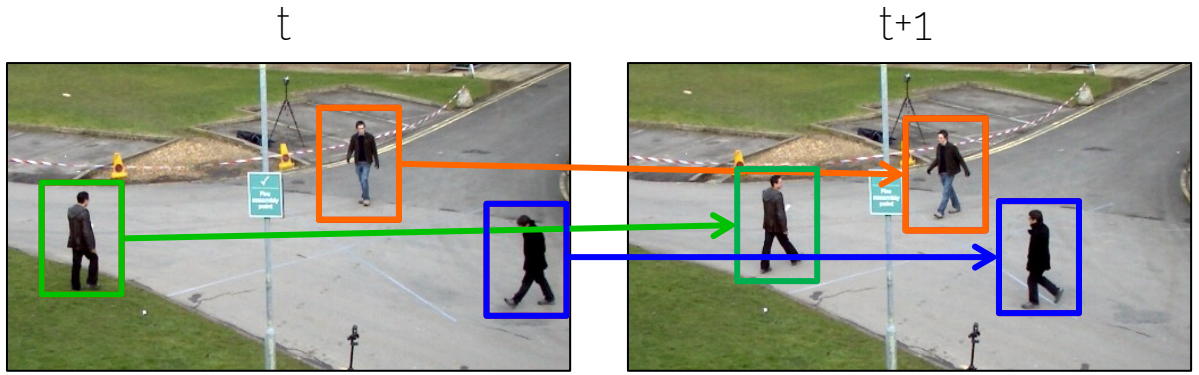
Object tracking

Problem statement

- Given a video, find out which parts of the image depict the same object in different frames
- Often we use detectors as starting points



Why do we need tracking?

- To model objects when detection fails:
  - Occlusions
  - Viewpoint/pose/blur/illumination variations (in a few frames of a sequence)
  - Background clutter
- To reason about the dynamic world, e.g., trajectory prediction (is the person going to cross the street?)

Tracking is….

- Similarity measurement
- Correlation
- Correspondence
  - Story time: „A young graduate student asked Takeo Kanade what are the three most important problems in computer vision. Kanade replied: “Correspondence, correspondence, correspondence!”
- Matching/retrieval
- Data association

Tracking is also….

- Learning to model our target's

- Appearance: we need to know how the target looks like
  - Single object tracking
  - Re-identification

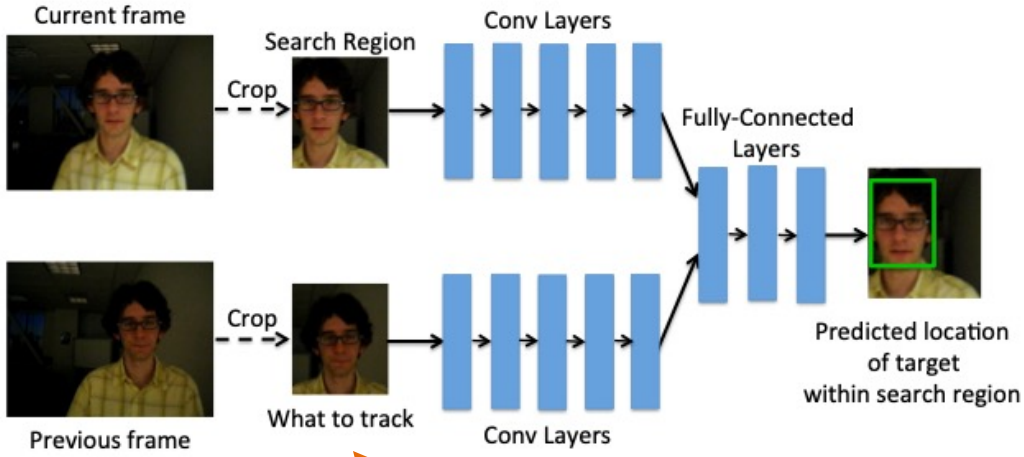
- Motion: to make predictions of where the targets goes
  - Trajectory prediction (lecture 6)

Single Target Tracking

- STT (1) as a matching/correspondence problem:
  - GOTURN: no online appearance modeling
- STT (2) as an appearance learning problem:
  - MDNet: quick online finetuning of the network
- STT (3) as a (temporal) prediction problem:
  - ROLO = CNN + LSTM

Single Target Tracking 1

- Input: what to track?



Crop the object to be tracked – Initialization of our tracker

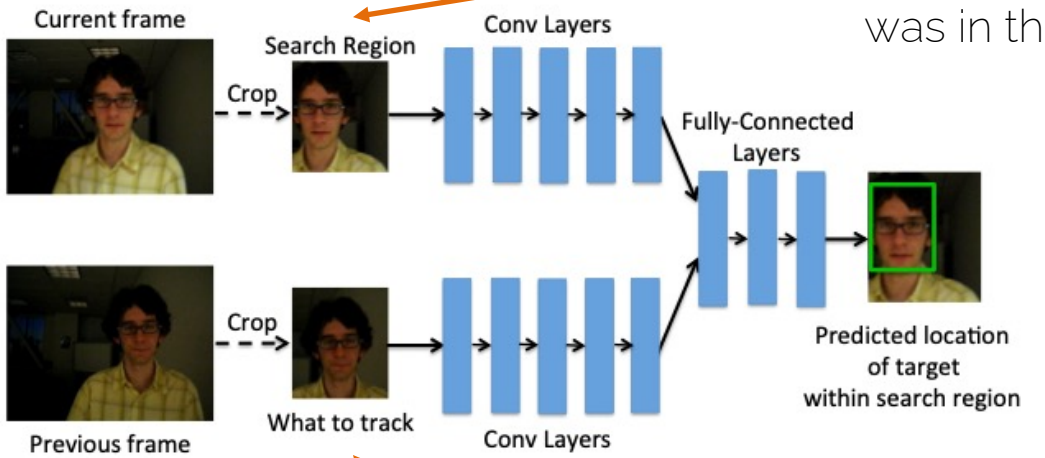
D. Held, S. Thrun, S. Savarese. "Learning to Track at 100 FPS with Deep Regression Networks". ECCV 2016.

Single Target Tracking 1

- Where do I search for the object?

Assume smooth "slow" motion. The object cannot be far away from where it was in the previous frame.

Use the position of t-1 to crop frame t

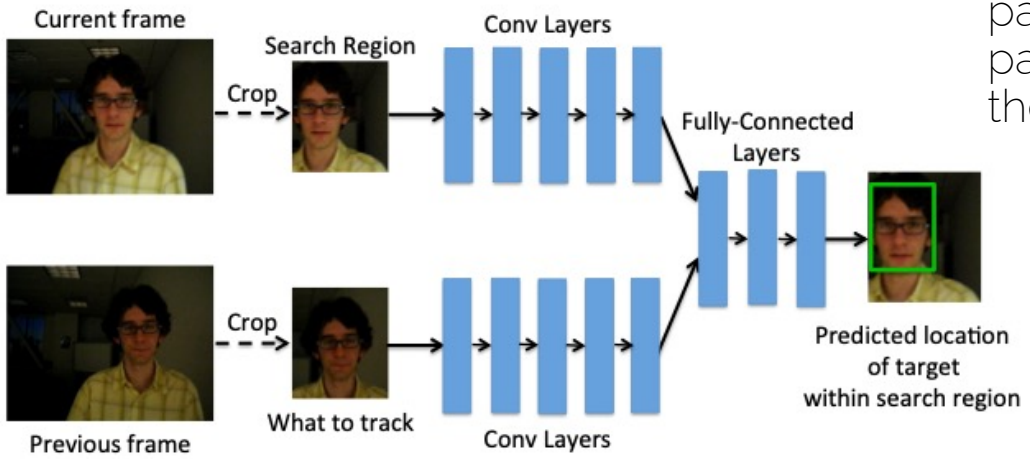


Crop the object to be tracked – Initialization of our tracker

D. Held, S. Thrun, S. Savarese. "Learning to Track at 100 FPS with Deep Regression Networks". ECCV 2016.

Single Target Tracking 1

- Architecture: conv + concatenate + FC



Check the original paper for the exact parameterization of the output

D. Held, S. Thrun, S. Savarese. "Learning to Track at 100 FPS with Deep Regression Networks". ECCV 2016.

Single Target Tracking 1

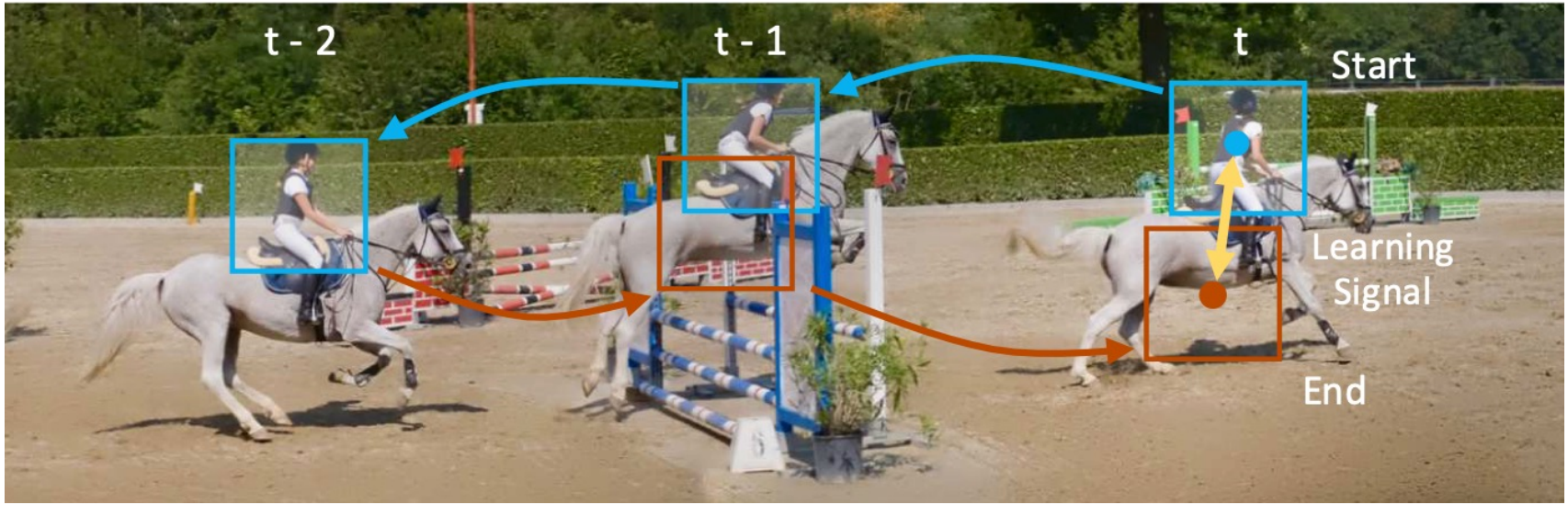
- PROS of GOTURN:
  - No online training required.
  - Tracking is done by comparison, so we do not need to retrain or finetune our model for every new object.
  - Close to the template matching approach that we saw in the first lectures for object detection
  - This makes it very fast!

- CONS:
  - We have a motion assumption. If the object moves fast and goes out of our search window, we cannot recover.

D. Held, S. Thrun, S. Savarese. "Learning to Track at 100 FPS with Deep Regression Networks". ECCV 2016.

SOT 1.2 - Unsupervised

- Forward cycle and backward cycle should be consistent!



X. Wang, A. Jabri, A. Efros. "Learning correspondence from the cycle-consistency of time". CVPR 2019

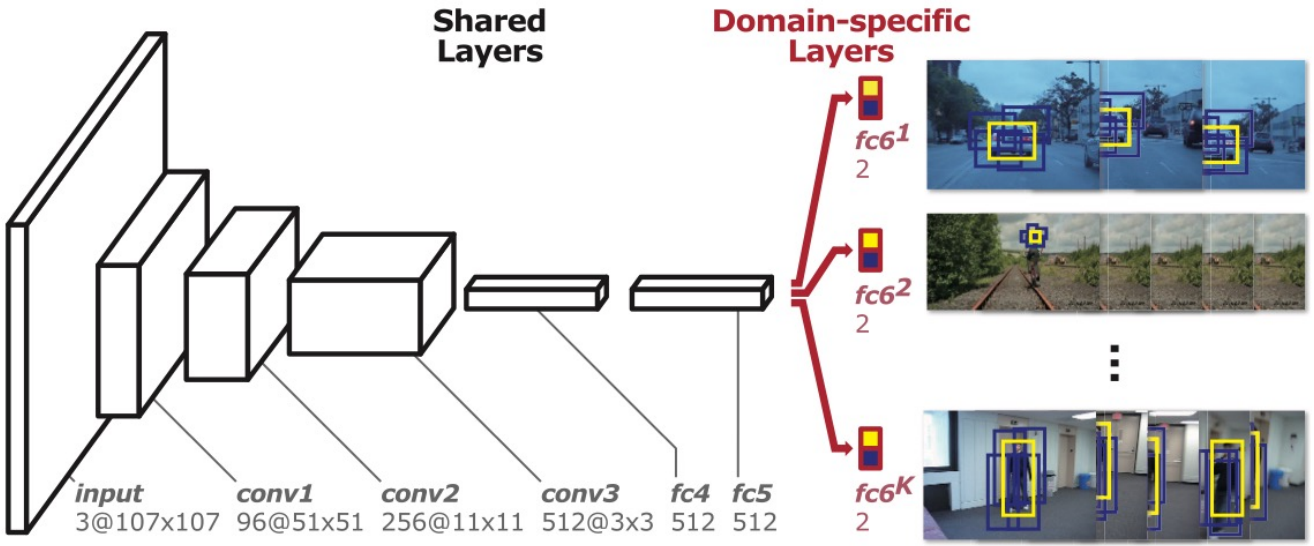
Single Target Tracking 2

- Online appearance model learning entails training your CNN at test time.
  - Slow: not suitable for real-time applications
  - Solution: train as few layers as possible

H. Nam and B. Han. „Learning Multi-Domain Convolutional Neural Networks for Visual Tracking". CVPR 2016

Single Target Tracking 2

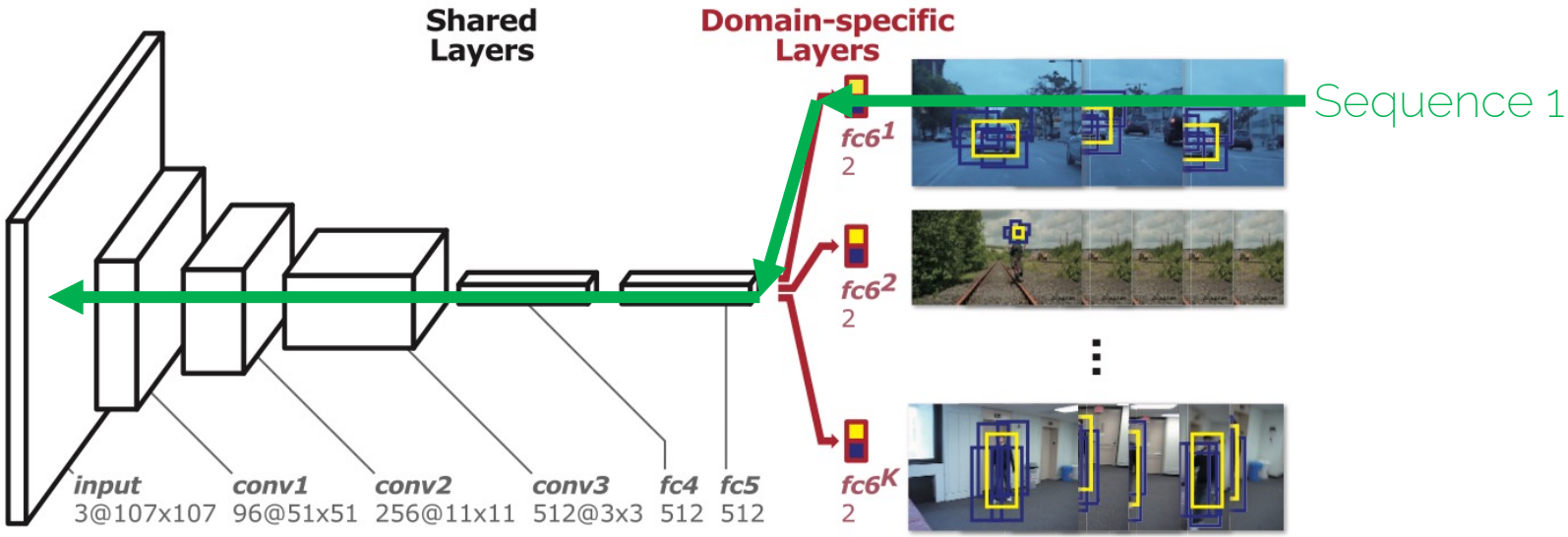
- Shared layers + scene-specific layers



H. Nam and B. Han. „Learning Multi-Domain Convolutional Neural Networks for Visual Tracking". CVPR 2016

Single Target Tracking 2

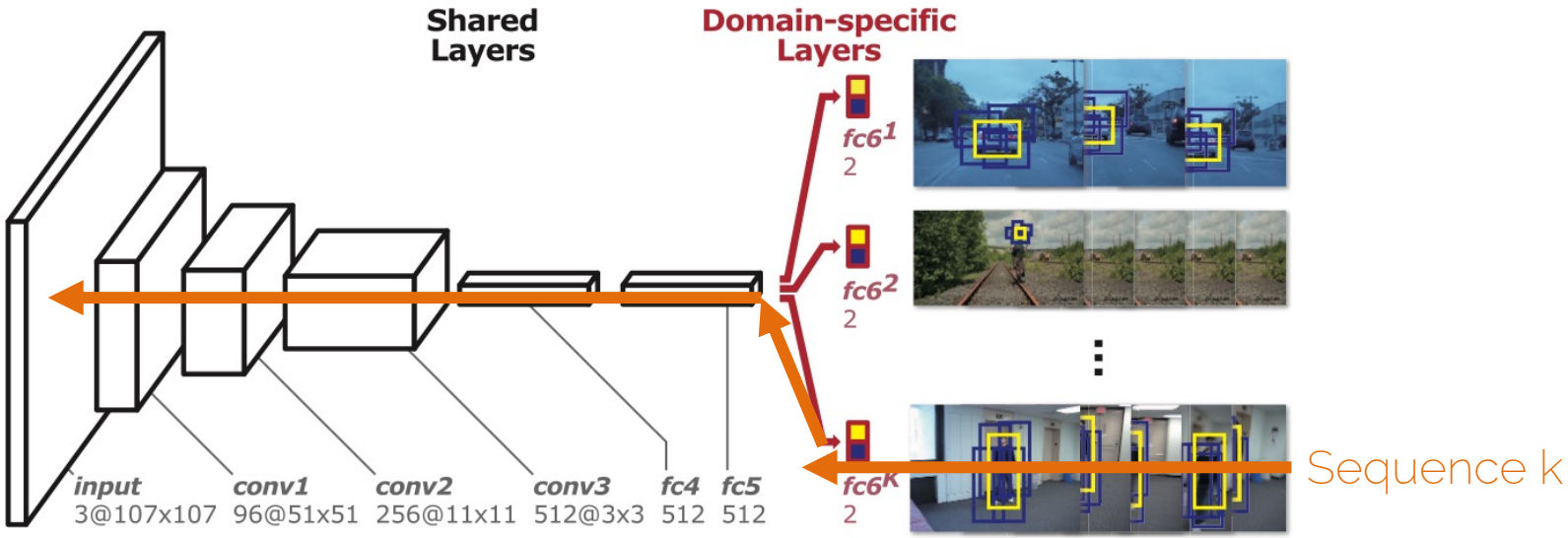
- Backpropagation is independent per sequence



H. Nam and B. Han. „Learning Multi-Domain Convolutional Neural Networks for Visual Tracking". CVPR 2016

Single Target Tracking 2

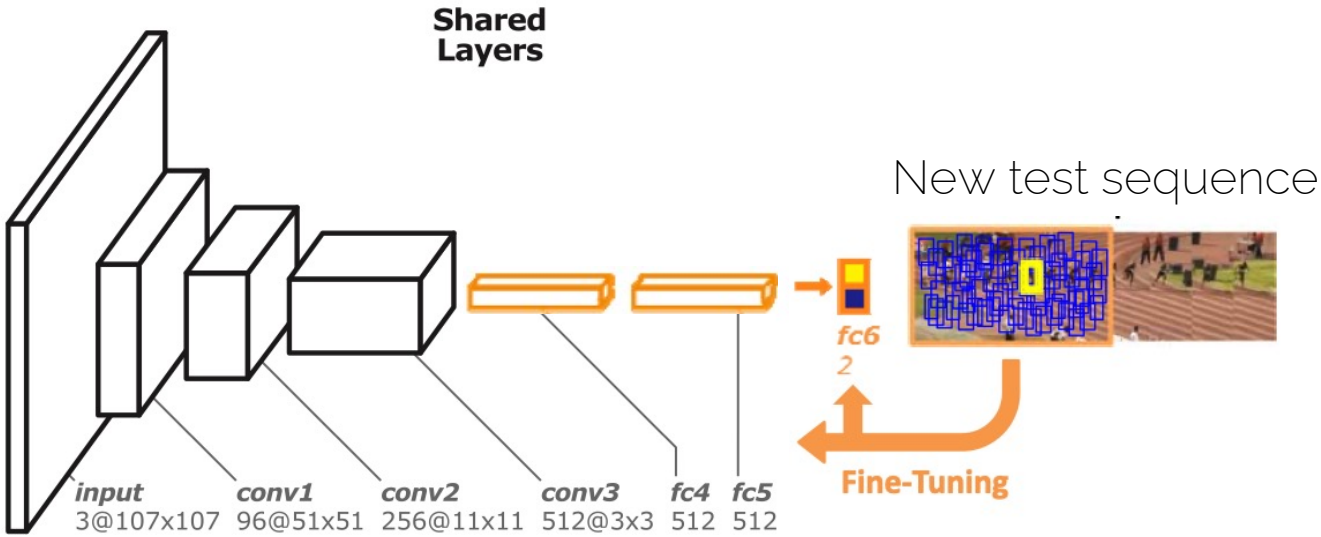
- Backpropagation is independent per sequence



H. Nam and B. Han. „Learning Multi-Domain Convolutional Neural Networks for Visual Tracking". CVPR 2016

Single Target Tracking 2

- At test time, we need to train fc6 (up to fc4 if wanted).

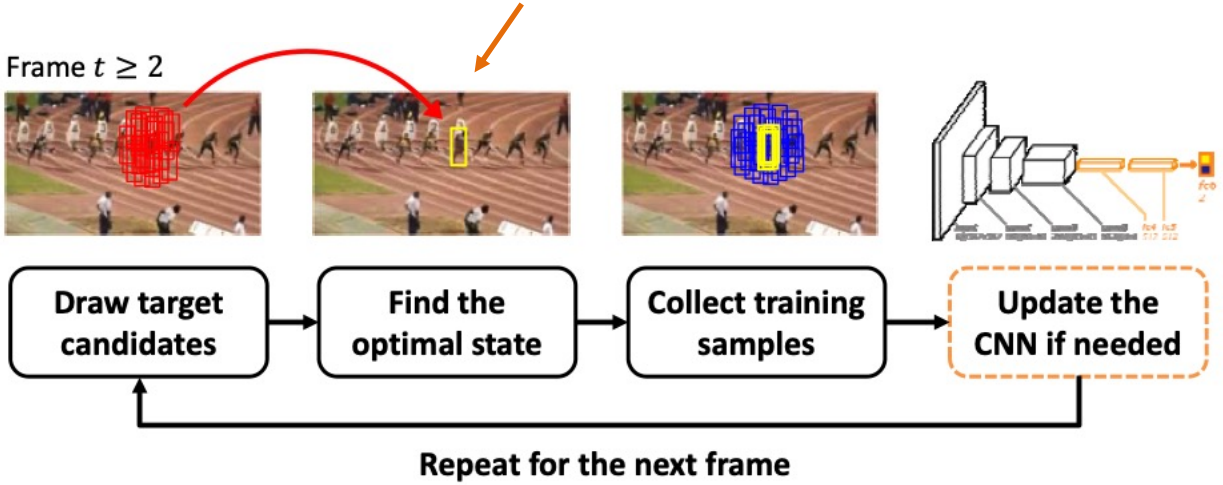


H. Nam and B. Han. „Learning Multi-Domain Convolutional Neural Networks for Visual Tracking". CVPR 2016

Single Target Tracking 2

- Online tracking

R-CNN type of regression



H. Nam and B. Han. „Learning Multi-Domain Convolutional Neural Networks for Visual Tracking". CVPR 2016

Single Target Tracking 2

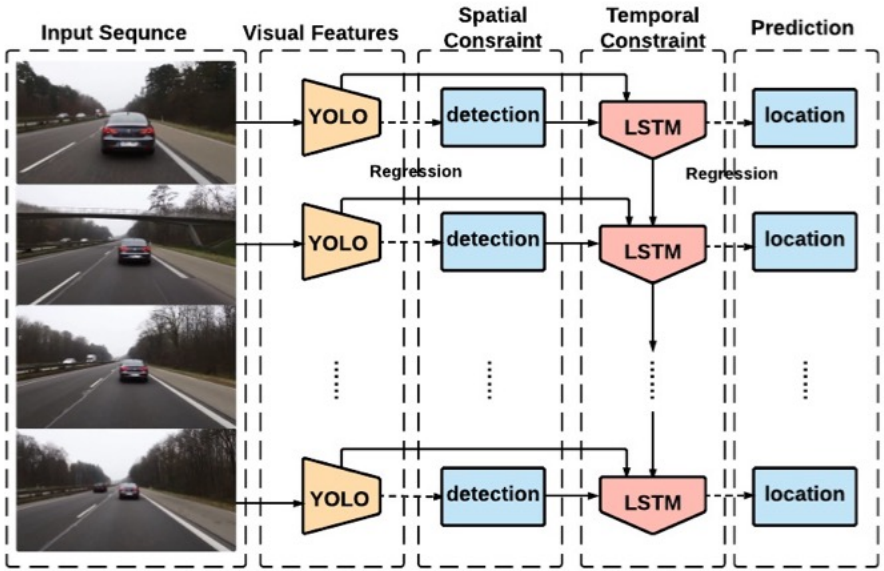
- PROS of MDNet:
  - No previous location assumption, the object can move anywhere in the image
  - Fine-tuning step is comparatively cheap
  - Winner of the VOT Challenge 2015 (http://www.votchallenge.net)

- CONS:
  - Not as fast as GOTURN

H. Nam and B. Han. „Learning Multi-Domain Convolutional Neural Networks for Visual Tracking". CVPR 2016

Single Target Tracking 3

- CNN for appearance + LSTM for motion



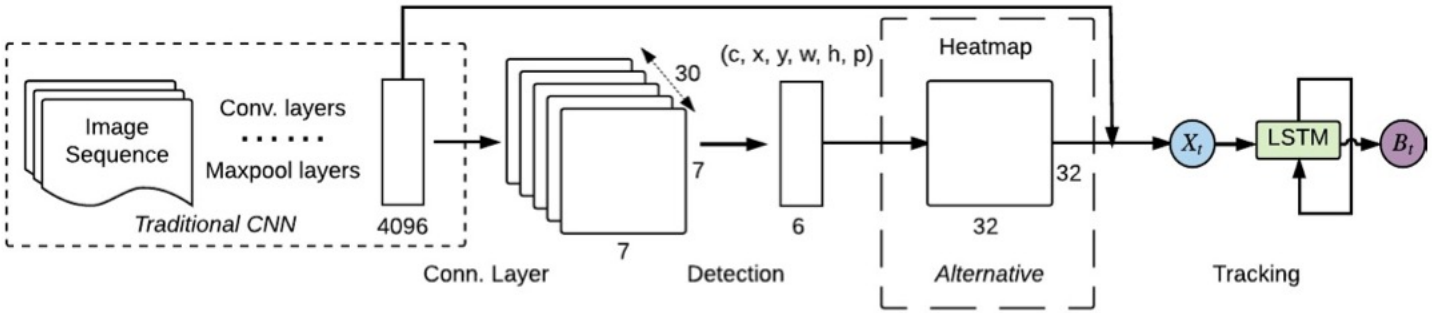
Recurrent YOLO

ROLO

G. Ning, Z. Zhang, C. Huang, Z. He. „Spatially Supervised Recurrent Convolutional Neural Networks for Visual Object Tracking". arXiv:1607.05781. 2016

Single Target Tracking 3

- LSTM receives the heatmap for the object's position and the 4096 descriptor of the image



G. Ning, Z. Zhang, C. Huang, Z. He. „Spatially Supervised Recurrent Convolutional Neural Networks for Visual Object Tracking". arXiv:1607.05781. 2016

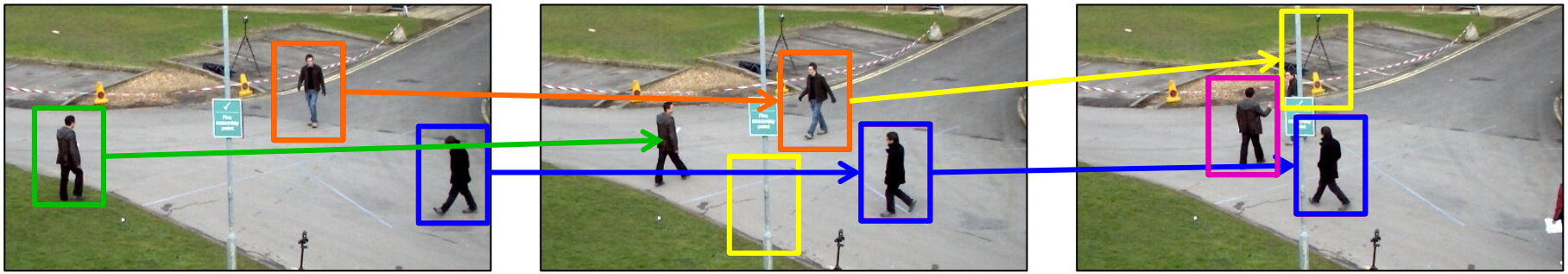
Multiple object tracking

Different challenges

- Multiple objects of the same type
- Heavy occlusions
- Appearance is often very similar

Tracking-by-detection

- We will focus on algorithms where a set of detections is provided
  - Remember detections are not perfect!



Find detections that match and form a trajectory

Online vs offline tracking

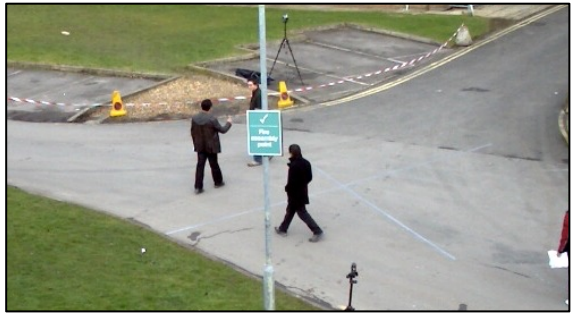
- Online tracking
  - Processes two frames at a time
  - For real-time applications
  - Prone to drifting → hard to recover from errors or occlusions
- Offline tracking
  - Processes a batch of frames
  - Good to recover from occlusions (short ones as we will see)
  - Not suitable for real-time applications
  - Suitable for video analysis

Online tracking

t

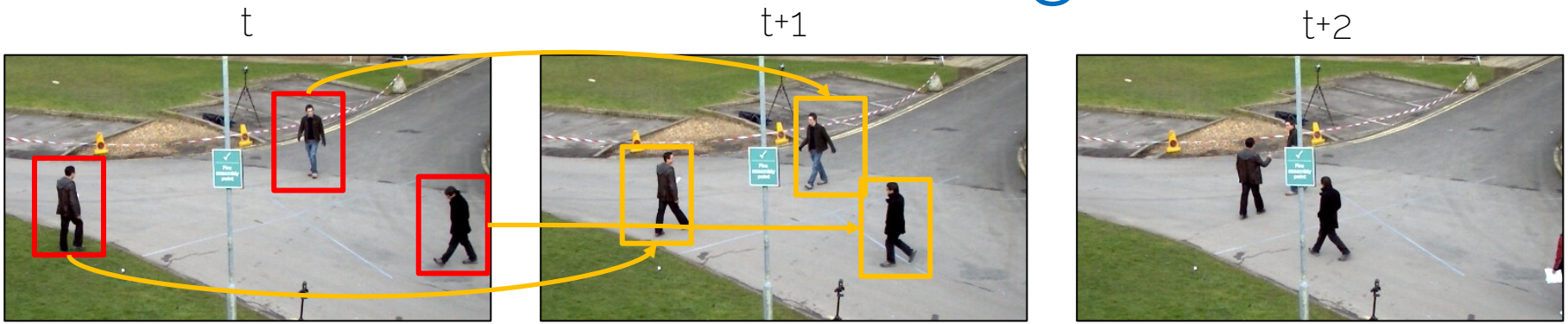
t+1

t+2



- 1. Track **initialization** (e.g. using a detector) 

Online tracking



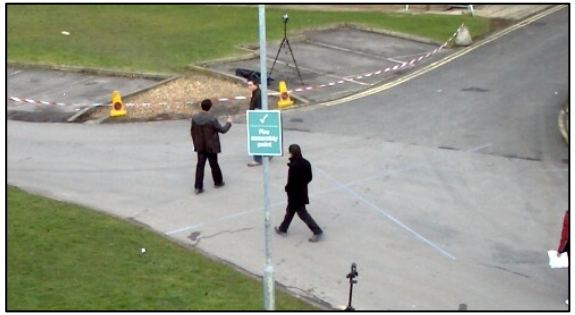
- 1. Track initialization (e.g. using a detector) 
- 2. **Prediction** of the next position (motion model) →

Online tracking

t

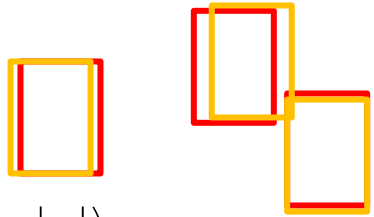
t+1

t+2



- 1. Track initialization (e.g. using a detector)

- 2. Prediction of the next position (motion model)



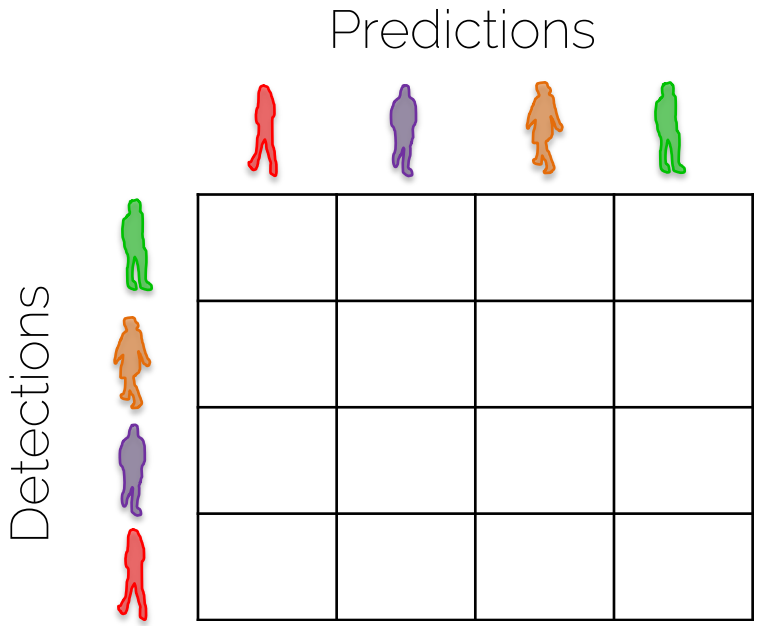
- 3. **Matching** predictions with detections (appearance model)

Online tracking

- 2. Prediction of the next position (motion model)
  - Classic: Kalman filter
  - Nowadays: Recurrent architecture
  - For now: we will assume a constant velocity model (spoiler alter: it works really well at high framerates and without occlusions!)

Online tracking

- 3. Matching predictions with detections



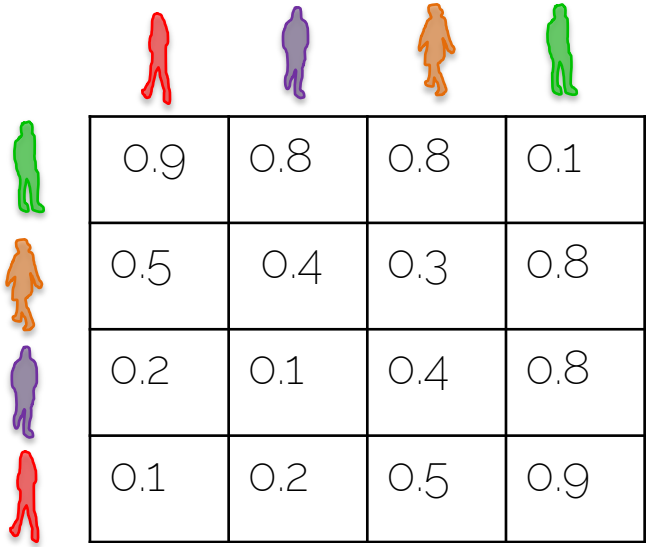
Online tracking

- Bipartite matching
  - Define distances between boxes

    (e.g., IoU, pixel distance, 3D distance)

Predictions

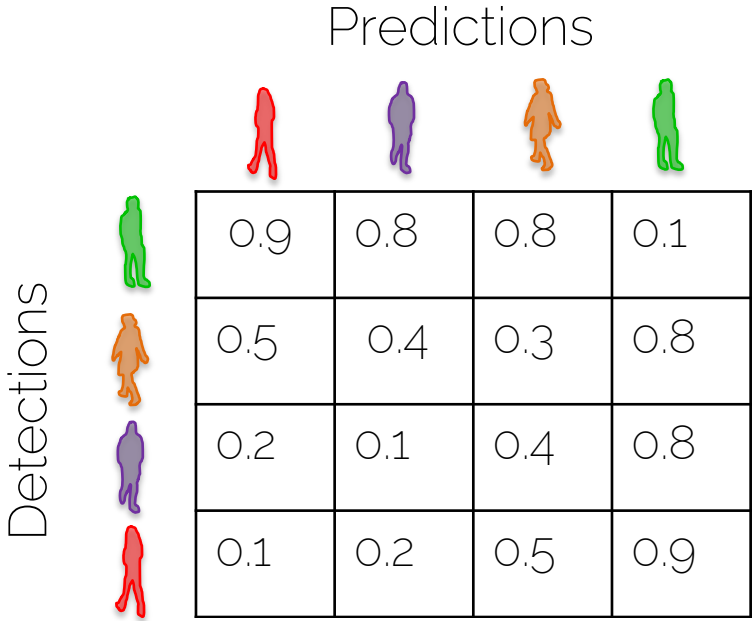
Detections



0.9	0.8	0.8	0.1
0.5	0.4	0.3	0.8
0.2	0.1	0.4	0.8
0.1	0.2	0.5	0.9

Online tracking

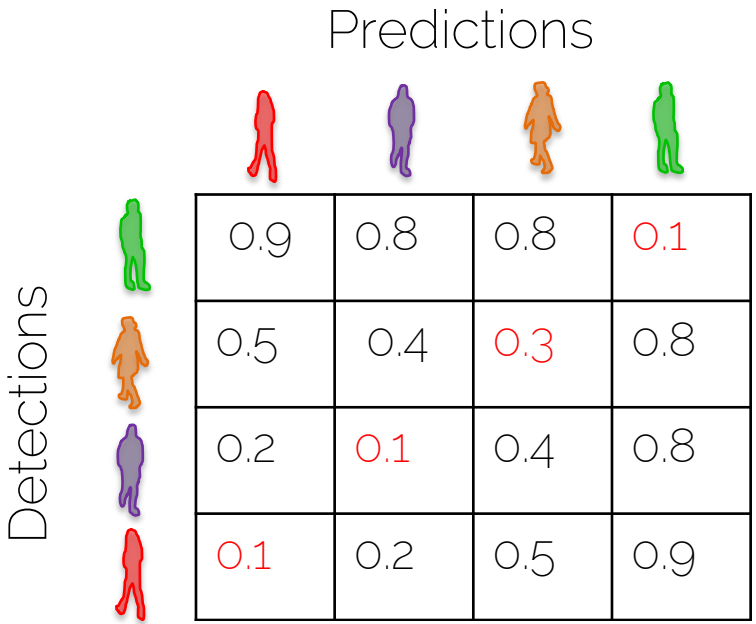
- Bipartite matching
  - Define distances between boxes
    (e.g., IoU, pixel distance, 3D distance)
  - Solve the unique matching with e.g., the Hungarian algorithm*



*Demo: http://www.hungarianalgorithm.com/solve.php

Online tracking

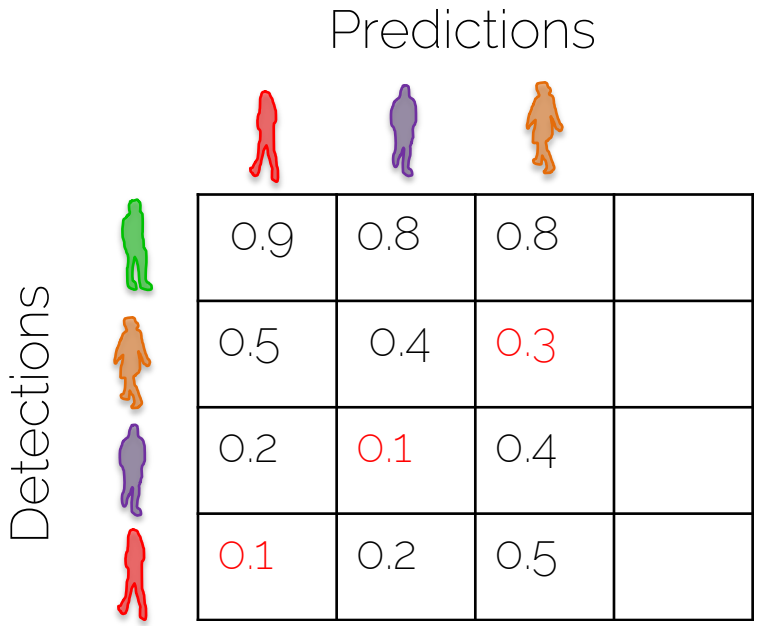
- Bipartite matching
  - Define distances between boxes
    (e.g., IoU, pixel distance, 3D distance)
  - Solve the unique matching with e.g., the Hungarian algorithm*
  - Solutions are the unique assignments that minimize the total cost



*Demo: http://www.hungarianalgorithm.com/solve.php

Online tracking

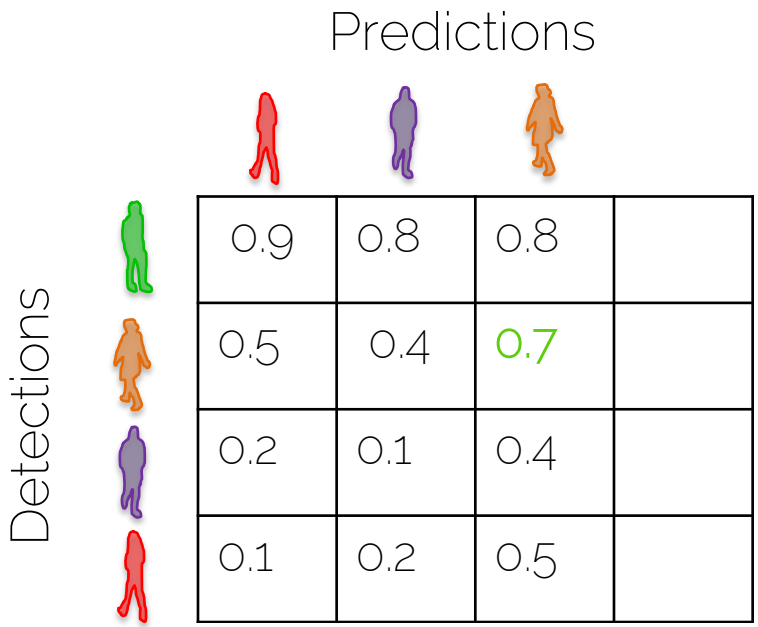
- Bipartite matching
  - What happens if we are missing a prediction?



*Demo: http://www.hungarianalgorithm.com/solve.php

Online tracking

- Bipartite matching
  - What happens if we are missing a prediction?
  - What happens if no prediction is suitable for the match?



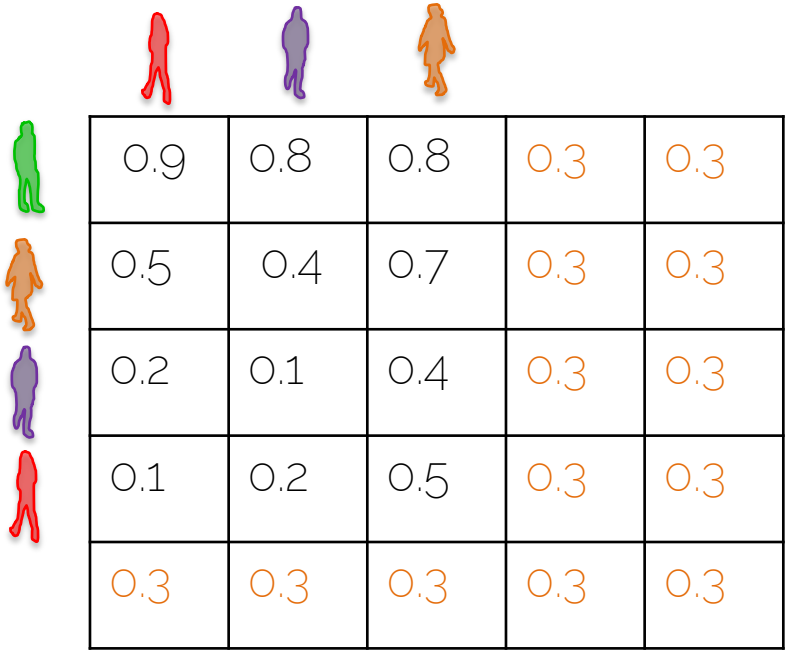
*Demo: http://www.hungarianalgorithm.com/solve.php

Online tracking

- Bipartite matching
  - What happens if we are missing a prediction?
  - What happens if no prediction is suitable for the match?
  - Introducing extra nodes with a threshold cost

Predictions

Detections



0.9	0.8	0.8	0.3	0.3
0.5	0.4	0.7	0.3	0.3
0.2	0.1	0.4	0.3	0.3
0.1	0.2	0.5	0.3	0.3
0.3	0.3	0.3	0.3	0.3

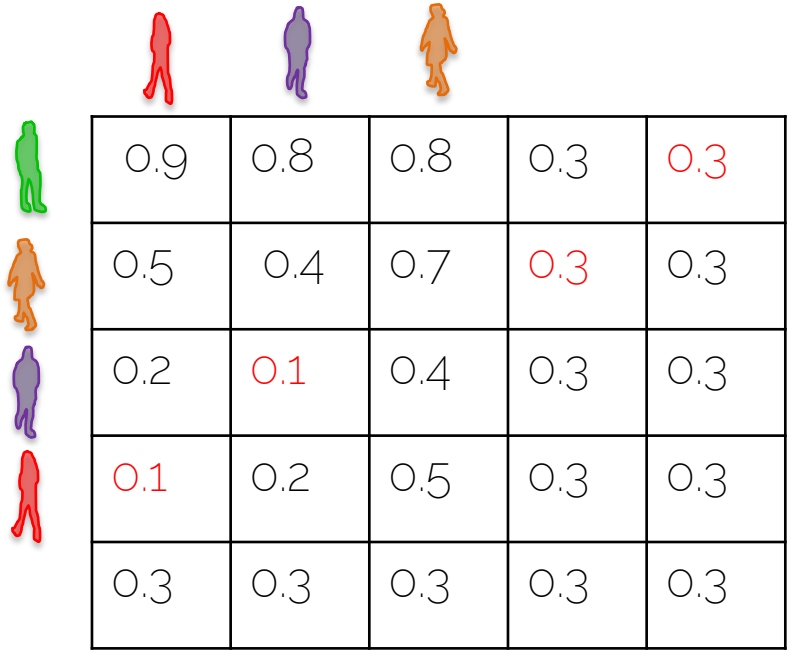
*Demo: http://www.hungarianalgorithm.com/solve.php

Online tracking

- Bipartite matching
  - What happens if we are missing a prediction?
  - What happens if no prediction is suitable for the match?
  - Introducing extra nodes with a threshold cost
  - Apply Hungarian
  - Result: two detections have no matched prediction

Predictions

Detections



0.9	0.8	0.8	0.3	0.3
0.5	0.4	0.7	0.3	0.3
0.2	0.1	0.4	0.3	0.3
0.1	0.2	0.5	0.3	0.3
0.3	0.3	0.3	0.3	0.3

*Demo: http://www.hungarianalgorithm.com/solve.php

Online tracking: what is DL's role?

- 1. Track initialization (e.g. using a detector)
  - Deep Learning has provided us with better detectors ✓

- 2. Prediction of the next position (motion model)
  - Trajectory prediction will be covered in Lecture 6

Adding temporal complexity

- 3. Matching predictions with detections (appearance model)
  - Improving appearance models → Re-Identification (in a few slides)
  - Matching still happens separately from learning, we will see how to couple both steps in the next lecture → MOT as a graph problem

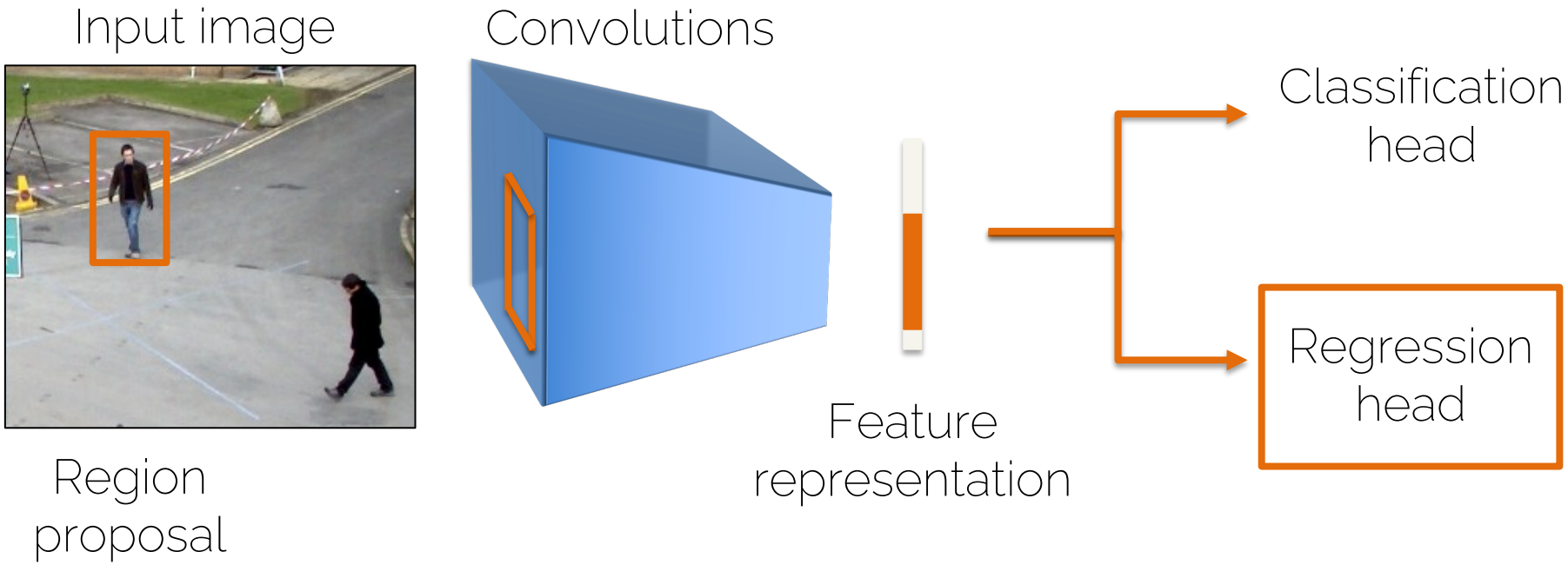
Adding feature complexity

Adding computational complexity

Tracktor: striving for simplicity

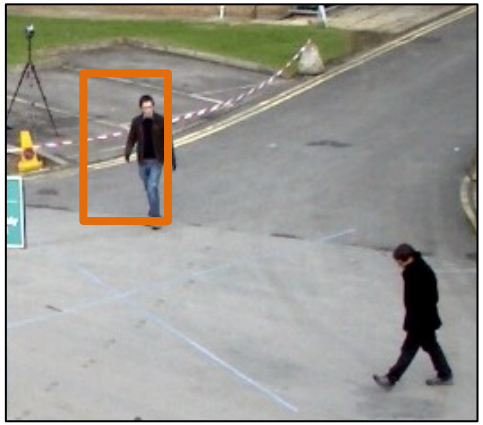
P. Bergmann, T. Meinhardt and L. Leal-Taixé. ICCV, 2019

Recall two step-detectors



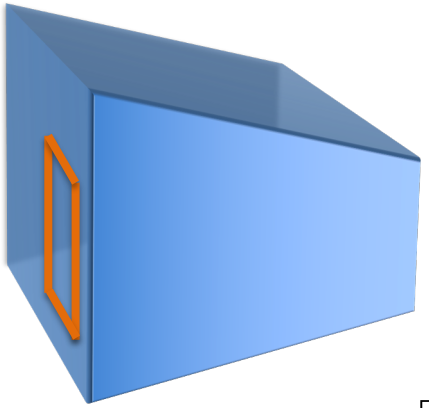
Recall two step-detectors

Input image



Region proposal

Convolutions



Feature representation

Classification head

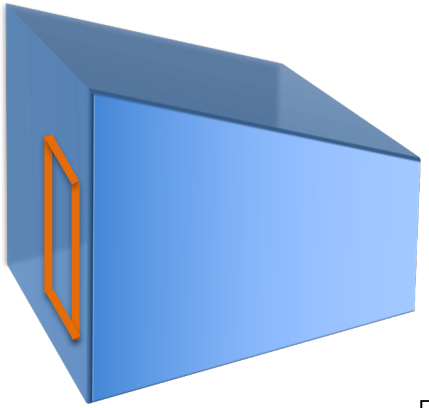
Regression head

Recall two step-detectors

Input image

Regressed bounding box

Convolutions



Feature representation

Classification head

Regression head

Making a detector into a tracktor

- Tracktor: a method trained as a detector but with tracking capabilities

P. Bergmann, T. Meinhardt and L. Leal-Taixé. ICCV, 2019

Making a detector into a tracktor

Frame t+1

Use detections of frame t as proposals

P. Bergmann, T. Meinhardt and L. Leal-Taixé. ICCV, 2019

Making a detector into a tracktor

Bounding box regression

Where did the detection with ID 1 go in the next frame? ✓ Tracking!

P. Bergmann, T. Meinhardt and L. Leal-Taixé. ICCV, 2019

Pros and cons

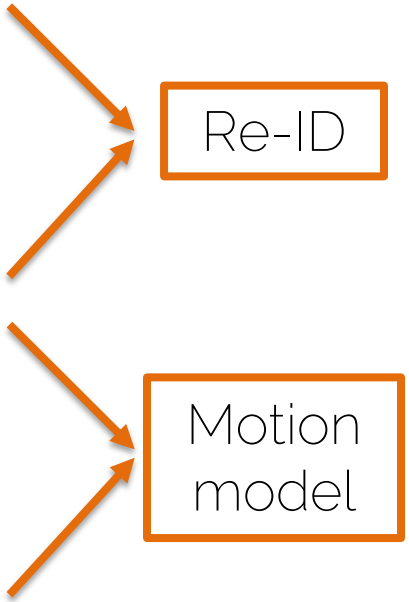
- PRO We can reuse an extremely well-trained regressor
  - We get well-positioned bounding boxes

- PRO We can train our model on still images → easier annotation!

- PRO Tracktor is online

Pros and cons

- CON There is no notion of "identity" in the model
  - Confusion in crowded spaces
- CON As any online tracker, the track is killed if the target becomes occluded
  - Need to close small gaps and occlusions
- CON The regressor only shifts the box by a small quantity
  - Large camera motions
  - Large displacements due to low framerate



Back to….

Modeling appearance → Re-ID

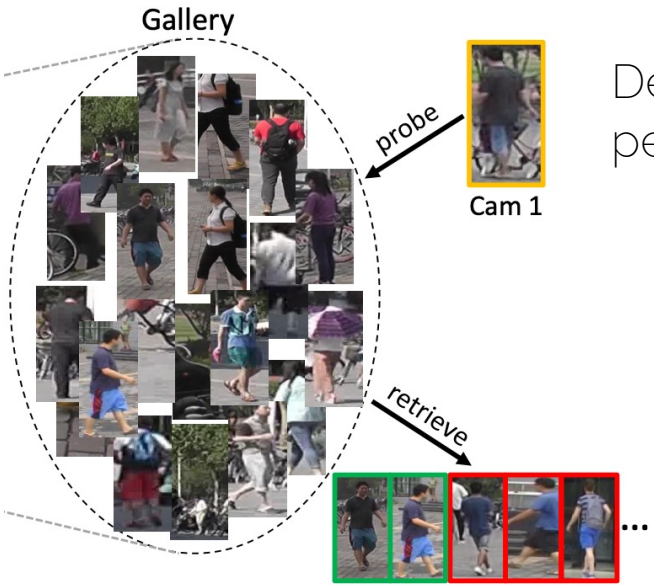
Modeling motion → Motion model

Re-ID

Problem statement

- Viewing tracking as a retrieval problem

Images of different identities



Detected person

How to measure (learn) the distance between two images?

Retrieve the top matches

Can we use classification?

- We can do image classification to get attributes

A 

Classification: person, face, female, brunette

B 

Classification: person, face, female, blonde

(Deep) similarity learning

- What if I just want to answer the question

A 

B 

Is it the same person?

(Deep) similarity learning

- Another type of problems:

A 

B 

  - Similarity Learning: learn a function that measure how similar two objects are.
  - Deep Metric Learning: Learning a distance function over objects.

Similarity Learning: when and why?

- Application: unlocking your iPhone with your face

Training

Similarity Learning: when and why?

- Application: unlocking your iPhone with your face

A  YES

B  NO

Testing

Can be solved as a classification problem

Similarity Learning: when and why?

- Application: face recognition system so students can enter the exam room without the need for ID check

Training

Person 1

Person 2

Person 3

Similarity Learning: when and why?

- Application: face recognition system so students can enter the exam room without the need for ID check

What is the problem
with this approach?

Scalability – we need to retrain our model every time a new student is registered to the course

Similarity Learning: when and why?

- Application: face recognition system so students can enter the exam room without the need for ID check

Can we train one model and use it every year?

Similarity Learning: when and why?

- Learn a similarity function

A

B

Low similarity score (large distance)

A

B

High similarity score (small distance)

Similarity Learning: when and why?

- Learn a similarity function: testing

A

B

$$d(A, B) > \tau$$

Not the same person

Similarity Learning: when and why?

- Learn a similarity function

A

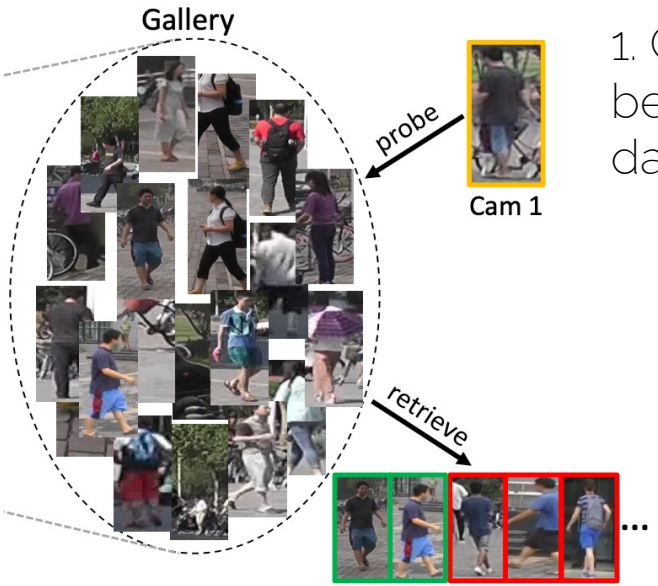
Same person

$$d(A, B) < \tau$$

B

Similarity Learning: when and why?

- How about retrieval and re-ID?



1. Compute distance between query image and database images

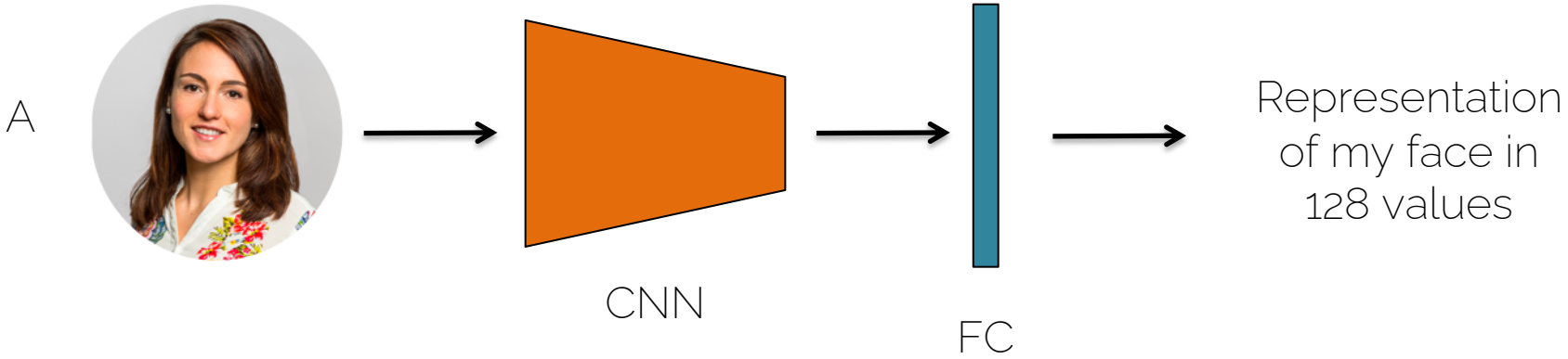
2. Use k-nearest neighbors, i.e., retrieve top images based on distance

Similarity learning

- How do we train a network to learn similarity?

Similarity learning

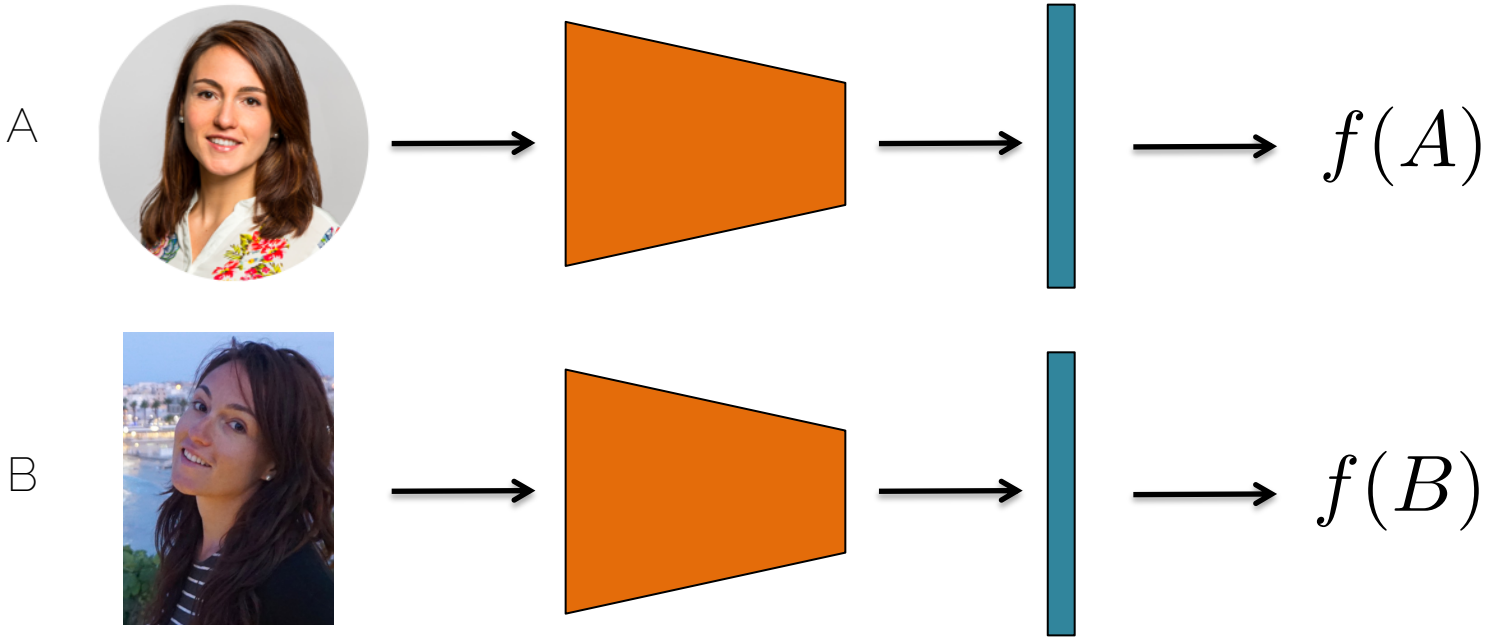
- How do we train a network to learn similarity?



Taigman et al. „DeepFace: closing the gap to human level performance". CVPR 2014

Similarity learning

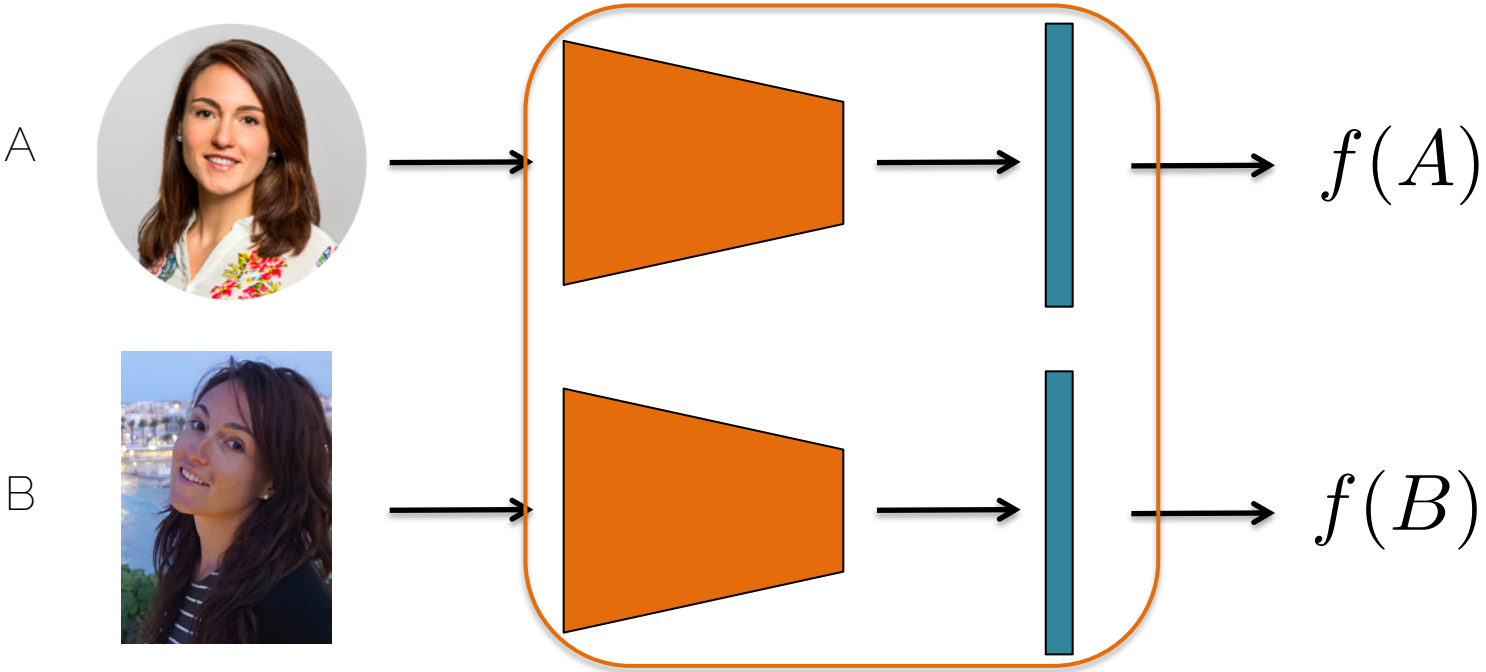
- How do we train a network to learn similarity?



Taigman et al. „DeepFace: closing the gap to human level performance". CVPR 2014

Similarity learning

- Siamese network = shared weights



Taigman et al. „DeepFace: closing the gap to human level performance". CVPR 2014

Similarity learning

- Siamese network = shared weights
- We use the same network to obtain an encoding of the images, $f(A)$ and $f(B)$ respectively.
- To be done: compare the encodings

Taigman et al. „DeepFace: closing the gap to human level performance". CVPR 2014

Similarity learning: losses

- Distance function $d(A, B) = ||f(A) - f(B)||^2$

- Training: learn the parameter such that

  - If A and B depict the same person, $d(A, B)$ is small

  - If A and B depict a different person, $d(A, B)$ is large

Taigman et al. „DeepFace: closing the gap to human level performance". CVPR 2014

Similarity learning: losses

- Loss function for a positive pair:
  - If A and B depict the same person, $d(A, B)$ is small

$$\mathcal{L}(A, B) = ||f(A) - f(B)||^2$$

Similarity learning: losses

- Loss function for a negative pair:
  - If A and B depict a different person, $d(A, B)$ is large
  - Better use a Hinge loss:

margin

$$\mathcal{L}(A, B) = \max(0, m^2 - ||f(A) - f(B)||^2)$$

If two elements are already far away, do not spend energy in pulling them even further away

Similarity learning: losses

- Contrastive loss:

$$\mathcal{L}(A, B) = y^* ||f(A) - f(B)||^2 + (1 - y^*) max(0, m^2 - ||f(A) - f(B)||^2)$$

Positive pair, reduce the distance between the elements

Negative pair, brings the elements further apart up to a margin

Chopra, Hadsell and LeCun. "Learning a similarity metric discriminatively, with application to Face verification", CVPR 2005.

Triplet loss

- Triplet loss allows us to learn a ranking

Anchor (A)

Positive (P)

Negative (N)

We want: $||f(A) - f(P)||^2 < ||f(A) - f(N)||^2$

Schroff et al „FaceNet: a unified embedding for face recognition and clustering“. CVPR 2015

Triplet loss

- Triplet loss allows us to learn a ranking

$$||f(A) - f(P)||^2 < ||f(A) - f(N)||^2$$

$$||f(A) - f(P)||^2 - ||f(A) - f(N)||^2 < 0$$

$$||f(A) - f(P)||^2 - ||f(A) - f(N)||^2 + m < 0$$

margin

Schroff et al „FaceNet: a unified embedding for face recognition and clustering". CVPR 2015

Triplet loss

- Triplet loss allows us to learn a ranking

$$||f(A) - f(P)||^2 < ||f(A) - f(N)||^2$$

$$||f(A) - f(P)||^2 - ||f(A) - f(N)||^2 < 0$$

$$||f(A) - f(P)||^2 - ||f(A) - f(N)||^2 + m < 0$$

$$\mathcal{L}(A, P, N) = max(0, ||f(A) - f(P)||^2 - ||f(A) - f(N)||^2 + m)$$

Schroff et al „FaceNet: a unified embedding for face recognition and clustering“. CVPR 2015

Triplet loss

- Triplet loss allows us to learn a ranking
  - If $d(A, P) > d(A, N)$ the loss is going to be positive
  - If $d(A, P) < d(A, N)$ then I look at the margin

$$\mathcal{L}(A, P, N) = max(0, ||f(A) - f(P)||^2 - ||f(A) - f(N)||^2 + m)$$

Schroff et al „FaceNet: a unified embedding for face recognition and clustering". CVPR 2015

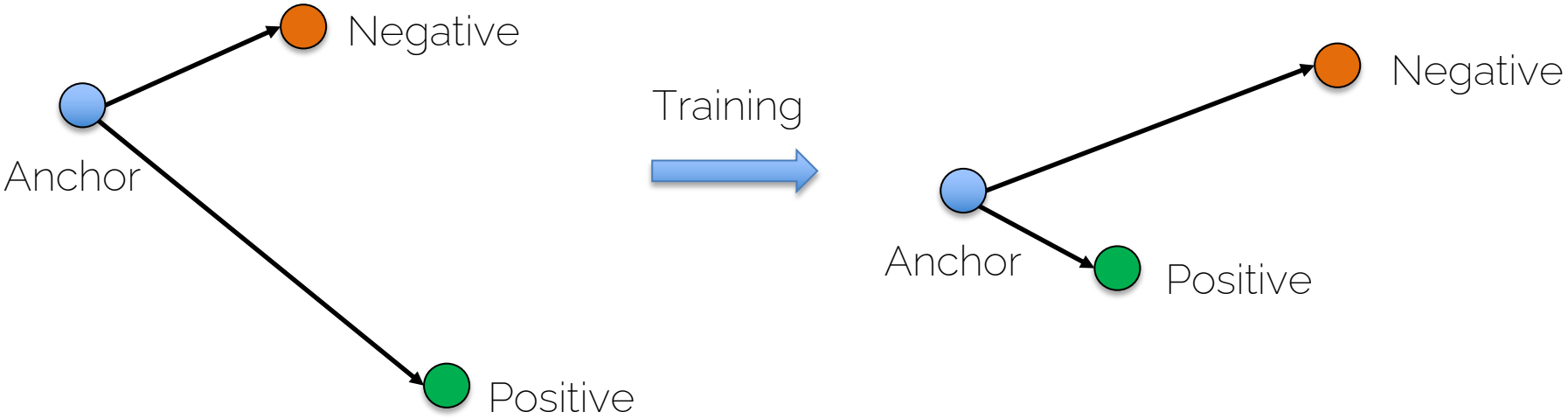
Triplet loss

- Training with hard cases

$$\mathcal{L}(A, P, N) = max(0, ||f(A) - f(P)||^2 - ||f(A) - f(N)||^2 + m)$$

- Train for a few epochs
- Choose the hard cases where $d(A, P) \approx d(A, N)$
- Train with those to refine the distance learned

Triplet loss



Problems with these losses

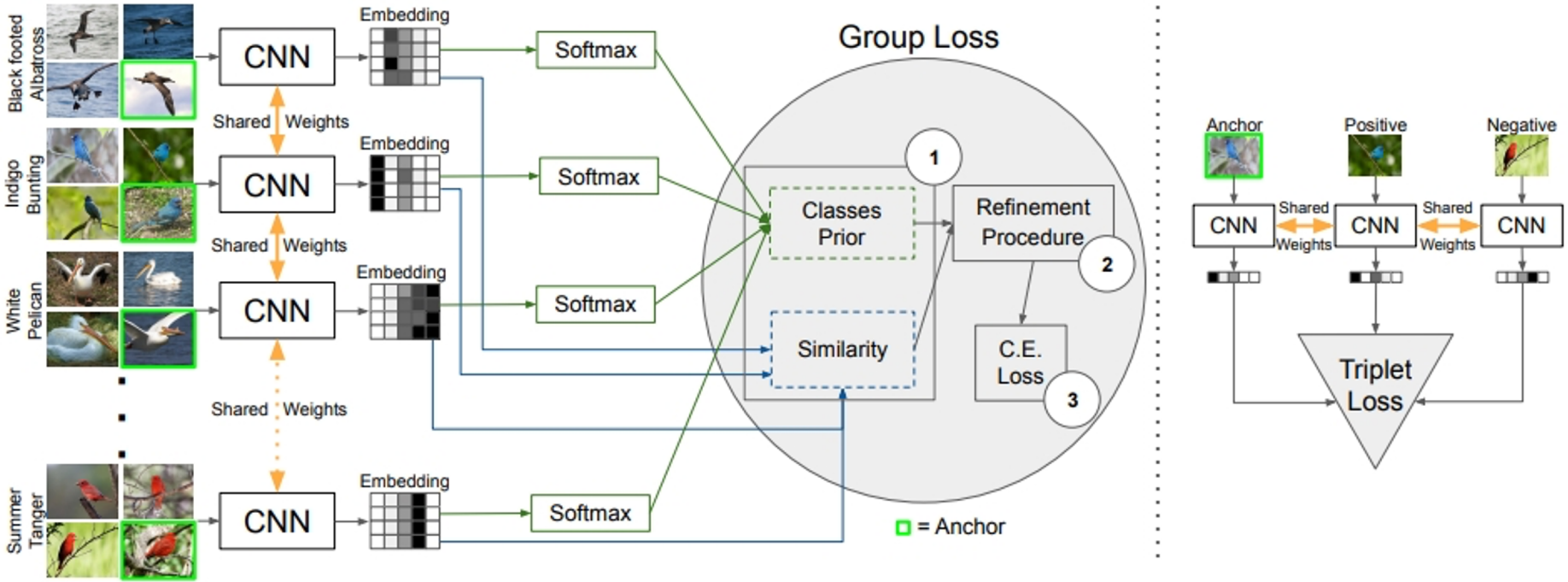
- Number of pairwise relations in a mini-batch is O(n^2), but contrastive loss considers only O(n/2) relations, and triplet loss considers only O(2n/3) relations.
- Too much information is thrown away.
- In order to train these networks many tricks are required: hard-negative mining, intelligent sampling and multi-task learning.

Group Loss

Elezi et al „The Group Loss for deep metric learning". Arxiv: 1912.00385. 2019

Overview

We want to take into account the similarity of all samples wrt all other samples!



Elezi et al „The Group Loss for deep metric learning". Arxiv: 1912.00385. 2019

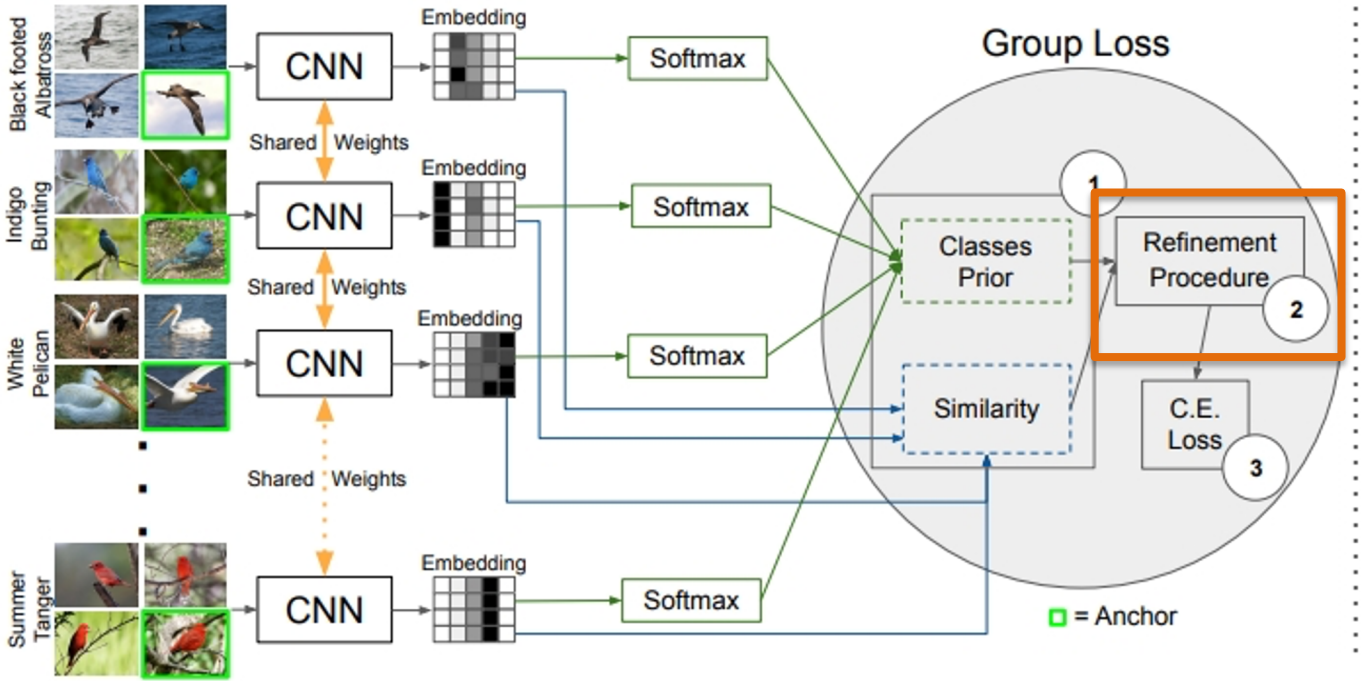
Summary of the model

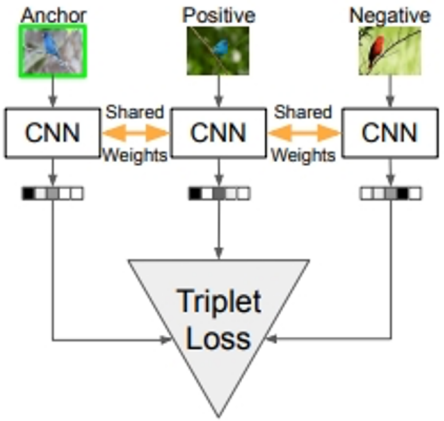
1) **Initialization:** Initialize X, the image-label assignment using the softmax outputs of the neural network. Compute the n x n pairwise similarity matrix W using the neural network embedding.

2) **Refinement:** Iteratively, refine X considering the similarities between all the mini-batch images, as encoded in W, as well as their labeling preferences.

3) **Loss computation:** Compute the cross-entropy loss of the refined probabilities and update the weights of the neural network using backpropagation.

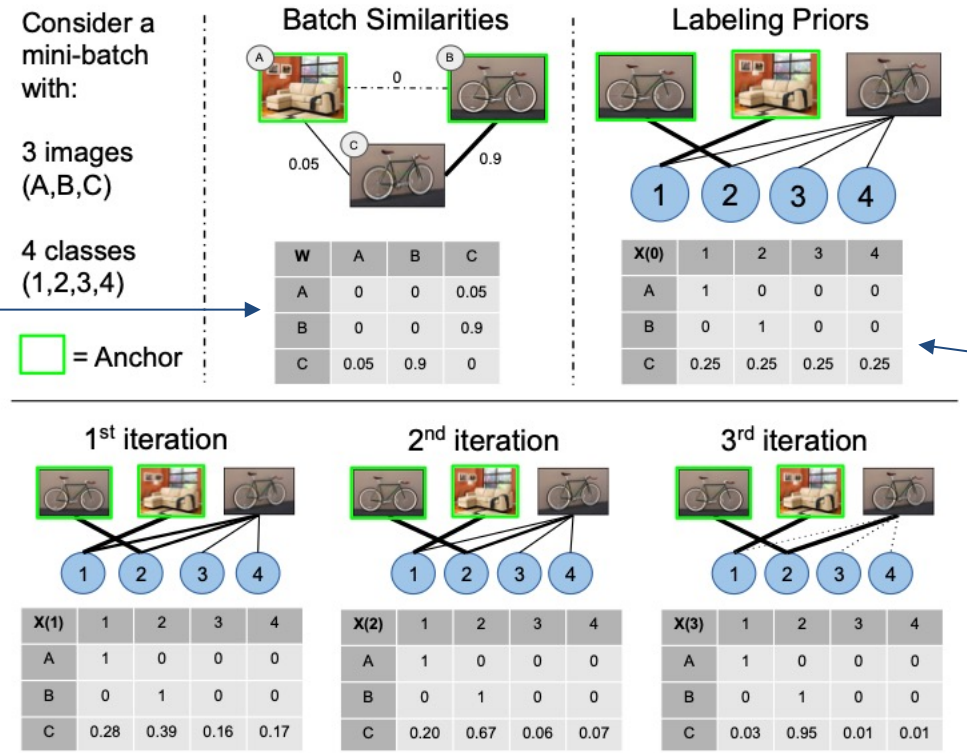
Step 2: Refinement





Step 2: Refinement

B and C are much more similar than A and B, or A and C



Maybe the net is untrained and initialized the probabilities to an uniform distribution.

Step 2: Refinement - math

- Use replicator dynamics*

Iterate t + 1 times

Propagate information

$$x_{i\lambda}(t+1) = \frac{x_{i\lambda}(t)\pi_{i\lambda}(t)}{\sum_{\mu=1}^{m} x_{i\mu}(t)\pi_{i\mu}(t)}$$

Lambda is the class

Normalize in order to stay in standard simplex.

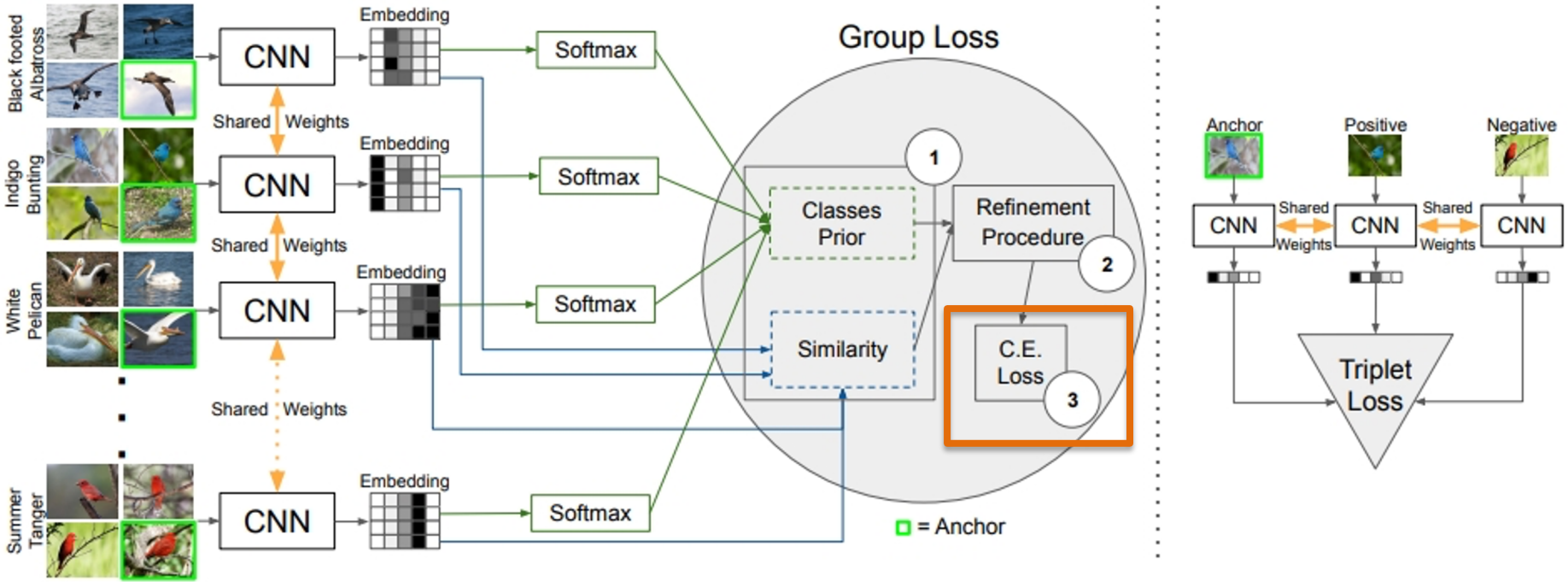
From the similarity matrix

$$\pi_{i\lambda} = \sum_{j=1}^{n} w_{ij} x_{j\lambda}$$

This measures the support that the current mini-batch gives to image I belonging to class lambda

*J.W. Weibull. Evolutionary Game Theory. MIT Press, 1997

Step 3: Loss and backprop



Step 3: Loss

- Compute cross-entropy over the refined probabilities.
- Backpropagate over the entire net
  - Group Loss has no parameters to learn, but it propagates the gradients over the network.
- Result: state-of-the-art results in retrieval

Elezi et al „The Group Loss for deep metric learning". Arxiv: 1912.00385. 2019

Overview

- Mostly focused on online tracking
- Better appearance models with similarity learning (re-ID)
- How about offline tracking? Taking into account more than two frames to compute the matching (to recover from errors, missing detections, wrong predictions)

Lecture 5: MOT with graphs

- Better motion models?

Lecture 6: Trajectory prediction

Object tracking