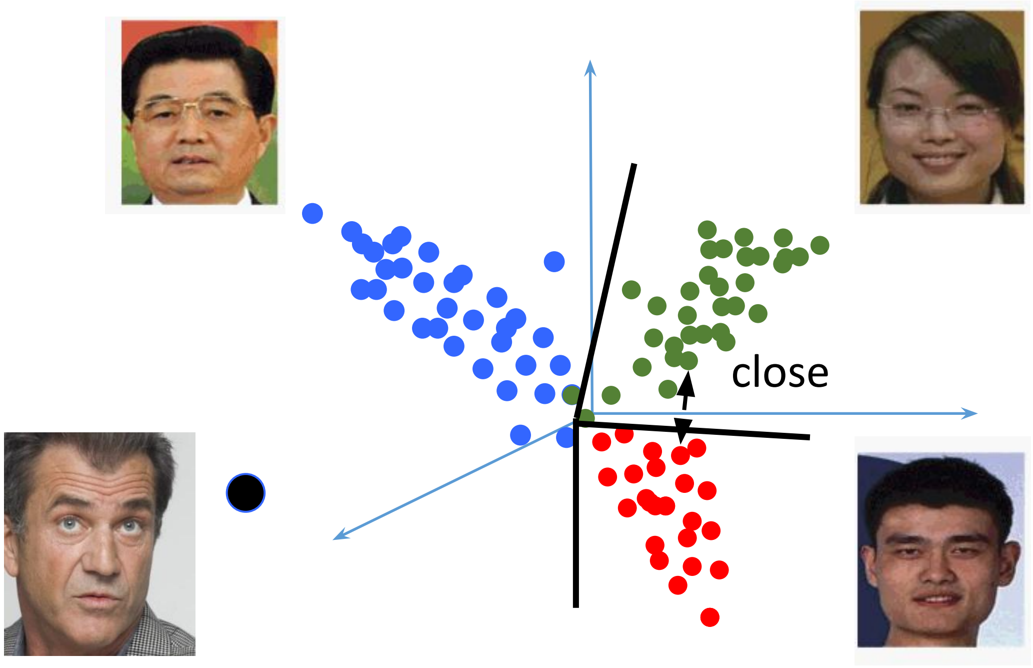
Classification

- Train CNN to predict classes
- Pray for good latent space



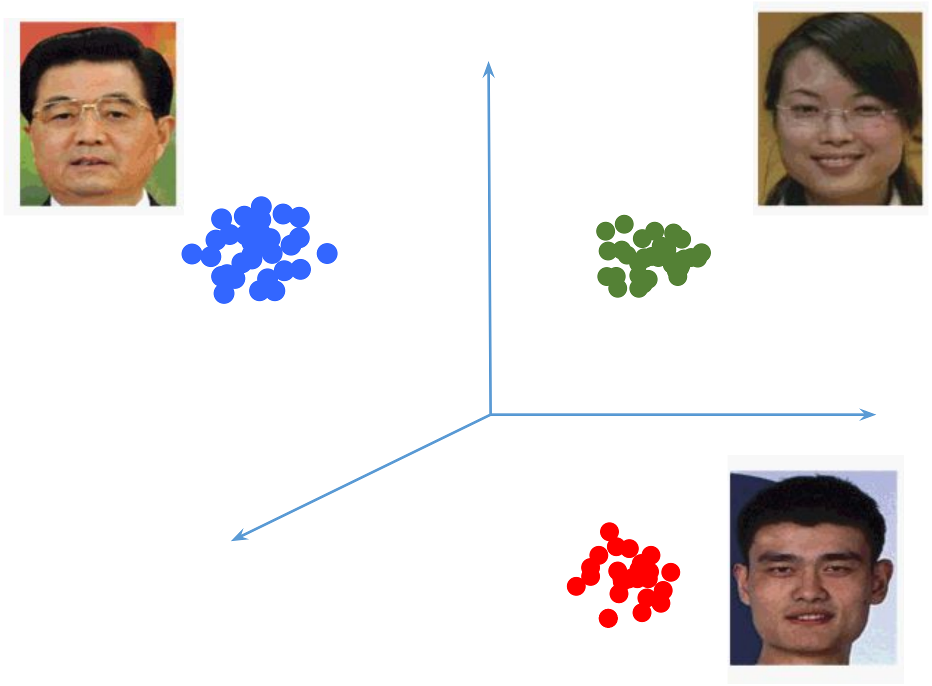
Softmax

- Learned features only separable but not discriminative
- The resulting features are not sufficiently effective

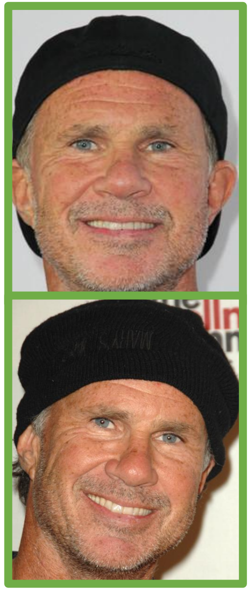


We need metric learning

- Tightness of the cluster
- Discriminative features



Triplet loss



Positive

minimize

Anchor

maximize

Negative

$$\text{positive} + \alpha < \text{negative}$$

Features

- Identity -> single point
- Enforces a margin between persons

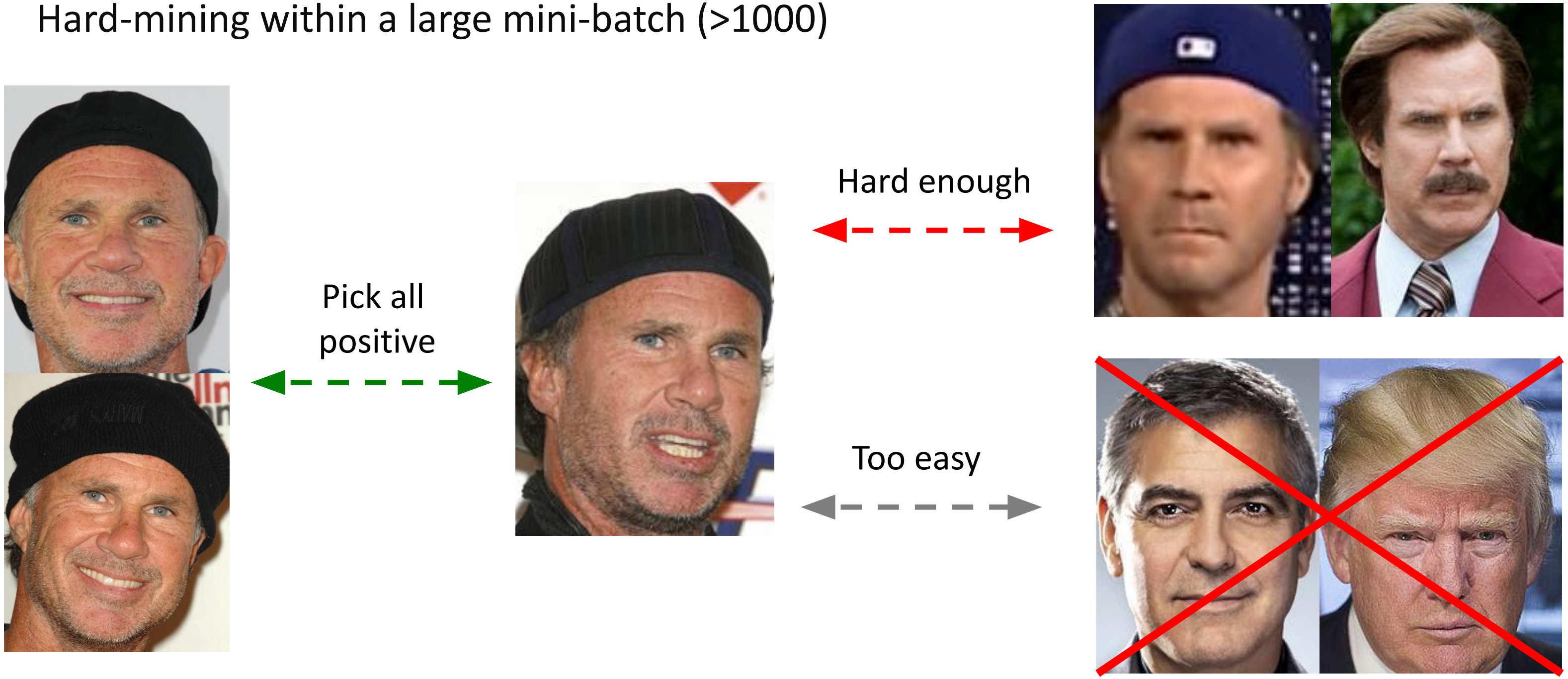
Choosing triplets

Crucial problem

How to choose triplets ? Useful triplets = hardest errors

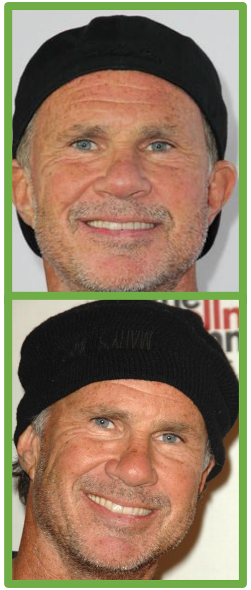
Solution

Hard-mining within a large mini-batch (>1000)



Choosing triplets: trap

Choosing triplets: trap



Positive

minimize

Anchor

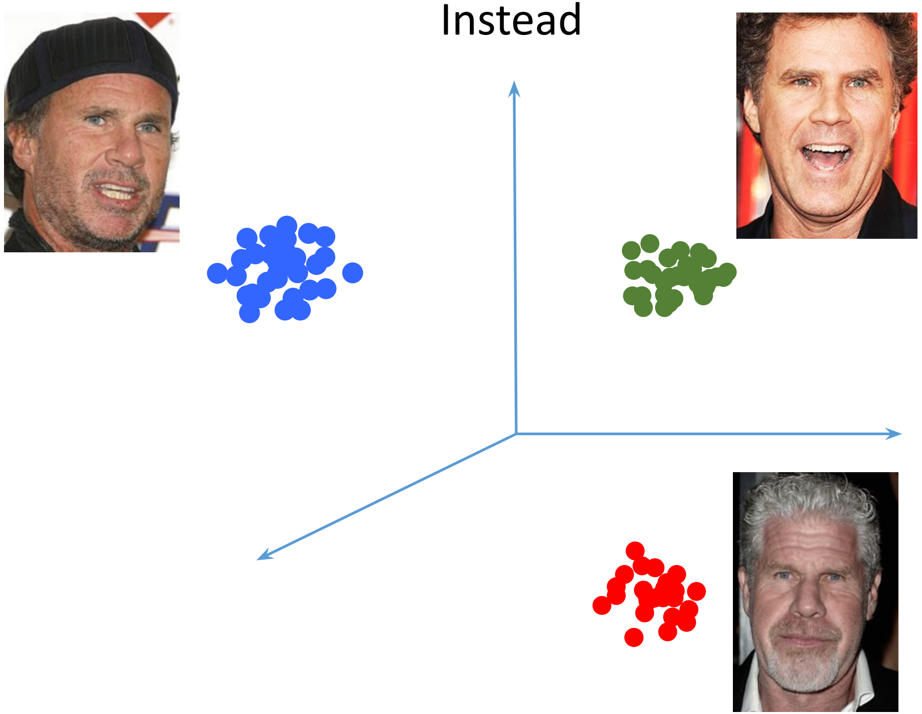
maximize

Negative

positive ~ negative

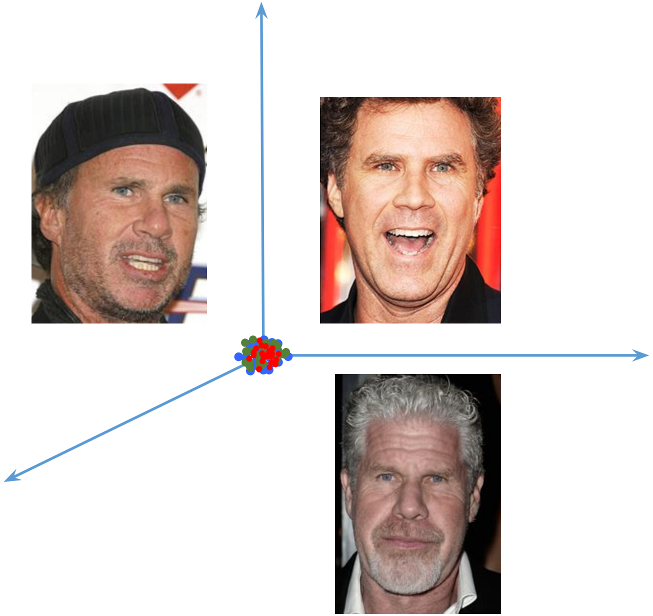
Choosing triplets: trap

Instead

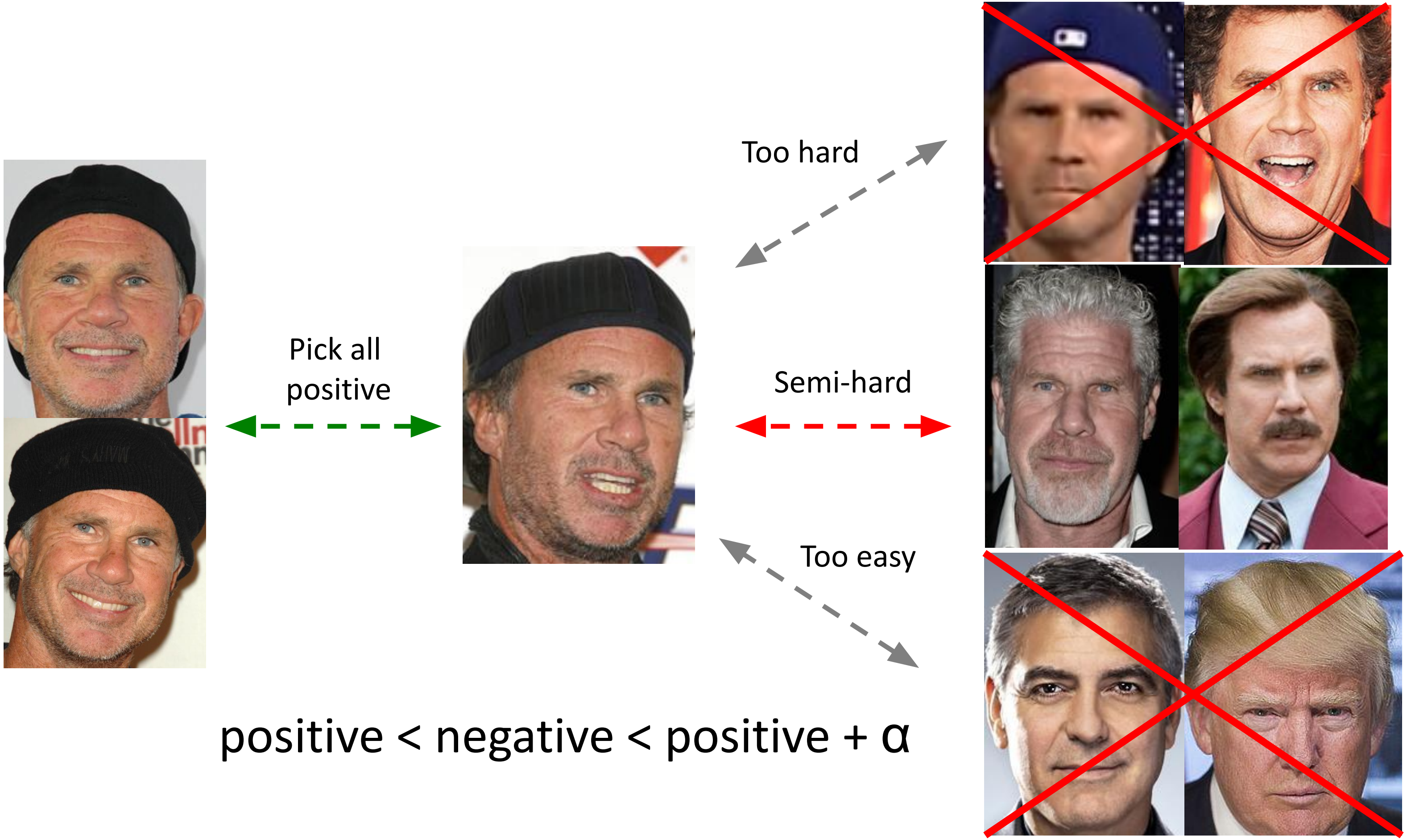


Choosing triplets: trap

Selecting hardest negative may lead to the collapse early in training



Choosing triplets: semi-hard



$$\text{positive} < \text{negative} < \text{positive} + \alpha$$

Triplet loss: summary

Overview

- Requires large batches, margin tuning
- Slow convergence

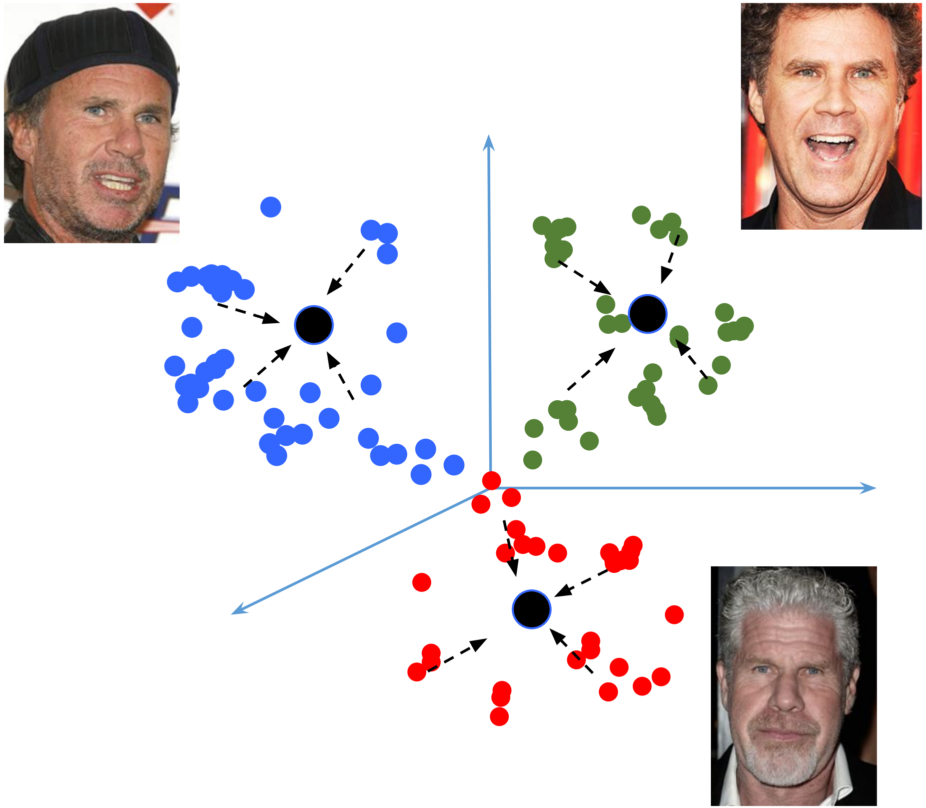
Opensource Code

- Openface (Torch)
  - suboptimal implementation
- Facenet, not original (TensorFlow)

	LFW, %	Megaface
Openface (Torch)	92	-
Our (Torch)	99.35	**65**
Google’s Facenet	99.63	70.5

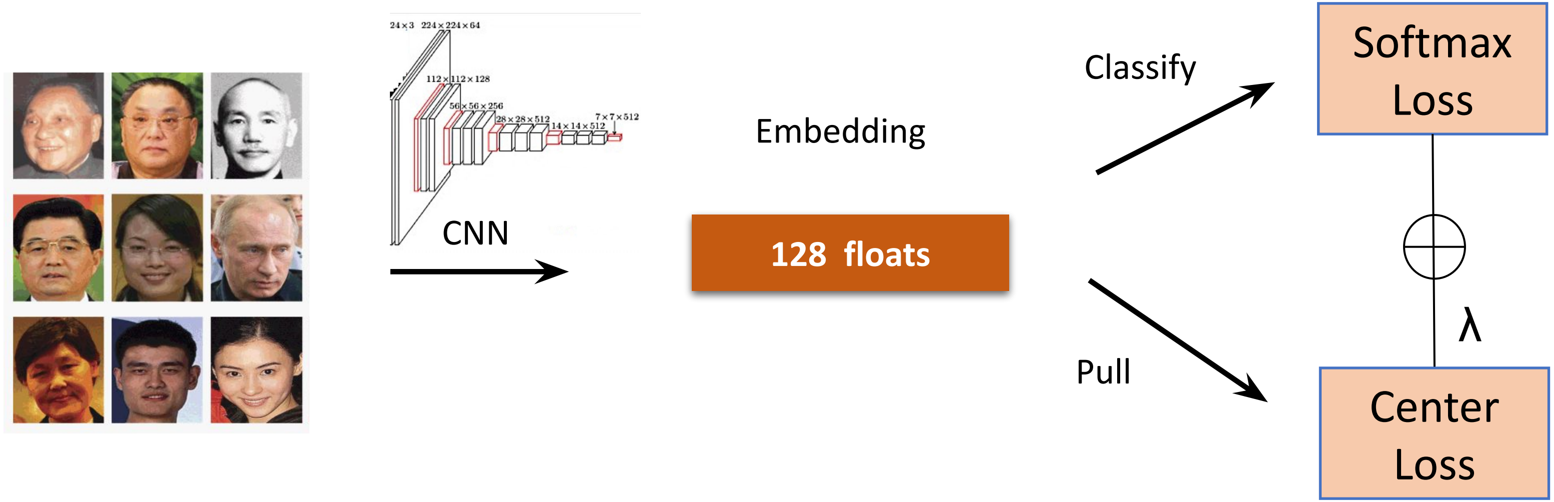
Center loss

Idea: pull points to class **centroids**



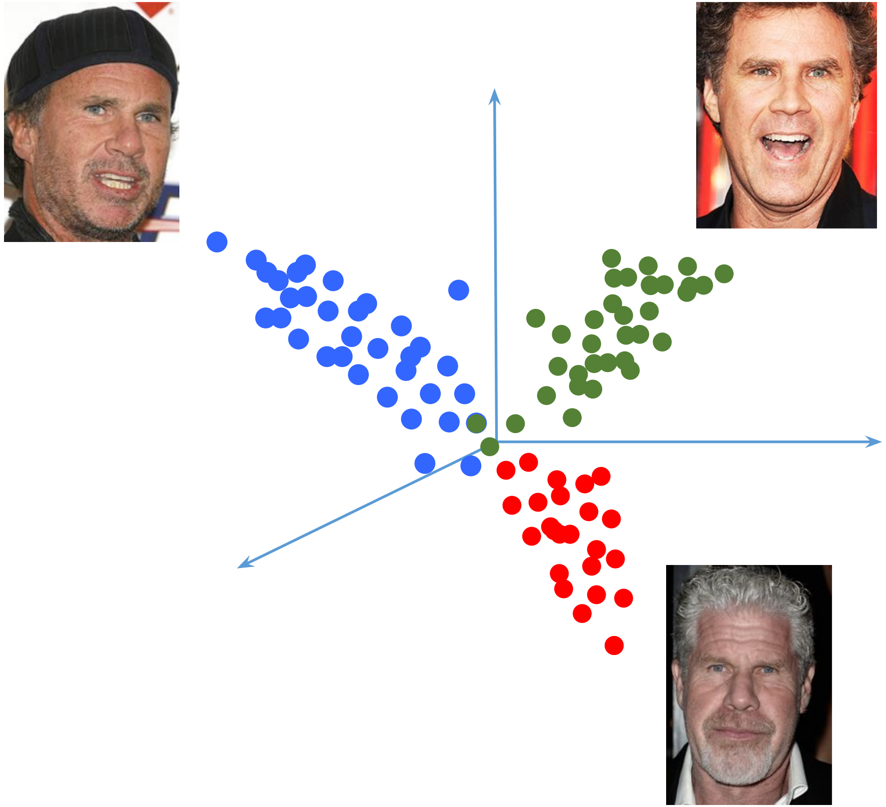
Center loss: structure

- Without classification loss – collapses
- Final loss = Softmax loss + λ Center loss



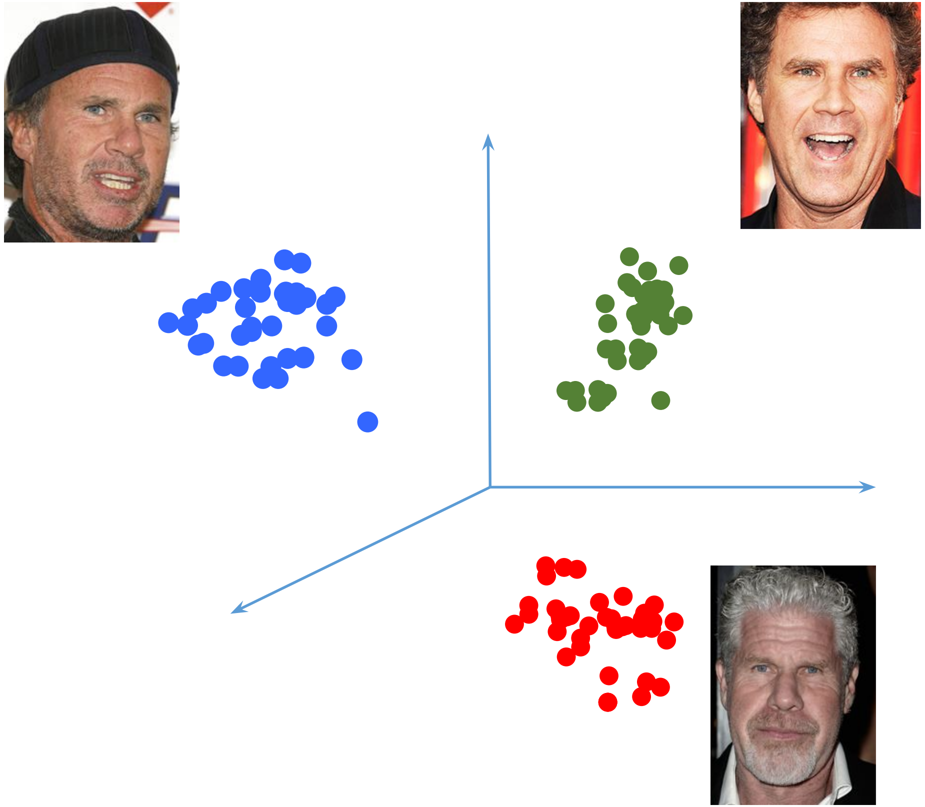
Center Loss: different lambdas

$\lambda = 10^{-7}$



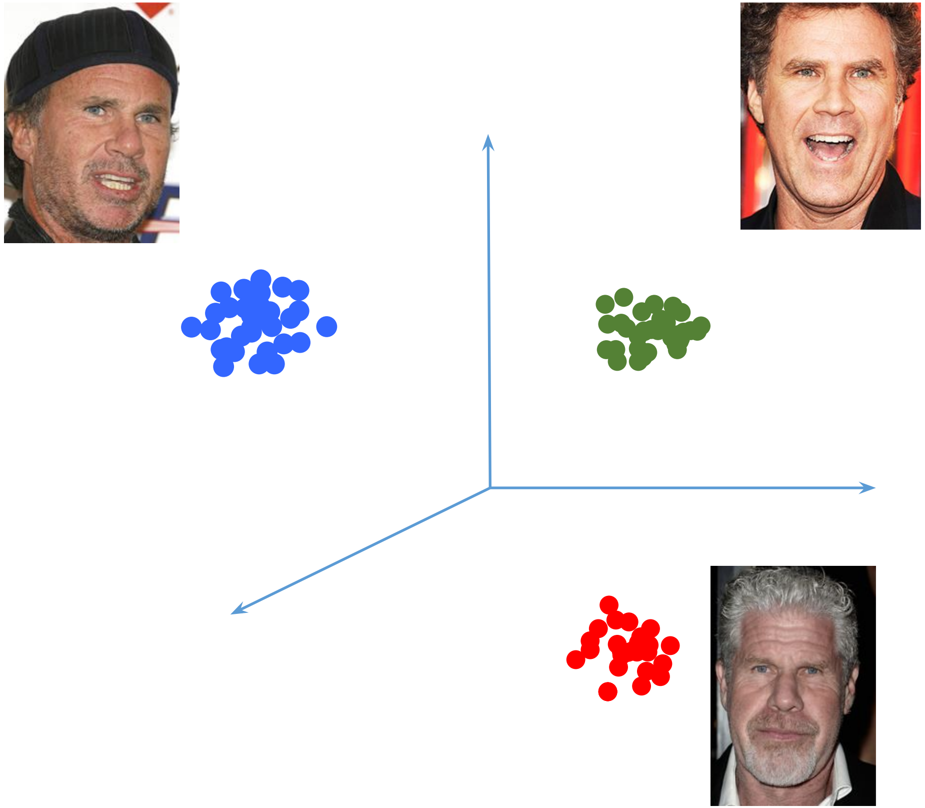
Center Loss: different lambdas

$\lambda = 10^{-6}$



Center Loss: different lambdas

$\lambda = 10^{-5}$



Center loss: summary

Overview

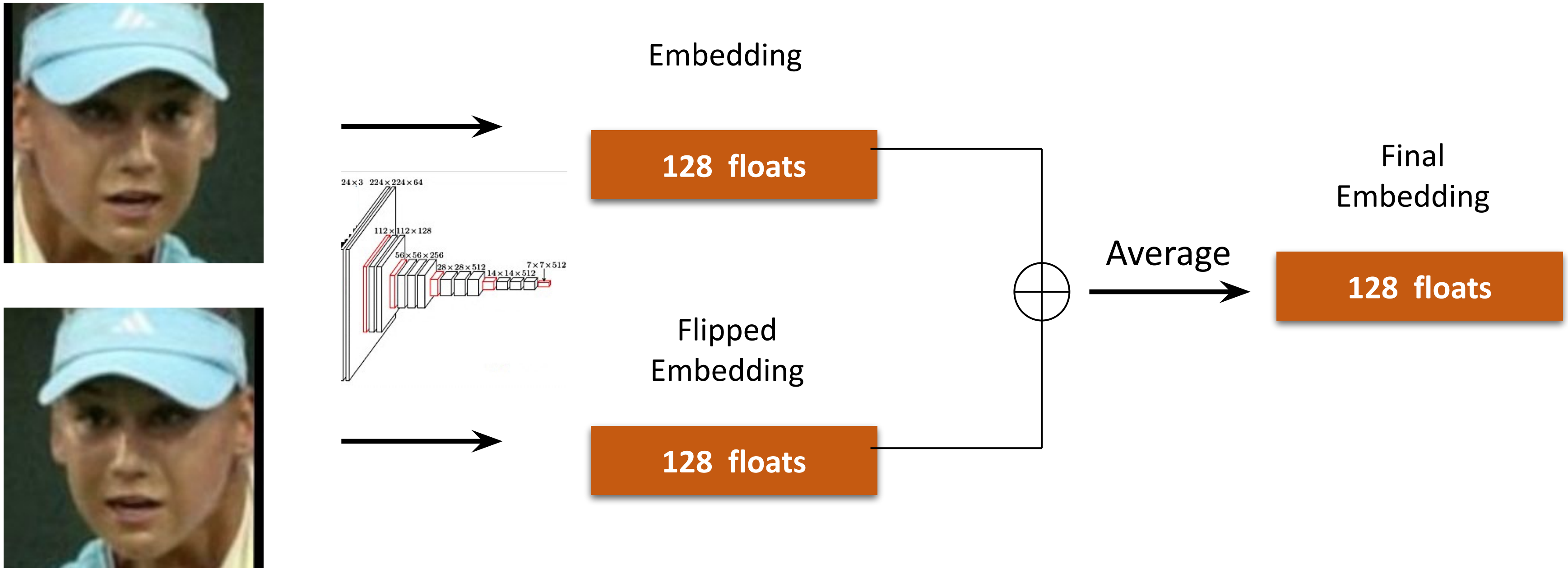
- Intra-class compactness and inter-class separability
- Good performance at several other tasks

	LFW, %	Megaface
Triplet Loss	99.35	65
Center Loss (Torch, ours)	99.60	**71.7**

Opensource Code

- Caffe (original, Megaface - 65%)

Tricks: augmentation



Test time augmentation

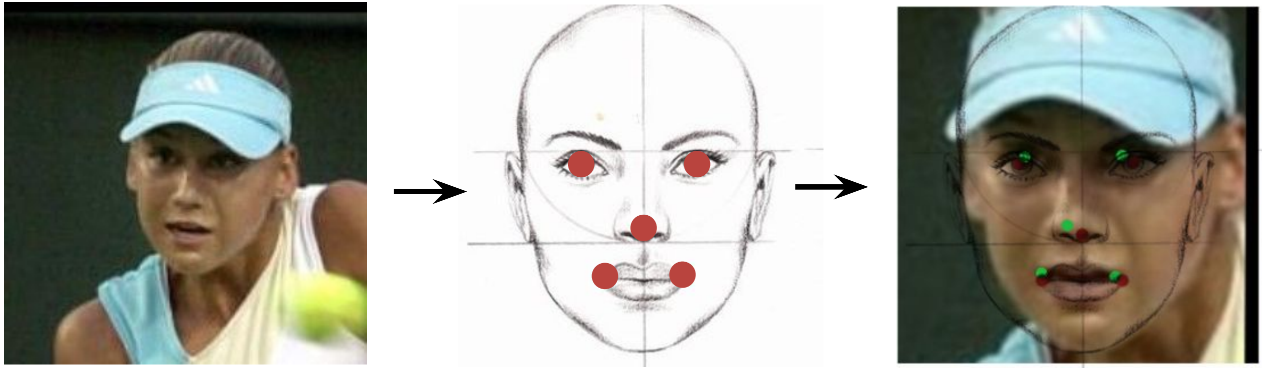
- Flip image
- Compute 2 embeddings
- Average embeddings

Tricks: alignment

	LFW, %	Megaface
Center Loss	99.6	71.7
Center Loss + Tricks	99.68	**73**

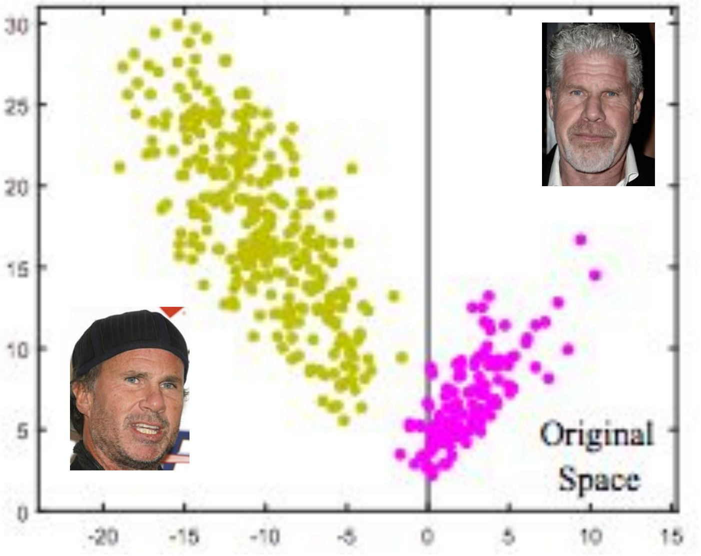
Rotation

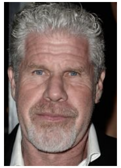
Kabsch algorithm - the optimal rotation matrix that minimizes the RMSD

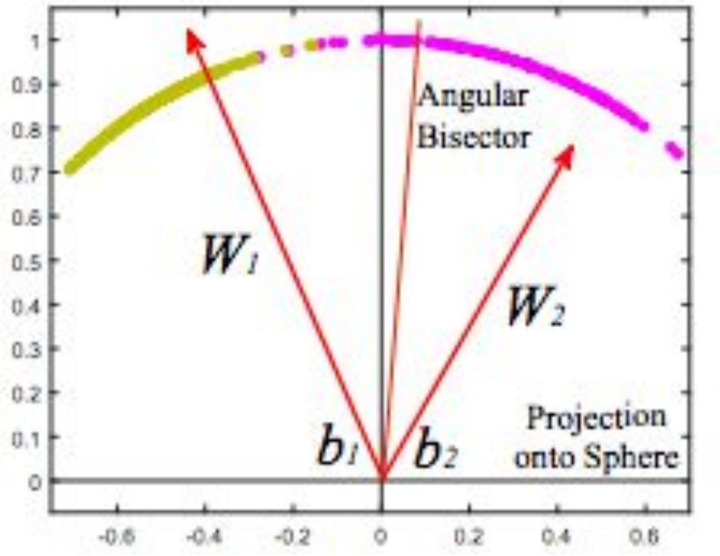


Angular Softmax

On sphere
Angle discriminates







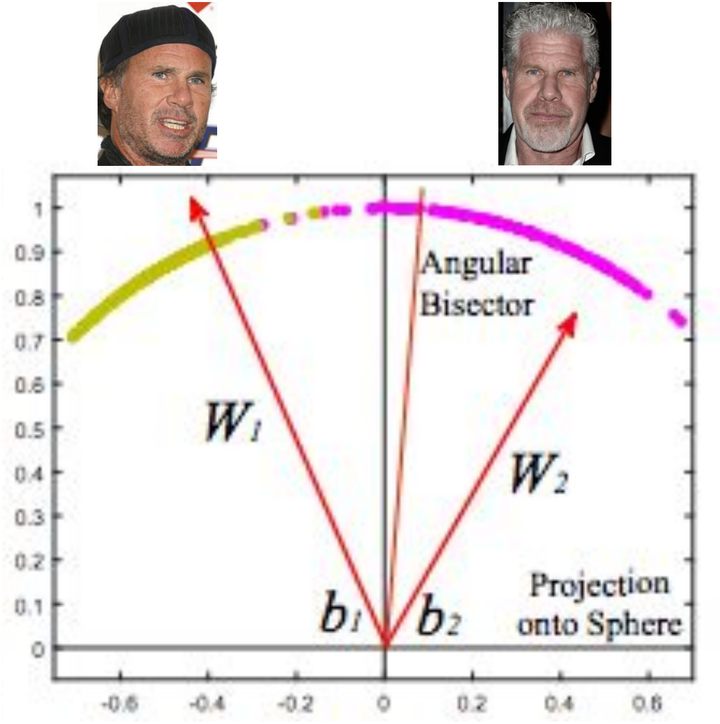
|| X||
= 1

|| W||
= 1
b=0

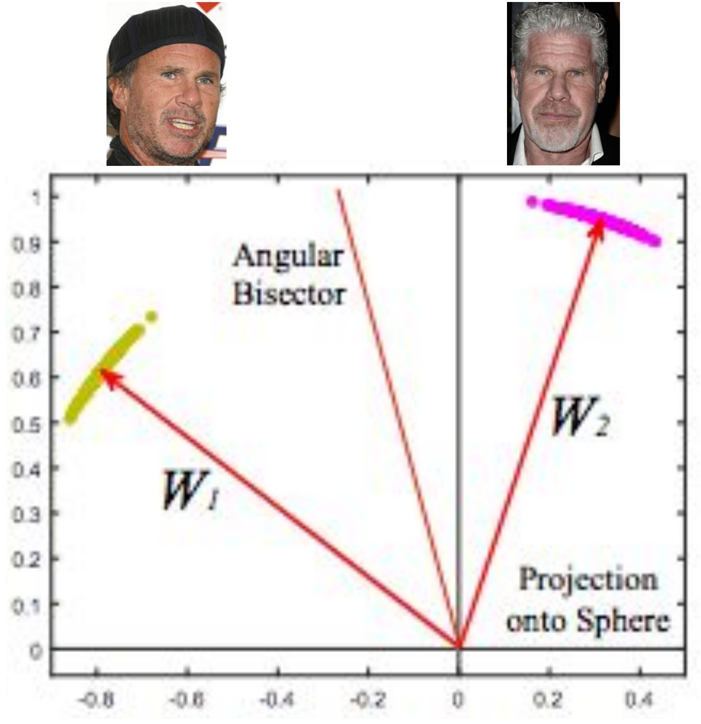
$$(\boldsymbol{W}_1 - \boldsymbol{W}_2)\boldsymbol{x} + b_1 - b_2 = 0$$

$$\|\boldsymbol{x}\|(\cos(\theta_1) - \cos(\theta_2)) = 0$$

Angular Softmax



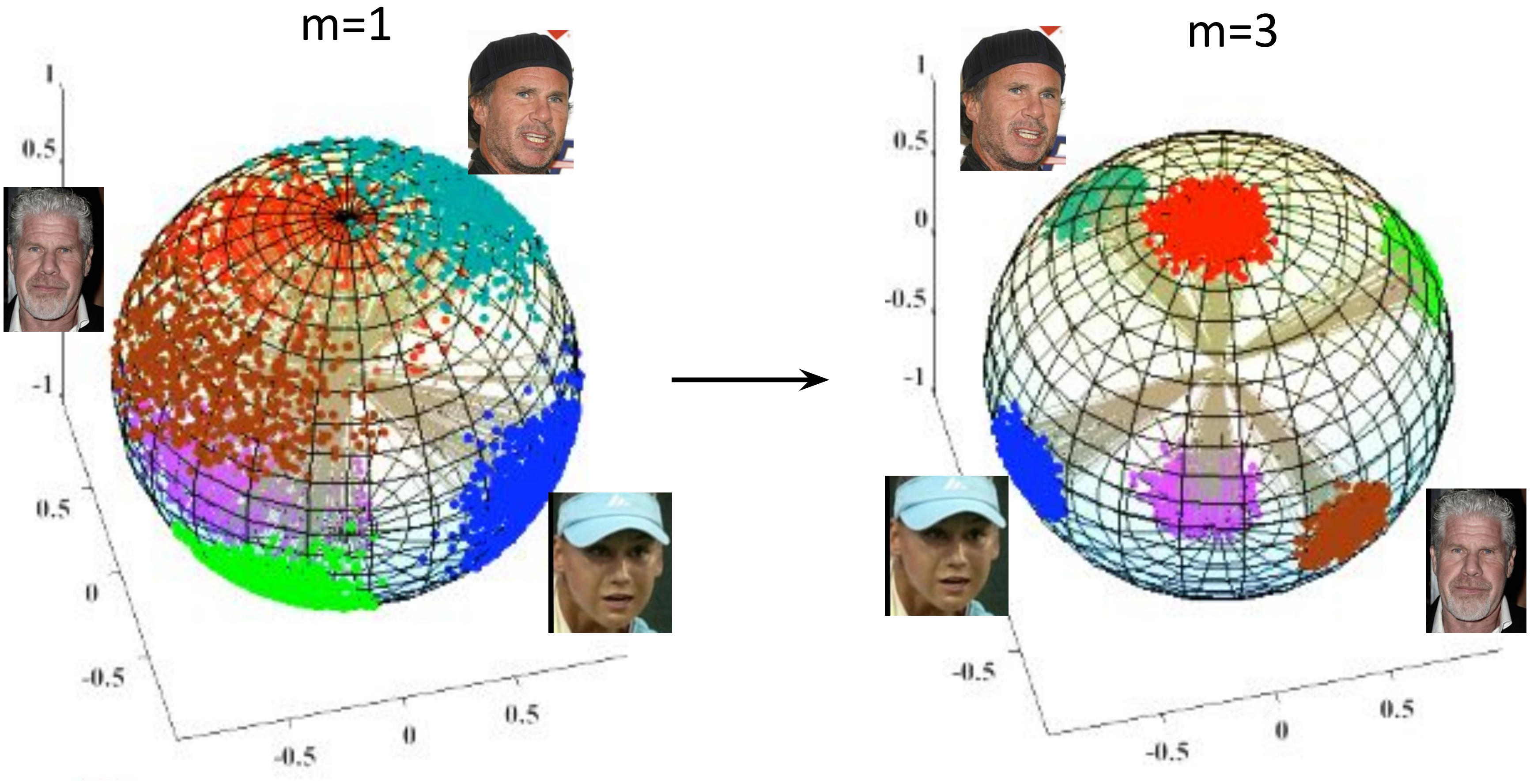
Enforce larger angle



$$\|\boldsymbol{x}\|(\cos(\theta_1)-\cos(\theta_2))=0$$

$$\|\boldsymbol{x}\|(\cos(m\theta_1)-\cos(\theta_2))=0$$

Angular Softmax: different «m»



Angular softmax: summary

Overview

- Works only on small datasets
- Slight modification of the loss yields **74.2**%
- Various modification of the loss function

	LFW, %	Megaface
Center Loss	99.6	73
A-Softmax (Torch)	99.68	**74.2**

CosineFace

$$s(\cos\theta_1 - m - \cos\theta_2) = 0$$

ArcFace

$$s(\cos(\underline{\theta_1 + m}) - \cos\theta_2) = 0$$

Metric learning: summary

Softmax < Triplet < Center < A-Softmax

Center loss

A-Softmax

- With bells and whistles better than center loss

Overall

- Rule of thumb: use **Center loss**
- Metric learning may improve classification performance

Fighting Errors

Errors after MSCeleb: children

Problem

Children all look alike

Consequence

Average embedding ~ single point in the space

person11

person12

Errors after MSCeleb: asian

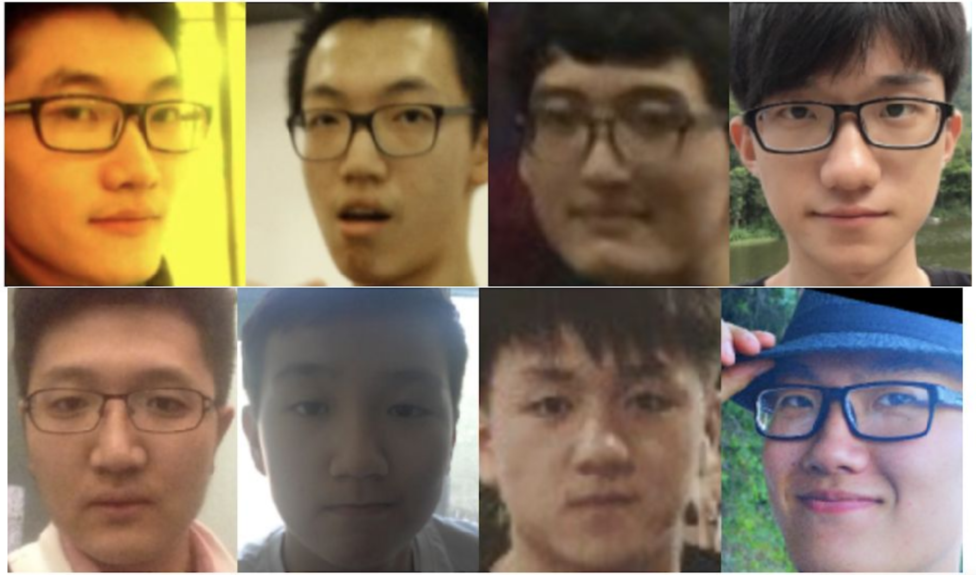
Problem

Face Recognition's intolerant to **Asians**

Reason

Dataset doesn't contain enough photos of these categories

person2

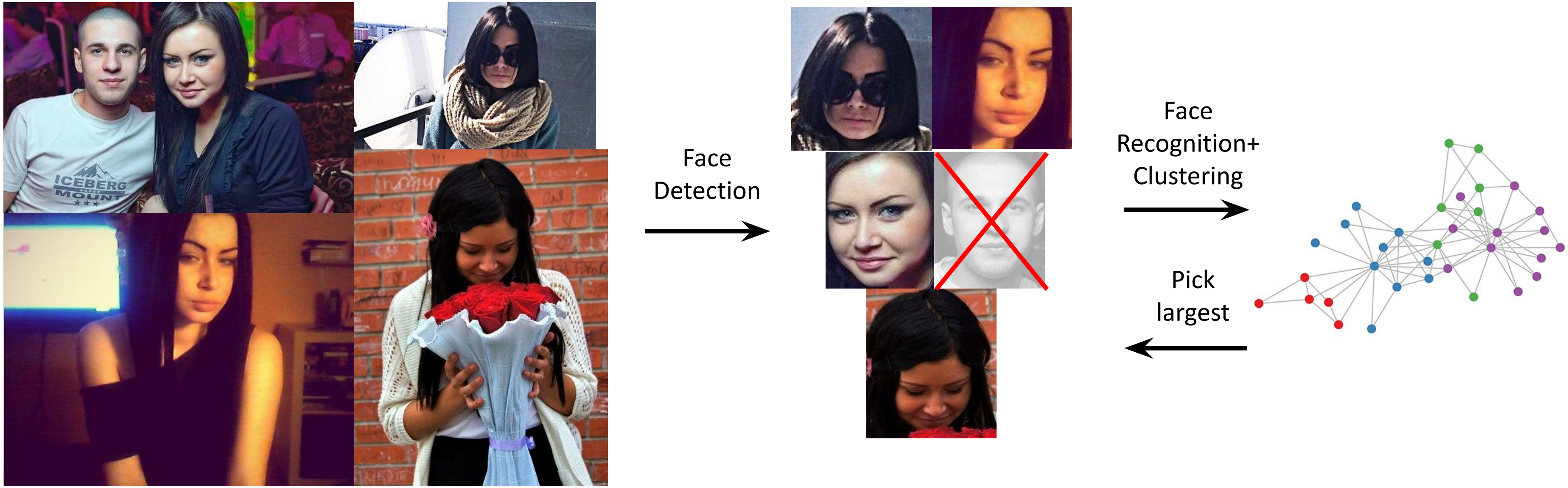


How to fix these errors ?

It’s all about data, we need diverse **dataset**!

Natural choice – avatars of social networks

A way to construct dataset



Cleaning algorithm

Iterate after each model improvement

MSCeleb dataset's errors

MSCeleb is constructed by leveraging search engines

Joe Eszterhas

=

Mel Gibson

Joe Eszterhas and Mel Gibson public confrontation leads to the error

MSCeleb dataset’s errors

Female
+
Male

MSCeleb dataset’s errors

Asia
Mix

MSCeleb dataset’s errors

Dataset has been shrinked from **100k to 46k** celebrities

Random search engine

Corrected

Results on new datasets

Datasets

- Train:
  - MSCeleb (46k)
  - VK-train (200k)
- Test
  - MegaVK
  - Sets for children and asians

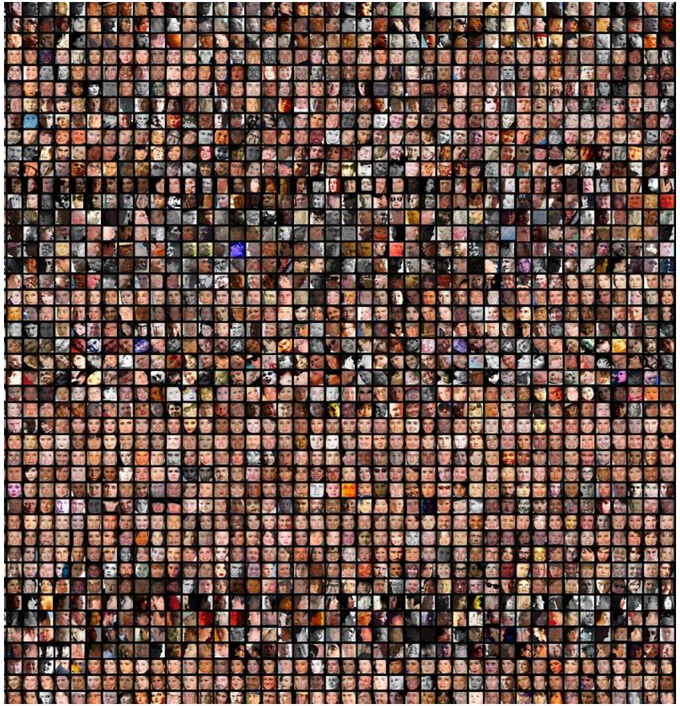
A-Softmax on dataset	Megaface	MegaVK
MSCeleb	74.2	58.4
MSCeleb cleaned	**81.1**	60
+ ArcFace	**83**	**62.5**
+ VK	79	**90**

How to handle big dataset

It seems we can add more data infinitely, but no.

Problems

- Memory consumption (Softmax)
- Computational costs
- A lot of noise in gradients



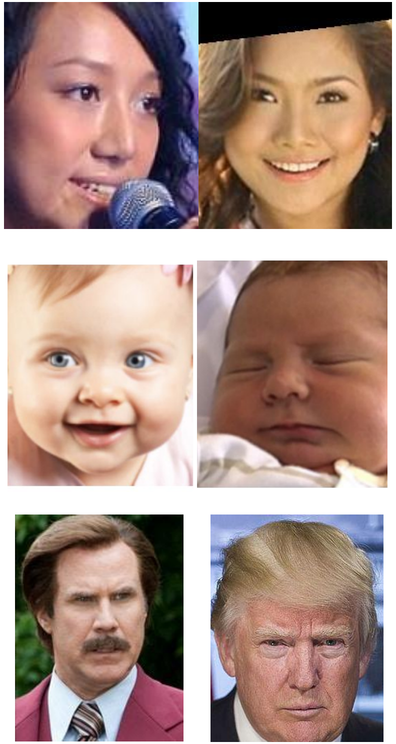
Softmax Approximation

Algorithm

1. Perform K-Means clustering using current FR model

Dataset

K-Means



Women

Children

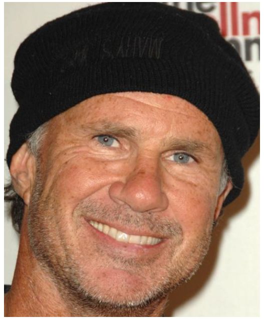
Men

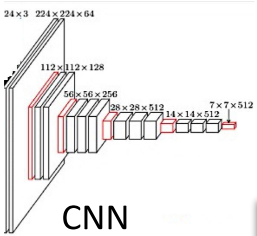
Smaller sets

Softmax Approximation

Algorithm

1. Perform K-Means clustering using current FR model
2. Two Softmax heads:
   1. Predicts cluster label
   2. Class within the true cluster





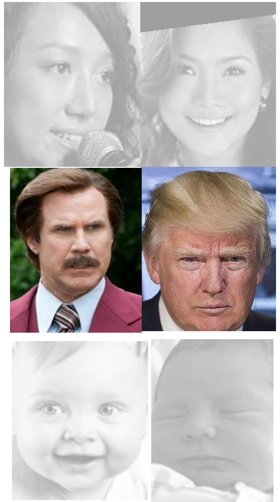
CNN

Embedding

128 floats

Predict cluster

Cluster Softmax



Men

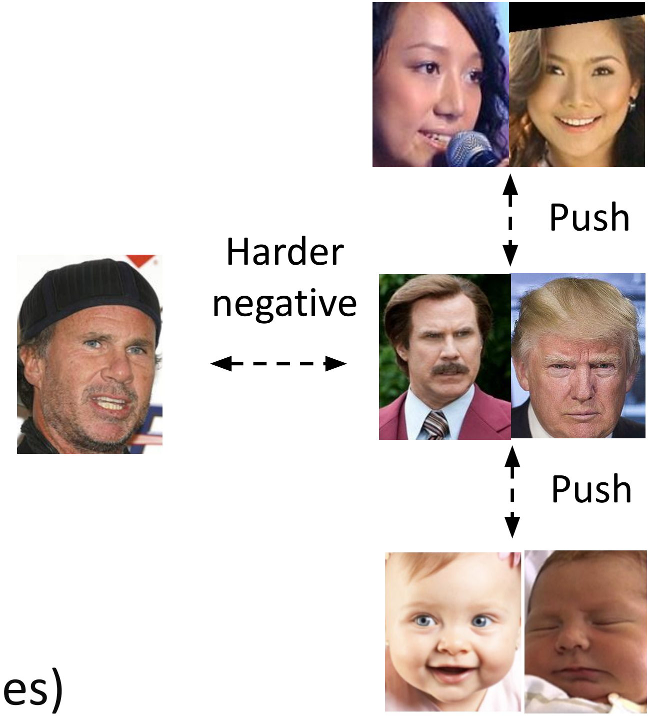
Predict person

Person Softmax

Softmax Approximation

Pros

1. Prevents fusing of the clusters
2. Does hard-negative mining
3. Clusters can be specified
   - Children
   - Asian



Results

– Doesn't improve accuracy
– Decreases memory consumption (**K** times)

Fighting errors on production

Errors: blur

Problem

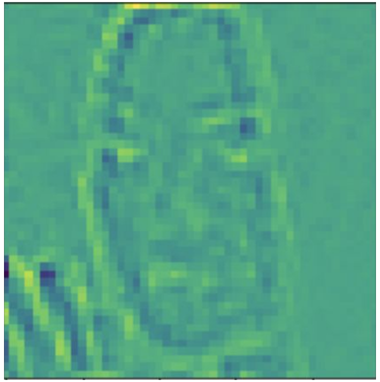
- Detector yields blurry photos
- Recognition forms «blurry clusters»

Solution

Laplacian – 2^{nd} order derivative of the image

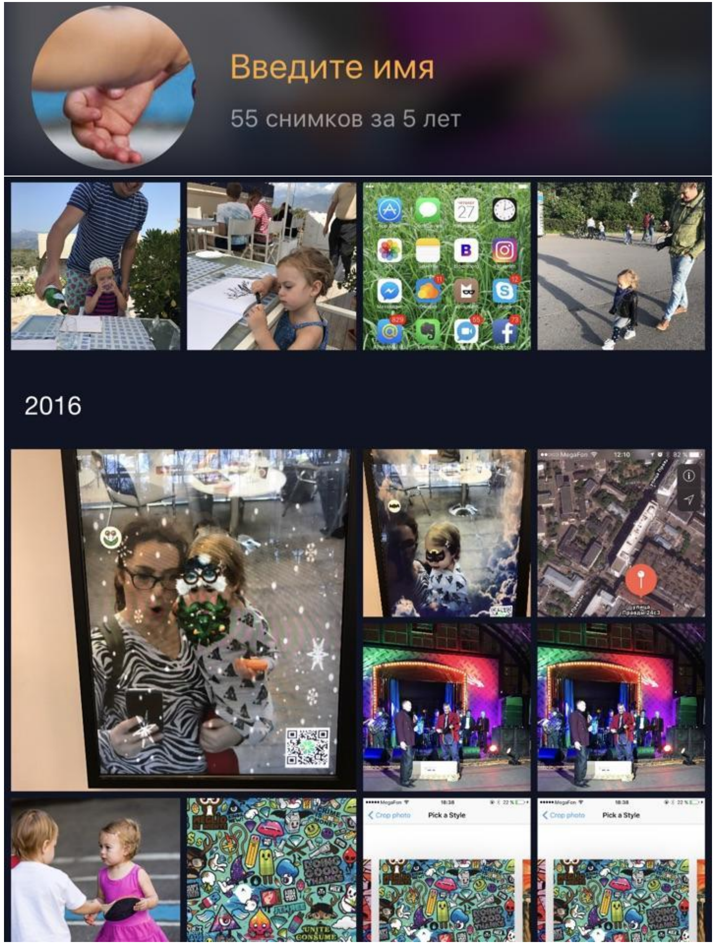
Laplacian in action

Low
variance

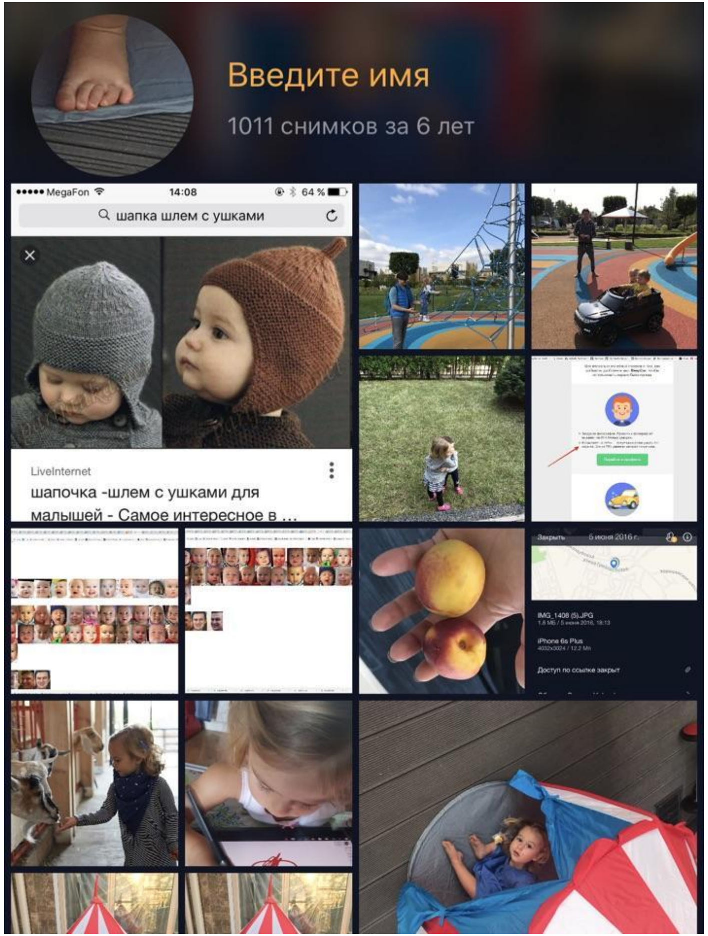


High
variance

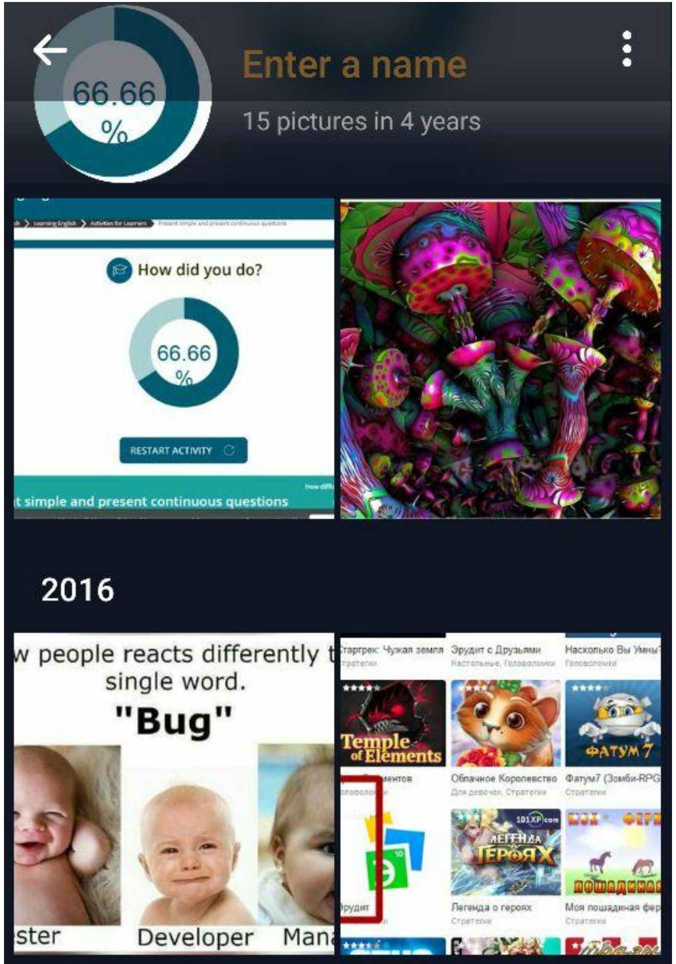
Errors: body parts

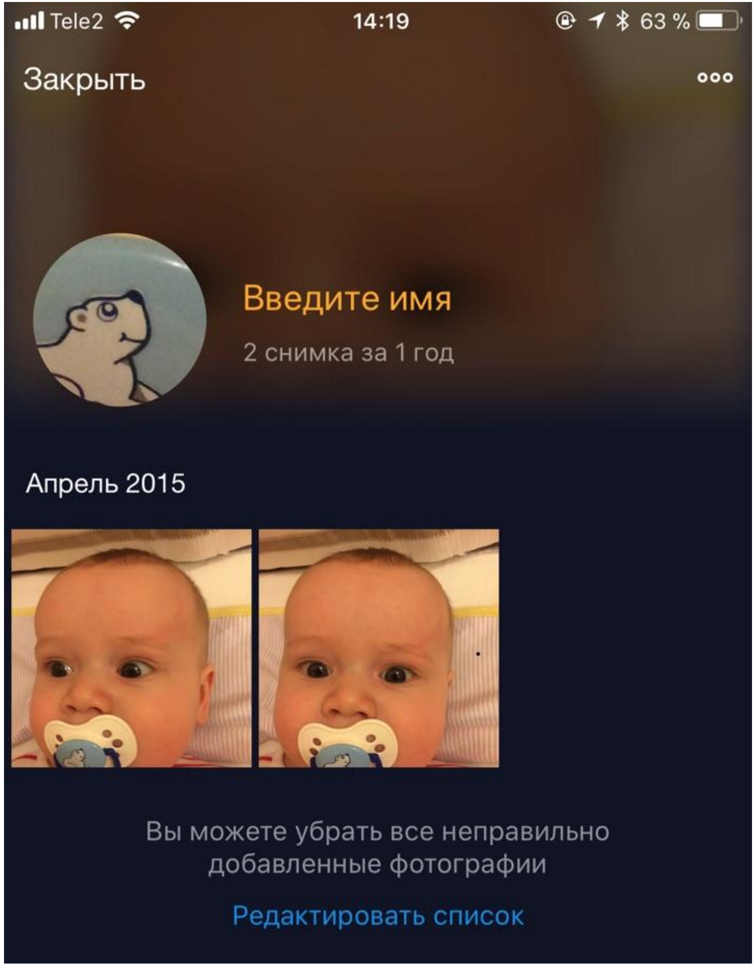


Detection mistakes form clusters



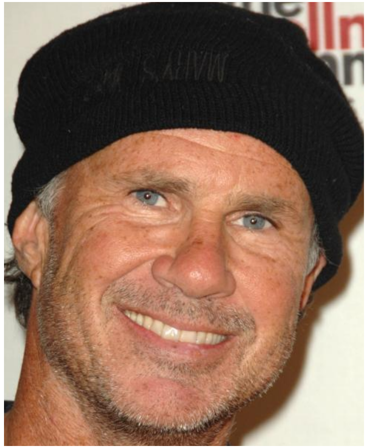
Errors: diagrams & mushrooms

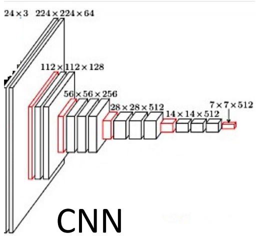




Fixing trash clusters

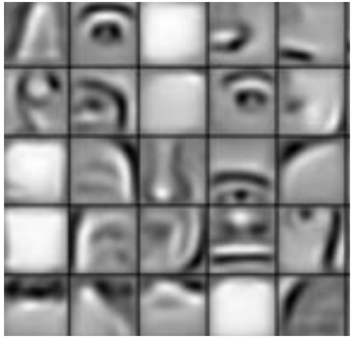
There is similarity between “no faces”!





CNN

Embedding

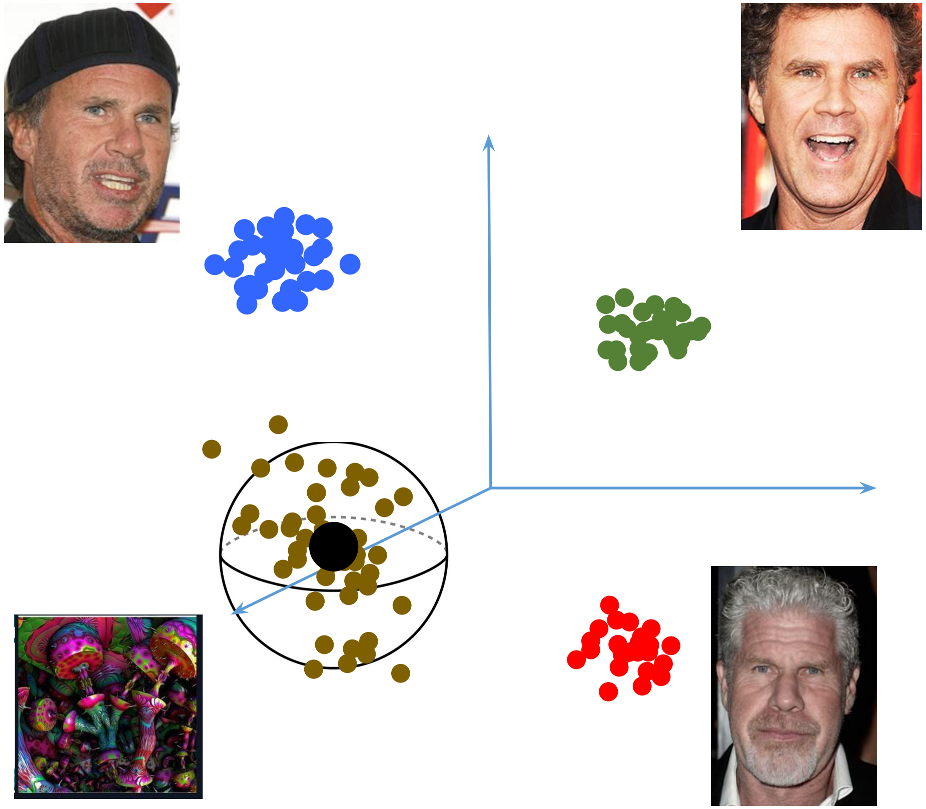


Specific activations

Workaround

Algorithm

1. Construct «trash» dataset
2. Compute average embedding
3. Every point inside the sphere – trash



Results

– ROC AUC 97%

Spectacular results

Fun: new governors

Recently appointed governors are almost twins, but FR distinguishes them

Dmitriy

Gleb

Over years

Face recognition algorithm captures similarity across years

Although we didn’t focus on the problem

Over years


ОВД РАЙОНА ХОРОШЕВО-МНЕВНИКИ
ГОР.МОСКВЫ
25.09.2003
772-139
45 06 602311
772-139
АРТАМОНОВА
МАРИЯ
АНАТОЛЬЕВНА
ЖЕН.
17.03.1942
45 06


VISADO
ESTADOS SCHENGEN
28-04-16
23-10-16
C
MULT
MOSCU
08-04-16
650511330
ARTAMONOVA, MARIA
VIS ESTANCIA
RUS1P1EST

Summary

1. Use TensorRT to speed up inference
2. Metric learning: use Center loss by default
3. Clean your data thoroughly
4. Understanding CNN helps to fight errors

{ "smartmail_hack": 20.18 }

Thank you!

Eduard Tyantov

Head of Machine Learning group at Mail.Ru

@mail.ru

Auxiliary

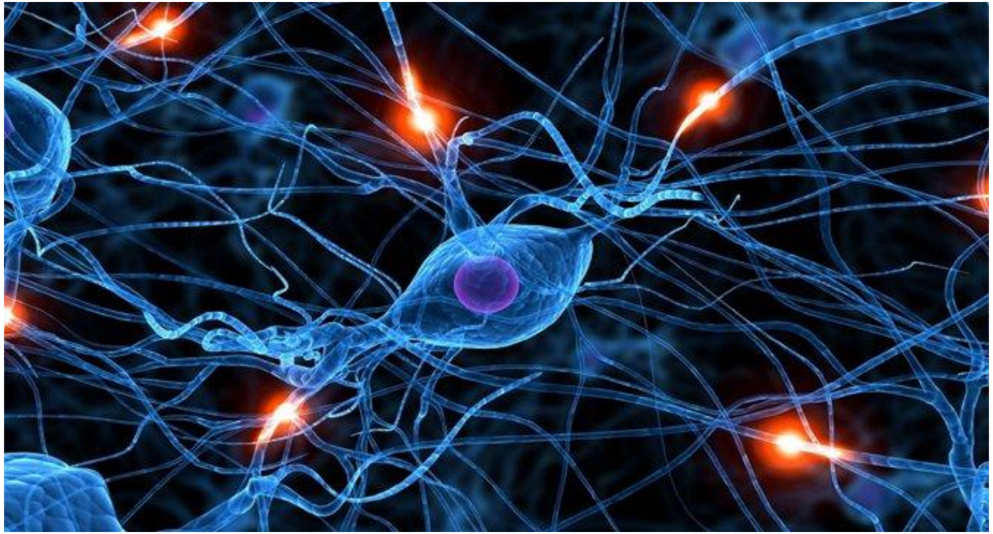
Best avatar

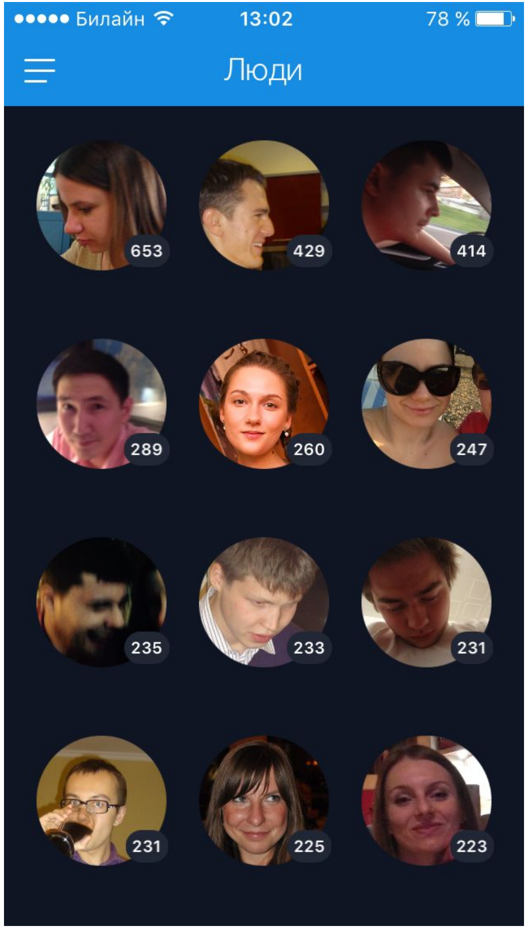
Problem

How to pick an avatar for a person ?

Solution

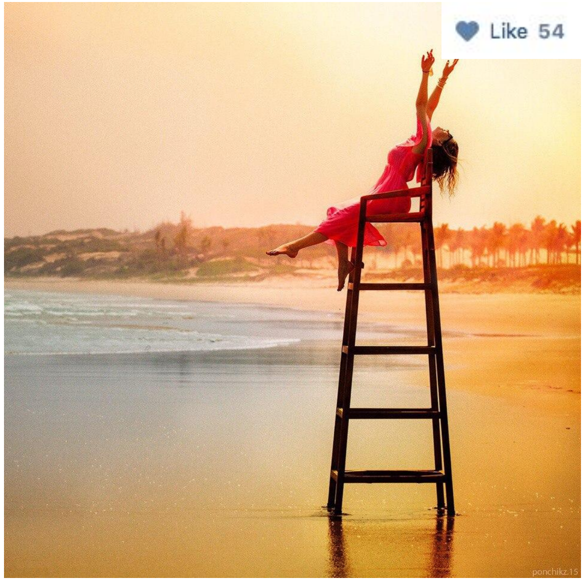
Train model to predict awesomeness of photo

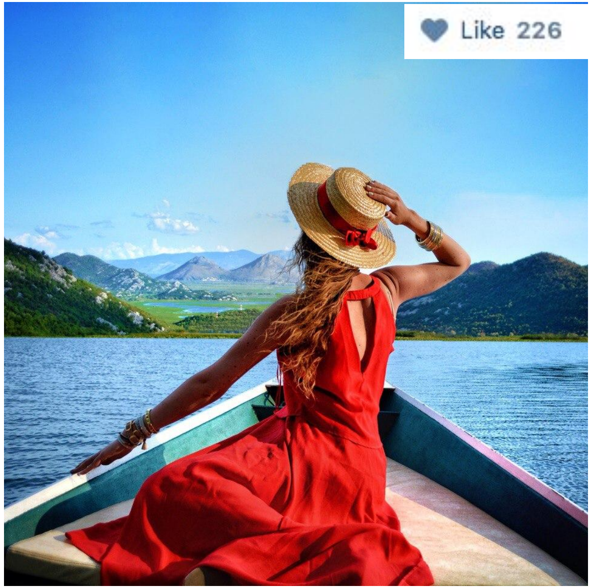




Predicting awesomeness: how to approach

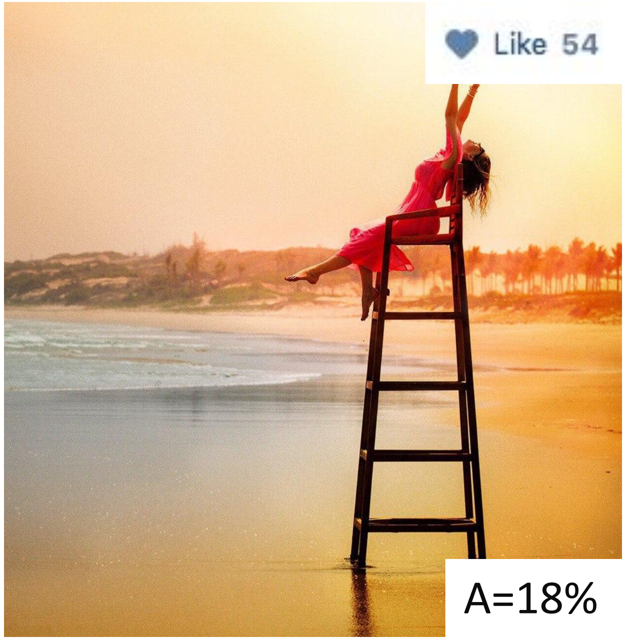
Social networks – not only photos, but likes too





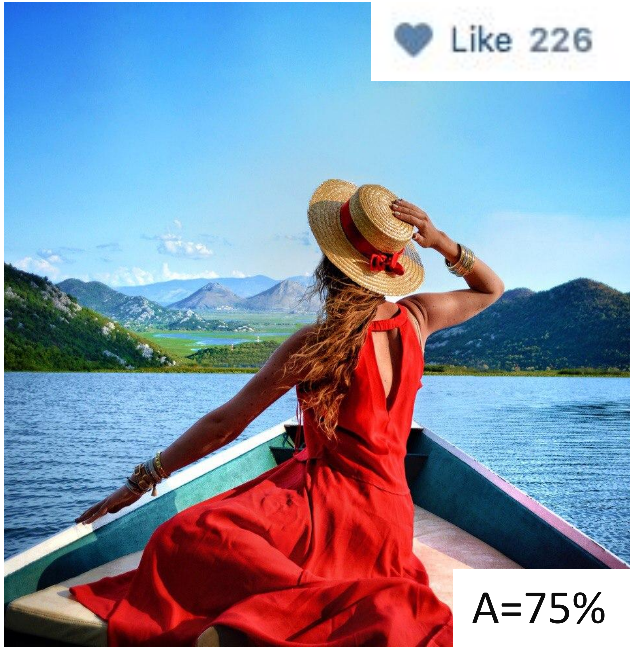
Predicting awesomeness: dataset

Awesomeness (A) = likes/audience



A=18%

A=27%



A=75%

Predicting awesomeness: summary

Results

- Mean Aveage Precision @5: 25%
- Data and metric are noisy => human evaluation

High score

Low score

Predicting awesomeness: incorporating into FR

One more branch in Face Recognition CNN

Small overhead