Nizhny Novgorod State University

Institute of Information Technologies, Mathematics and Mechanics

Department of Computer software and supercomputer technologies

Educational course
«Modern methods and technologies
of deep learning in computer vision»

Deep models for tracking objects in videos

Supported by Intel

Vasiliev Evgeny

Content

- Goals
- Object tracking problem statement
- Public datasets
- Quality metrics
- Deep models for tracking objects
- Conclusion

Goals

- ***The goal*** is to study the problem of object tracking in video and deep learning models for solving this problem

OBJECT TRACKING PROBLEM STATEMENT

Problem statement (1)

- Object tracking involves detecting start location of the object (usually at the time the object first appears in the frame) and predict the location of this object in subsequent frames of the video

Problem statement (2)

- $I_0, I_1, \dots, I_{N-1}$ is a sequence of video frames, where N is a number of frames
- To track objects, it is required to create the mapping ψ of the set of locations B_k in the frame I_k to the set of locations B_{k+1} in the frame I_{k+1}:

$$\psi: B_k \to B_{k+1} \cup \{b\}, \qquad b = \big((-1,-1),(-1,-1)[,s,c]\big)$$

where b is a fake location used to indicate losing of an object by the tracker

- If I_k is a first frame on which the object was detected, $r_0(k)$ is a location identifier, q is a number of frames, then a ***track*** is a sequence of object locations

$$T^k_{r_0(k)} = \left(b^k_{r_0}, b^{k+1}_{r_1}, \dots, b^{k+q-1}_{r_{q-1}}\right), \qquad b^{k+i}_{r_i} = \psi\left(b^{k+i-1}_{r_{i-1}}\right), i = \overline{1, q-1}$$

PUBLIC DATASETS

Public datasets (1)

Dataset	Number of videos	Number of frames	Number of tracks
Tracking of a single object (a single object in a video)			
Long-Term Visual Object Tracking Benchmark [https://amoudgl.github.io/tlp]	50	676 000	50
TrackingNet [https://tracking-net.org]	30 643	14 220 000	30 643
VOT Challenge [http://www.votchallenge.net]	60	21 455	60

Public datasets (2)

Dataset	Number of videos	Number of frames	Number of tracks	Number of objects
Tracking of multiple objects				
Multiple Object Tracking Benchmark [https://motchallenge.net]	21	33 705	3 993	901 119
CityFlow [https://www.aicitychallenge.org]	40	~117 000	666	229 680

Public datasets (3)

- CityFlow dataset contains more than 25 hours of video from 40 on-road cameras in the same city (USA); the videos are time synchronized
- TrackingNet consists of YouTube videos, so this dataset contains a large number of videos of various quality, including a large number of low-resolution videos
- Several videos of the VOT Challenge Benchmark in addition to RGB images include video from depth and infrared cameras
- Several videos of the VOT Challenge Benchmark contain annotation of bounding boxes, sides of these bounding boxes are not parallel to the coordinate axes

Multiple Object Tracking Benchmark (1)

- Multiple Object Tracking Challenge (MOT) is a contest for object tracking in video, held since 2015
- Official web-page [https://motchallenge.net]
- The latest dataset version in the public domain is MOT17
  - 21 video of moving vehicles and people in the city
  - 15 948 images and 1 638 tracks in the train dataset, 17 757 images and 2 355 tracks in the test dataset
- The dataset contains videos from static and moving cameras

* Leal-Taixé L., Milan A., Reid I., Roth S., Schindler K. MOTChallenge 2015: Towards a Benchmark for Multi-Target Tracking. – 2015. – [https://arxiv.org/pdf/1504.01942.pdf].

Multiple Object Tracking Benchmark (2)

Examples of video frames with tracks and bounding boxes

* Multiple Object Tracking Benchmark. MOT17 [https://motchallenge.net/data/MOT17].

** Leal-Taixé L., Milan A., Reid I., Roth S., Schindler K. MOTChallenge 2015: Towards a Benchmark for Multi-Target Tracking. – 2015. – [https://arxiv.org/pdf/1504.01942.pdf].

Long-Term Visual Object Tracking Benchmark

- Long-Term Visual Object Tracking Benchmark is a benchmark containing a set of long videos with single objects
- Official web-page [https://amoudgl.github.io/tlp]
- The dataset consists of 50 videos of different context with single objects
- More than 400 minutes, 676 000 frames

Example of the first frames in videos

* Moudgil A., Gandhi V. Long-Term Visual Object Tracking Benchmark. – 2019. – [https://arxiv.org/abs/1712.01358].

TrackingNet

- TrackingNet is a large dataset of YouTube videos of various quality and resolution
- Official web-page [https://tracking-net.org]
- More than 30 thousand videos, 140 hours
- Over 14 million detected objects

Examples of two sequences with tracked objects

* Müller M., Bibi A., Giancola S., Al-Subaihi S., Ghanem B. TrackingNet: A Large-Scale Dataset and Benchmark for Object Tracking in the Wild. – 2018. – [https://arxiv.org/abs/1803.10794].

QUALITY METRICS

Quality metrics

- Criteria for choosing quality metrics of object tracking:
  - Quality metric has to confirm the quality of location prediction on each frame of the video containing the tracked object
  - Quality metric has to confirm the quality of tracking throughout the sequence of all frames containing the object
  - For each tracked object, the track has to be a single one
  - Quality metric has to ensure comparability of metrics for different types of tracking algorithms (2D, 3D trackers, centroid trackers, area trackers, etc.)

Presented quality metrics

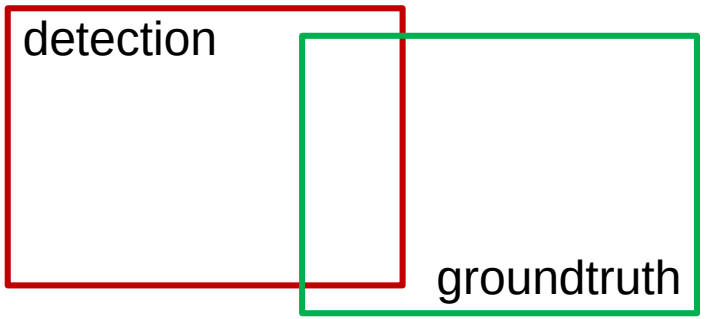
- Quality metrics for single-object tracking:
  - Accuracy
  - Robustness

- Quality metrics for multiple-object tracking:
  - Multiple Object Tracking Accuracy (MOTA)
  - Multiple Object Tracking Precision (MOTP)

Accuracy

- ***Accuracy**** is a mean ratio of overlapping bounding boxes, predicted by a tracker (detection) and annotated by an expert (groundtruth) in all video frames

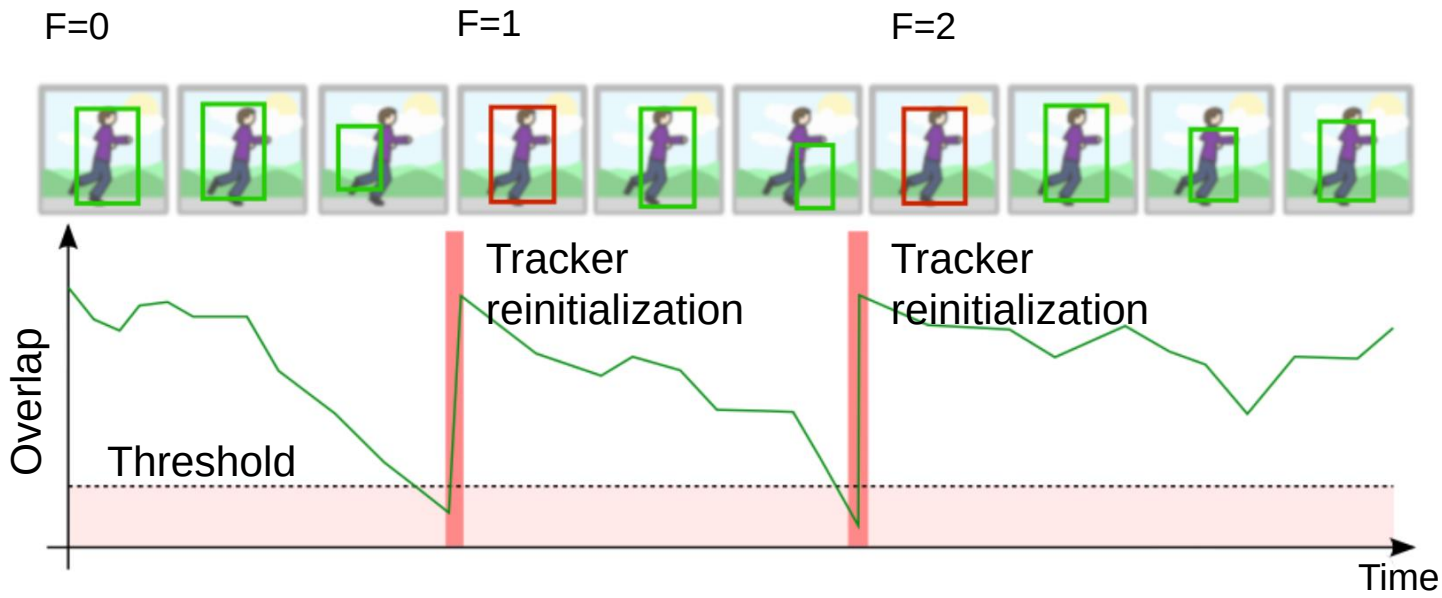
$$A = \frac{1}{N}\sum_{t=1}^{N}\frac{S_{d\cap g}}{S_{d\cup g}}$$



* Kristan M., et al. The Visual Object Tracking VOT2014:Challenge and results. – 2014. – [https://votchallenge.net/vot2014/download/vot_2014_presentation.pdf].

Robustness

- ***Robustness**** shows how many times the tracker loses the tracked object and has to be reinitialized
- A tracked object is considered as lost when the overlap between the detection and groundtruth is less than a certain threshold



* Kristan M., et al. The Visual Object Tracking VOT2014:Challenge and results. – 2014. – [https://votchallenge.net/vot2014/download/vot_2014_presentation.pdf].

MOTA (1)

- ***Multiple Object Tracking Accuracy**** (MOTA)
- One of the well-known and common metrics that is used to compare tracking algorithms in different contests and benchmarks
- Notation:
  - t is an identifier of the current frame in the video
  - $\{o_1, o_2, \dots, o_n\}$ is a set of observed objects in the frame t (groundtruth)
  - $\{h_1, h_2, \dots, h_m\}$ is a set of object locations predicted by the tracking algorithm in the frame t

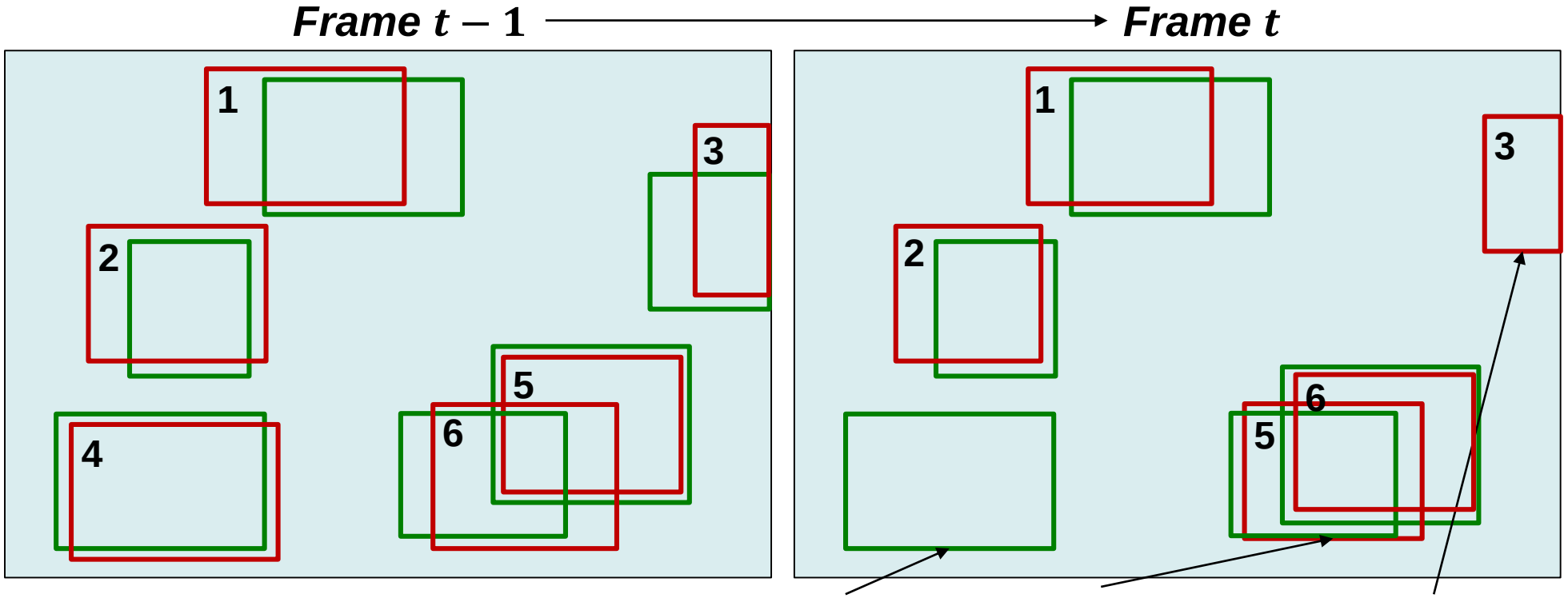
* Bernardin K., Stiefelhagen R. Evaluating Multiple Object Tracking Performance: The CLEAR MOT Metrics // Image and Video Processing. – 2008.

MOTA (2)

- ***Multiple Object Tracking Accuracy**** (MOTA)
- Metric calculation for each frame t:
  1. Calculating an error of object location prediction and searching for the best matching between the constructed locations $\{h_1, h_2, \dots, h_m\}$ and groundtruth $\{o_1, o_2, \dots, o_n\}$
  2. Accumulating the following errors:
     1. Counting ***misses***. Misses are annotated objects for which there is no matched predicted locations
     2. Counting ***false positives***. False positives are predicted locations for which there is no real objects in the frame
     3. Counting ***mismatch errors***. Mismatch errors occur when object track is reinitialized with a new track identifier

* Bernardin K., Stiefelhagen R. Evaluating Multiple Object Tracking Performance: The CLEAR MOT Metrics // Image and Video Processing. – 2008.

MOTA (3)



– groundtruth

– predicted location

m **– miss** (object is lost)

mme **– mismatch error** (track identifier was changed)

fp **– false positive** (real object disappeared from the frame)

MOTA (4)

- ***Multiple Object Tracking Accuracy**** (MOTA) is calculated as follows:

$$MOTA = 1 - \frac{\sum_t (m_t + fp_t + mme_t)}{\sum_t g_t}$$

where m_t is a number of misses on the frame t,
fp_t is a number of false positives on the frame t,
mme_t is a number of mismatches on the frame t,
g_t is a number of real objects on the frame t

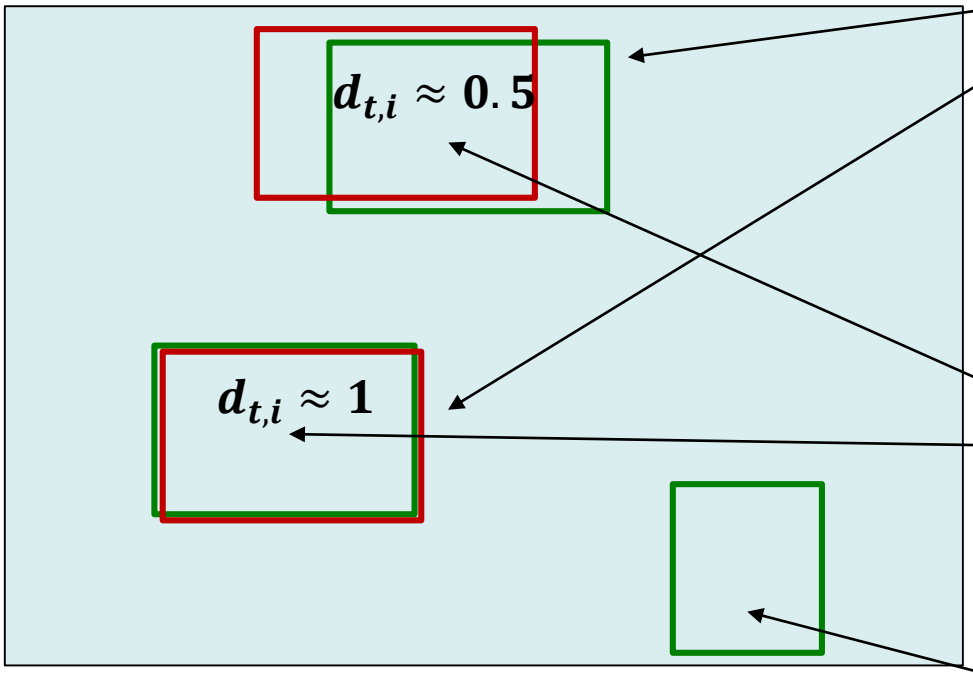
- ***Note:*** there are various approaches for matching predicted and annotated locations (for example, the Hungarian algorithm for solving the assignment problem based on a similarity matrix between annotated and predicted locations)

* Bernardin K., Stiefelhagen R. Evaluating Multiple Object Tracking Performance: The CLEAR MOT Metrics // Image and Video Processing. – 2008.

MOTP (1)

- ***Multiple Object Tracking Precision*** (MOTP)
- The metric shows the quality of detecting objects in the problem of object tracking
- Notation:
  - $\{o_1, o_2, \dots, o_n\}$ is a set of annotated bounding boxes (groundtruth) at the frame t
  - $\{h_1, h_2, \dots, h_m\}$ is a set of object locations predicted by the tracking algorithm at the frame t
  - $\{P_0^t, P_1^t, \dots, P_k^t\}$ is a set of matches between annotated and constructed object locations in tracks. Matches can be found by the Hungarian algorithm

MOTP (2)



$d_{t,i}$ is the ratio of overlapping bounding boxes from the pair P_i^t

c_t is the number of matches between annotated and predicted object locations

$c_t = 2$

An object without a pair, it is not involved in the metric calculation

– annotated location

– predicted location

MOTP (3)

- ***Multiple Object Tracking Precision*** (MOTP) is calculated as follows:

$$MOTP = \frac{\sum_{t,i} d_{t,i}}{\sum_t c_t} = \frac{\sum_{t,i} \frac{o_i \cap h_i}{o_i \cup h_i}}{\sum_t c_t}$$

  - c_t is the number of matches between annotated and predicted object locations at the frame t
  - $d_{t,i}$ is the ratio of overlapping bounding boxes from the pair P_i^t
  - $o_i \cap h_i$is the intersection of bounding boxes belonging the pair P_i^t
  - $o_i \cup h_i$is the union of bounding boxes belonging the pair P_i^t

MOTP (4)

- In some benchmarks, $d_{t,i}$ in the previous formula means the Euclidean distance between the centers of the bounding boxes:

$$MOTP = \frac{\sum_{t,i} d_{t,i}}{\sum_t c_t} = \frac{|Center(o_i) - Center(h_i)|}{\sum_t c_t}$$

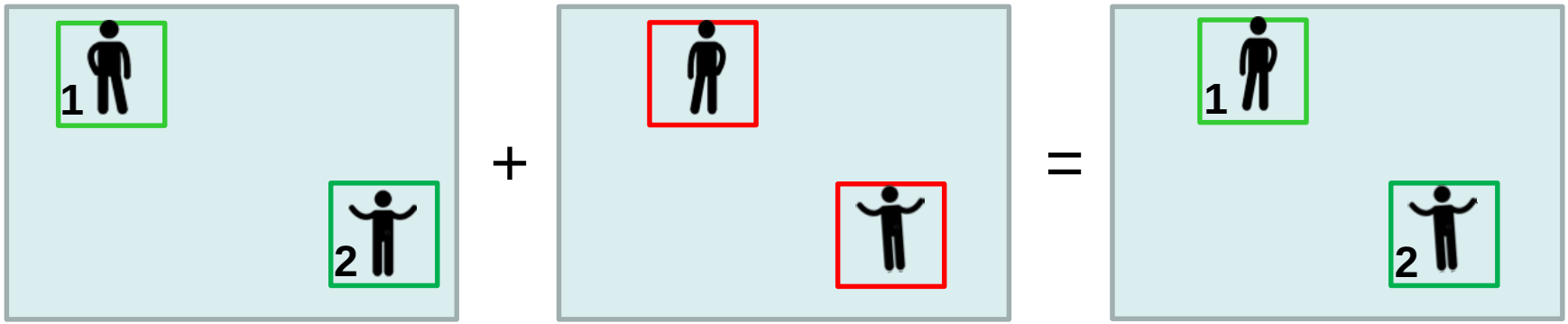
- ***Note***: MOTP does not take into account losing the object during tracking

* Bernardin K., Elbs A., Stiefelhagen R. Multiple Object Tracking Performance Metrics and Evaluation in a Smart Room Environment. – 2006. – [https://cvhci.anthropomatik.kit.edu/~stiefel/papers/ECCV2006WorkshopCameraReady.pdf].

DEEP MODELS FOR TRACKING OBJECTS

Scheme of multiple object tracking by object matching (1)

- Supposed object tracks are constructed in the frame t
- Detecting objects in the frame $t + 1$ (object detection is independent of previous frames)
- Matching tracks in the frame t and detections in the frame $t + 1$ to make the best prediction of object tracks in the frame $t + 1$



Object locations in the tracks of the frame t

Detected objects in the frame $t + 1$

Object locations in the tracks of the frame $t + 1$

Scheme of multiple object tracking by object matching (2)

- To match tracks and detections an affinity matrix (similarity matrix) is constructed
- The similarity matrix A for N tracks and M objects is the $N \times M$ matrix, where each element a_{ij} is a coefficient of similarity between the track T_i and the object R_j
  - Affinity coefficient (similarity coefficient) a_{ij} can be calculated using the position, shape and appearance of the object in the track and the detected object
- Searching for the best matches based on the affinity matrix is an optimization problem, and it is solved using the Hungarian algorithm

Classification of methods and deep models for object tracking

- ***Classification of methods by data access***:
  - Offline-methods
  - Online-methods

- ***Classification of deep models by their role in tracking method***:
  - Supplementing the existing tracking method
  - Deep network embedding
  - End-to-end deep networks

Classification of methods by data access

- ***Offline-methods*** have access to all frames of the video at the same time, and construct object tracks for the entire video
- ***Online-methods*** have sequential access to the video frames, and predict tracks for the current frame only based on the previous frames
- The industry and researchers are more interested in online tracking algorithms, since these algorithms allow to process the input frames in real-time

Classification of deep models by their role in tracking method

- ***Supplementing the existing tracking method***
  - Constructing a deep description for object tracking
- ***Deep network embedding***
  - Deep models replace some stages of tracking algorithms (for example, the detection stage)
- ***End-to-end deep networks***
  - Deep models replace all stages of object tracking (detection, matching, creating and removing tracks)

Deep models (1)

Supplementing the existing tracking method

- ***DeepSORT (2017)***
  - Wojke N., Bewley A., Paulus D. Simple online and realtime tracking with a deep association metric // International Conference on Image Processing. – 2017. – P. 3645–3649. – [https://arxiv.org/pdf/1703.07402.pdf], [https://ieeexplore.ieee.org/document/8296962].

Deep models (2)

Deep network embedding

- ***SINT (2016)***
  - Tao R., Gavves E., Smeulders A. Siamese instance search for tracking. – 2016. – [https://arxiv.org/pdf/1605.05863.pdf], [https://ieeexplore.ieee.org/document/7780527].
- ***SiameseNET (2017)***
  - Leal-Taixé L., Canton-Ferrer C., Schindler K. Learning by tracking: siamese CNN for robust target association. – 2016. – [https://arxiv.org/pdf/1604.07866.pdf], [https://ieeexplore.ieee.org/document/7789549].
- ***GOTURN (2016)***
  - Held D., Thrun S., Savarese S. Learning to track at 100 FPS with deep regression networks. – 2016. – [https://arxiv.org/pdf/1604.01802.pdf].

Deep models (3)

End-to-end deep networks

- ***RNN-LSTM (2017)***
  - Milan A., Rezatofighi S.H., Dick A.R., et al. Online multi-target tracking using recurrent neural networks. – 2017. – [https://arxiv.org/pdf/1604.03635.pdf], [https://dl.acm.org/doi/10.5555/3298023.3298181].

DeepSORT (1)

- Comparing the tracks and the detected objects through matching, it is required to calculate the coefficient of similarity between the descriptions of these tracks and detected objects
- Objects may be compared by the following criteria:
  - Location
  - Shape
  - Appearance

DeepSORT (2)

- Comparison of objects in appearance increases the likelihood of correct object tracking in the case of object reidentification (intersections or occlusions)
- Initially, features for constructing target appearance were designed manually, but now features are extracted using neural networks
- To extract object features, models based on the well-known classification deep neural networks are often used

DeepSORT (3)

- The idea of DeepSORT is to extract a feature vector using a deep model to describe the object features
- The deep neural network is constructed and trained on the train dataset, and then the last layer responsible for the object classification is discarded
- Instead of constructing and training the model from scratch, existing classification models can be used. Therefore, this model allows to extract high-level features of the object

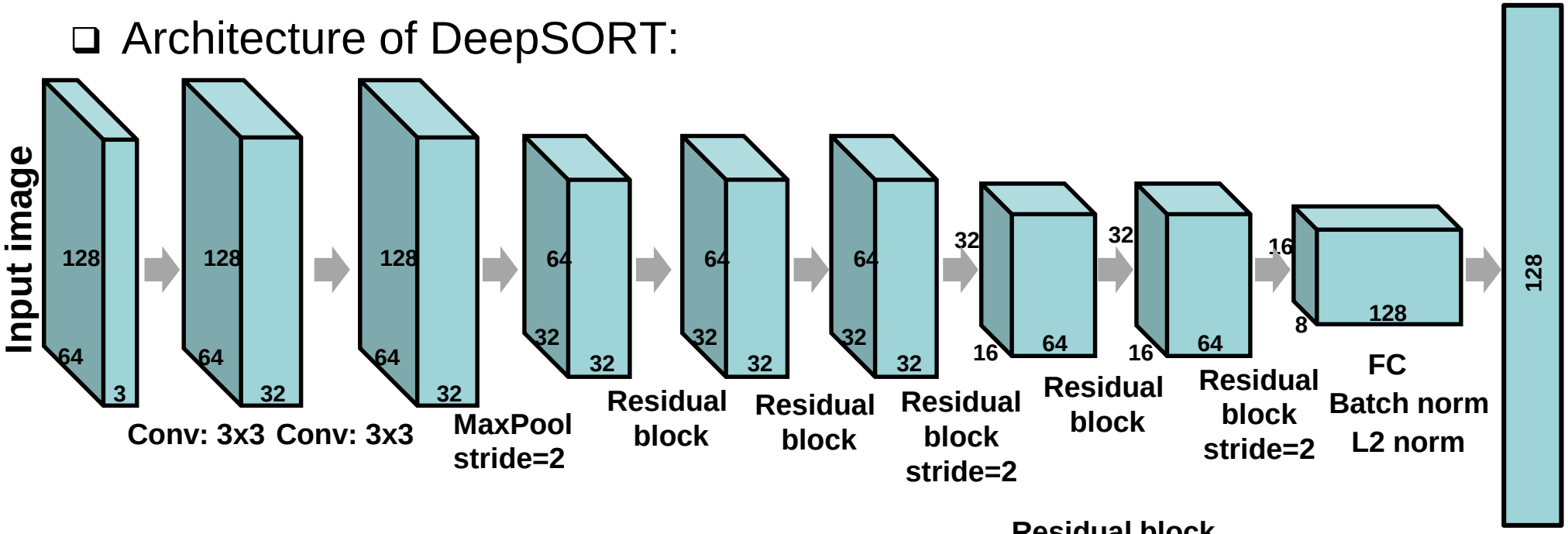
DeepSORT (4)

- DeepSORT (Deep Simple Online and Realtime Tracking) is a model which generates a ***descriptor*** (deep description of the detected object) represented by a vector of the size 128
- The model input:
  - RGB image of an object of the resolution 128x64
- The model output:
  - Object descriptor is a tensor of the shape 1x128

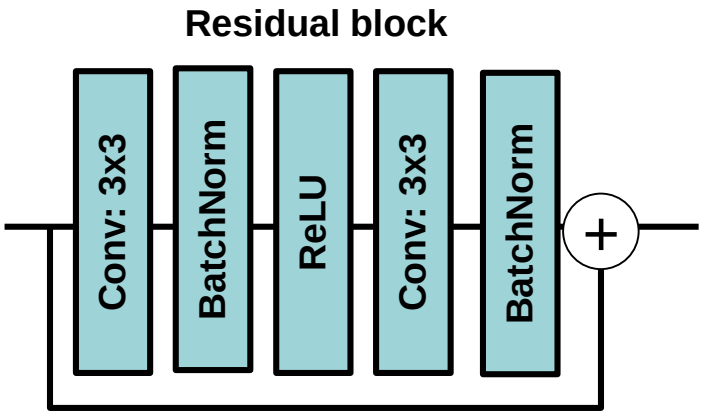
* Wojke N., Bewley A., Paulus D. Simple online and realtime tracking with a deep association metric. – 2017 – [https://arxiv.org/pdf/1703.07402.pdf].

DeepSORT (5)

- Architecture of DeepSORT:



- Architecture of residual blocks:
  - In blocks with stride=2 convolutions are calculated with the stride 2



DeepSORT (6)

- Object descriptor extracted by the DeepSORT model is involved in computing an affinity coefficient between the track and the new detection; it improves the quality of object tracking
- The appearance affinity coefficient D_a is calculated as follows:
  - Descriptors for the image of the object from the track a_i and the detected object a_j are calculated using a deep model
  - The affinity coefficient is calculated by the formula of the cosine distance between the descriptors of the objects a_i and a_j

$$D_a = cosine\_distance(a_i, a_j) = \frac{(a_i, a_j)}{\|a_i\| \cdot \|a_j\|}$$

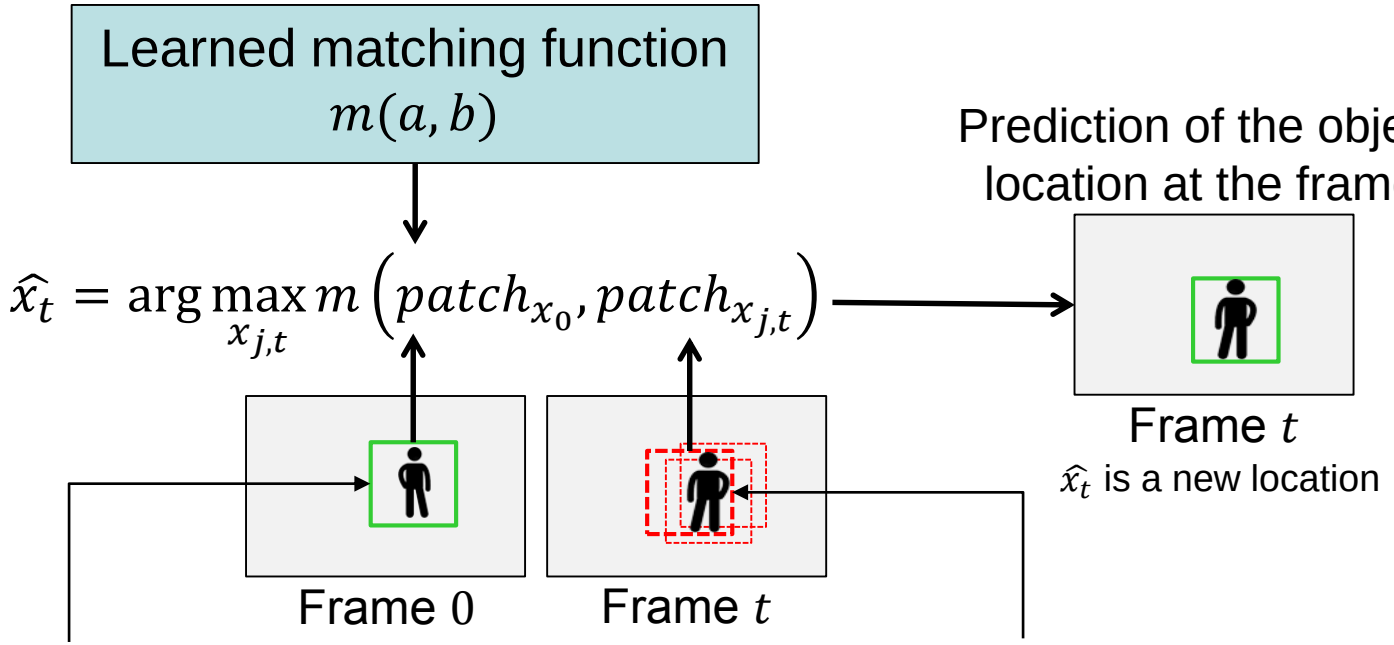
SINT (1)

- SINT (Siamese INstance search for Tracking) is a method that receives the first frame with the object location and the current frame with the hypotheses of object locations and returns a prediction of the object location on the current frame
- SINT is not specialized for the certain object class (pedestrians, vehicles, etc.), the method is designed for tracking objects of various classes

* Tao R., Gavves E., Smeulders A. Siamese instance search for tracking. – 2016. – [https://arxiv.org/pdf/1605.05863.pdf], [https://ieeexplore.ieee.org/document/7780527].

SINT (2)

- The idea of SINT is to construct a learned function that provides the best correspondences between the object at the first frame and the hypotheses about object location at the current frame



* Tao R., Gavves E., Smeulders A. Siamese instance search for tracking. – 2016. – [https://arxiv.org/pdf/1605.05863.pdf], [https://ieeexplore.ieee.org/document/7780527].

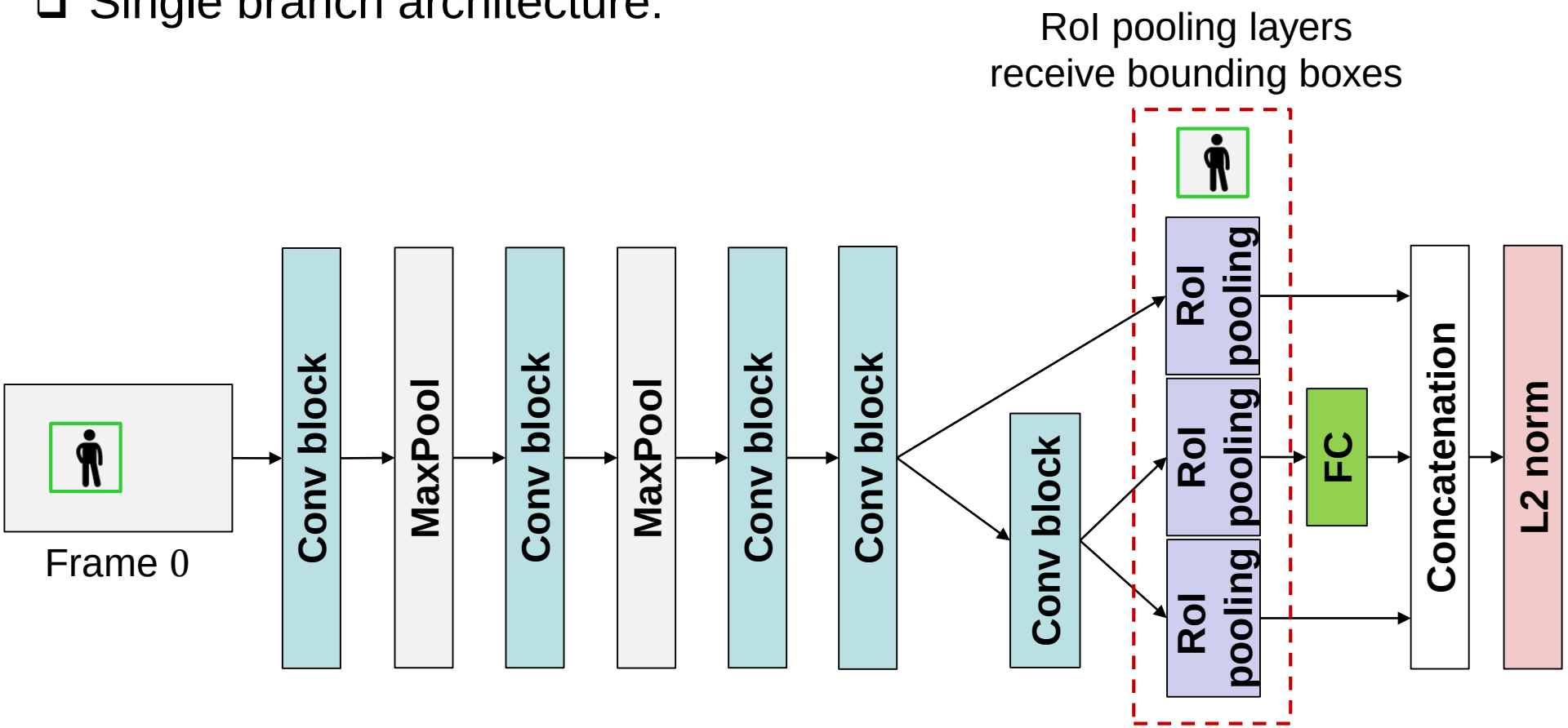
SINT (3)

- Siamese model consists of two symmetrical branches
- The first branch of the model receives the first frame and bounding box from the track, the second one receives the current frame and hypotheses (bounding boxes) about the object location on the current frame
- Input of a single branch:
  - Image is a tensor of the shape 1x3x512x512
  - Set of bounding boxes is a tensor of the size Nx4, where N is the number of hypotheses about the object location on the current frame
- Hypothesis generation is described in detail in the original paper*

* Tao R., Smeulders A., Chang S.-F. Attributes and categories for generic instance search from one example. – 2015. – [openaccess.thecvf.com/content_cvpr_2015/papers/Tao_Attributes_and_Categories_2015_CVPR_paper.pdf].

SINT (4)

- Single branch architecture:



SINT (5)

- Convolutional block (Conv block):
  - Conv: 3x3
  - Conv: 3x3
  - ReLU
- Parameters of the RoI pooling layer (RoI pooling):
  - Pooled width: 7
  - Pooled height: 7

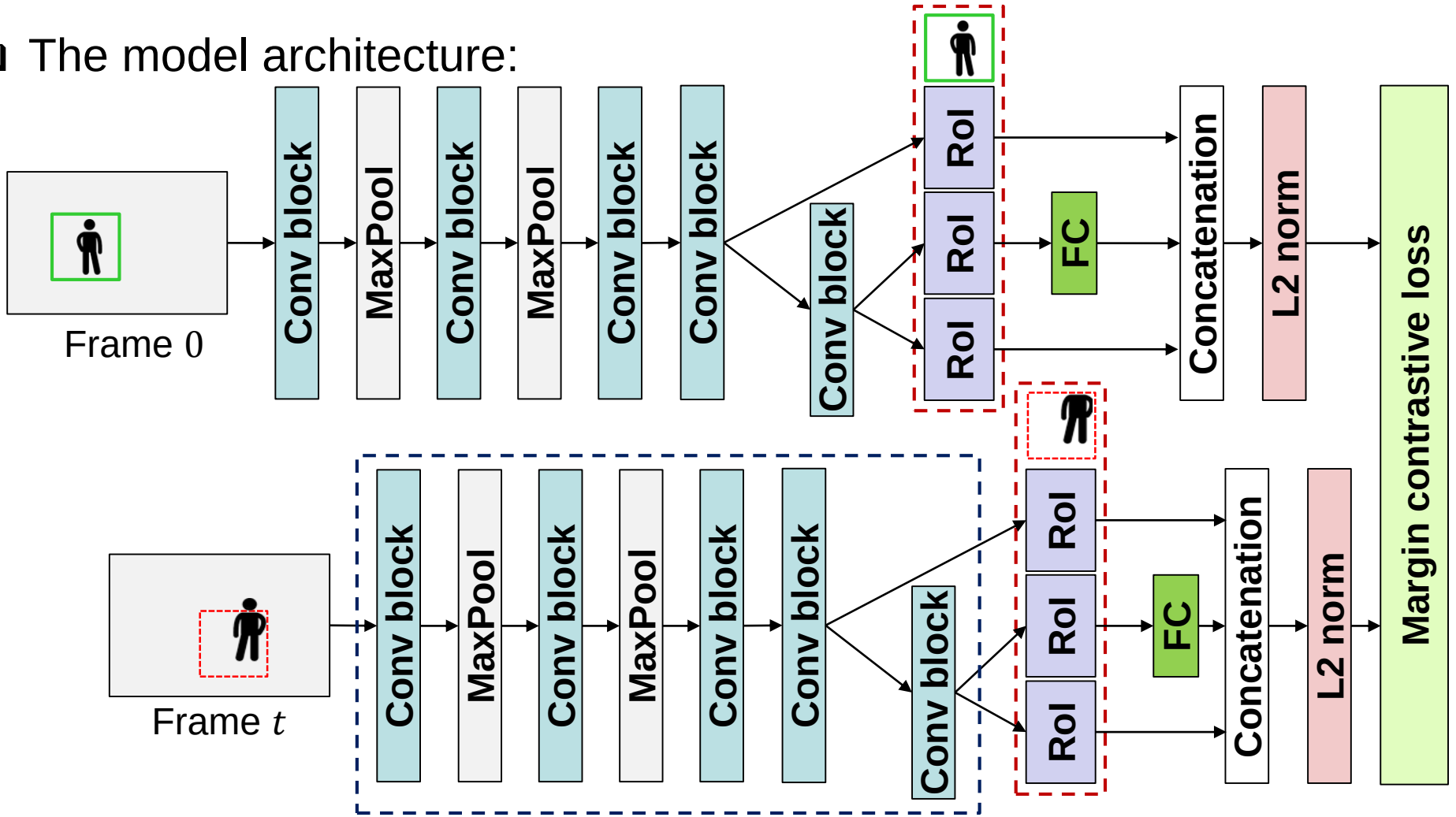
- The model contains a small number of layers to extract low-level features of the object (edges, angles, etc.)

SINT (6)

- An object description is generated at the output of the branch
- Branch output:
  - Object descriptor for a bounding box is a vector of the size 54 272 (7*7*512 + 4 096 + 7*7*512)
- An object descriptor is constructed by concatenation of outputs from three layers:
  - The output of the first RoI pooling layer of the spatial size 7x7 and 512 channels, flatten into a one-dimensional vector
  - Output of the fully connected layer is a vector of the size 4 096
  - The output of the third RoI pooling layer of the spatial size 7x7 and 512 channels, flatten into a one-dimensional vector

SINT (7)

- The model architecture:



Convolutions are calculated once for all hypotheses

SINT (8)

- For the model training, ***the margin contrastive loss*** is used:

$$L(x_j, x_k, y_{jk}) = \frac{1}{2} y_{jk} D^2 + \frac{1}{2}(1 - y_{jk}) \max\{0, \epsilon - D^2\},$$

where $D^2 = \|f(x_j) - f(x_k)\|$ is the Euclidean distance between descriptors x_j and x_k,

$$y_{jk} = \begin{cases} 1, \text{if } x_j \text{ and } x_k \text{ represent the same object} \\ 0, \text{if } x_j \text{ and } x_k \text{ represent different objects} \end{cases}$$

ϵ (margin) is the parameter above of which objects will be considered different

SINT (9)

- After calculating the descriptors for all pairs “object-hypothesis”, the pair whose descriptor is closest to the object descriptor at the first frame is selected

$$\widehat{x_t} = \arg\max_{x_{j,t}} m\left(patch_{x_0}, patch_{x_{j,t}}\right)$$

$$m(x, y) = f(x)^T f(y)$$

where $f(x)$ is a descriptor calculated for the frame with bounding box, t is a frame number, $x_{j,t}$ is a hypothesis j for the frame t,

$patch_{x_0}$ is the frame 0 and object location, i.e. bounding box (groundtruth)

$patch_{x_{j,t}}$ is the frame t and object location corresponding to the bounding box for the hypothesis j

SiameseNET (1)

- The idea of the SiameseNET* algorithm:
  - Objects on the frames M and N are detected; and for each pair of object images, a confidence that these images contain the same object is calculated using a deep model
  - For each pair of object images ***the context*** is also calculated
  - Further, the stochastic gradient boosting classifier** is trained based on the descriptor extracted from the last fully connected layer of the model and the context of the pair
  - This classifier calculates the correct mappings between objects on the frames M and N

* Leal-Taixé L., Canton-Ferrer C., Schindler. K. Learning by tracking: siamese CNN for robust target association. – 2016. – [https://arxiv.org/pdf/1604.07866.pdf].

** Friedman. J. Stochastic gradient boosting // Computational Statistics & Data Analysis. – 2002. P. 367–378.

SiameseNET (2)

- Assumed the first object location is $p_1 = (x, y)$, and its size is $S_1 = (w, h)$ at the time t_1 (for the second time the similar notation is introduced)
- The context for the pair of object locations consists of the following contextual features:
  - Relative size change:

$$\frac{S_1 - S_2}{S_1 + S_2} = \left(\frac{w_1 - w_2}{w_1 + w_2}, \frac{h_1 - h_2}{h_1 + h_2}\right)$$

  - Position change:

$$p_1 - p_2$$

  - Relative velocity:

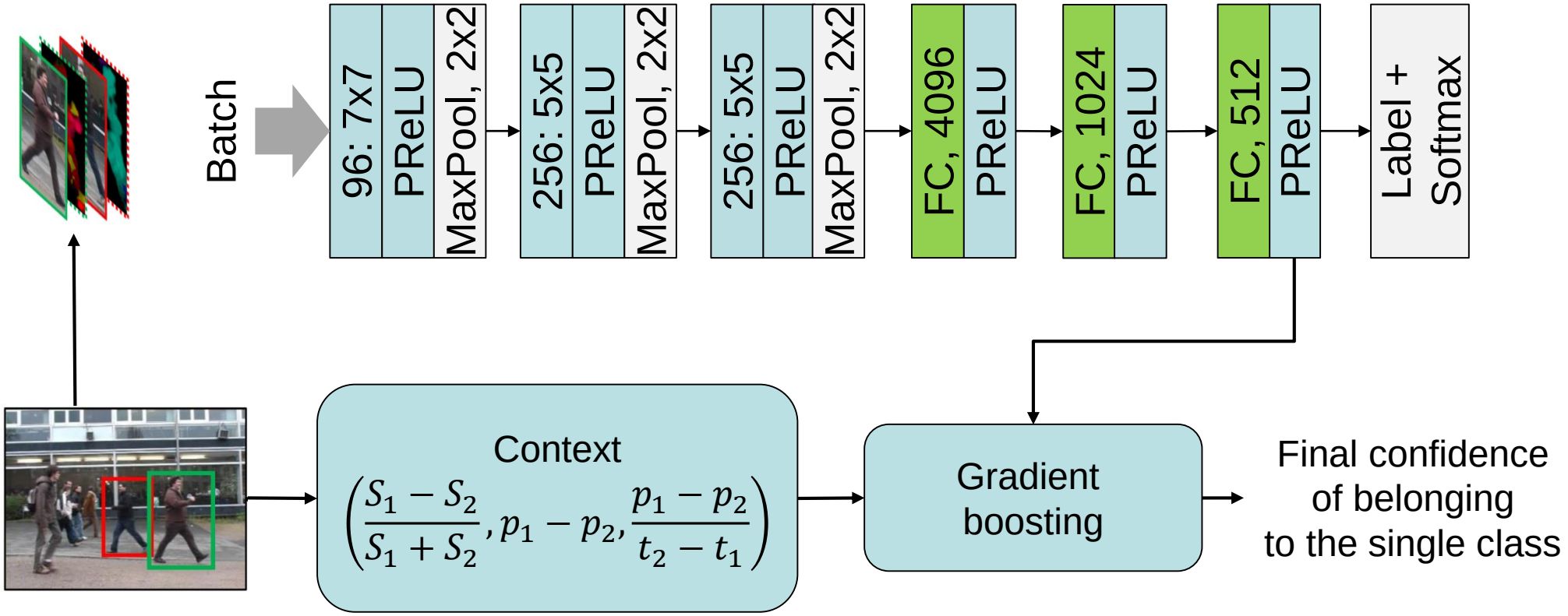
$$\frac{p_1 - p_2}{t_2 - t_1}$$

SiameseNET (3)

- SiameseNETs typically contain two branches with two inputs, each input is a set of images
- The developers of SiameseNET carried out experiments, and found that it is better to combine both branches of the Siamese network into a single one by combining the input images into a single batch

* Leal-Taixé L., Canton-Ferrer C., Schindler. K. Learning by tracking: siamese CNN for robust target association. – 2016. – [https://arxiv.org/pdf/1604.07866.pdf].

SiameseNET (4)



* Leal-Taixé L., Canton-Ferrer C., Schindler. K. Learning by tracking: siamese CNN for robust target association. – 2016. – [https://arxiv.org/pdf/1604.07866.pdf].

SiameseNET (5)

- The deep model receive images in LUV format, not in RGB
- The model input is a tensor of the shape 10x121x53, we get 10 channels by concatenating the following data:
  - Image of the object 1 in LUV format (3 channels)
  - Optical flow of the object 1 (2 channels)
  - Image of the object 2 in LUV format (3 channels)
  - Optical flow of the object 2 (2 channels)
- Optical flow is calculated using the Farnebäck method*

* Farnebäck G. Two-Frame Motion Estimation Based on Polynomial Expansion. – 2003. – [http://www.diva-portal.org/smash/get/diva2:273847/FULLTEXT01.pdf].

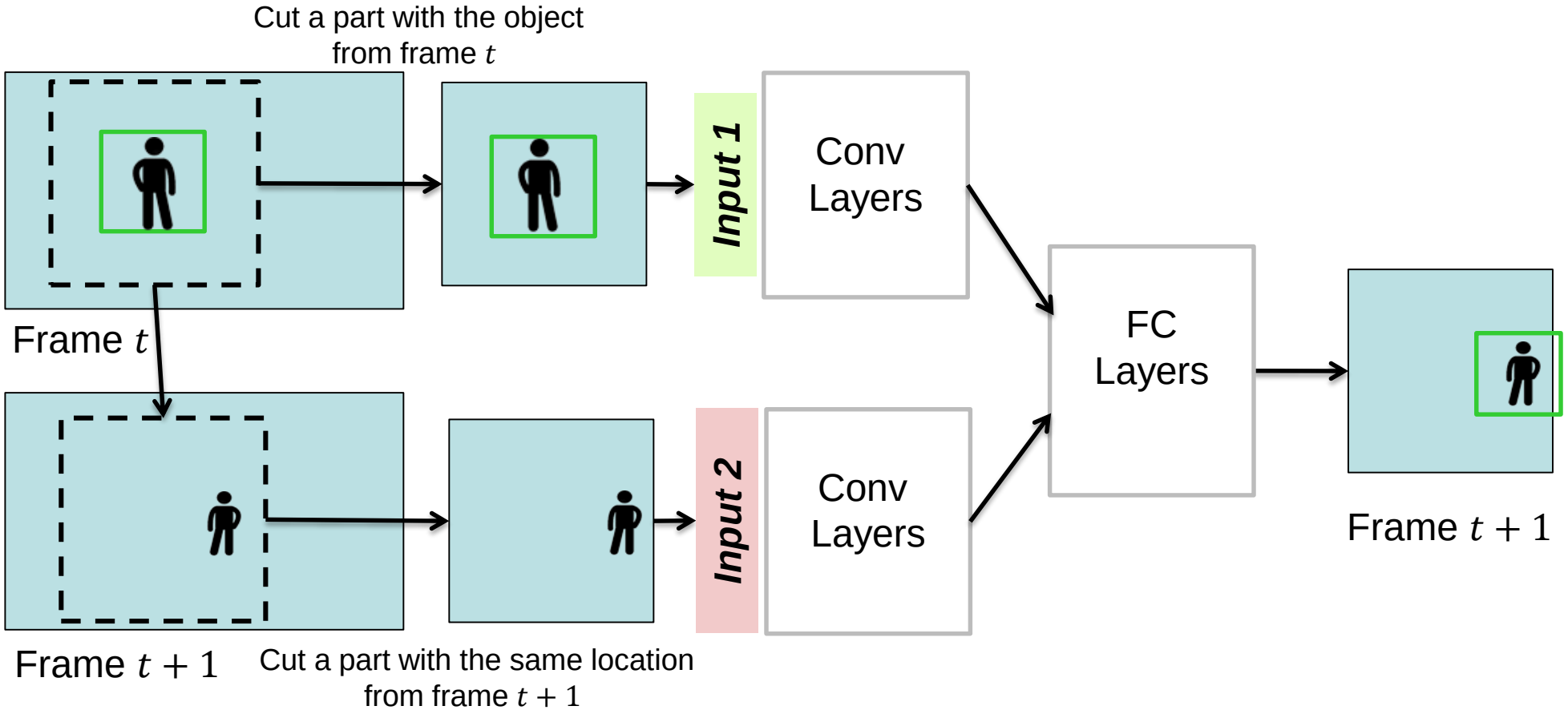
GOTURN (1)

- GOTURN (Generic Object Tracking Using Regression Network) is trained by comparing pairs of cropped frames:
  - Supposed the object location in the first frame is known, and the first frame and the second one are cropped to the size which in two times larger than the box bounded the object at the first frame, the cropped image is centered by the object center
  - Further, the algorithm predicts the location of the same object in the second frame
  - The convolutional neural network is trained to predict the location of the bounding box in the second frame

* Held D., Thrun S., Savarese S. Learning to track at 100 FPS with deep regression networks. – 2016. – [https://arxiv.org/pdf/1604.01802.pdf].

GOTURN (2)

- The model architecture:



* Held D., Thrun S., Savarese S. Learning to track at 100 FPS with deep regression networks. – 2016. – [https://arxiv.org/pdf/1604.01802.pdf].

GOTURN (3)

- The branch 1 receives the current frame with the object in the center; the image is cropped so that the object size is a half of the image size
- The branch 2 receives a new frame, but it is cropped just like the first one
- The model output is the bounding box in the second image

* Held D., Thrun S., Savarese S. Learning to track at 100 FPS with deep regression networks. – 2016. – [https://arxiv.org/pdf/1604.01802.pdf].

GOTURN (4)

- The GOTURN architecture is similar to CaffeNet (AlexNet) with a few changes:
  - GOTURN contains two branches of convolutional layers similar to the CaffeNet convolutional layers. The outputs of the convolutional layers of two branches are concatenated in front of the fully connected layers. The fully connected layers are identical to CaffeNet
  - The shape of the last fully connected layer changed from 1x1000 to 1x4 (the model output is a bounding box)
- GOTURN is a compact and fast convolutional network

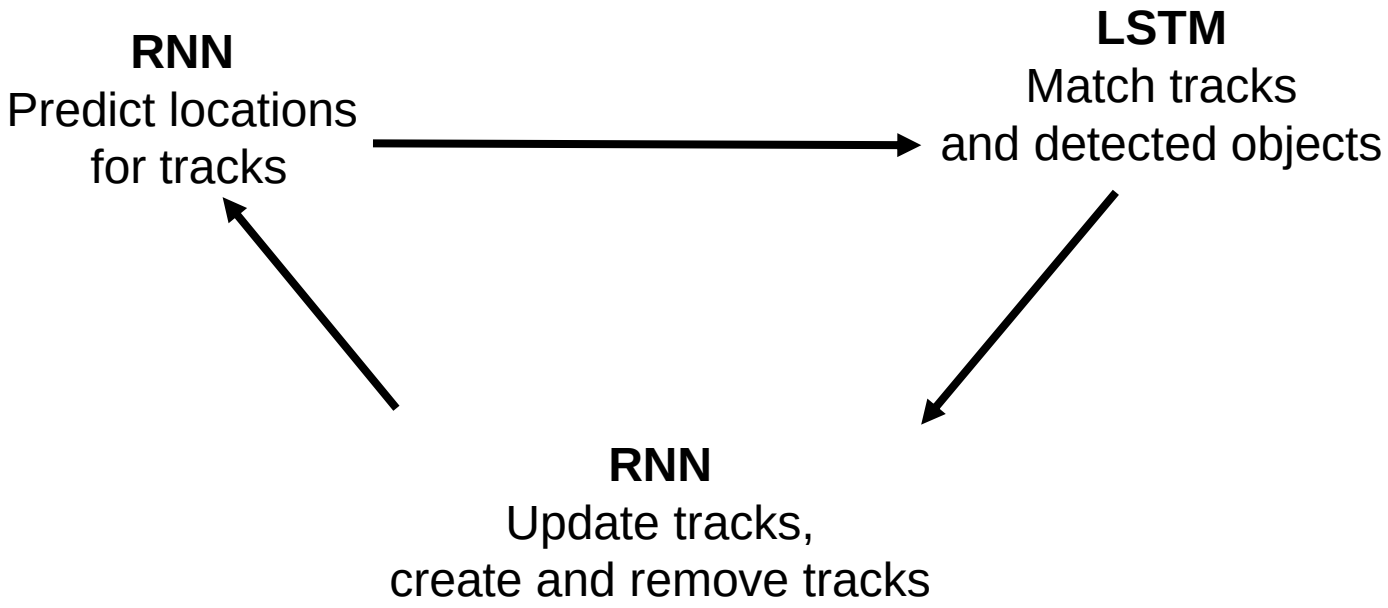
RNN-LSTM (1)

- The idea of this method is to implement object tracking using only deep neural networks
- Set of models with the LSTM architecture are used to calculate ***data association***, i.e. to find best matches between detected objects on the frame $t + 1$ and tracks on the frame t
- Model with the RNN architecture is used to predict object locations, update tracks, estimate the confidence of the objects existence, and create new tracks or remove old ones

* Milan A., Rezatofighi S.H., Dick A.R., et al. Online multi-target tracking using recurrent neural networks // AAAI Conference on Artificial Intelligence. – 2017. – P. 4225–4232. – [https://arxiv.org/pdf/1604.03635.pdf].

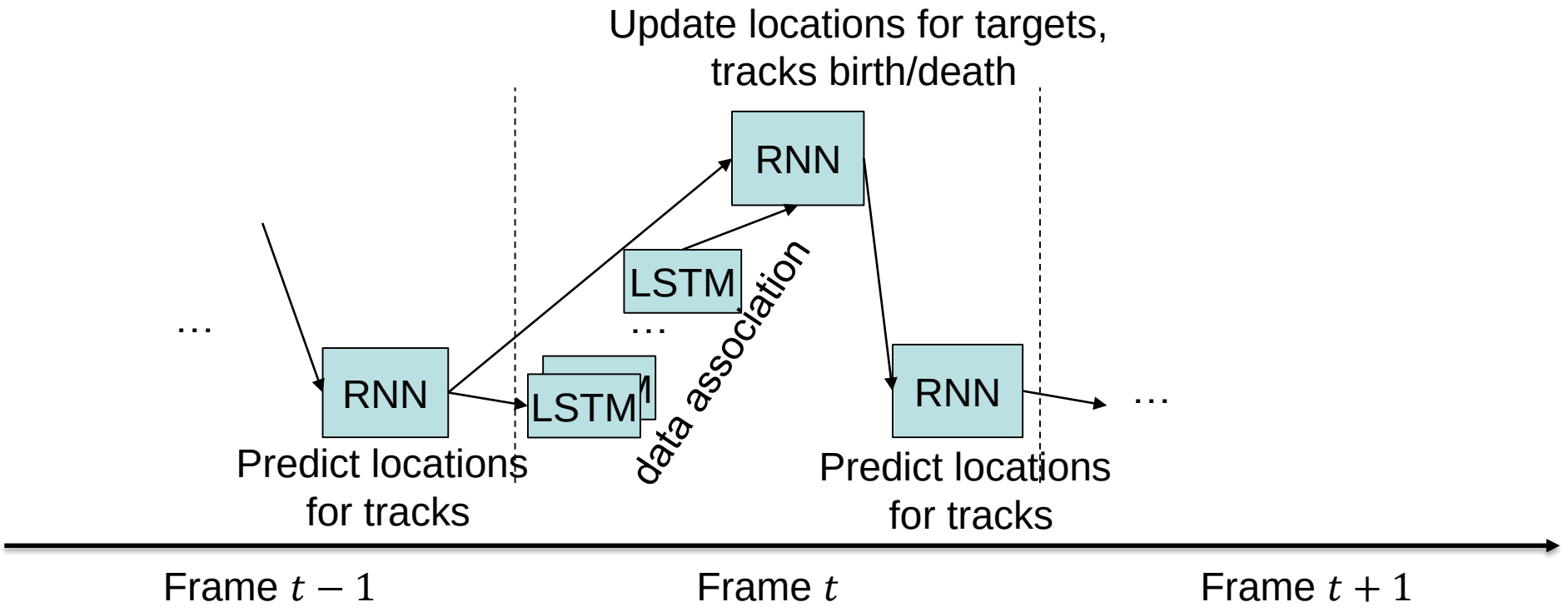
RNN-LSTM (2)

- The RNN output is used as the input of LSTM
- The LSTM output is used as input of RNN; therefore, there is a loop



RNN-LSTM (3)

- The algorithm pipeline:



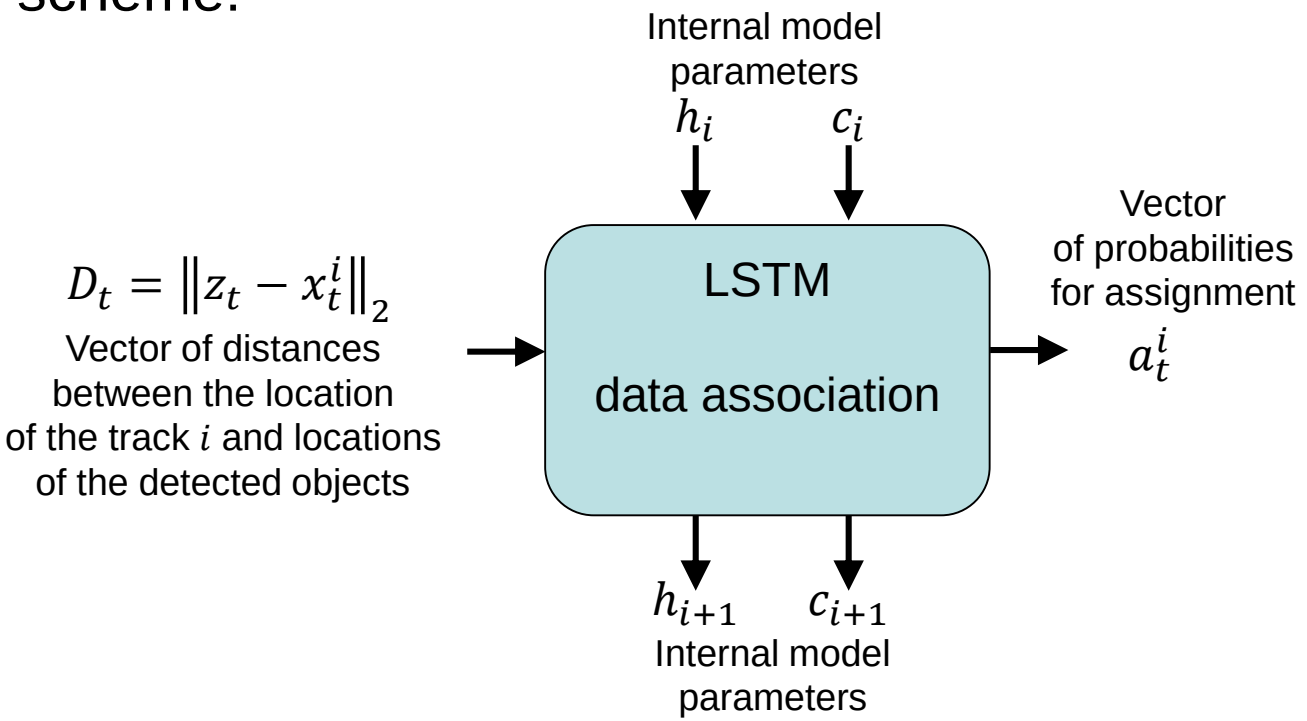
* Milan A., Rezatofighi S.H., Dick A.R., et al. Online multi-target tracking using recurrent neural networks. – 2017. – [https://arxiv.org/pdf/1604.03635.pdf], [https://dl.acm.org/doi/10.5555/3298023.3298181].

RNN-LSTM (4)

- The RNN model provides the following functionality:
  - Predict locations of tracks
  - Update locations of tracks using information about new detected objects
  - Create or remove tracks
- The LSTM model searches for the best matches between location prediction for tracks and bounding boxes corresponding to the detected objects using the matrix of distances between tracks and objects

RNN-LSTM (5)

- LSTM replaces greedy algorithms for matching tracks and detected objects, in particular, the Hungarian algorithm
- The single LSTM for the single track
- LSTM scheme:



RNN-LSTM (6)

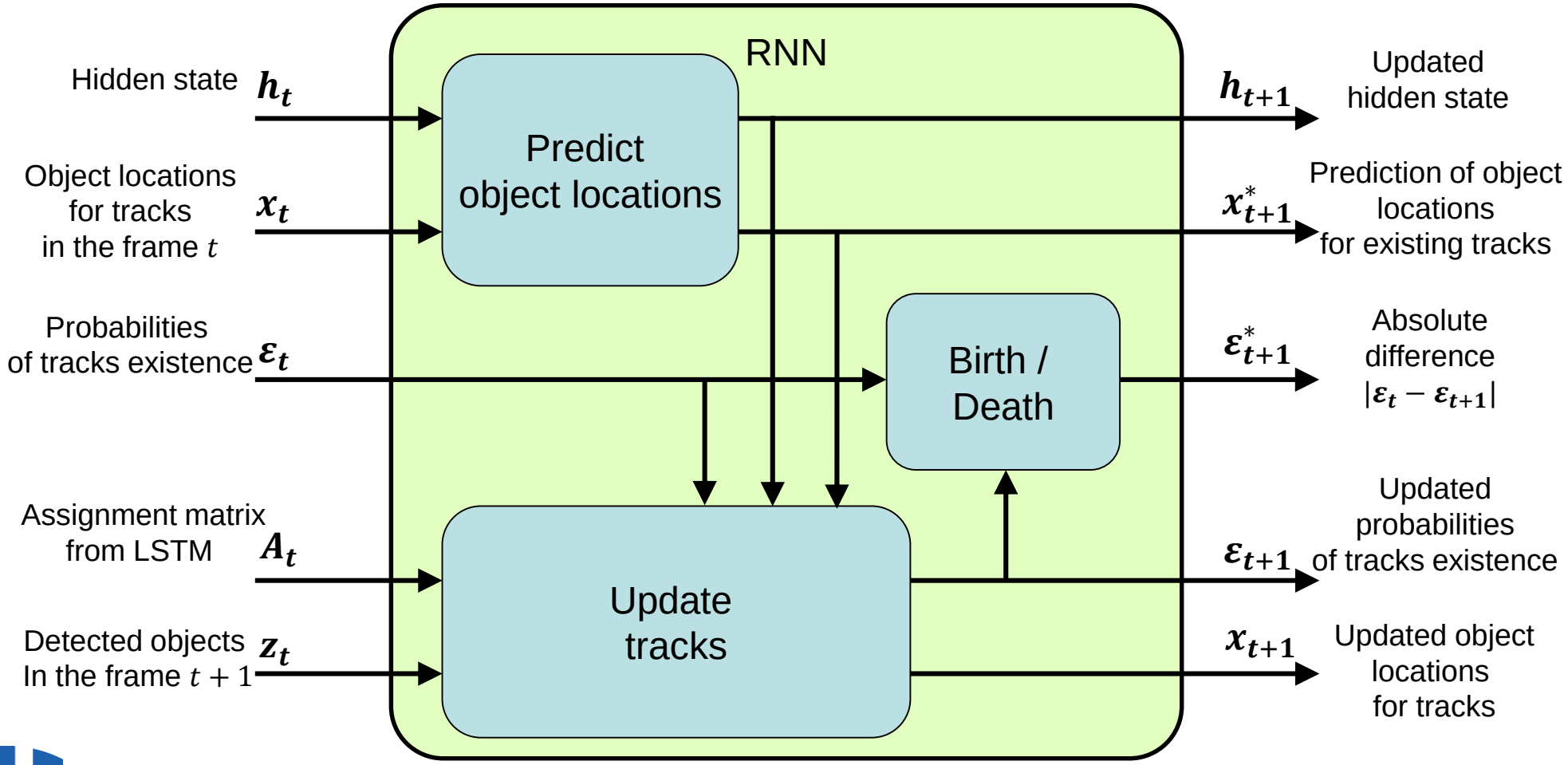
- The LSTM inputs:
  - LSTM state c_i
  - Hidden state h_i
  - Vector of distances between the location of the track i and the bounding boxes of the detected objects $D_t = \left\| z_t - x_t^i \right\|_2$
- The LSTM outputs:
  - LSTM state c_{i+1}
  - Hidden state h_{i+1}
  - The vector of probabilities for assignment between the tracks and objects on the new frame a_t^i

RNN-LSTM (7)

- The assignment matrix A_t is constructed, $A_t \in [0,1]^{N \times (M+1)}$, based on the probability vectors a_t^i of the LSTM models
  - Each element $a_t^{i,j}$ of the vector a_t^i is a probability that the object j belongs to the track i
  - The sum of the vector a_t^i elements equals 1
  - The number of rows of the matrix is N, where N is the number of tracks
  - The number of columns of the matrix is $M + 1$ (the number of detected objects + extra column for the situations when the probability is missing)
- The assignment matrix A_t is an input of the RNN model

RNN-LSTM (8)

- The RNN model:



RNN-LSTM (9)

- The algorithm:
  1. The block ”Predict object locations”. Object locations x_{t+1}^{*} for the tracks in the new frame are predicted using the locations x_t for the tracks in the current frame
  2. The block “Update tracks”. New locations x_{t+1} for the tracks and updated probabilities of the tracks existence ε_{t+1} are calculated based on the new detected objects z_t, the assignment matrix A_t, the current probabilities of the tracks existence ε_t and the hypotheses about new locations x_{t+1}^{*}. The matrix $A_t$is required to handle track occlusions and other ambiguous situations

RNN-LSTM (10)

- The algorithm:
  3. The block “Birth/Death”. The absolute difference $\varepsilon^*_{t+1} = |\varepsilon_t - \varepsilon_{t+1}|$ is calculated based on the probabilities of the tracks existence ε_t and the updated probabilities of the tracks existence ε_{t+1}. This difference is required for the model training to handle situations when the object detection is absent, but it is not supposed to complete object track since the object will appear on the next frame

RNN-LSTM (11)

- The RNN inputs:
  - Hidden state h_t
  - Object locations x_t for the tracks is a vector of bounding boxes (x, y, w, h) of the size N, where x, y are coordinates of the center of the bounding box, and w, h are width and height of the bounding box
  - Probabilities of tracks existence ε_t
  - Assignment matrix between the tracks and detections on the new frame $A_t \in [0,1]^{N \times (M+1)}$
  - Detected objects z_t in the new frame is a vector of bounding boxes of the size M. The vector is calculated using an object detector (for example, a deep model to solve the problem of object detection)

RNN-LSTM (12)

- The RNN outputs:
  - Updated hidden state h_{t+1}
  - Hypotheses about new object locations in the tracks x^*_{t+1}
  - Absolute difference $\varepsilon^*_{t+1} = |\varepsilon_{t+1} - \varepsilon_t|$
  - Updated probabilities of the tracks existence $\varepsilon_{t+1} \in (0,1)^N$. Inferring the model, if this value becomes less than 0.6, then the track terminates
  - Updated object locations of the tracks x_{t+1}, x_{t+1} is a vector of bounding boxes of the size N

Conclusion

- Currently, new object tracking algorithms are being developed, new large datasets are being collected and deep models are being developed
- The problem of object tracking cannot be considered completely solved
- However, software that solves the problem of object tracking for the specific object classes (cars, pedestrians, etc.) with high accuracy already exists and is used in industry, manufacturing, and retail

Literature (1)

- Milan A., et al. MOT16: A benchmark for multi-object tracking. – 2016. – [https://arxiv.org/pdf/1603.00831.pdf].
- Wojke N., Bewley A., Paulus D. Simple online and realtime tracking with a deep association metric // International Conference on Image Processing. – 2017. – P. 3645–3649. – [https://arxiv.org/pdf/1703.07402.pdf], [https://ieeexplore.ieee.org/document/8296962].
- Tao R., Gavves E., Smeulders A. Siamese instance search for tracking. – 2016. – [https://arxiv.org/pdf/1605.05863.pdf], [https://ieeexplore.ieee.org/document/7780527].
- Leal-Taixé L., Canton-Ferrer C., Schindler K. Learning by tracking: siamese CNN for robust target association. – 2016. – [https://arxiv.org/pdf/1604.07866.pdf], [https://ieeexplore.ieee.org/document/7789549].

Literature (2)

- Held D., Thrun S., Savarese S. Learning to track at 100 FPS with deep regression networks. – 2016. – [https://arxiv.org/pdf/1604.01802.pdf].
- Milan A., Rezatofighi S.H., Dick A.R., et al. Online multi-target tracking using recurrent neural networks. – 2017. – [https://arxiv.org/pdf/1604.03635.pdf], [https://dl.acm.org/doi/10.5555/3298023.3298181].
- Ciaparrone G., et al. Deep learning in video multi-object tracking: a survey. – 2019. – [https://arxiv.org/pdf/1907.12740.pdf].

Authors

- **Turlapov Vadim Evgenievich,** Dr., Prof., department of computer software and supercomputer technologies
  vadim.turlapov@itmm.unn.ru
- **Vasiliev Engeny Pavlovich,** lecturer, department of computer software and supercomputer technologies
  evgeny.vasiliev@itmm.unn.ru
- **Getmanskaya Alexandra Alexandrovna,** lecturer, department of computer software and supercomputer technologies
  alexandra.getmanskaya@itmm.unn.ru
- **Kustikova Valentina Dmitrievna**
  Phd, assistant professor, department of computer software and supercomputer technologies
  valentina.kustikova@itmm.unn.ru