基于深度回归网络的视觉目标跟踪技术

马超 博士
助理教授
上海交通大学

2019年8月20日

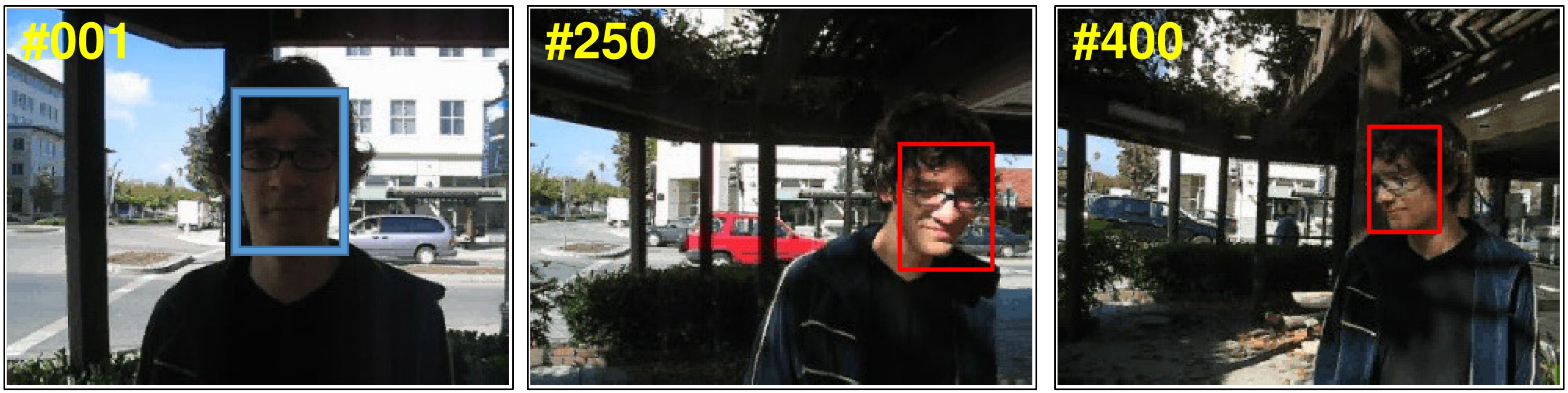
计算机视觉&图像理解

高层语义识别

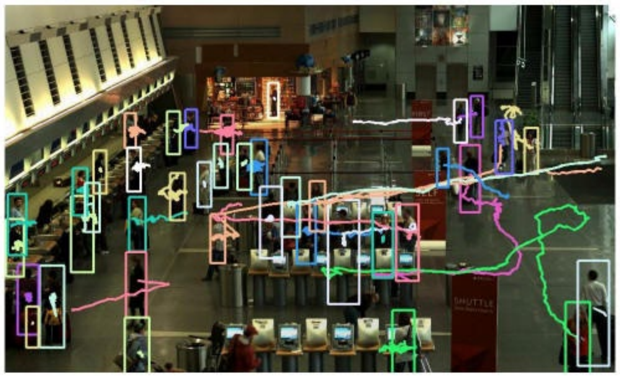
目标检测、目标跟踪

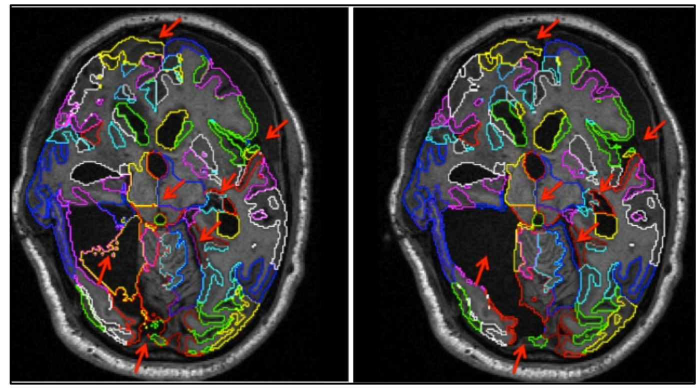
特征提取

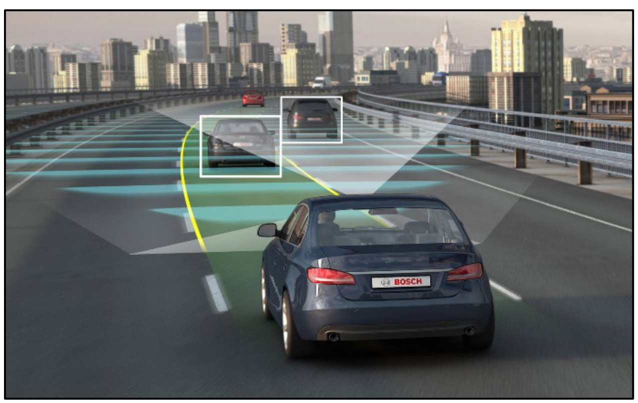
问题定义：视觉目标跟踪



目标跟踪典型应用场景

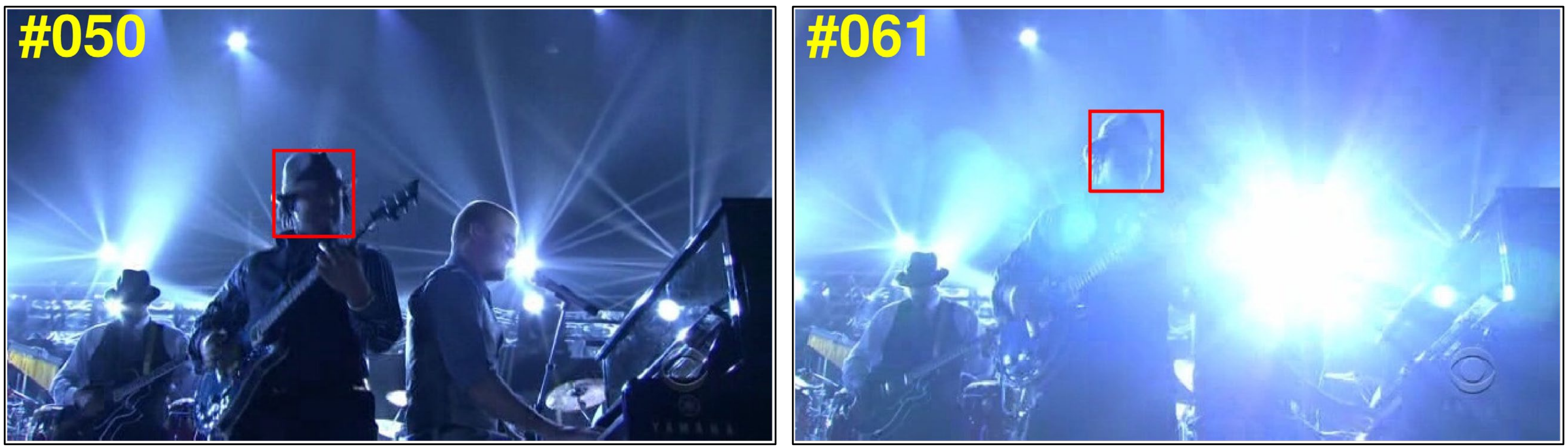






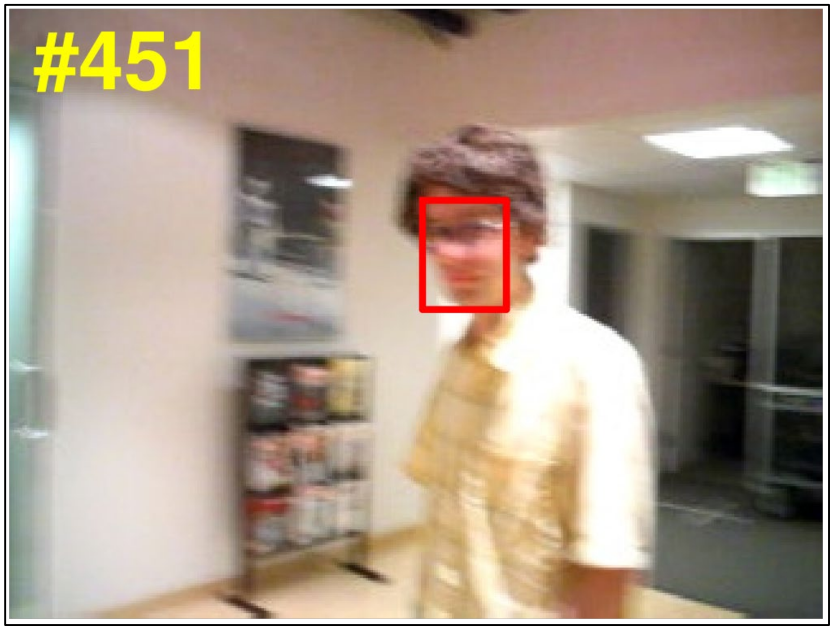
Images from Google Search

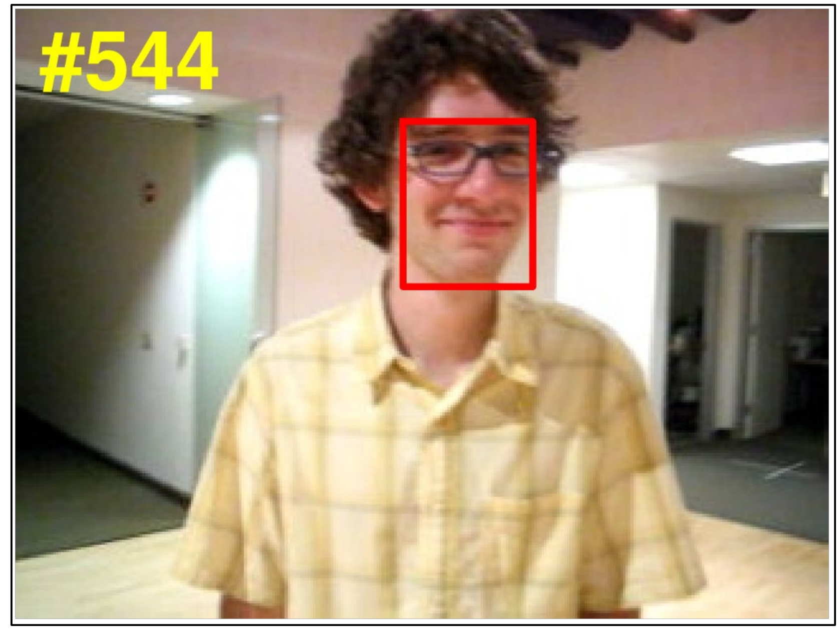
目标跟踪难点示意：光照变化



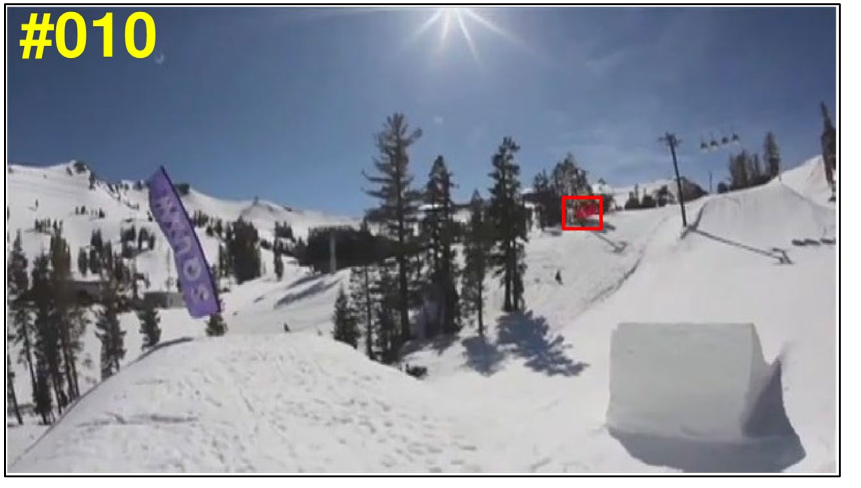
目标跟踪难点示意：严重遮挡

目标跟踪难点示意：运动模糊

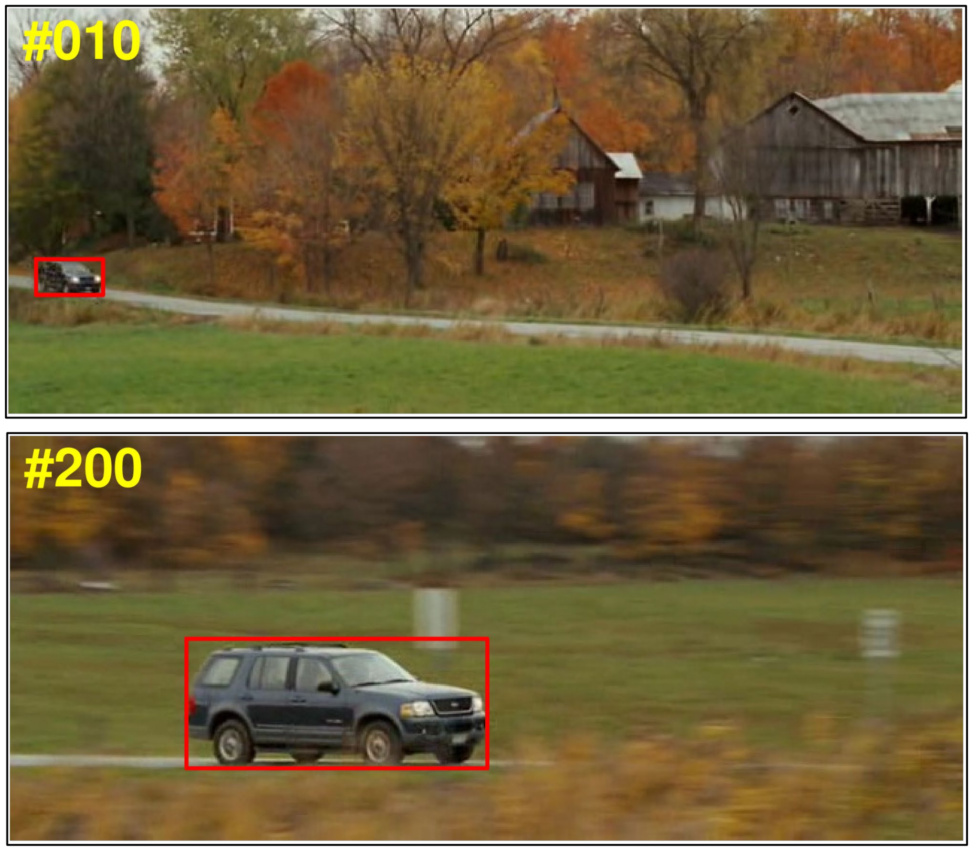




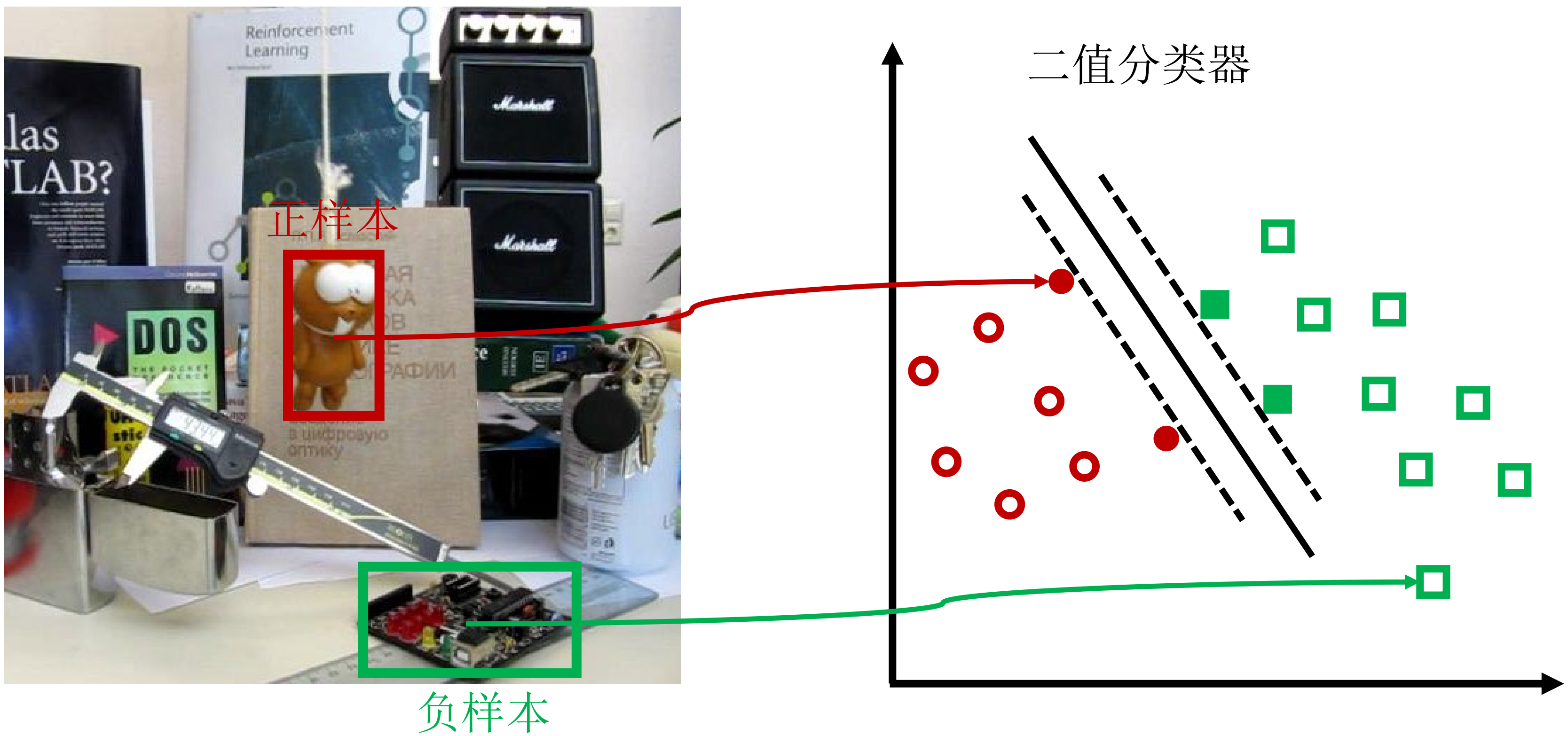
目标跟踪难点示意：运动目标过小



目标跟踪难点示意：尺度变化过大



主流跟踪方法回顾：基于分类模型

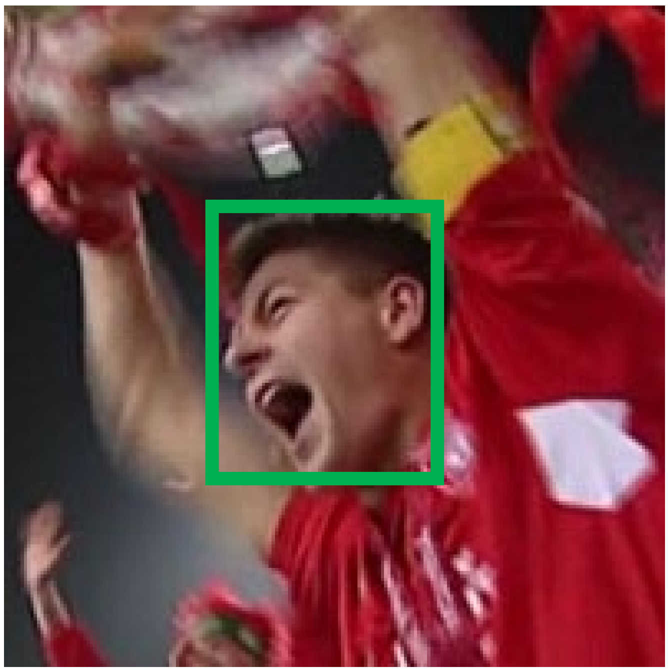


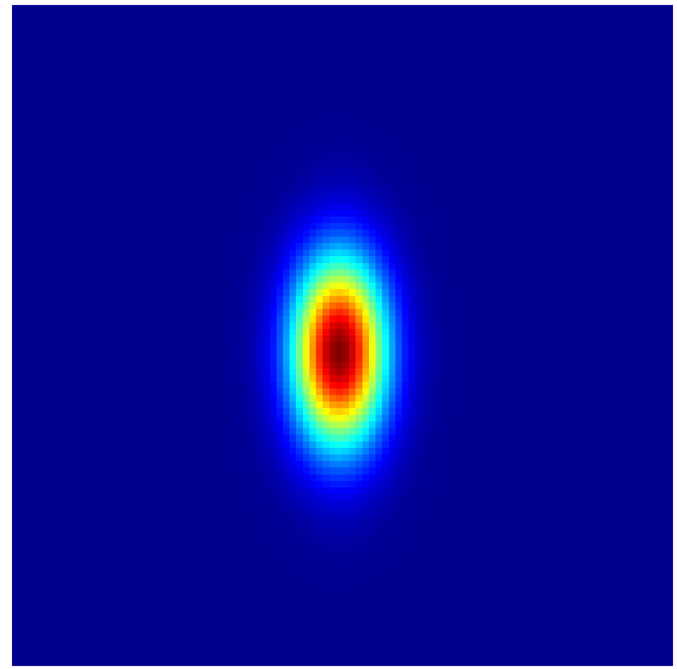
训练阶段

主流跟踪方法回顾：基于分类模型

测试阶段

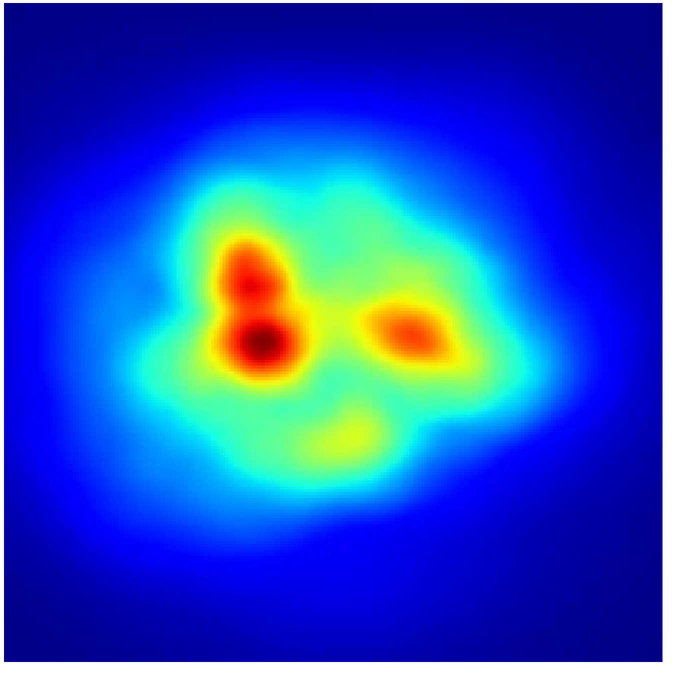
主流跟踪方法回顾：基于回归模型





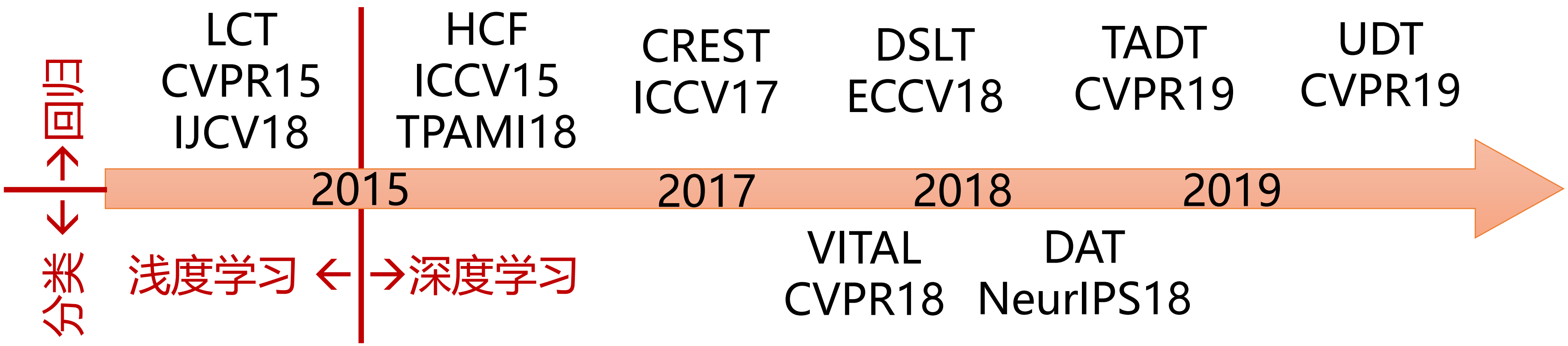
训练阶段

主流跟踪方法回顾：基于回归模型



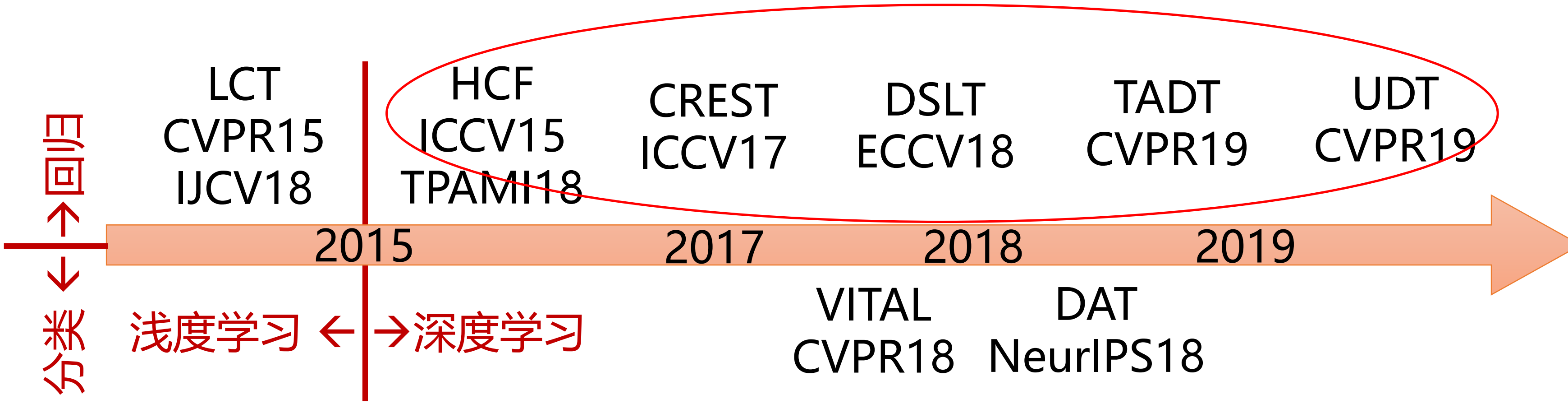
测试阶段

回归 VS 分类



回归模型	分类模型
• 输出密集响应图 • 方便利用多层深度特征 • 尺度不敏感	• 输出稀疏响应图 • 依赖随机采样 • 尺度敏感

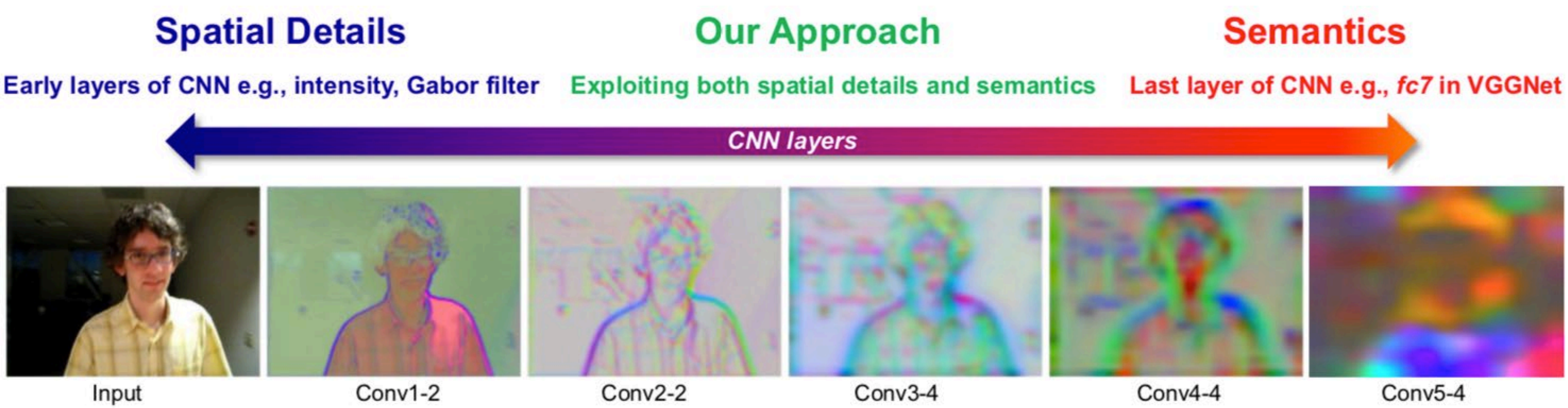
回归 VS 分类



回归模型	分类模型
• 输出密集响应图 • 方便利用多层深度特征 • 尺度不敏感	• 输出稀疏响应图 • 依赖随机采样 • 尺度敏感

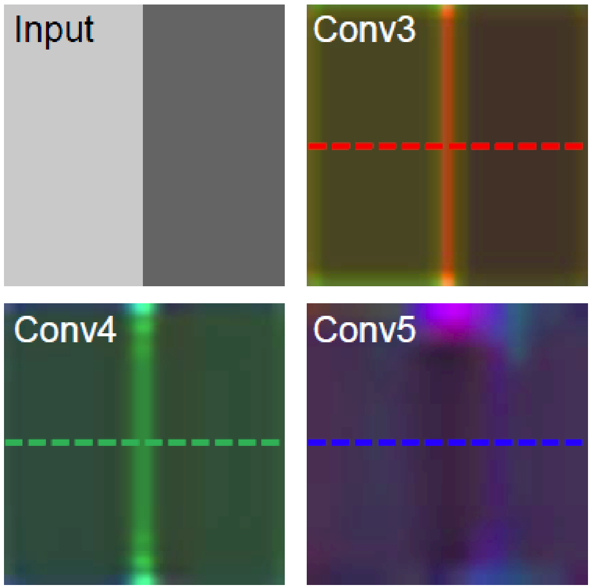

#001

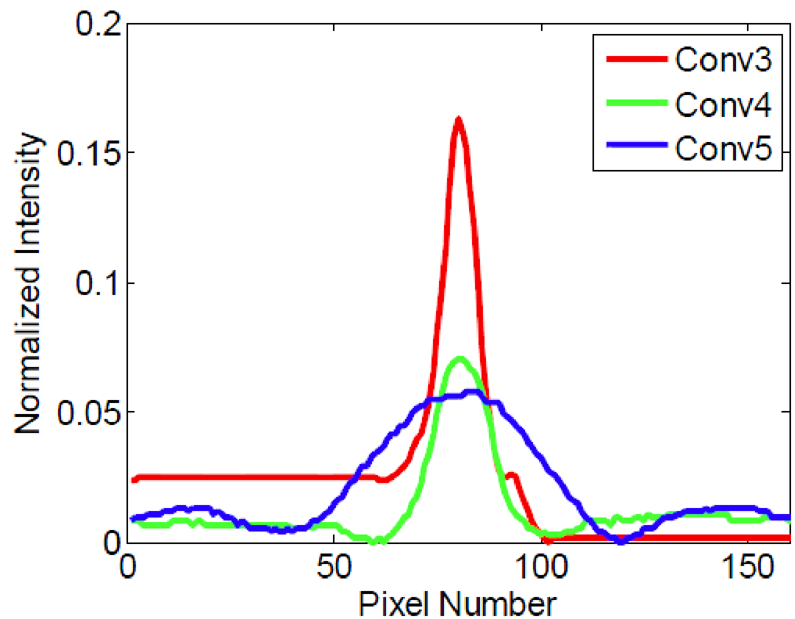
Observations



- Earlier layers retain higher spatial resolution for precise localization
- Latter layers capture more semantic information and are robust to appearance changes
- Exploit the rich hierarchies for robust visual tracking

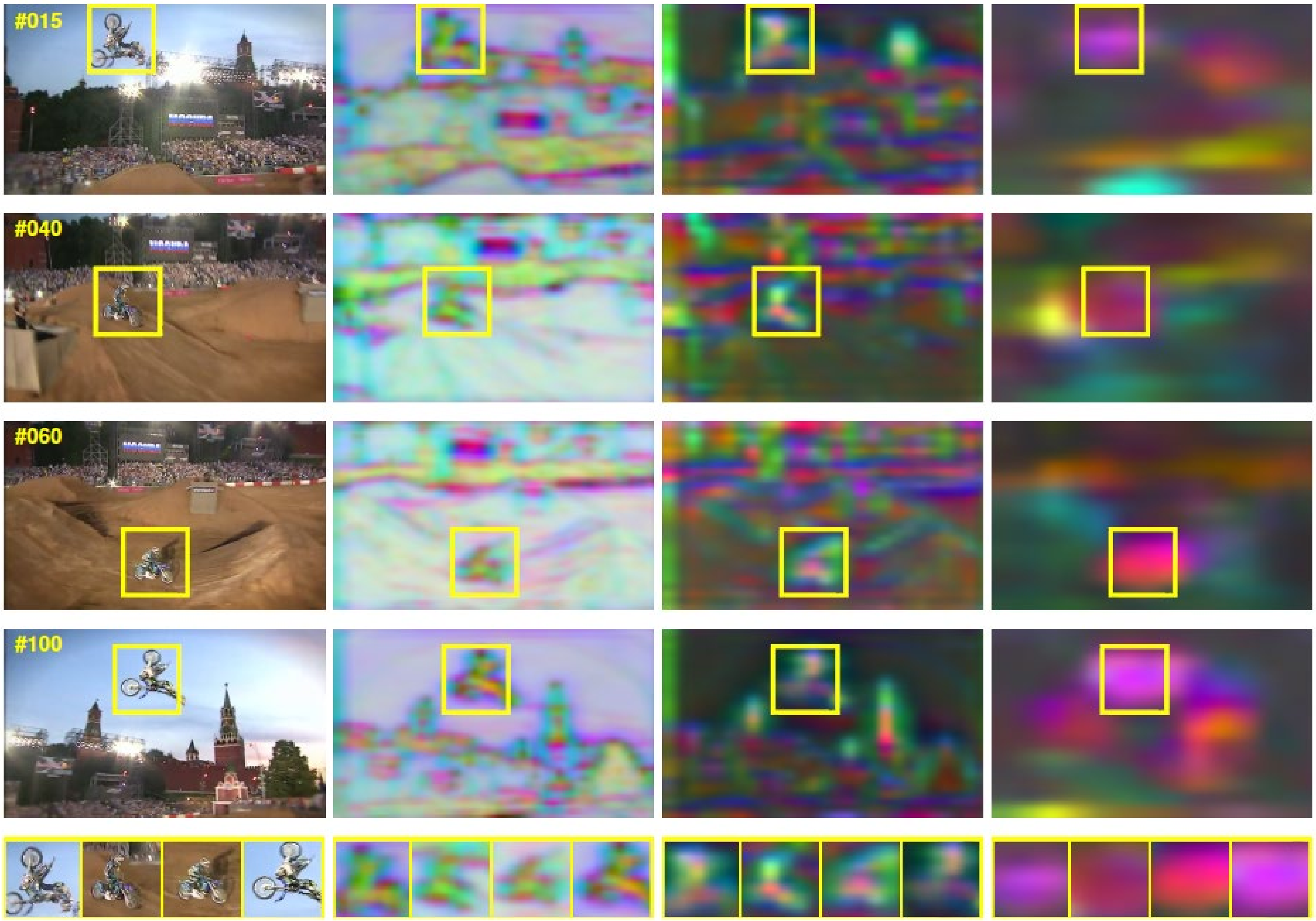
Toy Example





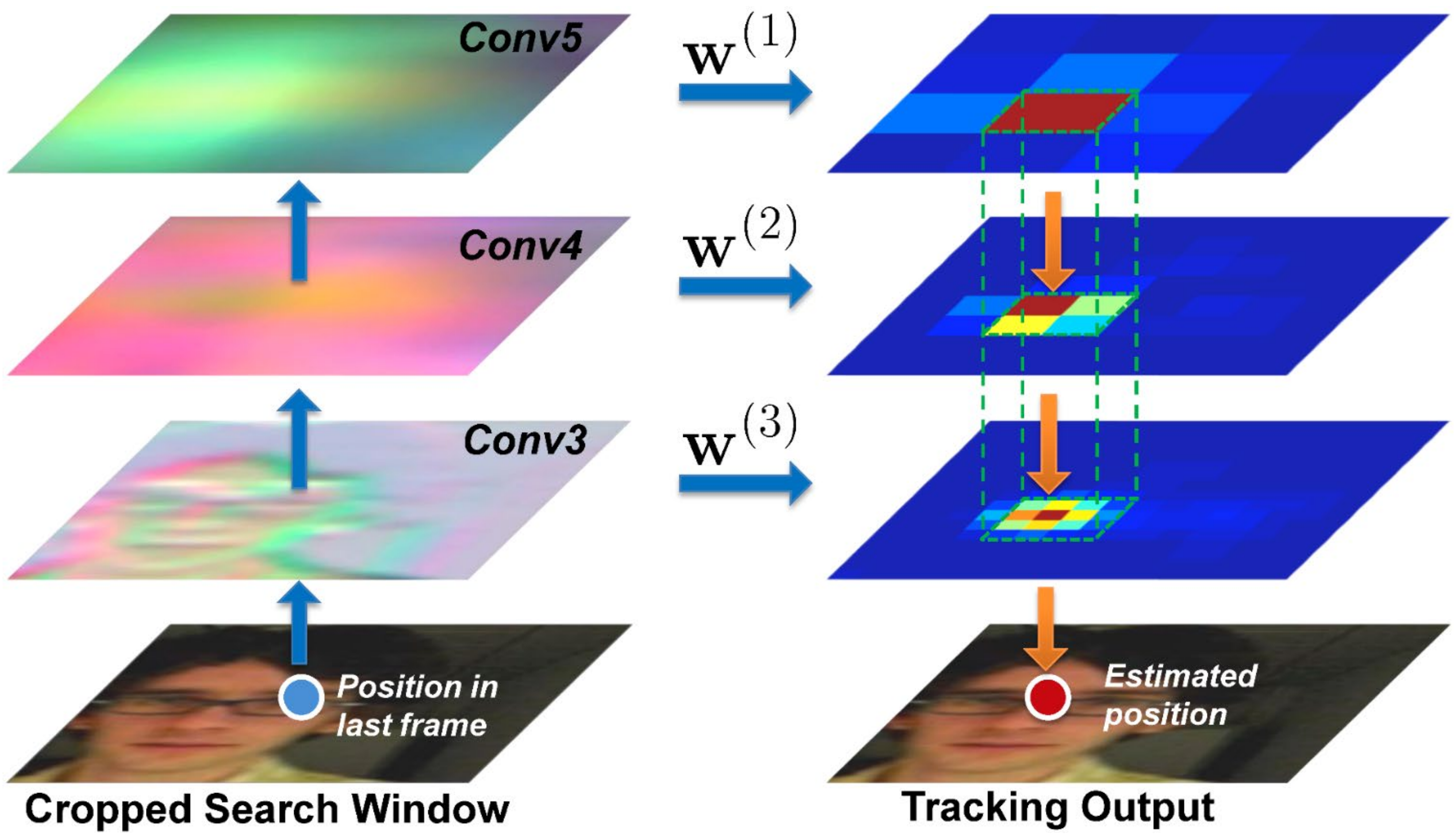
- Layer *conv5* robust to appearance change: insensitive to the sharp step edge
- Layer *conv3* is useful for precise localization: sensitive to the edge position

VGG-19 Feature Visualization



(a) Input (b) *conv3-4* (c) *conv4-4* (d) *conv5-4*

Flowchart of HCF



- C. Ma et al, Hierarchical convolutional features for visual tracking, **ICCV 2015**
- C. Ma et al, Robust visual tracking via hierarchical convolutional feature, **TPAMI 2018**

同行引用

- ICCV15论文谷歌学术单篇引用820+次，历年所有ICCV论文引用排名第17位

Best-Buddies Similarity - Robust Template Matching using Mutual Nearest Neighbors

Shaul Oron, Tali Dekel, Tianfan Xue, William T. Freeman, Shai Avidan

using deep features taken from a pre-trained neural net. Using such deep features is motivated by recent success in applying features taken from deep neural nets to different applications [37], [38].

[37] C. Ma, J.-B. Huang, X. Yang, and M.-H. Yang, "Hierarchical convolutional features for visual tracking," in *Proceedings of the IEEE International Conference on Computer Vision)*, 2015. 4

MIT计算机视觉实验室主任Bill Freeman教授在2018年发表于TPAMI论文明确指出其最新工作受本人工作启发

Detect to Track and Track to Detect

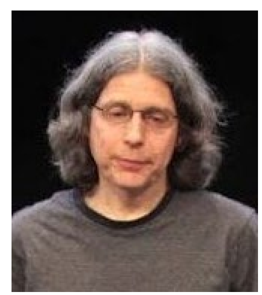
Christoph Feichtenhofer *
Graz University of Technology
feichtenhofer@tugraz.at

Axel Pinz
Graz University of Technology
axel.pinz@tugraz.at

Andrew Zisserman
University of Oxford
az@robots.ox.ac.uk

ConvNet. To achieve this we propose to extend the R-FCN [3] detector with a tracking formulation that is inspired by current correlation and regression based trackers [1, 13, 25]. We train a fully convolutional architecture end-to-end us-

[25] C. Ma, J.-B. Huang, X. Yang, and M.-H. Yang. Hierarchical convolutional features for visual tracking. In *Proc. ICCV*, 2015. 2, 3, 8



牛津大学视觉几何实验室（VGG）主任Andrew Zisserman教授发表于CVPR 2017论文明确指出其最新工作受本人工作启发

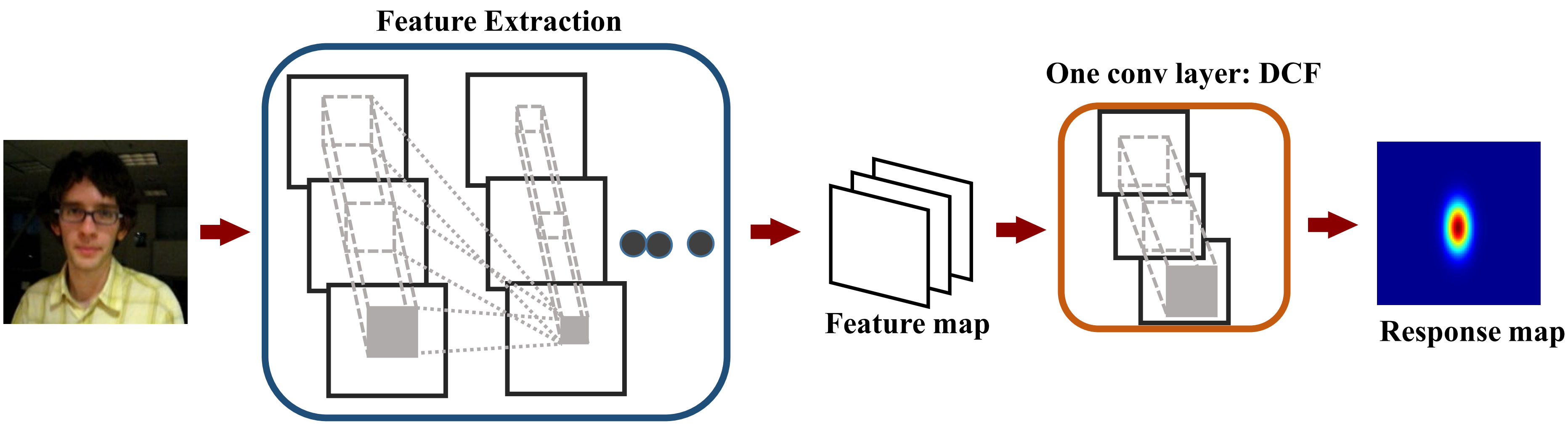
Visualization

Problems of HCF

- Existing correlation filter frameworks are empirically designed:
  - Filter weights training, model update, convolutional feature integration, etc.
- The whole framework has not been optimized end-to-end
- The deep architecture has not been fully exploited.

Why not integrate both as a whole?

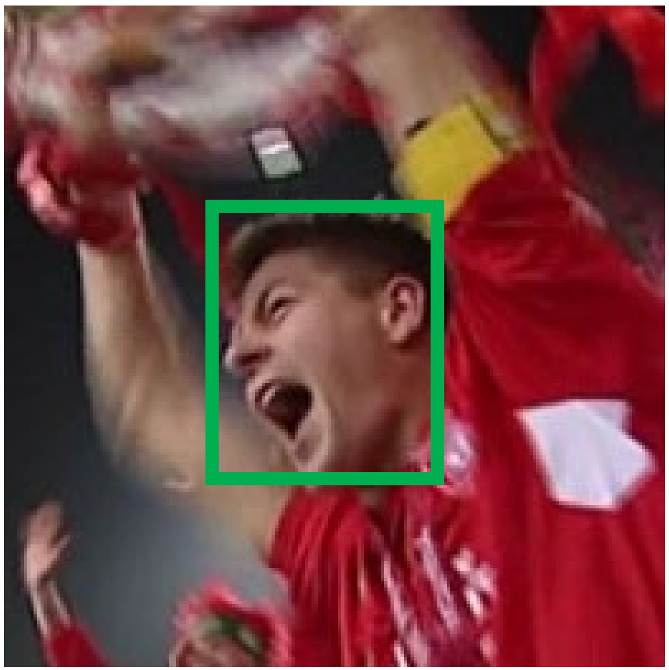
Our Method: Overview

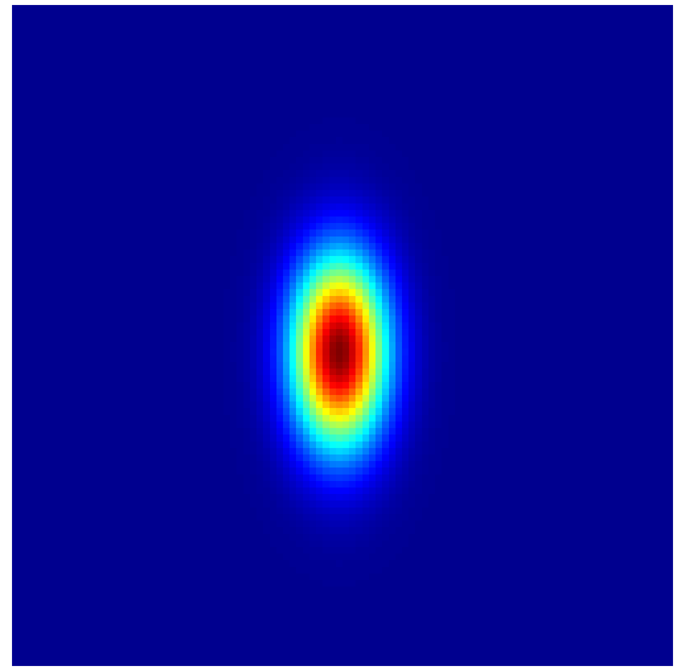


End-to-end prediction and optimization

Key idea: reformulate correlation filter as one convolutional layer

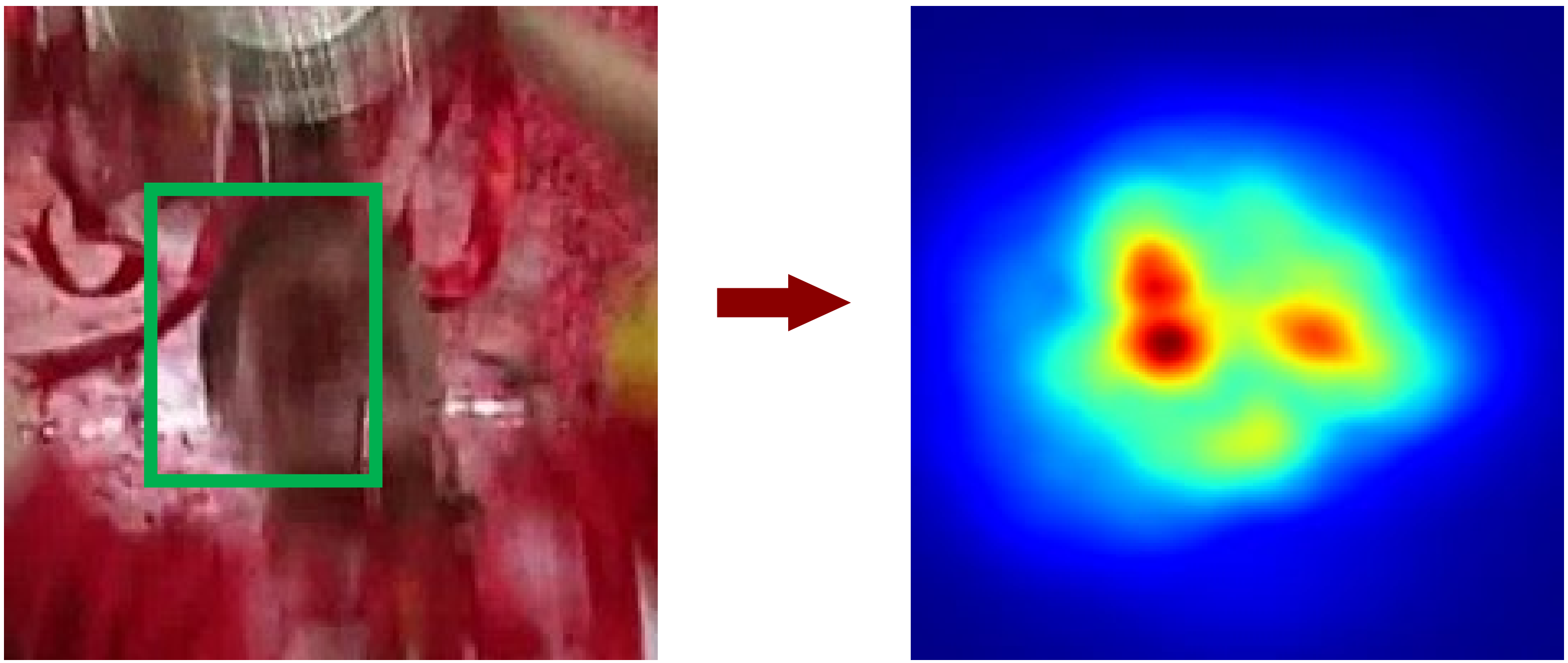
Observations





Training Step

Observations



Prediction Result

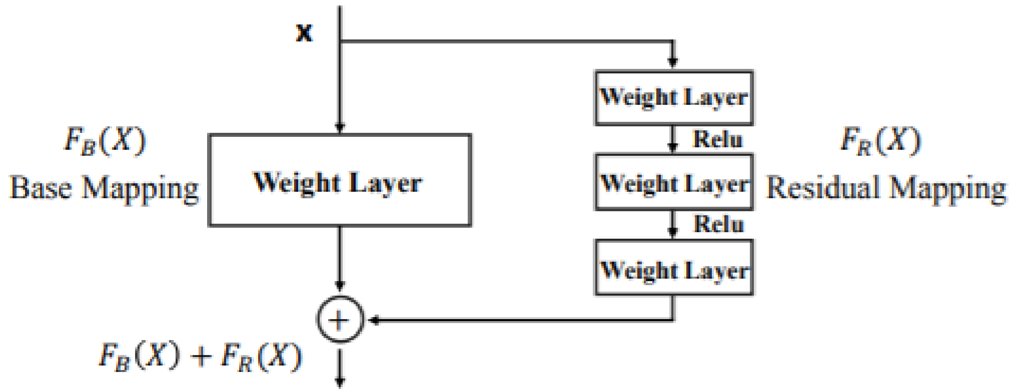
Our Method: Motivation

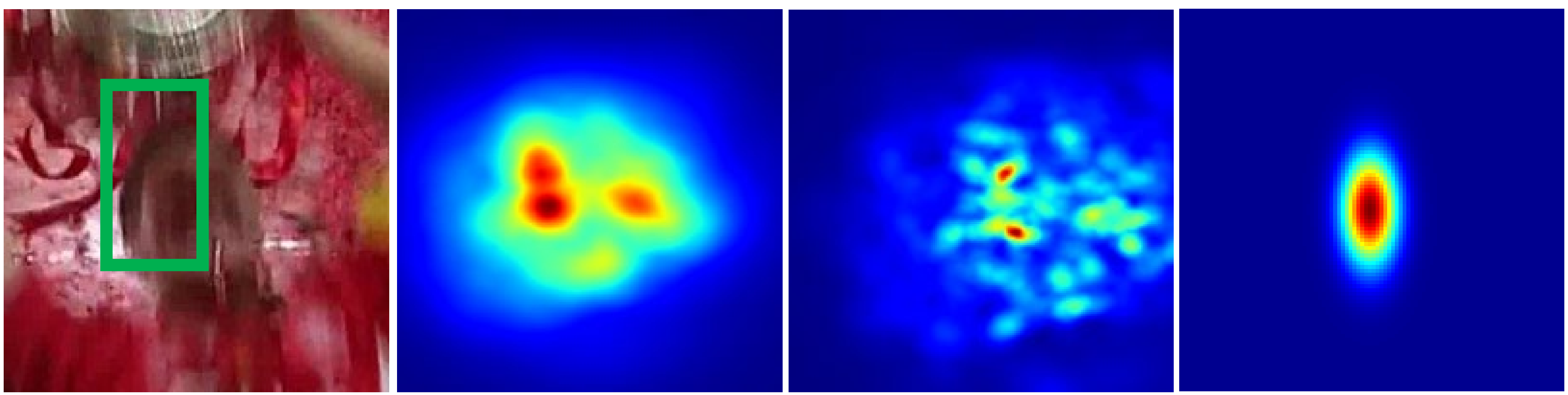
$X \Rightarrow \mathcal{H}(X)$ Optimal ground truth prediction

$X \Rightarrow F(X)$ Practical prediction

$X \Rightarrow H(X) - F(X)$ Residual prediction

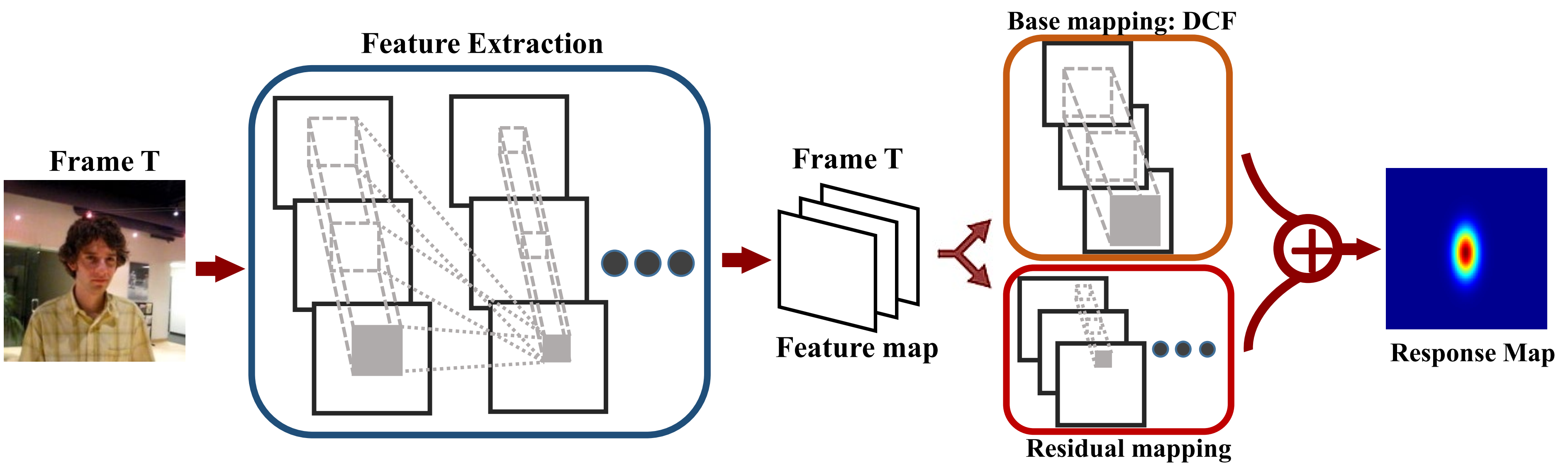
Our Method: Residual Layer



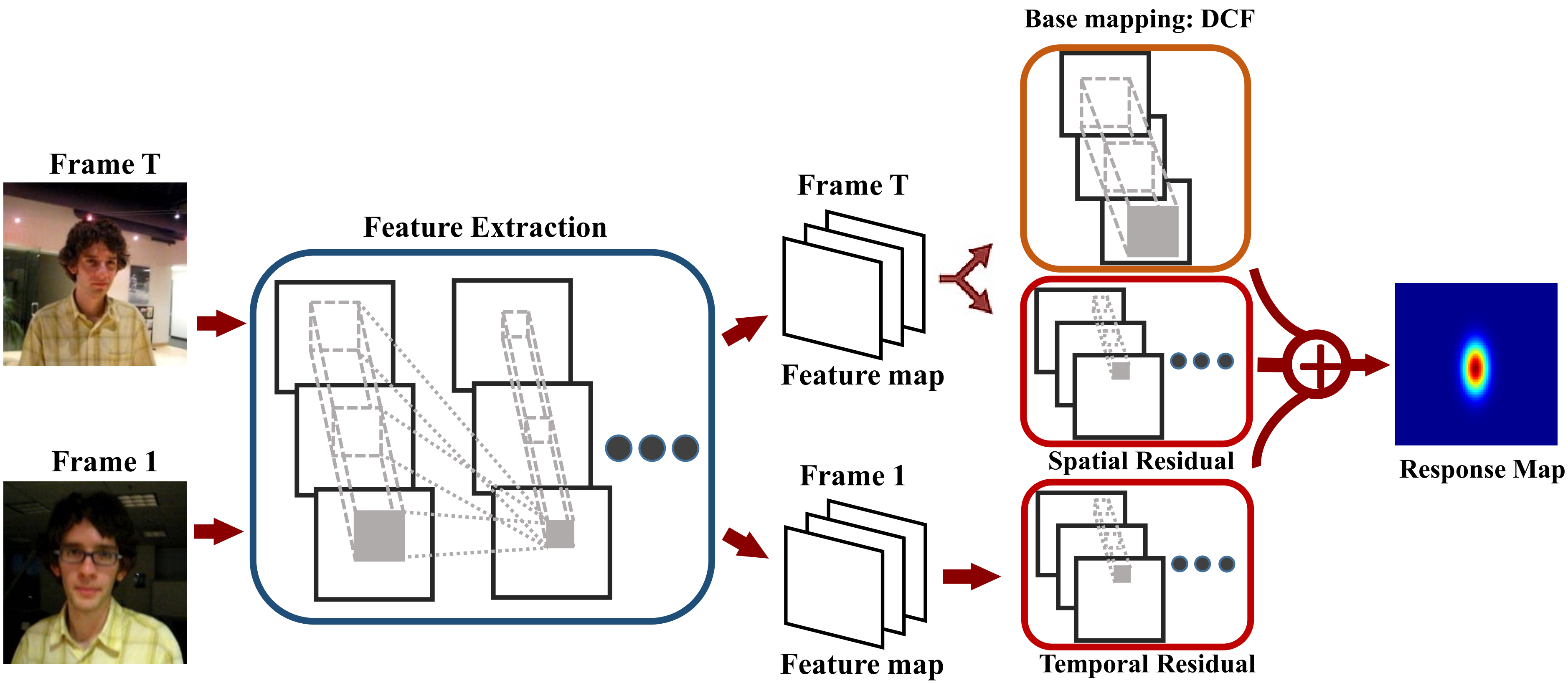


输入帧 基路输出 残差输出 完整网络输出

Base and Spatial Residual Learning

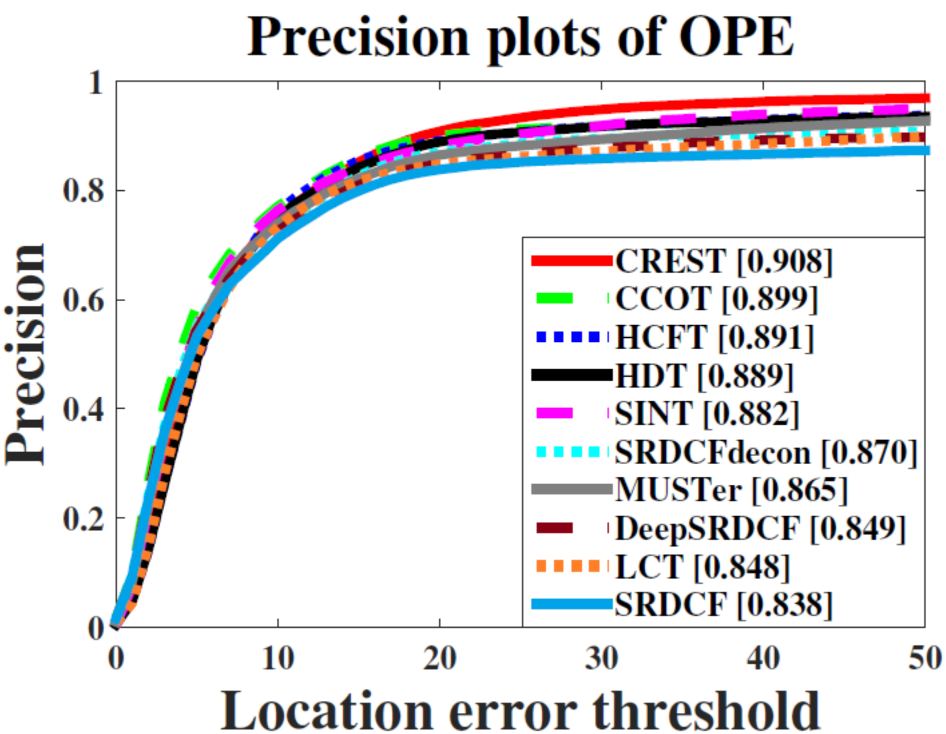


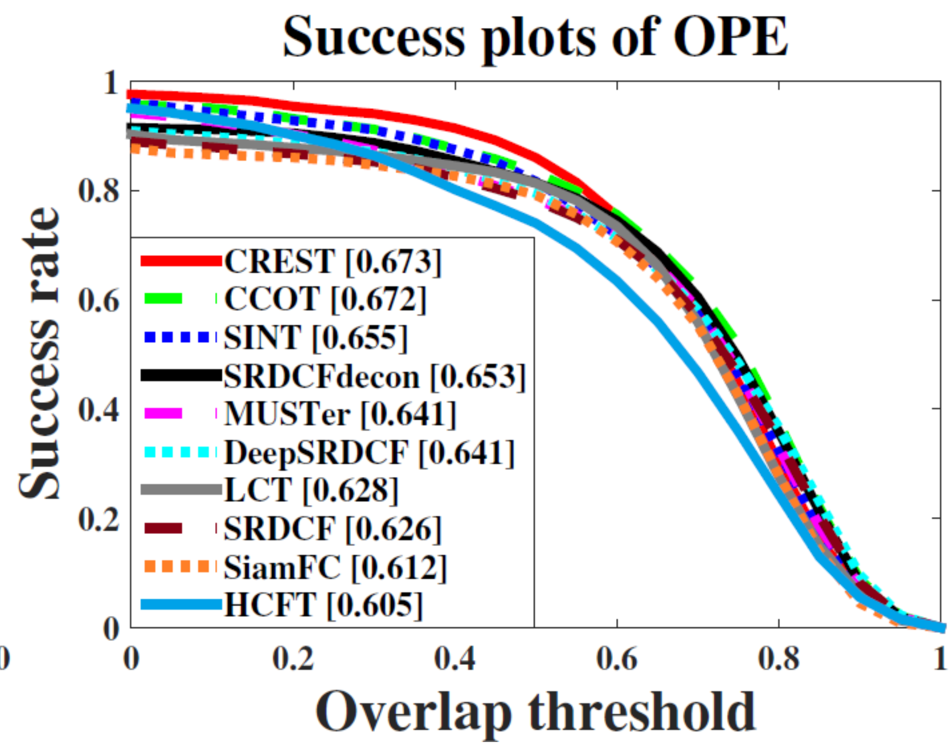
Temporal Residual Integration



Y. Song, **C. Ma**, L. Gong, J. Zhang, R. Lau, and M.-H. Yang, "CREST: Convolutional RESidula Learning for Visual Tracking," in ICCV 2017.

Results on OTB-2013



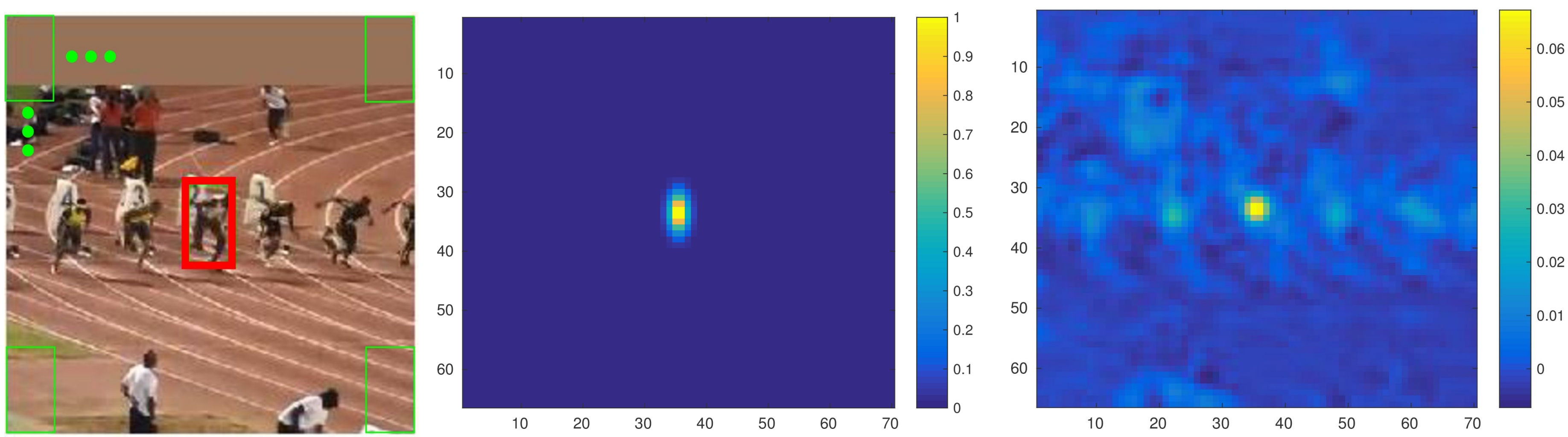


Problems of CREST

- One-stage deep regression trackers do not perform as well as correlation trackers
  - CREST: 90.9% vs ECO (CVPR' 17) 92.2% on OTB-2013
  - Data imbalance in regression learning
  - Residual response map learning vs residual feature learning

Data Imbalance Issue

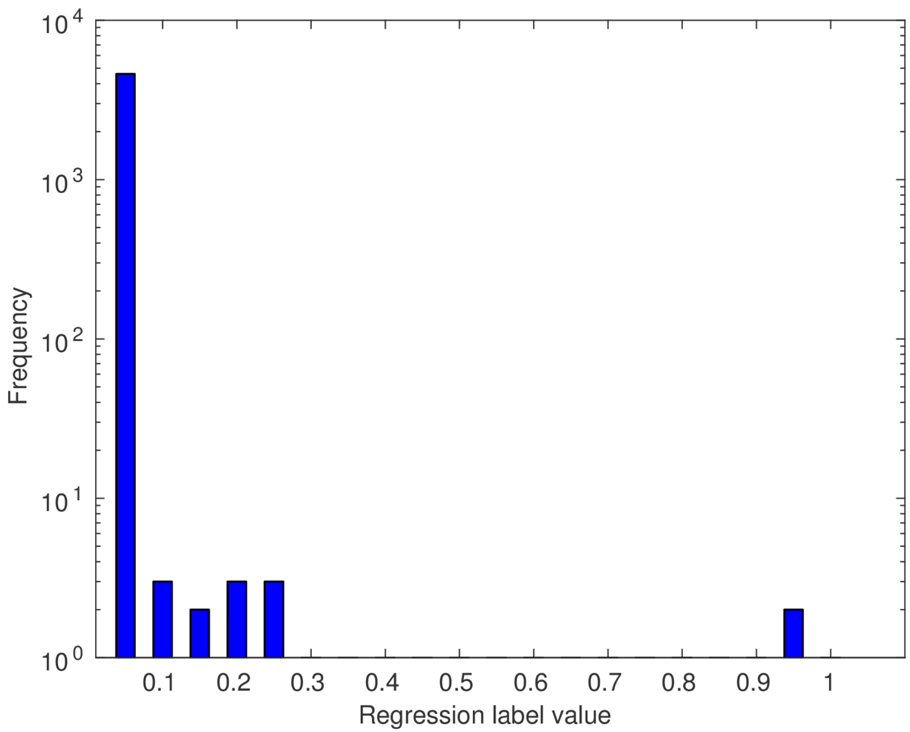
- Data imbalance in learning regression networks



Input Image, ground truth labels and regression outputs

Data Imbalance Issue

- Histogram of the difference values between regression outputs and labels



Easy training samples dominates the difference values

Regression Loss

$$L_2 = |p - y|^2 = l^2$$

p: regression out, *y*: ground truth label

$$L_{FL} = l^{\gamma} \times L_2$$

focal loss (ICCV 2017) in regression learning

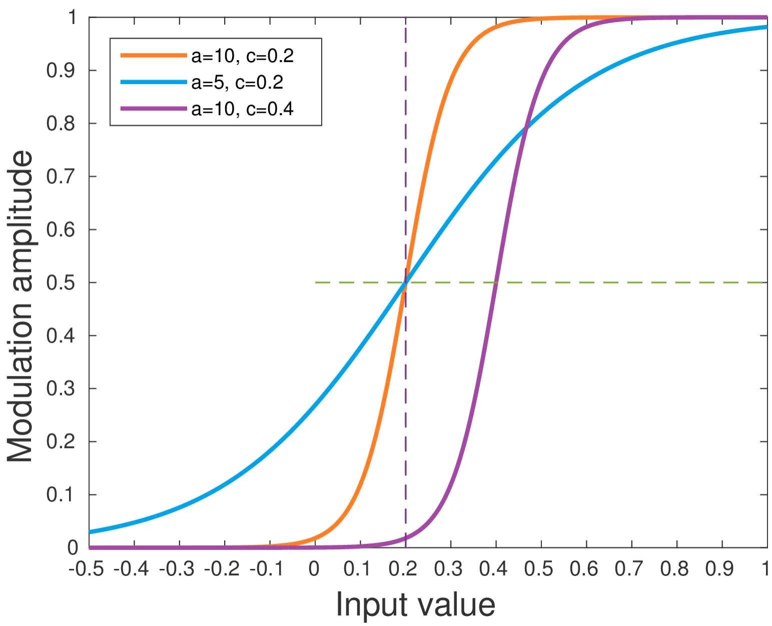
$$L_{FL} = L_3 = l \times L_2 = l^3$$

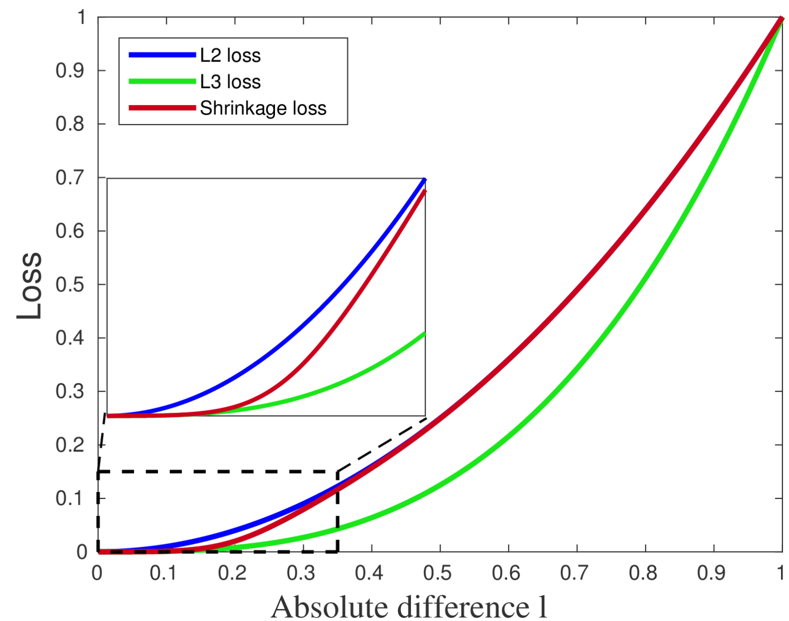
$when\ \gamma = 1,$

$$Ls = \frac{L_2}{1+exp(a*l-b)}$$

we use the Sigmoid function as the modulator

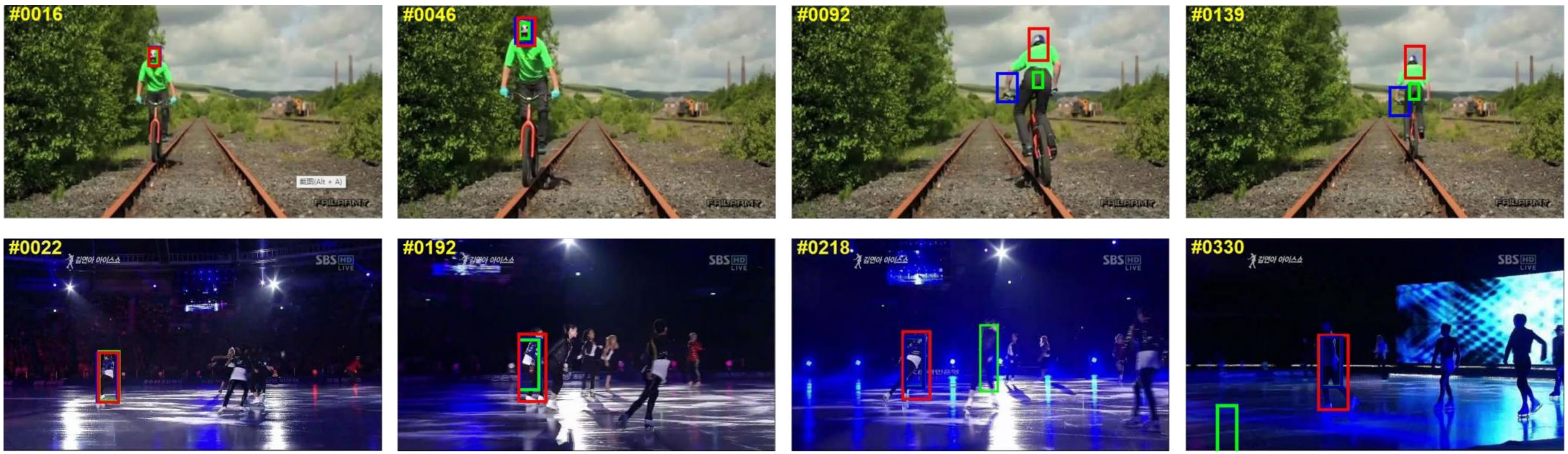
Shrinkage Loss





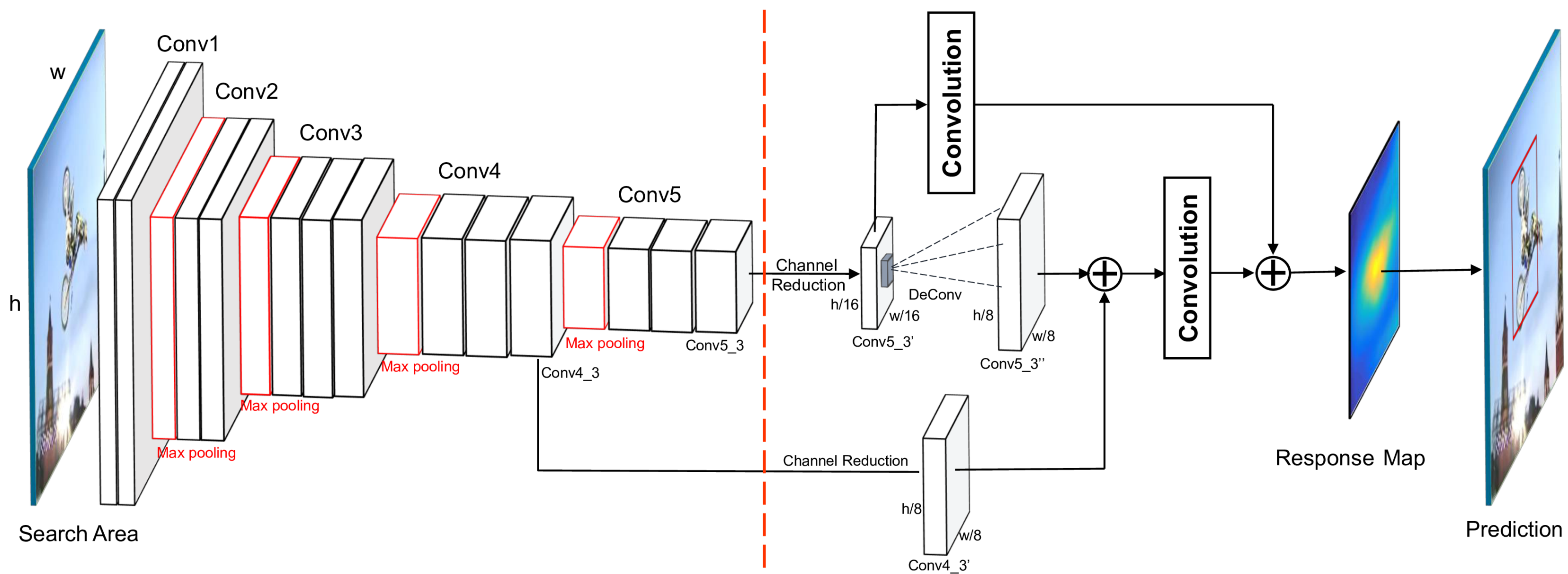
Our shrinkage loss penalizes the importance of the easy samples only, whereas the focal loss (L3) penalizes the importance of both easy and hard samples.

Visualization



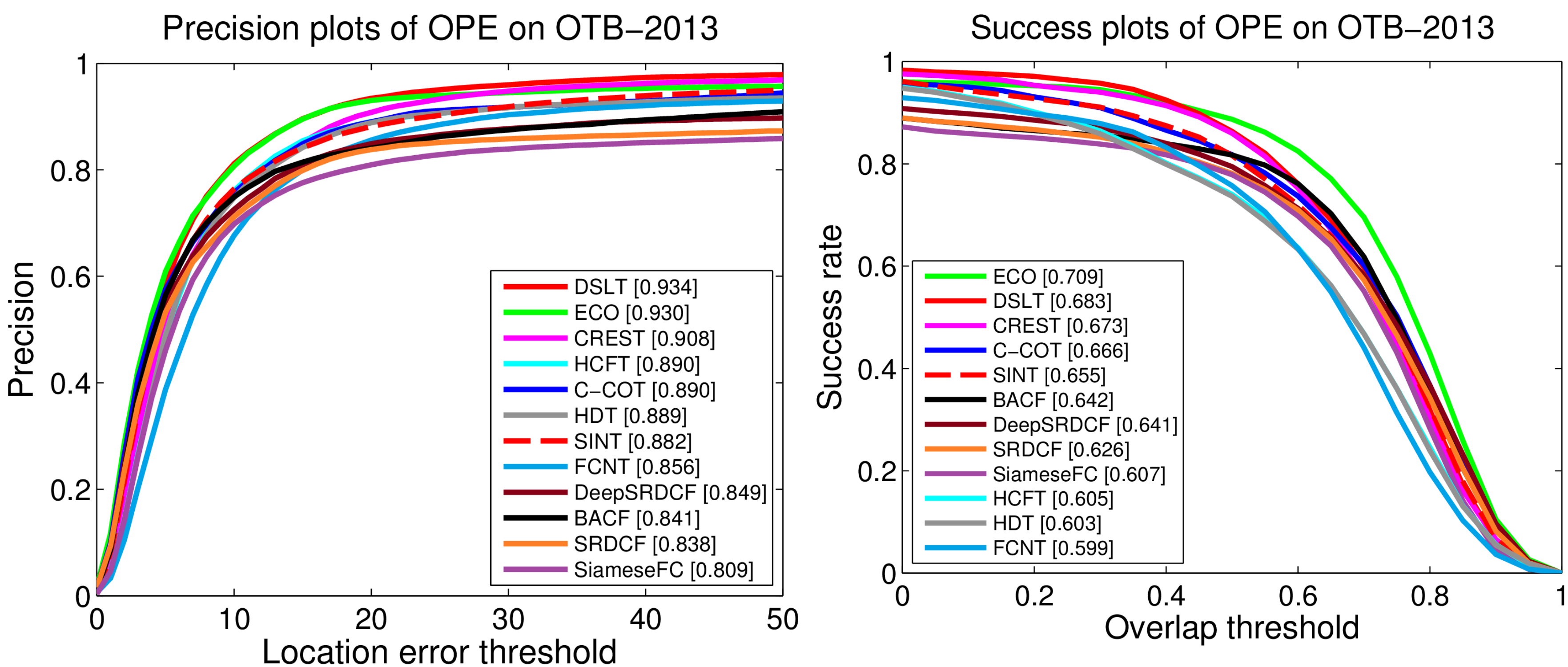
DSLT (shrinkage loss) L2 loss L3 loss

Architecture of Our network



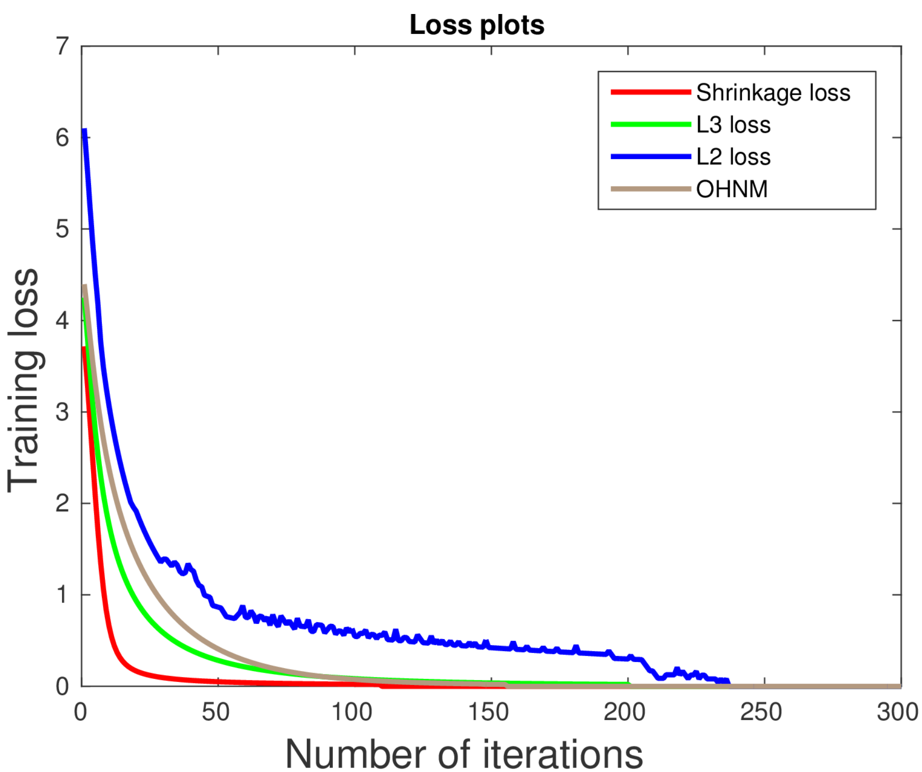
X. Lu*, **C. Ma***, B. Ni, X. Yang, I. Reid, and M.-H. Yang, "Deep Regression Tracking with Shrinkage Loss", in ECCV 2018

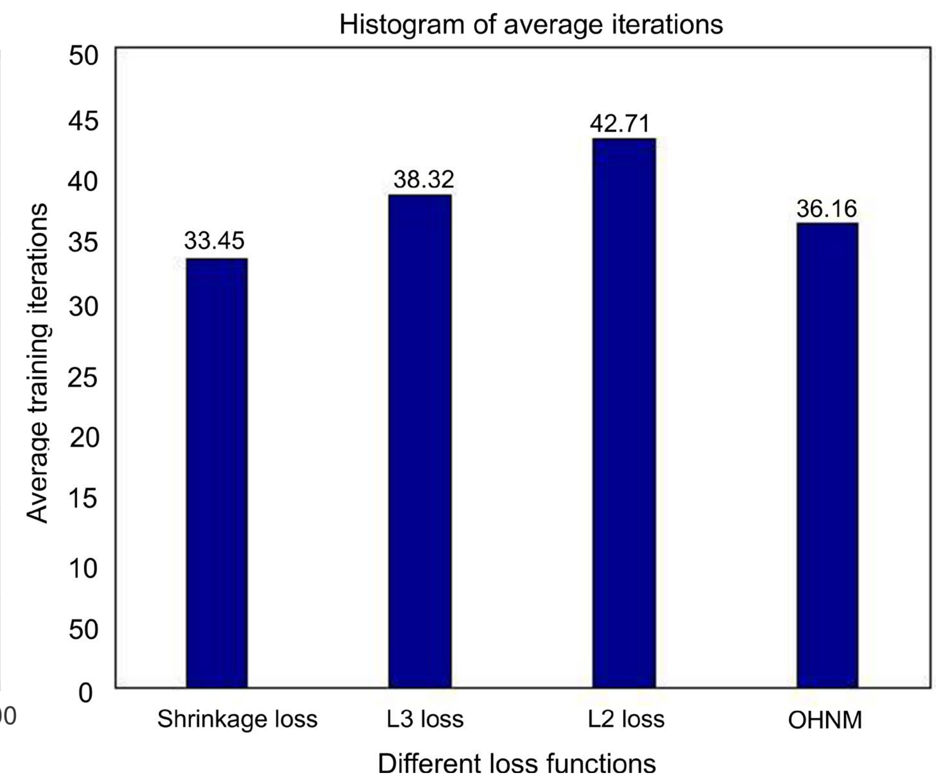
OTB-2013



Precision plots of OPE on OTB−2013
Precision
Location error threshold
0
0.2
0.4
0.6
0.8
1
0
10
20
30
40
50
DSLT [0.934]
ECO [0.930]
CREST [0.908]
HCFT [0.890]
C−COT [0.890]
HDT [0.889]
SINT [0.882]
FCNT [0.856]
DeepSRDCF [0.849]
BACF [0.841]
SRDCF [0.838]
SiameseFC [0.809]
Success plots of OPE on OTB−2013
Success rate
Overlap threshold
0
0.2
0.4
0.6
0.8
1
ECO [0.709]
DSLT [0.683]
CREST [0.673]
C−COT [0.666]
SINT [0.655]
BACF [0.642]
DeepSRDCF [0.641]
SRDCF [0.626]
SiameseFC [0.607]
HCFT [0.605]
HDT [0.603]
FCNT [0.599]

Convergence speed

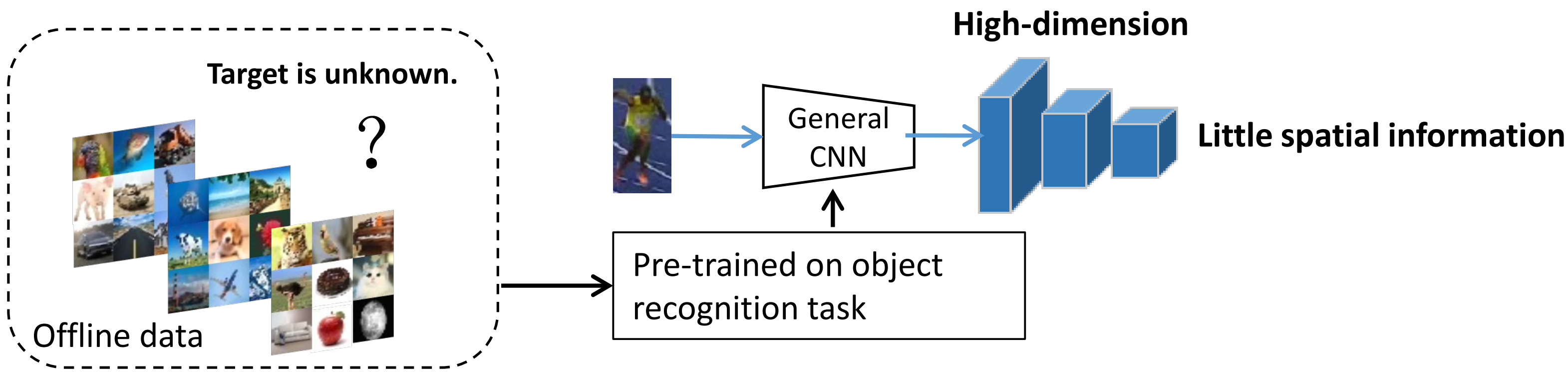




Conclusion

- Correlation filters can be reformulated as CNN layers via end-to-end learning
- Deep regression trackers do not perform as well as the DCFs trackers due to data imbalance
- Shrinkage loss can effectively deal with data imbalance in regression learning

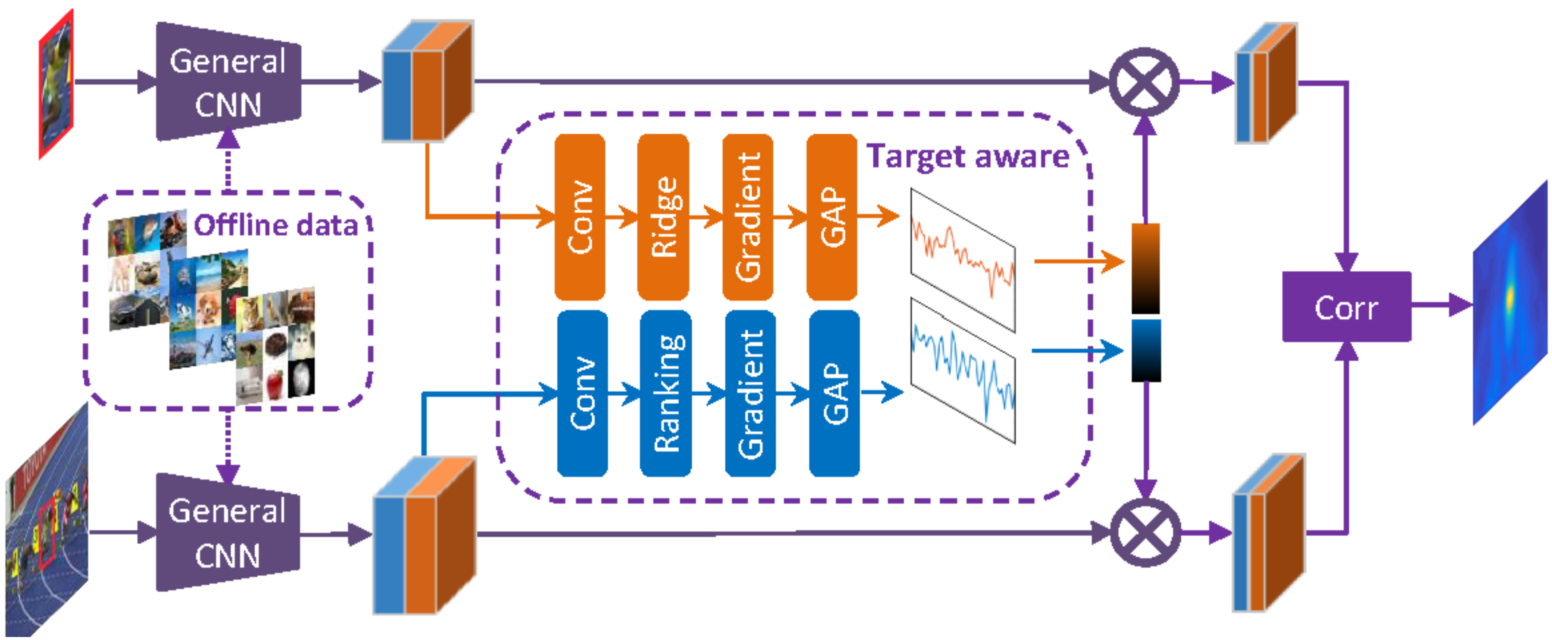
Problem of pre-trained features



Using pre-trained deep features on object recognition for visual tracking may lead to the following issues:

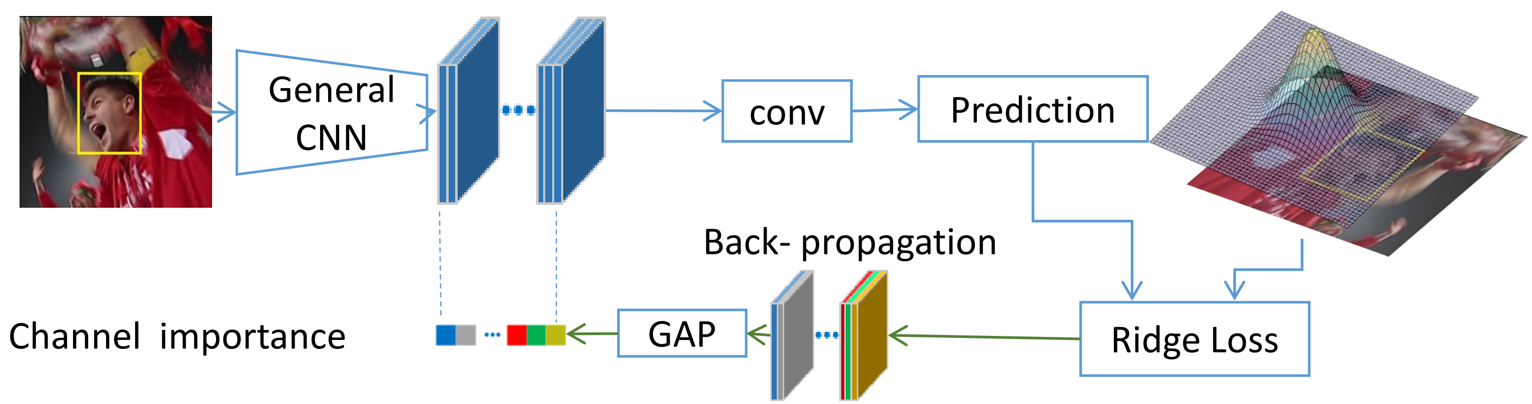
- Ineffective target representation.
- Inaccurate scale estimation.
- High computational loads.

Target-Aware Deep Tracker



X. Li, **C. Ma**, B. Wu, Z. He, and M.-H. Yang , "Target-Aware Deep Tracking ", in CVPR 2019

Target-Active Features via Regression



- Regressing the features of the candidates to a Gaussian label map indicating the target position.
- Finding the target-active channels based on the gradient values with global average pooling (GAP).

Target-Active Features via Regression

We define the regression loss as:

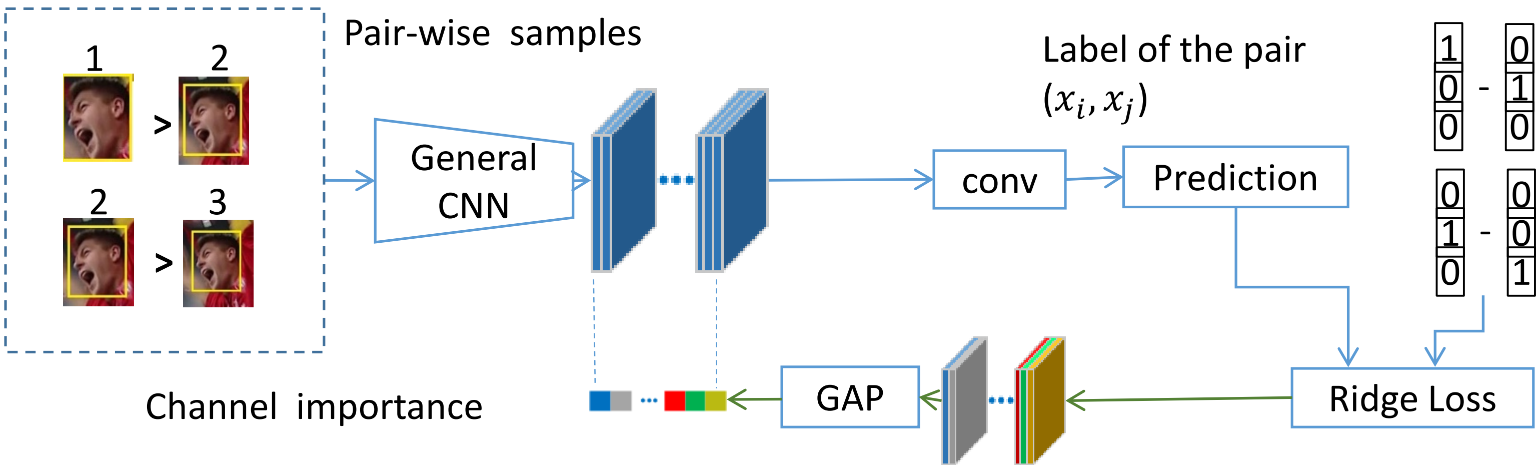
$$L_{reg} = \left\|Y - W * X_{\text{in}}\right\|^2 + \lambda\|W\|^2$$

where X_{in}, Y, and W denote the input features, the label map, and the weights.
The gradient of the regress loss with respect to the input feature is computed by

$$\frac{\partial L_{reg}}{\partial X_{in}} = \sum_{i,j} \frac{\partial L_{reg}}{\partial X_o(i,j)} \times \frac{\partial X_o(i,j)}{\partial X_{in}(i,j)}$$
$$= \sum_{i,j} 2(X_o(i,j) - Y(i,j)) \times W$$

where $X_o = W * X_{in}$ denotes for the output prediction of the regression convolutional layer.

Scale-Sensitive Features via Ranking



- Finding the scale-sensitive channels based on their contribution to scale changes.

Scale-Sensitive Features via Ranking

We exploit a smooth approximated ranking loss [1] as:

$$L_{rank} = \log\left(1 + \sum_{(x_i, x_j)\in\Omega} \exp(f(x_i) - f(x_j))\right),$$

where (x_i, x_j) is a sample pair and the size of x_j is closer to the target size compared to x_i, $f(x; w)$ is the prediction model, and Ω is the set of training pairs.

The gradient of L_{rank} with respect to the features is computed as:

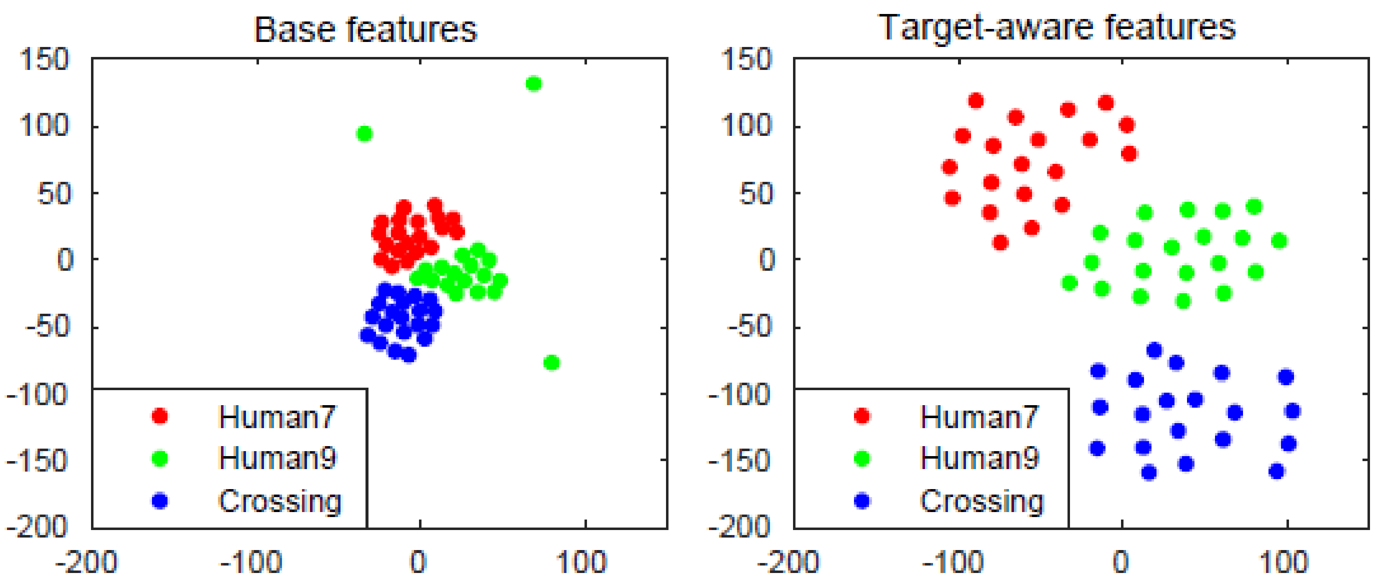
$$\frac{\partial L_{rank}}{\partial x_{in}} = \frac{\partial L_{rank}}{\partial x_o} \times \frac{\partial x_o}{\partial x_{in}} = \frac{\partial L_{rank}}{\partial f(x_{in})} \times W,$$

where $\frac{\partial L_{rank}}{\partial f(x)} = -\frac{1}{L_{rank}} \sum_{\Omega} \Delta z_{i,j} \exp(-f(x)\Delta z_{i,j})$, $\Delta z_{i,j} = z_i - z_j$ and z_k is a one-hot vector with the *k*-th element being 1 while others being 0.

[1] Yuncheng Li, Yale Song, and Jiebo Luo. ``Improving pairwise ranking for multi-label image classification." CVPR 2017

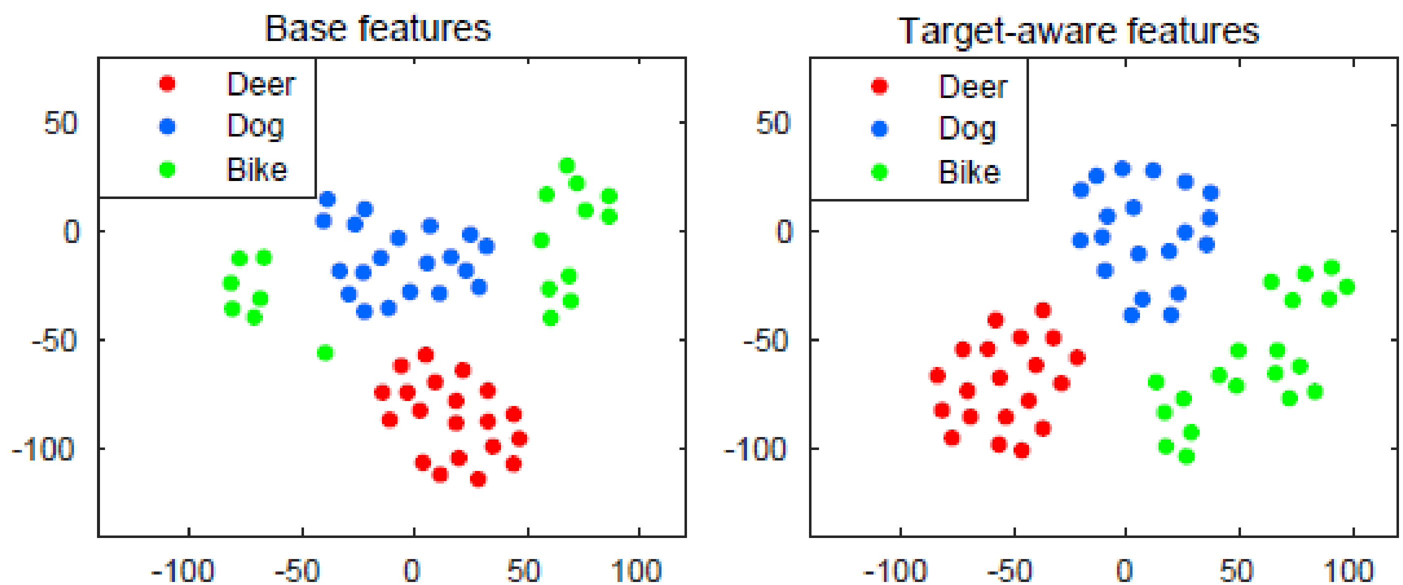
Analyses of the generated features

Distributions of the original and target-aware features using the t-SNE method.



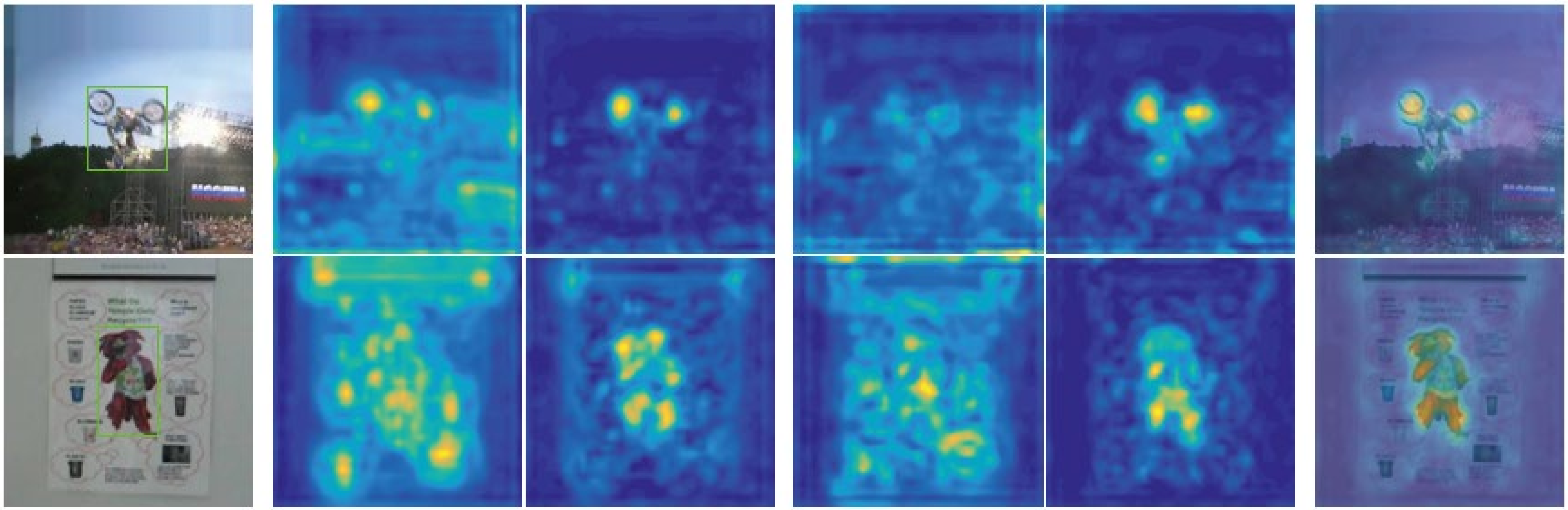
Distributions of intra-class targets (pedestrian).

Analyses of the generated features



Distributions of inter-class targets.

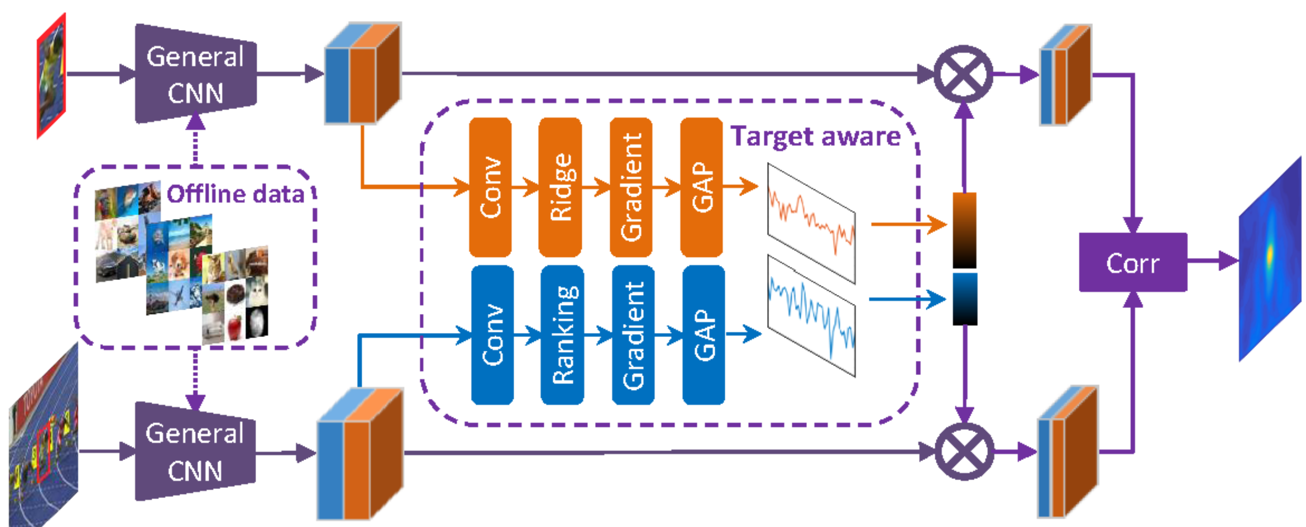
Visualization of the generated features



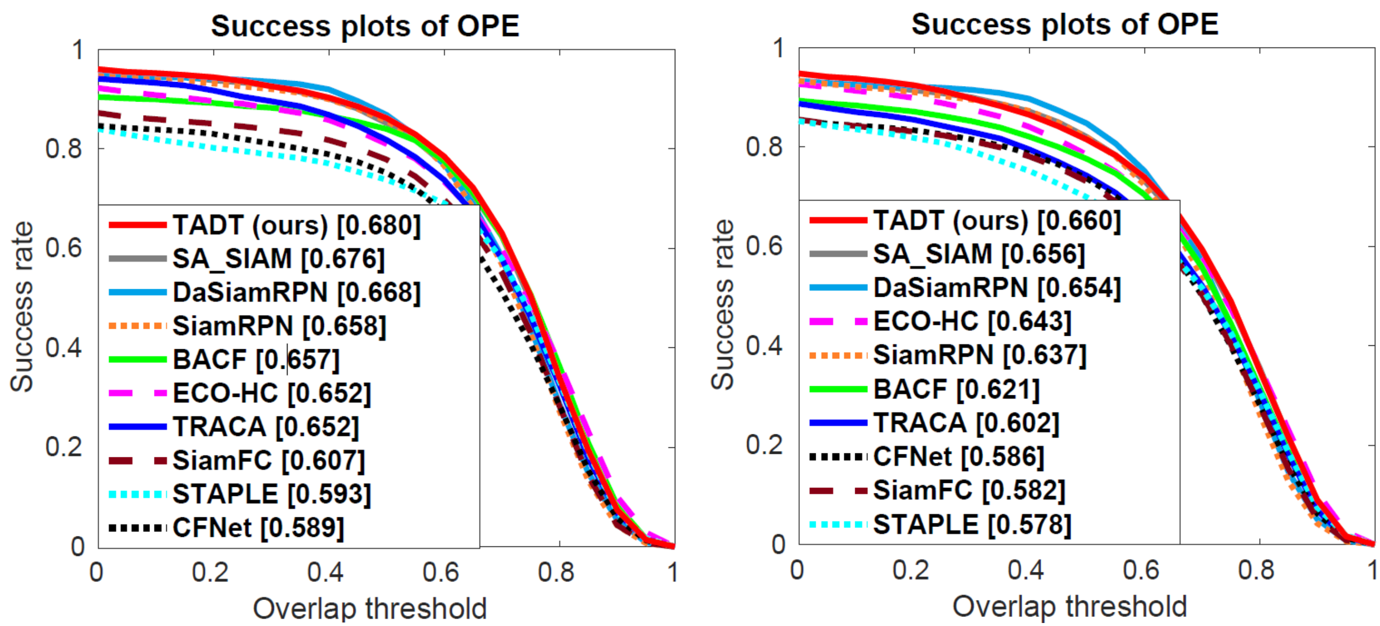
Input images | Conv4-1 w/o and w/ ranking and regression loss | Conv4-3 w/o and w/ regression loss | Target-aware

Tracking pipeline

- Model initialization:
  - fine-tuning the initial frame
  - Computing gradients
  - Selecting Filters
- Online detection:
  - Computing the similarity scores between the target template and the search region



Experimental results



Success plots on the OTB2013 and OTB2015 datasets

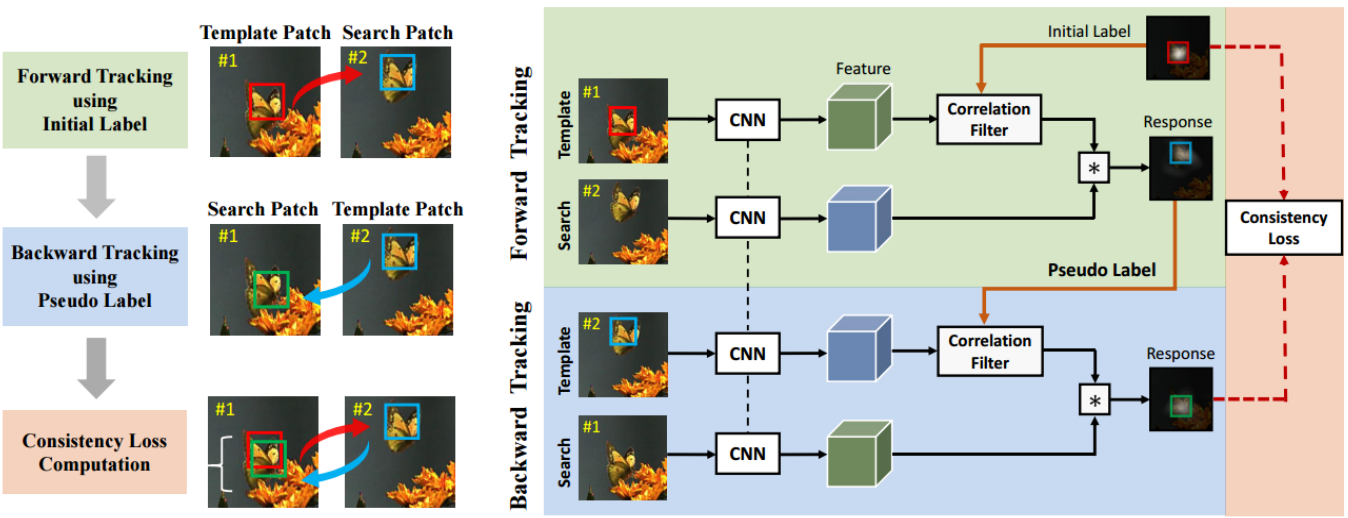
Introduction: Supervised vs. Unsupervised

- Recent trackers mostly rely on the deep convolutional neural network (CNN).
- Training a CNN model requires expensive annotated ground-truth labels
- Unlabeled videos are readily available on the Internet

Supervised Training:	Unsupervised Training:
Forward tracking	**Forward** and **Backward** tracking
Annotated sequences	**Unlabeled** sequences

Our Approach: Forward-backward Training Pipeline

- Forward tracking using initial label
- Backward tracking using pseudo label
- Consistency loss computation



(a) Unsupervised Learning Motivation (b) Unsupervised Learning Pipeline using a Siamese Network

Our Approach: Forward-backward Training Pipelin

- Correlation filter based tracking:

$$\min_{\mathbf{W}} \|\mathbf{W} * \mathbf{X} - \mathbf{Y}\|_2^2 + \lambda \|\mathbf{W}\|_2^2,$$

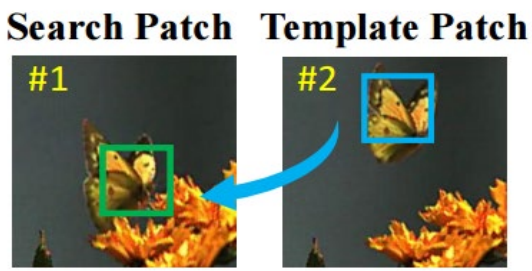
$$\mathbf{W} = \mathscr{F}^{-1}\left(\frac{\mathscr{F}(\mathbf{X}) \odot \mathscr{F}^\star(\mathbf{Y})}{\mathscr{F}^\star(\mathbf{X}) \odot \mathscr{F}(\mathbf{X}) + \lambda}\right),$$

- Forward tracking using Template and Initial Label:

$$\mathbf{W_T} = \mathscr{F}^{-1}\left(\frac{\mathscr{F}(\varphi_\theta(\mathbf{T})) \odot \mathscr{F}^\star(\mathbf{Y_T})}{\mathscr{F}^\star(\varphi_\theta(\mathbf{T})) \odot \mathscr{F}(\varphi_\theta(\mathbf{T})) + \lambda}\right),$$

$$\mathbf{R_S} = \mathscr{F}^{-1}(\mathscr{F}^\star(\mathbf{W_T}) \odot \mathscr{F}(\varphi_\theta(\mathbf{S}))).$$

- Backward tracking using original search patch and pseudo label:



Switch the role between *template* and *search* patches Treat **Rs** as the pseudo label

Our Approach: Forward-backward Training Pipeline

- Compute Consistency Loss :

  Ideally, R and Y should be similar:

$$\mathcal{L}_{\mathrm{un}} = \|\mathbf{R}_{\mathbf{T}} - \mathbf{Y}_{\mathbf{T}}\|_2^2.$$

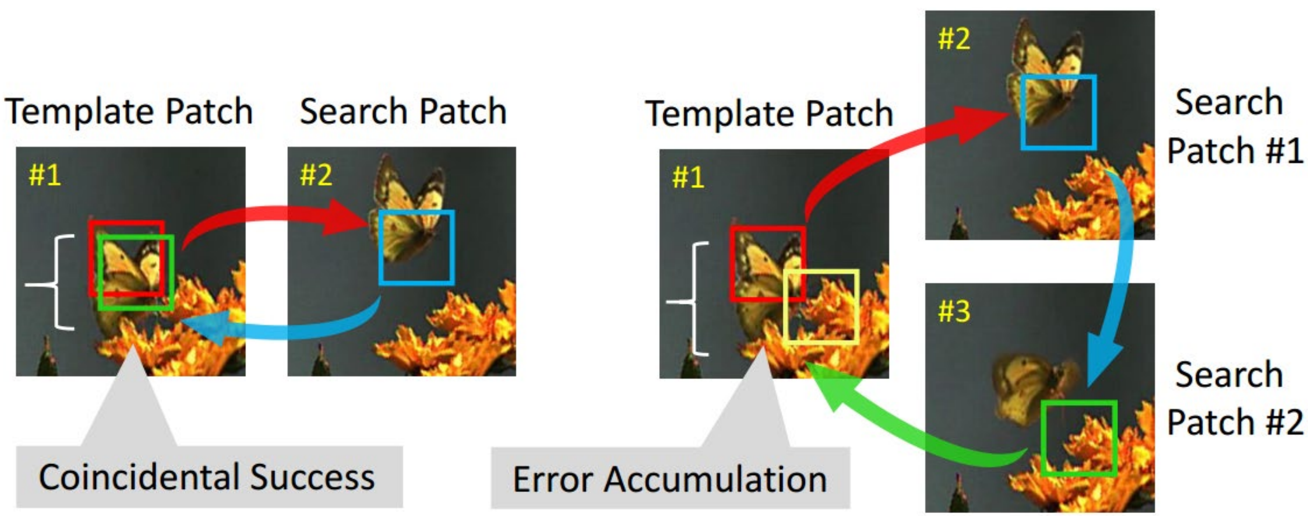
- Update the network parameters by back-propagation :

$$\frac{\partial \mathcal{L}_{\mathrm{un}}}{\partial \varphi_\theta(\mathbf{T})} = \mathscr{F}^{-1}\left(\frac{\partial \mathcal{L}_{\mathrm{un}}}{\partial\left(\mathscr{F}\left(\varphi_\theta(\mathbf{T})\right)\right)^{\star}} + \left(\frac{\partial \mathcal{L}_{\mathrm{un}}}{\partial\left(\mathscr{F}\left(\varphi_\theta(\mathbf{T})\right)\right)}\right)^{\star}\right),$$

$$\frac{\partial \mathcal{L}_{\mathrm{un}}}{\partial \varphi_\theta(\mathbf{S})} = \mathscr{F}^{-1}\left(\frac{\partial \mathcal{L}_{\mathrm{un}}}{\partial\left(\mathscr{F}\left(\varphi_\theta(\mathbf{S})\right)\right)^{\star}}\right).$$

Our Approach: Multiple Frames Validation

- The tracker may successfully return to the initial target location from a deflected or false position.
- By simply involving more search patches, the proposed consistency loss will be more effective to penalize the inaccurate localizations.



Our Approach: Cost-sensitive Loss

- To reduce the contributions of noisy pairs, we exclude 10% of the whole training pairs which contain a high loss value.
- The target with a large motion contributes more to the network training. Therefore, we compute the motion distance as follows,

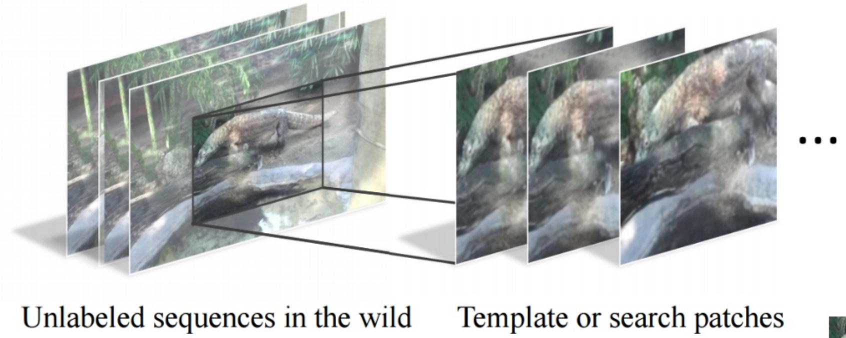
$$\mathbf{A}^{i}_{\text{motion}} = \left\|\mathbf{R}^{i}_{\mathbf{S}_1} - \mathbf{Y}^{i}_{\mathbf{T}}\right\|_2^2 + \left\|\mathbf{R}^{i}_{\mathbf{S}_2} - \mathbf{Y}^{i}_{\mathbf{S}_1}\right\|_2^2,$$

$$\mathbf{A}^{i}_{\text{norm}} = \frac{\mathbf{A}^{i}_{\text{drop}} \cdot \mathbf{A}^{i}_{\text{motion}}}{\sum_{i=1}^{n} \mathbf{A}^{i}_{\text{drop}} \cdot \mathbf{A}^{i}_{\text{motion}}},$$

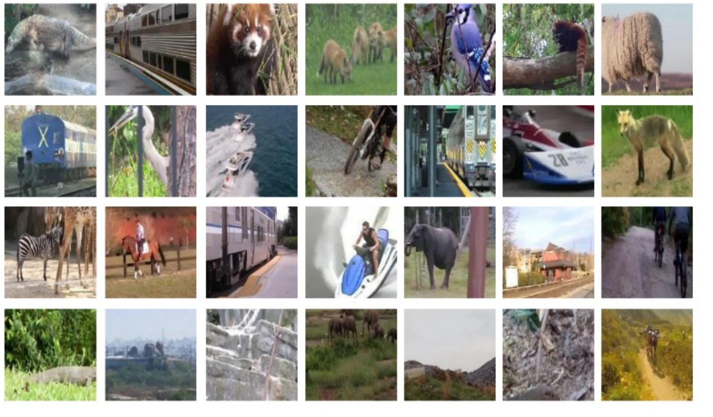
$$\mathcal{L}_{\text{un}} = \frac{1}{n}\sum_{i=1}^{n} \mathbf{A}^{i}_{\text{norm}} \cdot \left\|\widetilde{\mathbf{R}}^{i}_{\mathbf{T}} - \mathbf{Y}^{i}_{\mathbf{T}}\right\|_2^2.$$

Our Approach: Data Collection

- We simply crop the center patches in raw videos.

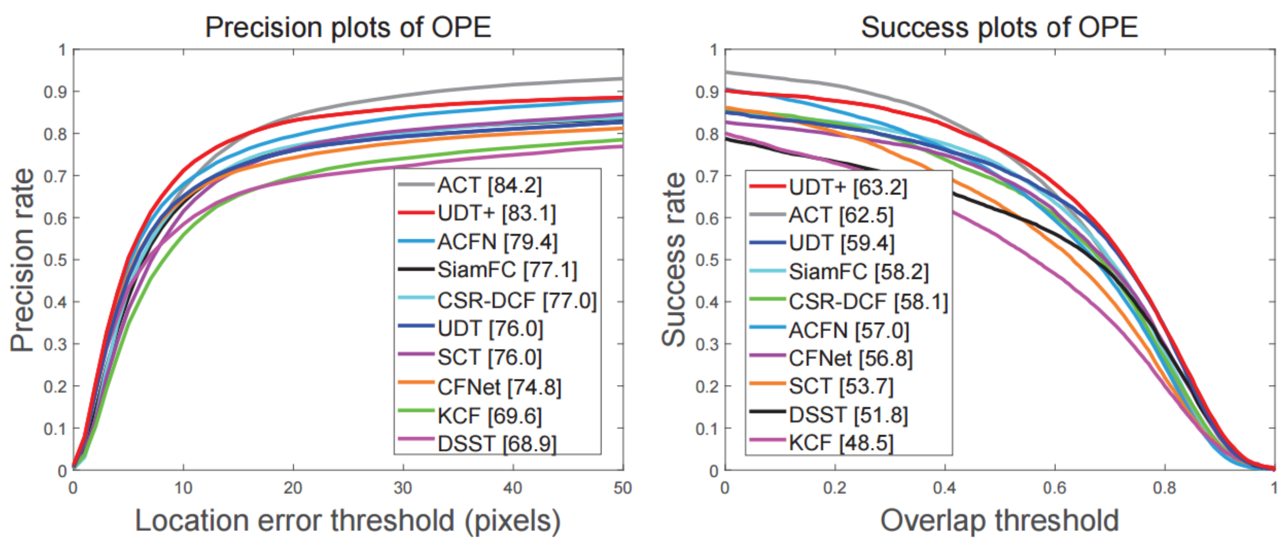


Some examples:



Experiments: State-of-the-art Comparison

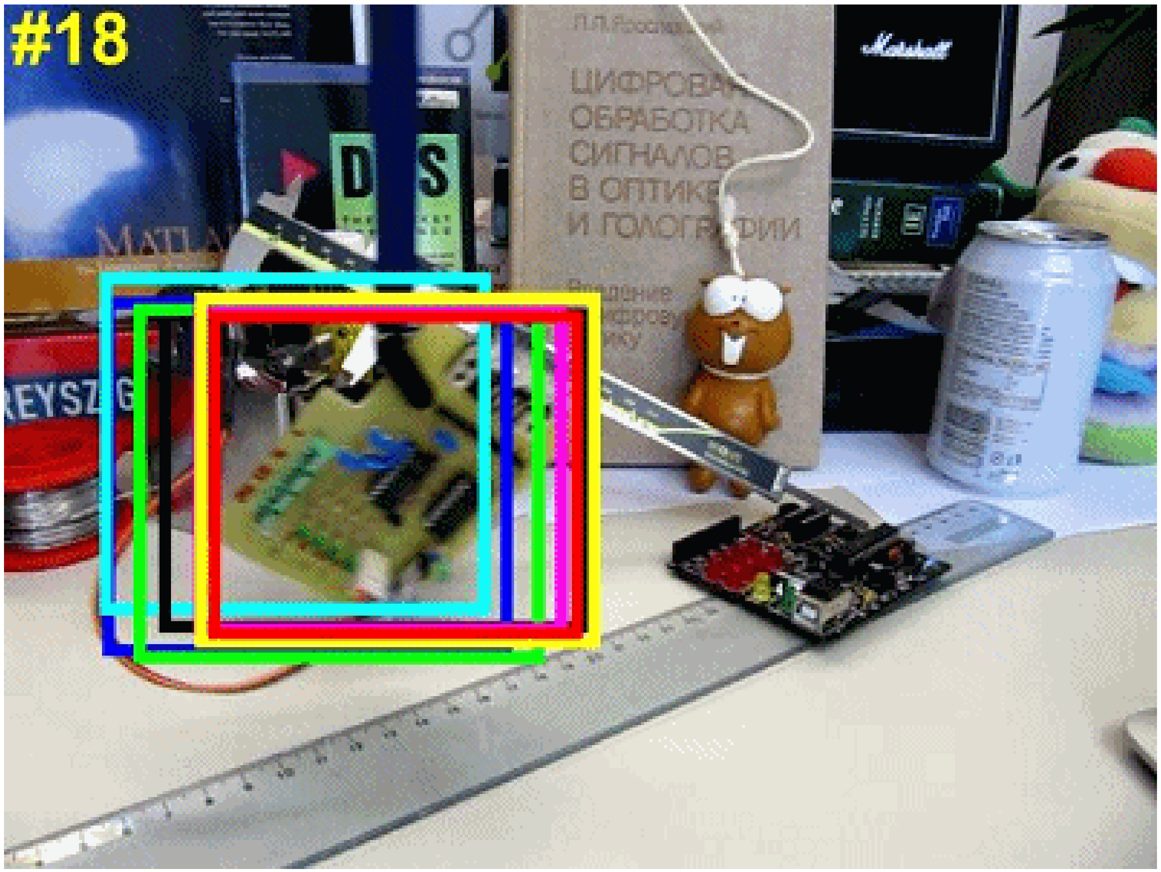
- OTB-2015 dataset: 100 challenging videos
- Our UDT tracker achieves comparable results with the baseline supervised methods. Adding some online tricks, our UDT+ achieves state-of-the-art performance



Experiments: Qualitative Comparison

UDT SiamFC CFNet ACFN DSST

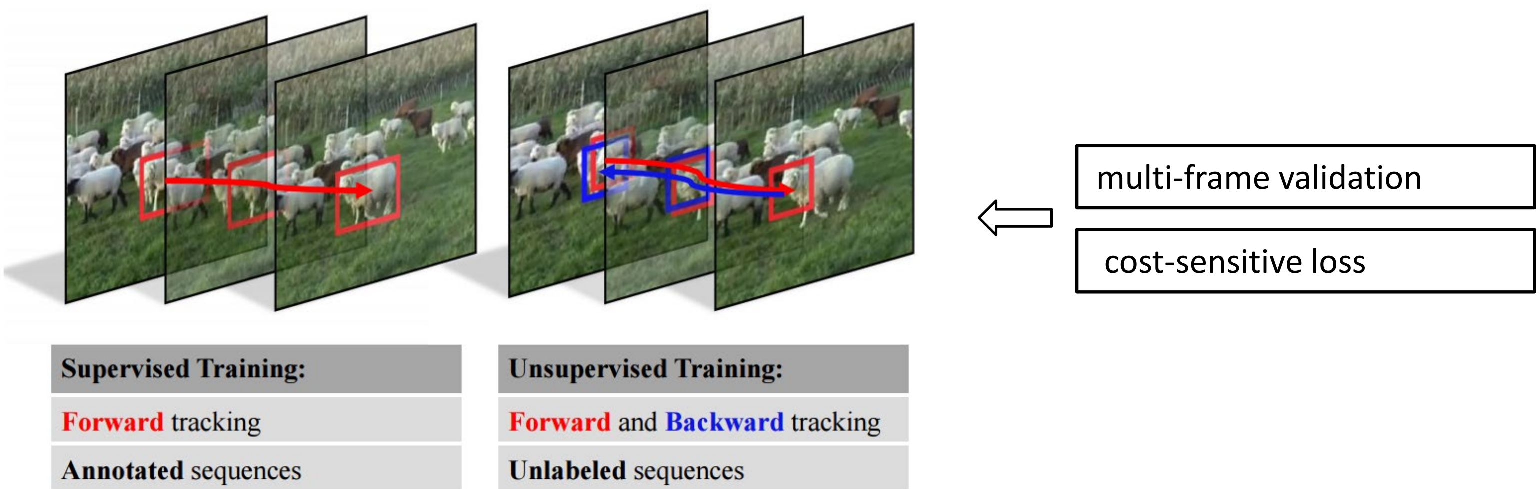
Experiments: Qualitative Comparison



UDT SiamFC CFNet ACFN DSST

Conclusion

- We propose an unsupervised tracker
- We propose a multiple frames validation and a cost-sensitive loss to facilitate the unsupervised training



Discussion

- Our method achieves baseline accuracy of the classic fully-supervised trackers

- Unsupervised framework shows potential in visual tracking:
  - ✓ For example: few-shot learning, adding more data, domain adaptation……

谢谢各位老师同学！

欢迎提问