Person Re-Identification, Face Recognition and Tracking/Clustering

A presentation by Abhijn, Atharva, Devinesh and Nishant

Problem Statement

We wanted to cluster images by recognising the persons in the image.

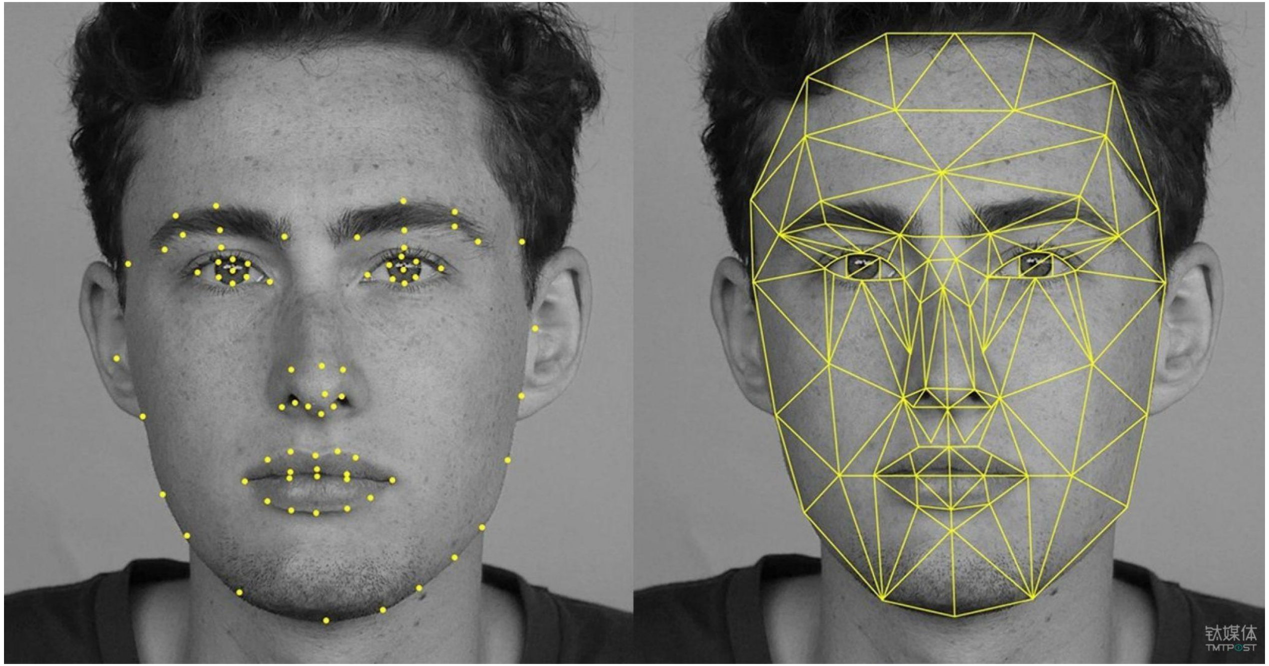
This problem deals with issues of Facial recognition, Person re-identification and Tracking objects through frames and images.

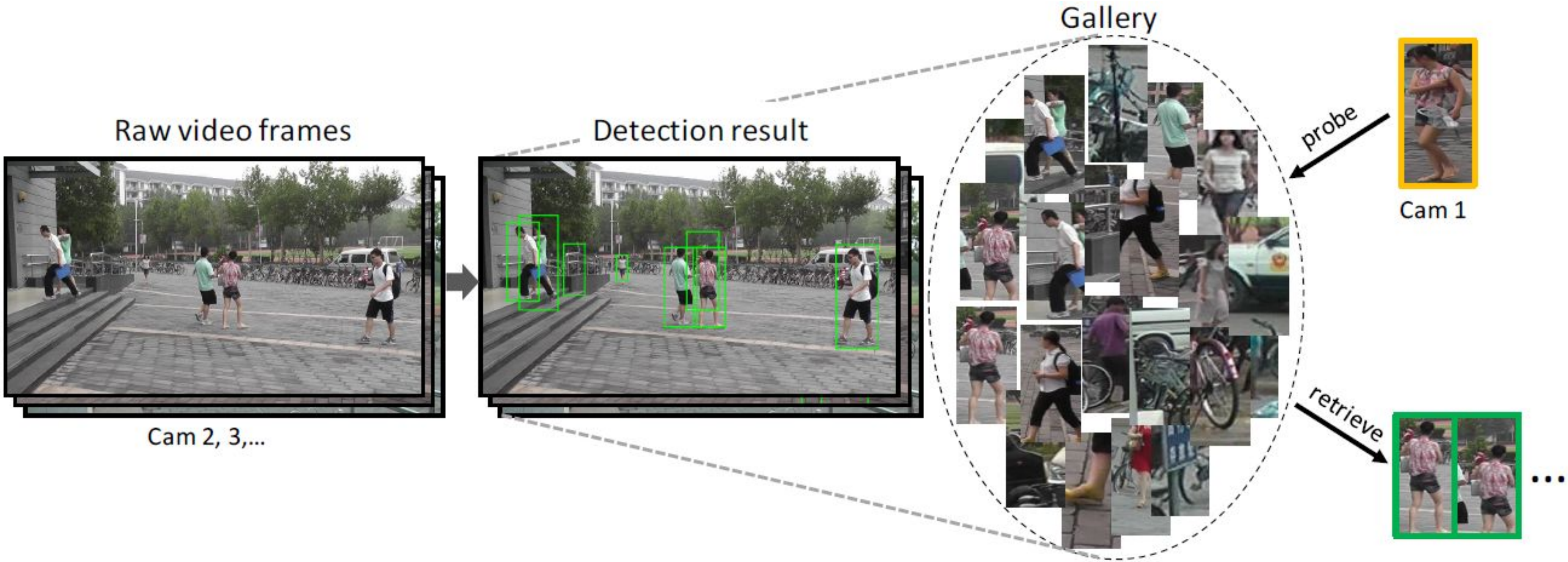
In today’s presentation we are going to go through some state of the art methods on the above mentioned topics.

Literature Survey

Person Re-identification

Facial Recognition?





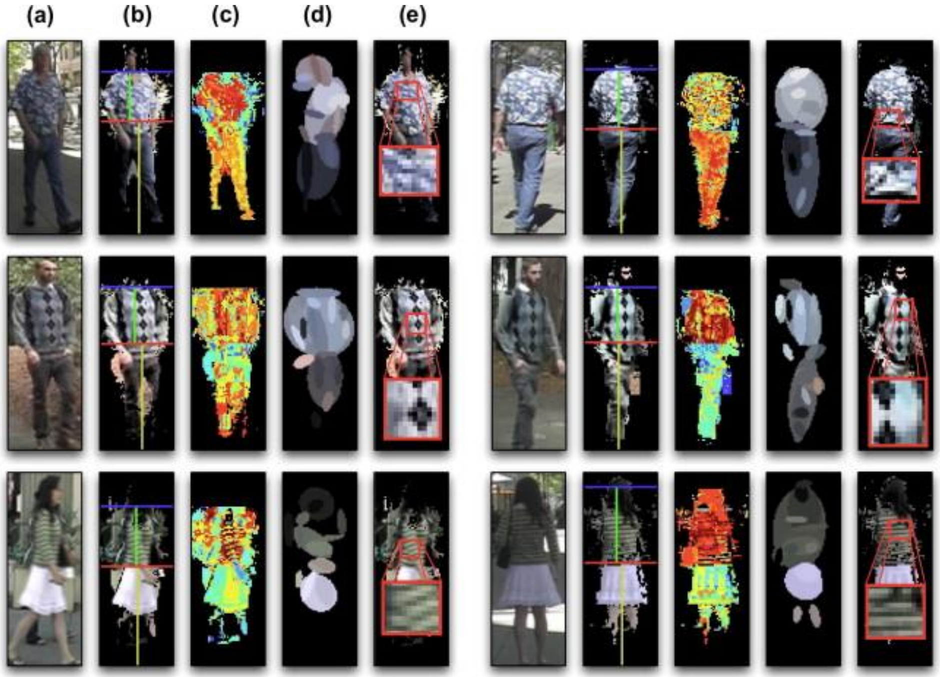
(a) Pedestrian Detection

(b) Person Re-identification

Classical Approaches

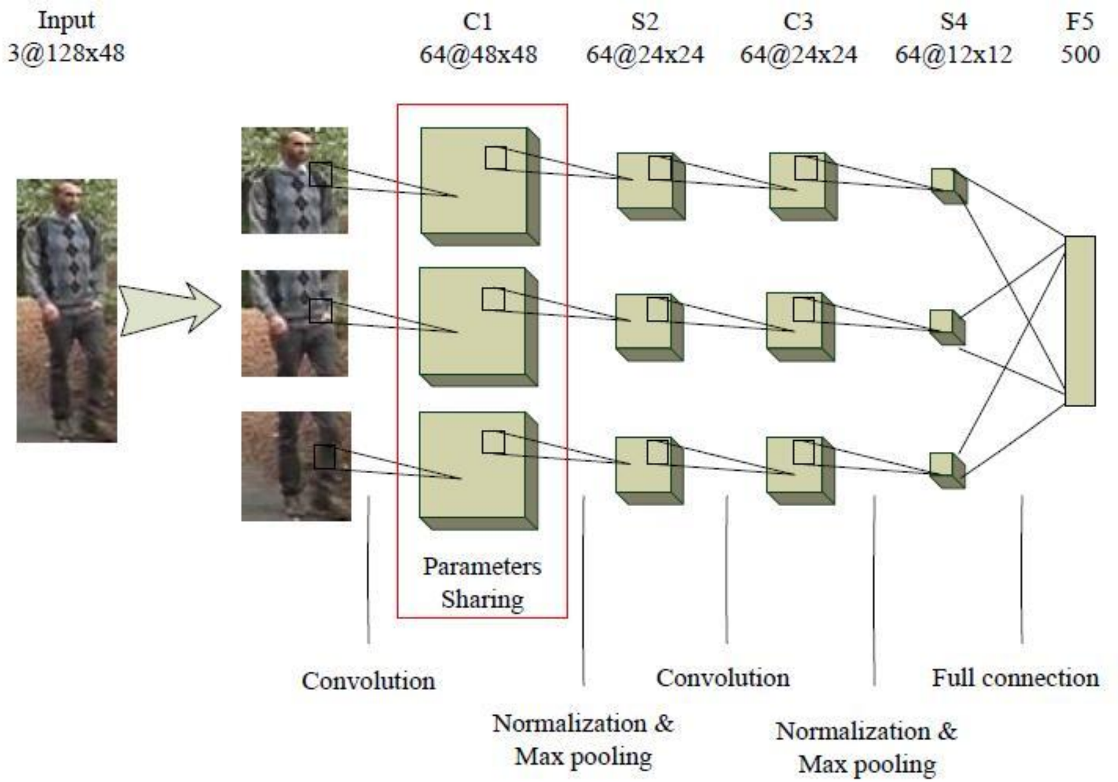
- Feature Definition
- Metric Learning

Classical Approach: SDALF



Symmetry-driven part-based feature accumulation

Deep Learning: Feature Representation

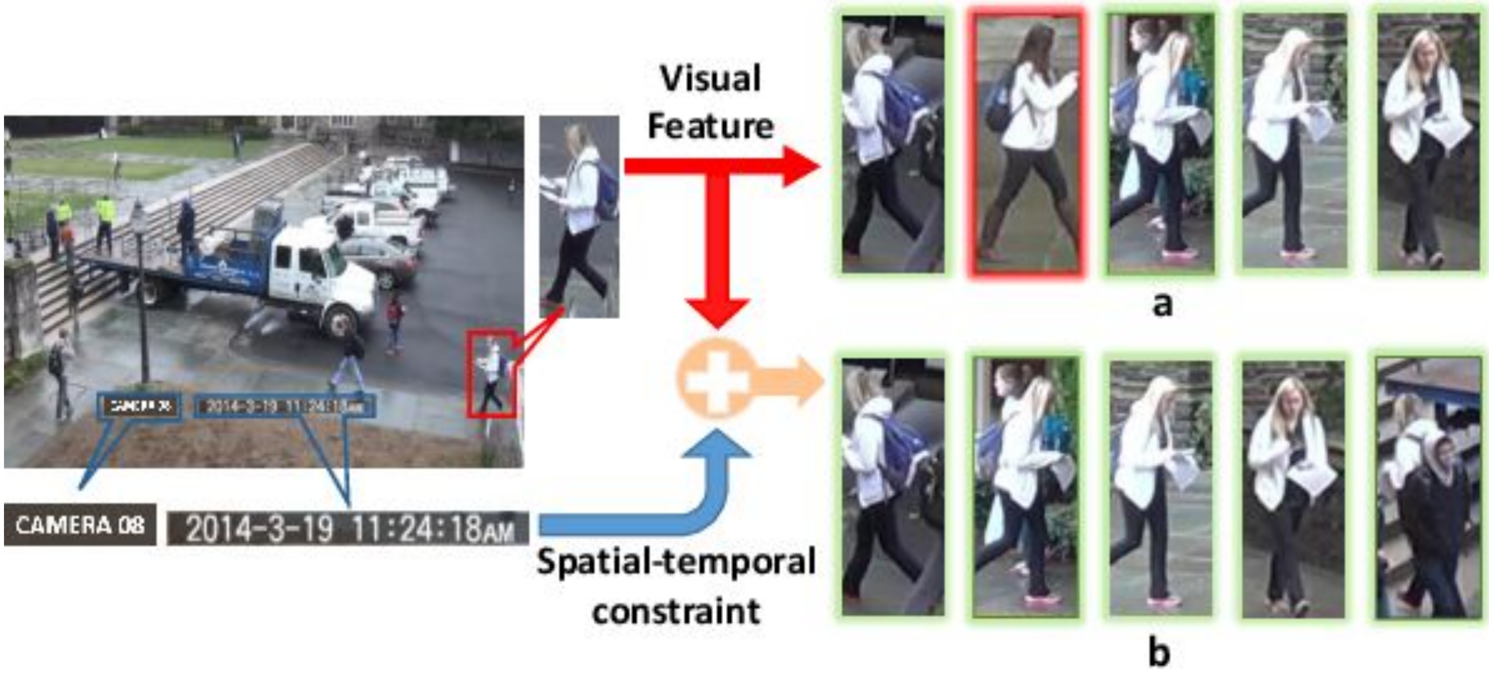


Challenges

(a)

(b)

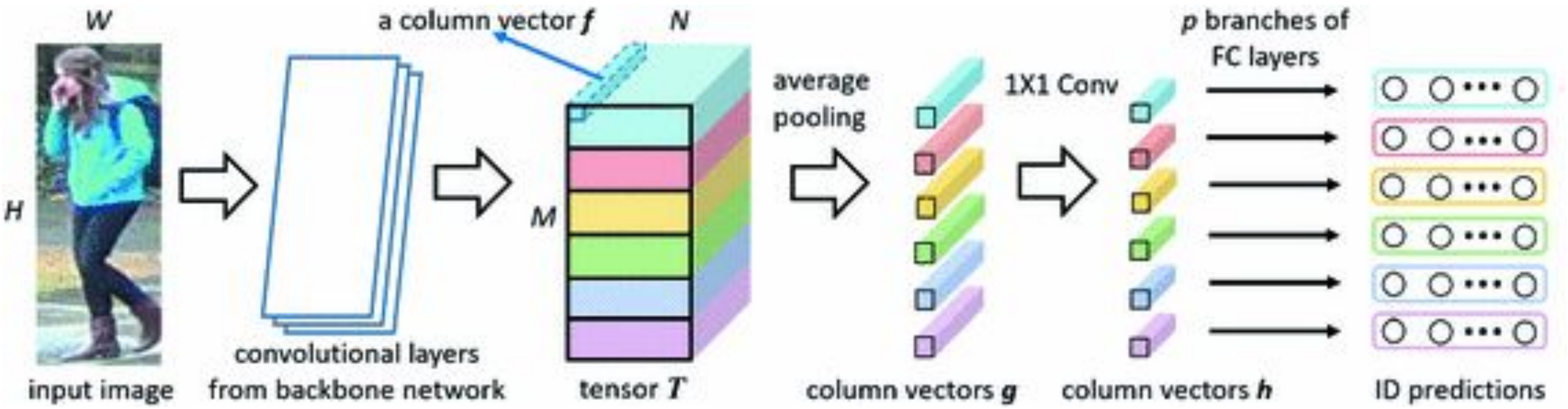
Spatial-Temporal Person Re-identification (st-reID)



Components of st-reID

- Visual Feature Stream
- Spatial-Temporal Stream
- Joint Metric

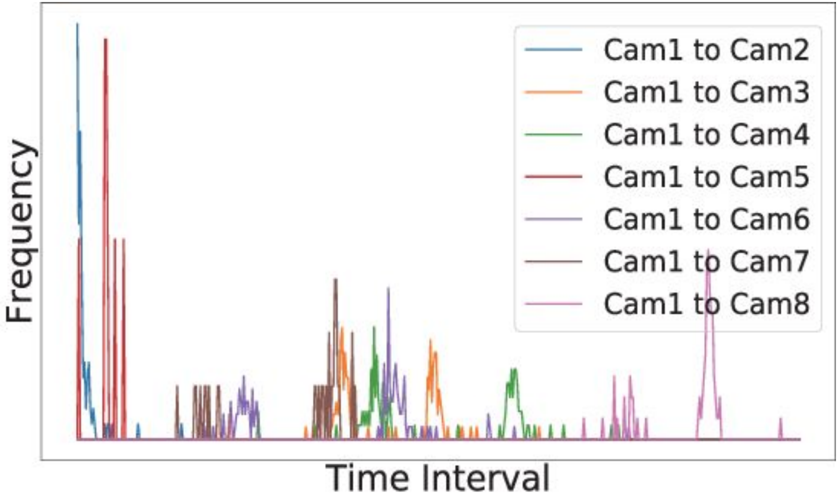
Visual Feature Stream



$$s(\mathbf{x_i}, \mathbf{x_j}) = \frac{\mathbf{x_i} \cdot \mathbf{x_j}}{||\mathbf{x_i}||\,||\mathbf{x_j}||}$$

Spatial-Temporal Stream

(a)



(b)

Spatial-Temporal Stream

$$\hat{p}(y=1|k,c_i,c_j)=\frac{n^k_{c_ic_j}}{\sum_l n^l_{c_ic_j}}$$

After Parzen Window method:

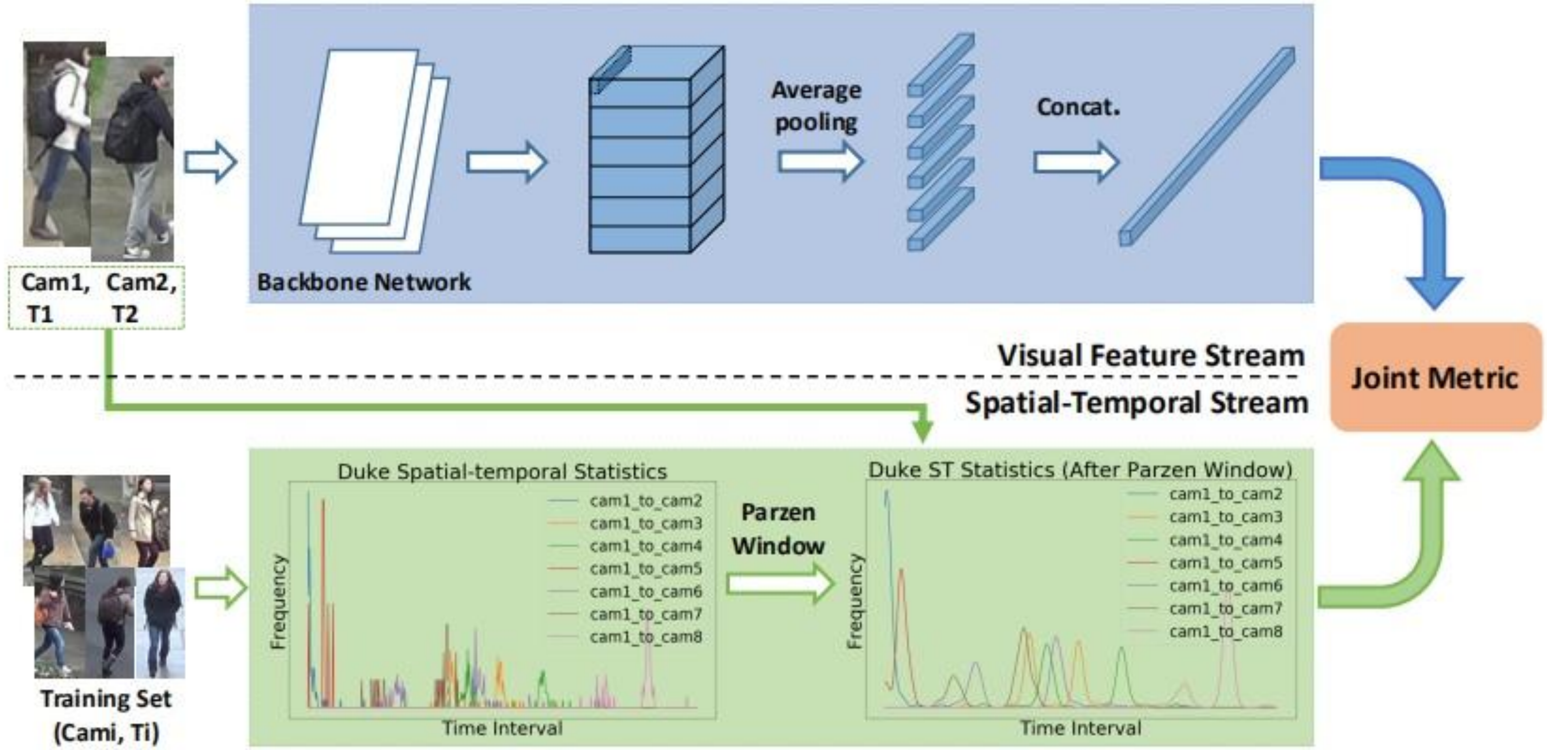
$$p(y=1|k,c_i,c_j)=\frac{1}{Z}\sum_l \hat{p}(y=1|l,c_i,c_j)K(l-k)$$

Joint Metric

$$p(y=1|\mathbf{x_i},\mathbf{x_j},k,c_i,c_j)=s(\mathbf{x_i},\mathbf{x_j})p(y=1|k,c_i,c_j)$$

$$p_{joint}=f(s;\lambda_0,\gamma_0)f(p_{st};\lambda_1,\gamma_1)$$

Two-Stream Architecture



Performance

Market-1501 Dataset

Methods	R-1	R-5	R-10	mAP
BoW+kissme	44.4	63.9	72.2	20.8
KLFDA	46.5	71.1	79.9	-
Null Space	55.4	-	-	29.9
WARCA	45.2	68.1	76.0	-
PAN	82.8	-	-	63.4
SVDNet	82.3	92.3	95.2	62.1
HA-CNN	91.2	-	-	75.7
SSDAL	39.4	-	-	19.6
APR	84.3	93.2	95.2	64.7
Human Parsing	93.9	98.8	99.5	-
Mask-guided	83.79	-	-	74.3
Background	81.2	94.6	97.0	-
PDC	84.1	92.7	94.9	63.4
PSE+ECN	90.3	-	-	84.0
MultiScale	88.9	-	-	73.1
Spindle Net	76.9	91.5	94.6	-
Latent Parts	80.3	-	-	57.5
Part-Aligned	81.0	92.0	94.7	63.4
PCB(*)	91.2	97.0	98.2	75.8
TFusion-sup	73.1	86.4	90.5	-
st-ReID	97.2	**99.3**	99.5	86.7
st-ReID+RE	**98.1**	**99.3**	**99.6**	87.6
st-ReID+RE+re-rank	98.0	98.9	99.1	**95.5**

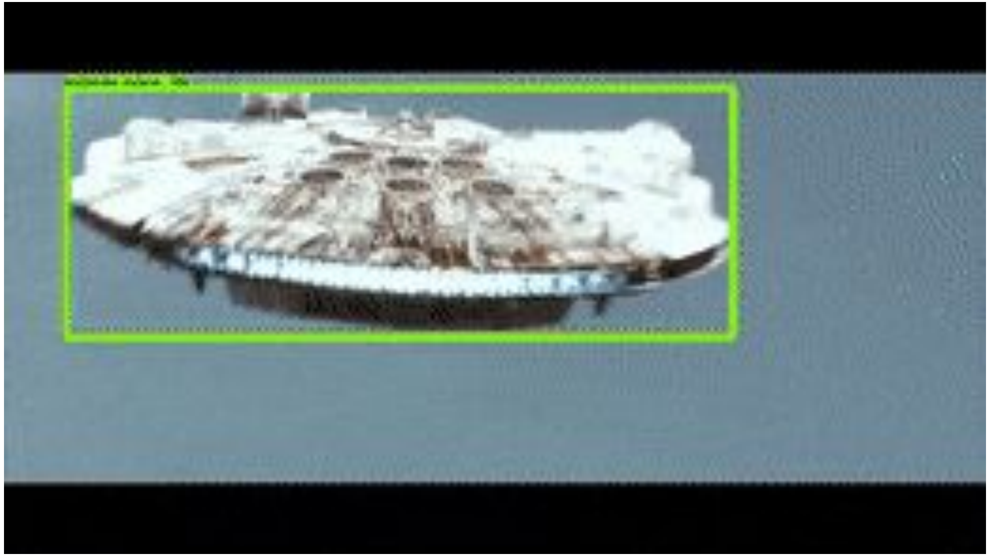
DukeMTMC-reID Dataset

Methods	R-1	R-5	R-10	mAP
BoW+kissme	25.1	-	-	12.2
LOMO+XQDA	30.8	-	-	17.0
PAN	71.6	-	-	51.5
SVDNet	76.7	-	-	56.8
HA-CNN	80.5	-	-	63.8
APR	70.7	-	-	51.9
Human Parsing	84.4	91.9	93.7	71.0
PSE+ECN	85.2	-	-	79.8
MultiScale	79.2	-	-	60.6
PCB(*)	83.8	91.7	94.4	69.4
st-ReID	94.0	97.0	97.8	82.8
st-ReID+RE	94.4	**97.4**	**98.2**	83.9
st-ReID+RE+re-rank	**94.5**	96.8	97.1	**92.7**

Object Tracking

Tracking

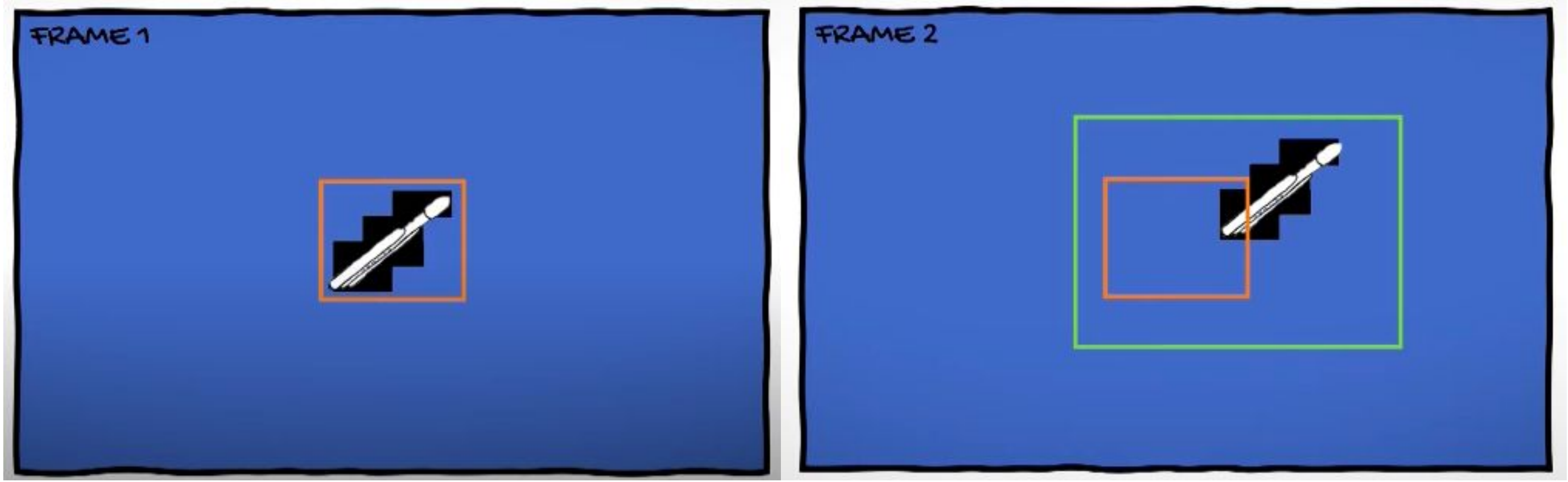
- Tracking objects through frames of a video



Mean Shift Tracking

- A video can be considered a set of images getting displayed in a sequence.
- We can cluster the pixels of high intensity of the object of interest and bound them using a bounding box. - IN ONE FRAME.
- Iteratively, we can get to the next frame, apply mean shift to the **neighbouring area** surrounding our previous bounding box and see if the mean of pixels around our bounding box has moved.
- If the mean has moved, we'll move the bounding box to the new mean.
- Iteratively doing it frame by frame will actually track your object of interest around the video.

Mean Shift Tracking

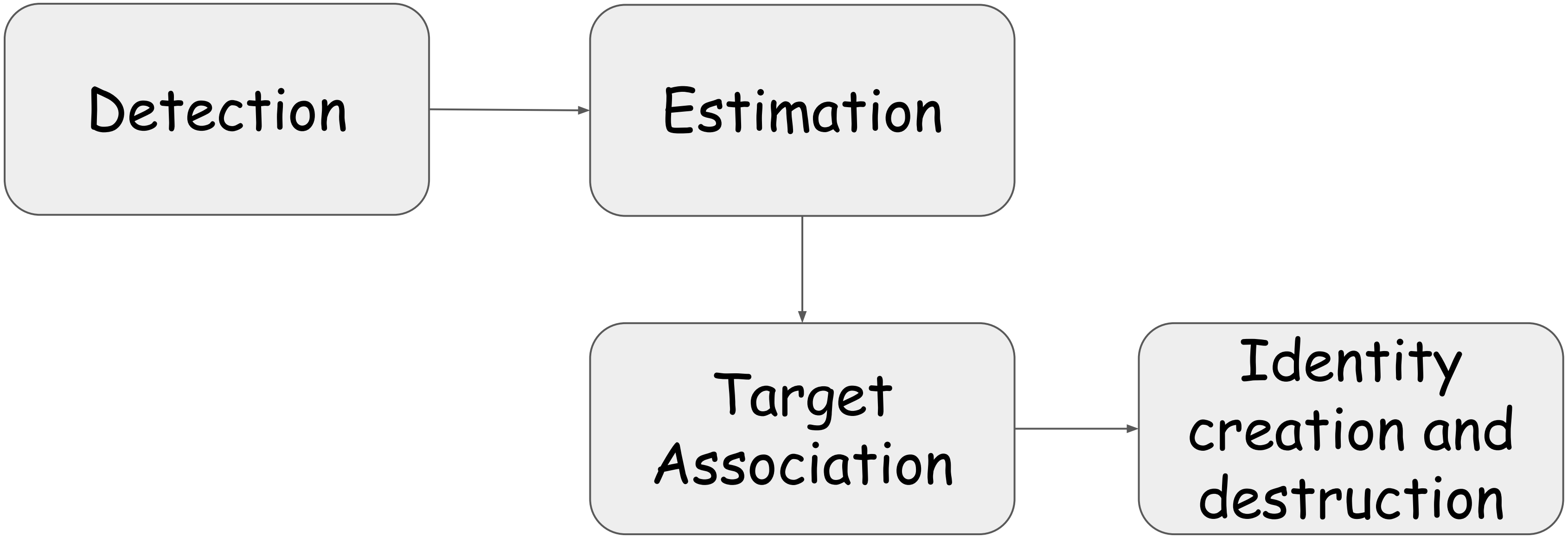


Mean Shift Tracking

- Issues with meanshift
- 1) Doesnt identify anything, has no context about the object
- 2) follows pixel density around the video, so if there is occlusion, it'll go haywire.
- 3) Cannot track multiple objects at once
- 4) Too sensitive to accelerating objects and window size that we give to track density.

State of the art - SORT

- SORT or Simple online realtime tracking algorithm
- SORT comprises of 4 components :



Detection

Faster Region CNN

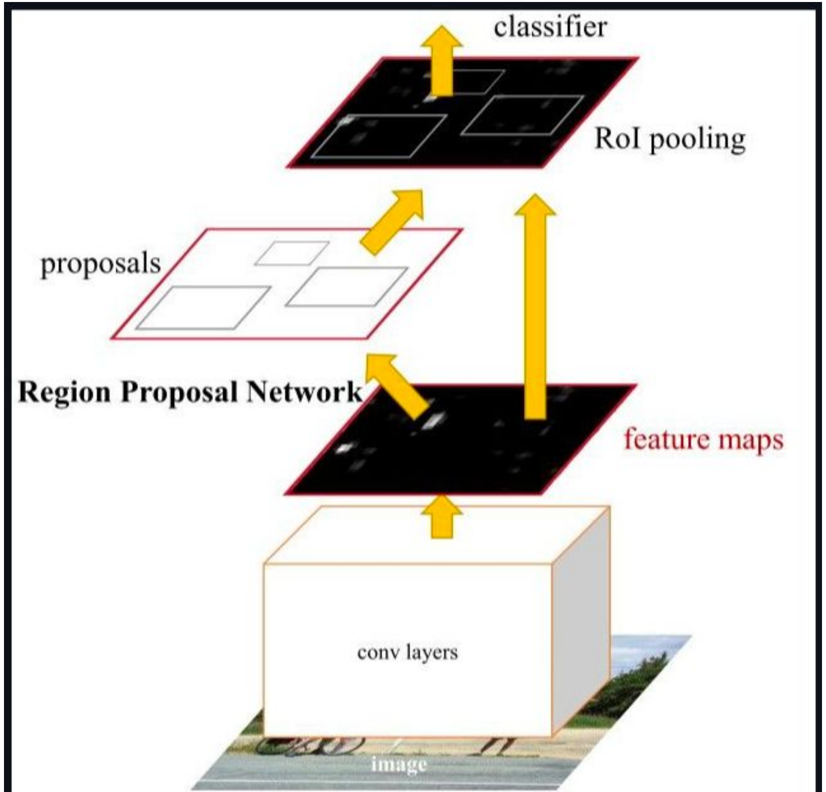
Detection

Faster region CNN

- A combination of Region Proposal Network and its predecessor Fast R CNN.
- Fast RCNN used to detect objects in an image in around 2 seconds per image.
- With the addition of Region Proposal network - Faster RCNN can detect objects in under 10 ms.
-

Detection

Faster region CNN



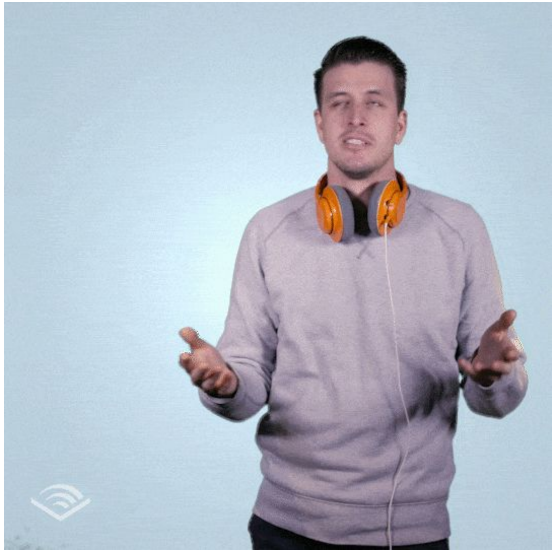
Detection

Faster region CNN

- The RPN generates region proposals.
- For all region proposals in the image, a fixed-length feature vector is extracted from each region using the ROI Pooling layer
- The extracted feature vectors are then classified using the Fast R-CNN.
- The class scores of the detected objects in addition to their bounding-boxes are returned.
-
-

Estimation

Kalman Filter



Estimation

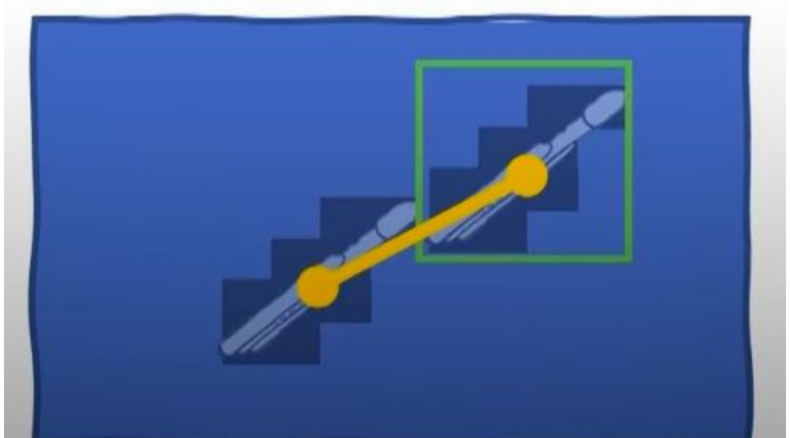
Kalman Filter

- Estimation is the process of propagating the detected object from current frame to next frame.
- The position of the object in the next state can be estimated based on one of the two methods
- Either by using a constant velocity model or some gaussian estimation of sensor data (any sensor which we are using in tracking)

Estimation

Kalman Filter

- In simple words kalman filter gives more weight to either of the estimation modes, depending on the conditions.
- For example, if there is occlusion in front of detected object, the kalman filter will give more weight to linear velocity model to predict the position of object in next frame.

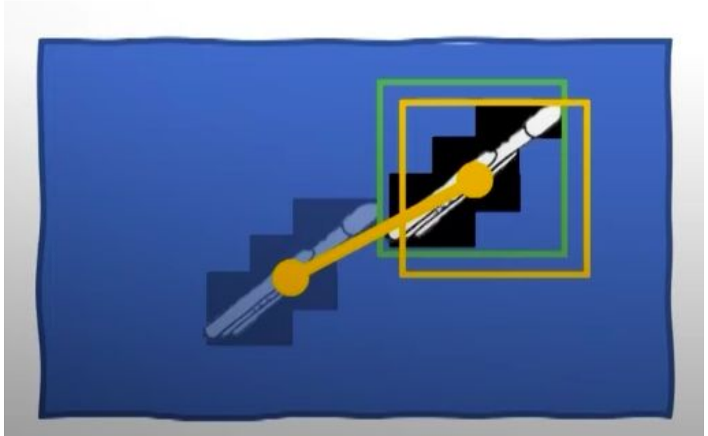


Target
Association

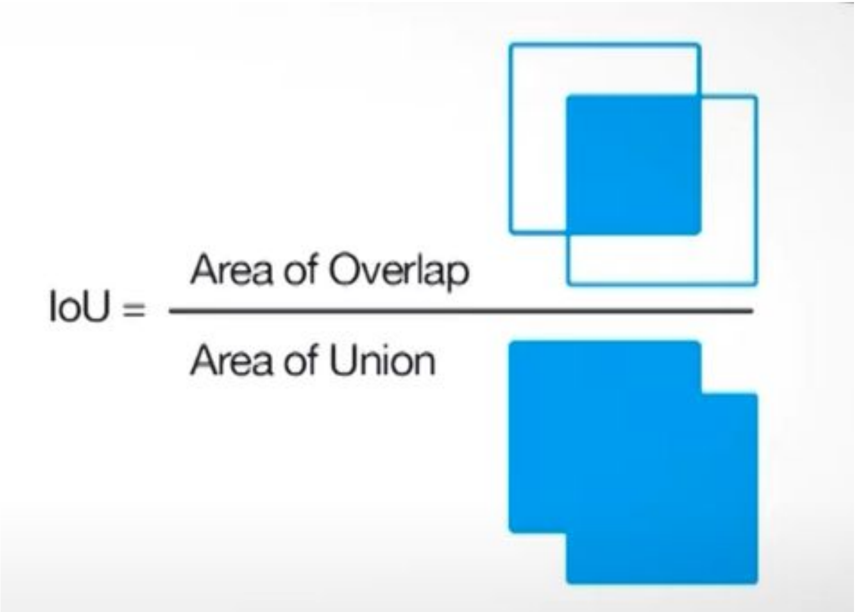
IOU and Hungarian Algorithm

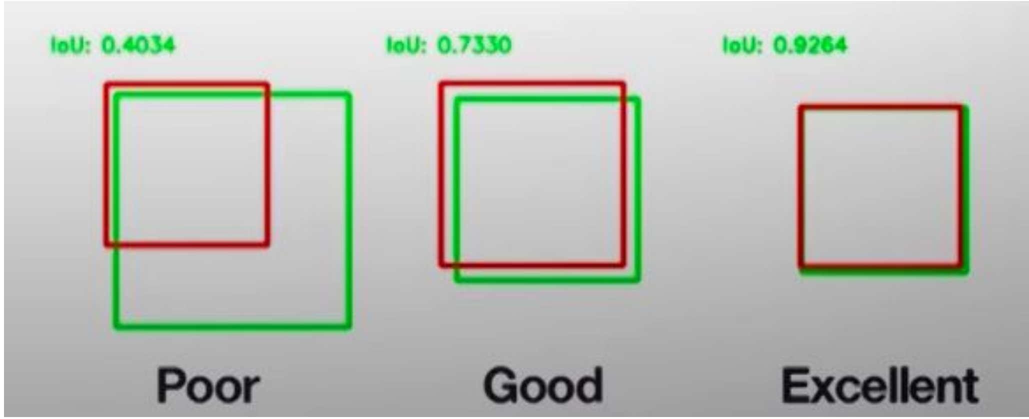
Target Association

- In this step, we try to associate the estimated position of the object from the previous frame, to the object detected in the current frame.
- We take the bounding box of propagated estimated object with the bounding box of new detected object and calculate IOU.



Target Association





Target Association

Using hungarian algo

- These IOU associations are solved by hungarian algorithms.
- If we are getting high overlap between predicted bounding box and detected bounding box, that means the object is same between the previous frame and the current frame.
- We can associate the same identity with both the objects.

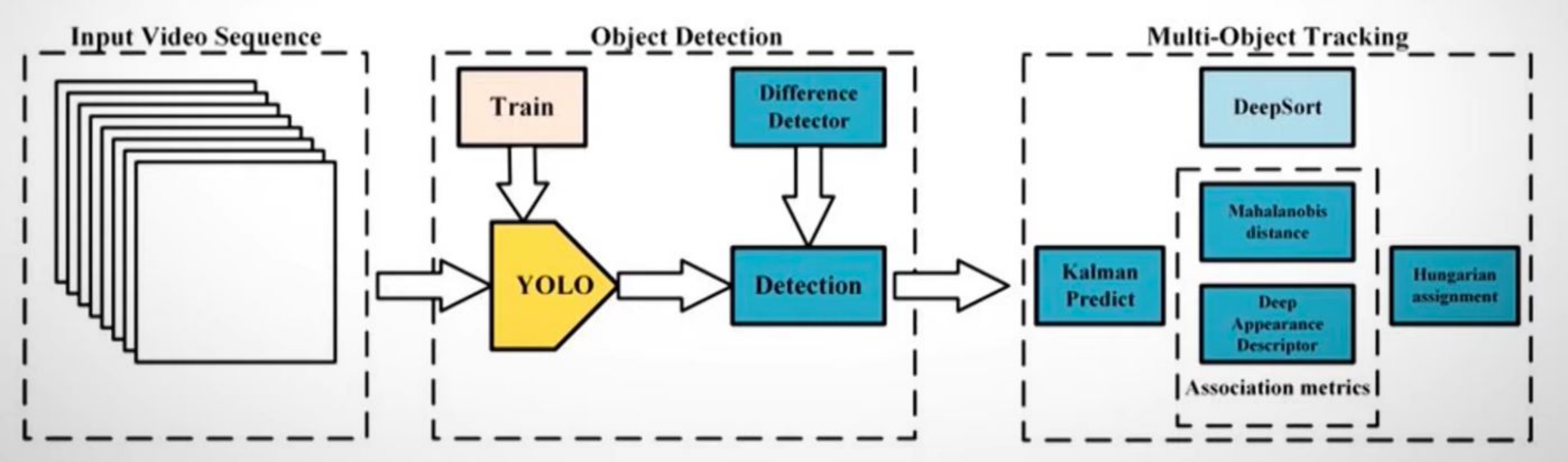
Identity creation and destruction

- When objects enter and leave the frame, unique identities must be created and destroyed accordingly.
- To create new identity we take any object which has less than ideal IOU, signifying it was not present in previous frame.
- Any lost association for more than t loss frames, this identity will be deleted.

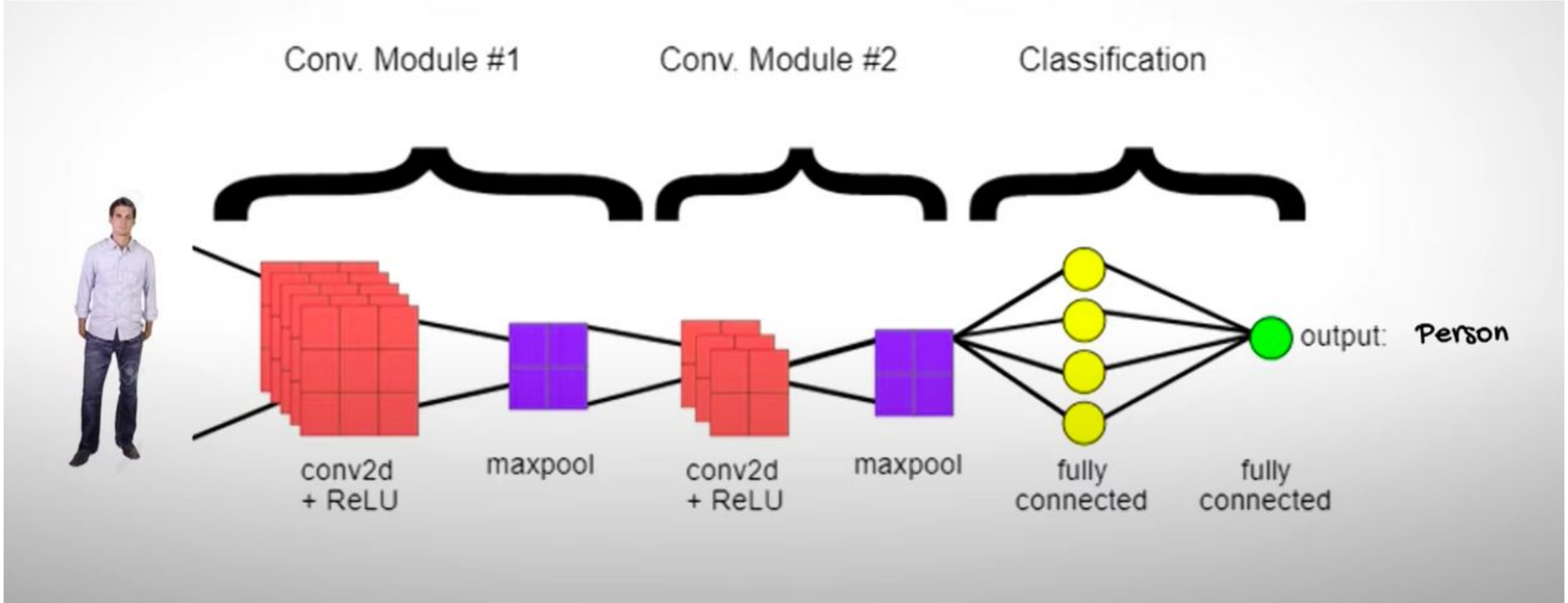
SORT → Deep SORT

Deep SORT

- Simple Online Realtime Tracking using Deep Association Metric.

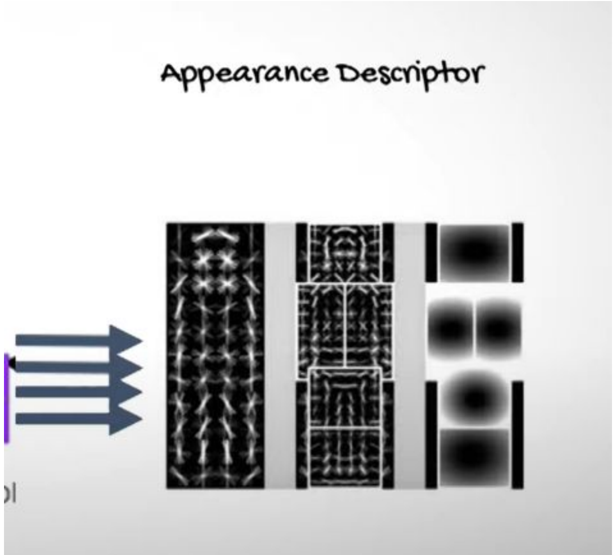


Deep SORT

- To improve the quality of detection and association, authors of deep sort added another distance metric which is called Deep Appearance Descriptor.
- 

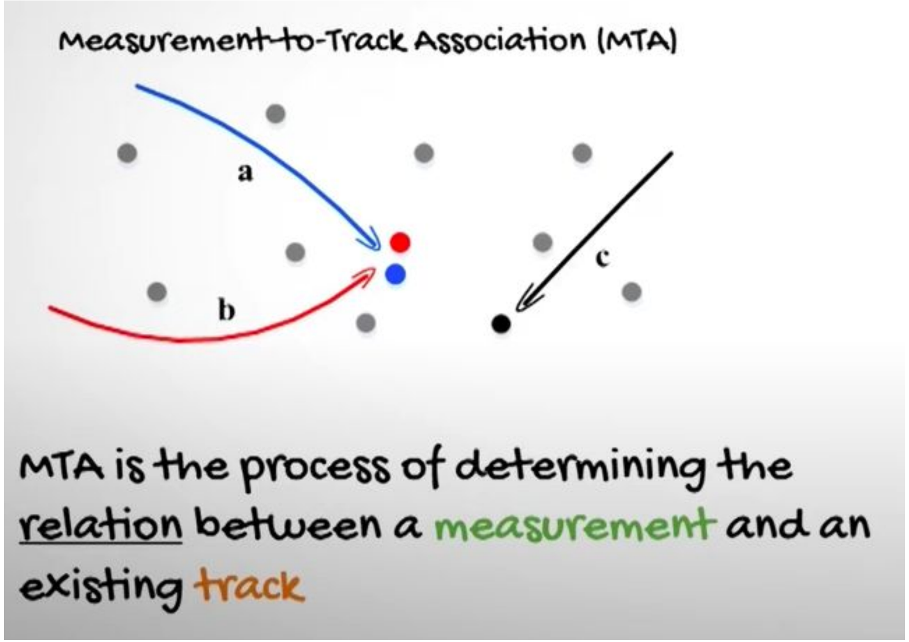
Deep SORT

- The deep appearance descriptor is another CNN which is trained meticulously on the data sets to achieve high accuracies,
- Once that is done, we strip away everything and we just take the fully connected dense layer which contains the features or “Appearance Descriptor”
-



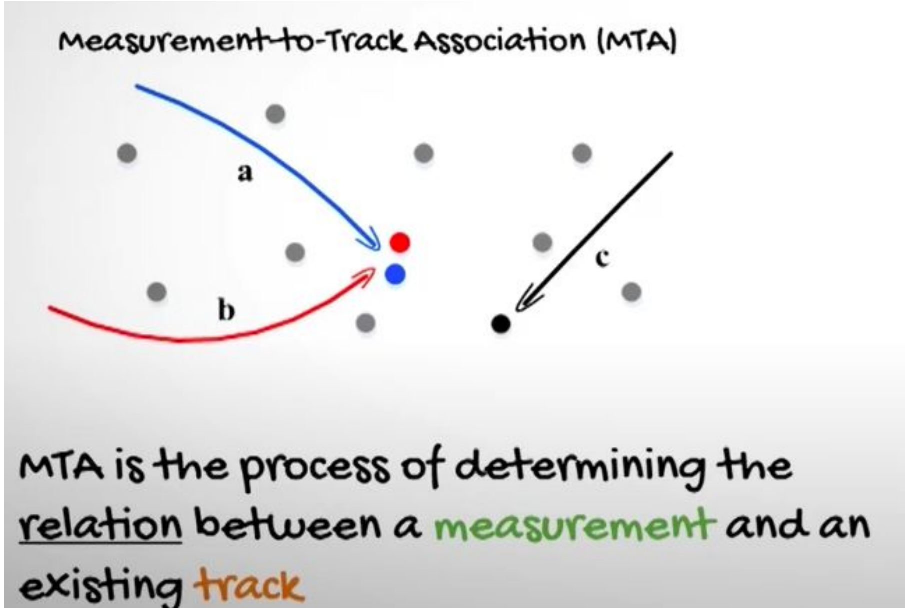
Deep SORT

- This appearance features are used while assigning associations by a method called Measurement to Track Association or MTA
-
-



Deep SORT

- The authors use k nearest neighbour algorithm to associate the identity of tracked object so far and the detected object using the mahalanobis distance as the distance metric.



Deep SORT

- The number of identity switches has gone down by 45% by the use of deep association metric.
- Also the performance of sort has improved through occlusions by its improvement in deep sort.

		MOTA ↑	MOTP ↑	MT ↑	ML ↓	ID ↓	FM ↓	FP ↓	FN ↓	Runtime ↑
KDNT [16]*	BATCH	68.2	79.4	41.0%	19.0%	933	1093	11479	45605	0.7 Hz
LMP_p [17]*	BATCH	**71.0**	**80.2**	**46.9%**	21.9%	434	**587**	7880	**44564**	0.5 Hz
MCMOT_HDM [18]	BATCH	62.4	78.3	31.5%	24.2%	1394	1318	9855	57257	35 Hz
NOMTwSDP16 [19]	BATCH	62.2	79.6	32.5%	31.1%	**406**	642	**5119**	63352	3 Hz
EAMTT [20]	**ONLINE**	52.5	78.8	19.0%	34.9%	910	**1321**	**4407**	81223	12 Hz
POI [16]*	**ONLINE**	**66.1**	79.5	**34.0%**	20.8%	805	3093	5061	**55914**	10 Hz
SORT [12]*	**ONLINE**	59.8	**79.6**	25.4%	22.7%	1423	1835	8698	63245	**60 Hz**
Deep SORT (Ours)*	**ONLINE**	61.4	79.1	32.8%	**18.2%**	**781**	2008	12852	56668	40 Hz

Facial Recognition

FaceNet: A Unified Embedding for Face Recognition
and Clustering

Introduction

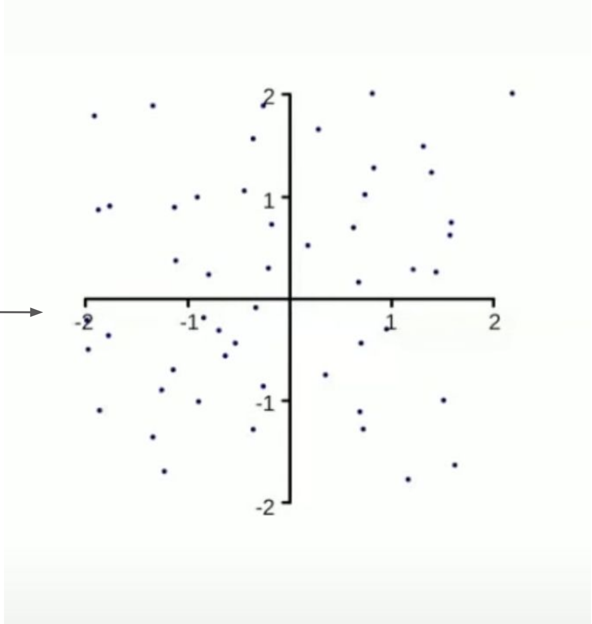
- FaceNet, used for recognition, verification and clustering
- Uses Deep Convolutional Networks with triplet loss
- Provides state of the art accuracy
- Generates embeddings for each images, uses these embeddings for all the related tasks
- Do not define any new algorithms, create embeddings and then use them for any task
- Squared L2 distance is used

How it works?

Input Image

1.205
0.125
0.009
1.345
0.255
1.988

Embedding (128 Numbers)

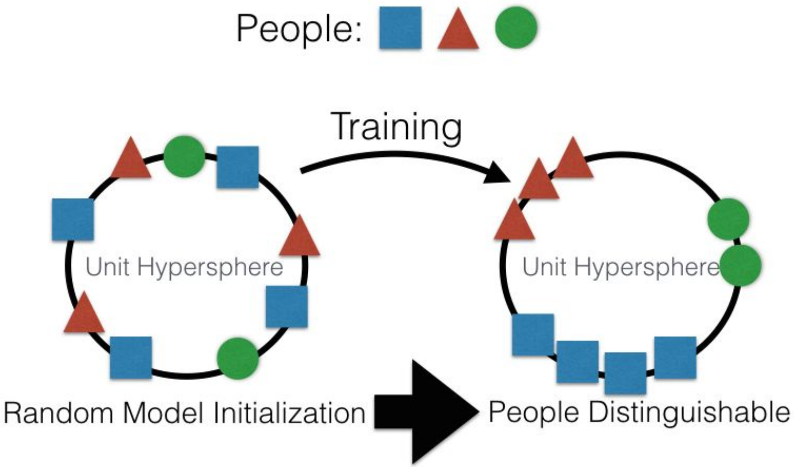


128 Dimensional Space

L2 Distance

- Squared L2 distance corresponds to similarity
- L2 distance further placed on unit hypersphere
  - Distance 0 == identical
  - Distance 4 == different

$$d(P,Q) = \|f(P) - f(Q)\|_2^2$$

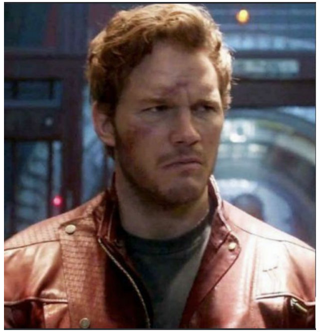


How embeddings help us?

- Verification - distance threshold

    Calculate the distance between two images

    If smaller than the threshold, new image contains the same person



Star Lord

Test

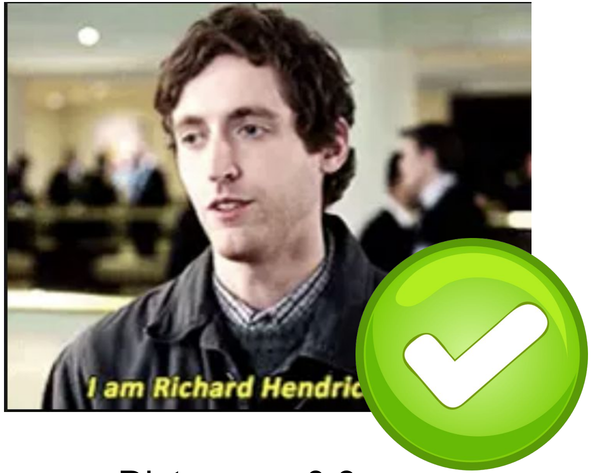
d(Star Lord, Test) <= τ

How embeddings help us?

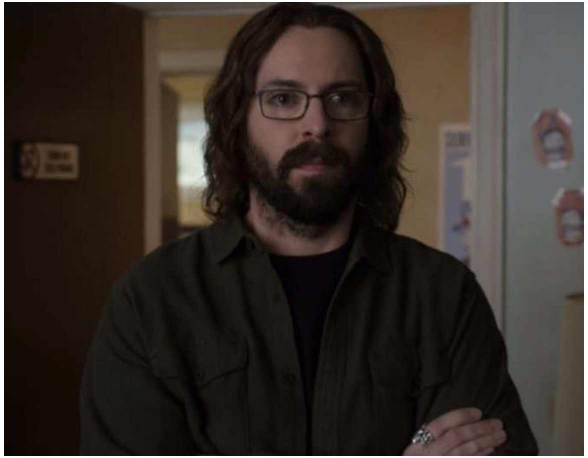
- Identification - lowest distance

  Search lowest distance

Rickard Hendricks



Distance = 0.3



Distance = 1.5

Triplet Loss

0.5

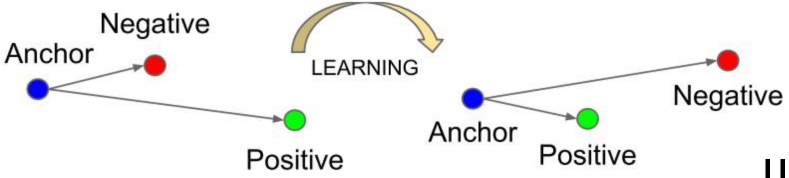
ANCHOR (A)

POSITIVE (P)

0.52

ANCHOR (A)

NEGATIVE (N)



$$d(A, P) <= d(A, N)$$

$$||f(A) - f(P)||^2 - ||f(A) - f(N)||^2 + \alpha <= 0$$

Cost Function

$$||f(A) - f(P)||^2 - ||f(A) - f(N)||^2 + \alpha <= 0$$

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

α represents the margin between positive and negative pairs

Architecture