Real-time detection and tracking of pedestrians in CCTV images using a deep convolutional neural network

Debaditya Acharya
acharyad@student.unimelb.edu.au

Kourosh Khoshelham
k.khoshelham@unimelb.edu.au

Stephan Winter
winter@unimelb.edu.au

Infrastructure Engineering, The University of Melbourne

Abstract

In this work, deep convolutional neural networks are used to automate the process of feature extraction from CCTV images. The extracted features serve as a strong basis for a variety of object recognition tasks and are used to address a tracking problem. The approach is to match the extracted features of individual detections in subsequent frames, hence creating a correspondence of detections across multiple frames. The developed framework is able to address challenges like cluttered scenes, change in illumination, shadows and reflection, change in appearances and partial occlusions. However, total occlusion and similar persons in the same frame remain a challenge to be addressed. The framework is able to generate the detection and the tracking results at the rate of four frames per second.

1 Introduction

Pedestrian tracking has gained a significant interest in the last two decades. The increasing interest is due to the availability of high-quality inexpensive CCTV video cameras and the need for automated video analysis. Recognising human actions in real-world environments finds applications in intelligent video surveillance, knowing customer attributes, customer shopping behaviour analysis (Chen et al., 2016), homeland security, crime prevention, hospitals, elderly and child care (Wang, 2013) and can be used for management of public places and handling emergency situations as well.

There is a rich literature (Yilmaz et al., 2006; Smeulders et al., 2014) that follows the conventional paradigm of pattern recognition that includes extraction of hand-crafted features (pre-defined features such as Histogram of oriented Gradients (HOG)) from the images for detecting pedestrians in a scene and their subsequent classification, using classifiers. The drawback of using such hand-crafted features for a tracking task is the limited ability of the hand-crafted features to adapt to variations of object appearance that are complex, highly non-linear and time-varying (Yilmaz et al., 2006; Chen et al., 2016). Additionally, to achieve accurate recognition, major challenges that are required to be addressed include occlusions, cluttered backgrounds, viewpoint variations, changes in appearance (scale, pose and shape), similar appearing pedestrians, illumination variations and unpredictable nature of pedestrian movements (Ji et al., 2013; Chen et al., 2016). However, most of the state-of-the-art trackers address specific challenges and the generalisation abilities of the trackers are not sufficient (Feris et al., 2013). Re-identification of pedestrians (in single camera and multi-camera views) still remains an open challenge.

In this work, pedestrians are detected in each frame of CCTV images using a state-of-the-art object detection framework Faster R-CNN (Ren et al., 2015). Subsequently, to overcome the limitations of using hand-crafted features, automatic feature extraction from the detected pedestrians with deep convolutional neural networks (CNNs) is performed. Donahue et al. (2014) state that the activations of the neurons in the late layers of a deep CNN serve as strong features for a variety of object recognition tasks. The hypothesis behind this work is that the extracted activations from the late layers of a deep CNN can be used to distinguish detected pedestrians

across the frames and can be used to address a tracking-by-detection problem accurately. So, in a novel way features are used to address a tracking problem. Tracking is formulated as the correspondence of the detections across multiple frames and is achieved by matching the extracted features of individual detections in subsequent frames. The main contributions are:

- A framework for real-time pedestrian detection and tracking using CNNs is developed
- A new algorithm is developed to establish correspondence between the detections across the frames

The framework addresses challenges such as partial occlusion, variations in illumination, changes in pose, shape and scale of pedestrians, cluttered backgrounds and total occlusions for short periods. The framework is not able to handle total occlusions of long periods and fails to address the problem of having similar appearing persons in the same frame.

2 Related work

Tracking is defined as the creation of trajectory of an object in an image plane and a tracker assigns correct labels to the tracked objects in different frames of a video. There are three fundamental aspects of pedestrian tracking that are analogous to object tracking: 1) detection of the pedestrian in the video frame, 2) tracking of the detection, and 3) analysis of the tracks for the specified purpose (Yilmaz et al., 2006). In the literature, for object detection point detectors, background subtraction methods, segmentation and supervised learning methods have been used. For accurate tracking, selection of suitable features plays a vital role and is related to object representation. Subsequently, the task of establishing correspondence of the detections is performed. This has been done in the past using deterministic or probabilistic motion models and appearance based kernel tracking models. Additionally, on-line adaptation methods have been proposed that adapt detectors to handle the variations in the appearances of the tracked objects over time. The detectors are trained and updated on-line during tracking, however these usually require a large number of instances for learning, which may not always be available. (Chen et al., 2016; Feris et al., 2013).

Recently, there has been a significant performance improvement in the field of image category classification and recognition by training a deep CNN with millions of images of different classes (Krizhevsky et al., 2012). The CNNs (Lecun et al., 1998) are a machine learning method that exploits the local spatial information in an image and learns a hierarchy of increasingly complex features, thus automating the process of feature construction. CNNs are relatively insensitive to certain variations on the inputs (Ji et al., 2013).

Motivated by the success of image classification and recognition, attempts have been made to exploit the usefulness of deep CNN for tracking tasks. Fan et al. (2010) design a CNN tracker with shift-variant architecture. The features are learned during off-line training that extracts both spatial and temporal information by considering image pairs of two consecutive frames rather than a single frame. The tracker extracts both local and global features to address partial occlusions and change in views. Ji et al. (2013) use a 3D CNN model for pedestrian action recognition. The model extracts features from both spatial and temporal dimensions by performing 3D convolutions and captures motion information across multiple frames. Jin et al. (2013) introduce a deep CNN for the task of tracking, which extracts features and transforms images to high dimensional vectors. A confidence map is generated by computing the similarities of two matches by using a radial basis function. Hong et al. (2015) propose using outputs from the last layer of a pre-trained CNN to learn discriminative appearance models using an on-line Support Vector Machine (SVM). Subsequently, tracking is performed using sequential Bayesian filtering with a target-specific saliency map, which is computed by back-projection of the outputs from the last layer. Wang et al. (2015) use features learned from a pre-trained CNN for on-line tracking. The CNN is fine-tuned during on-line tracking to adjust the appearance of an object specified in the first frame of the sequence and a probability map is generated instead of producing simple class labels. Wang and Yeung (2013) train a stacked de-noising auto-encoder off-line and follow a knowledge transfer from off-line training to on-line tracking process to adapt appearance changes of a moving target. Nam and Han (2015) propose a tracking algorithm that learns domain independent representations from pre-training, and captures domain-specific information through on-line learning during tracking. The network has a simple architecture compared to the one designed for image classification tasks. The entire network is pre-trained off-line, and the later fully connected layers including a single domain-specific layer are fine-tuned on-line. Li et al. (2016) propose a novel tracking algorithm using CNN to automatically learn the most useful feature representation of a particular target object. A tracking-by-detection strategy is followed to distinguish the target object from its background. The CNN generates scores of all possible hypotheses of object locations in a frame. The tracker learns on the samples obtained from the current image sequence. Chen et al. (2016) train a deep CNN and transfer the learned parameters

for the tracking task and construct an object appearance model. Initial and on-line training is used to update the appearance model. Despite such success of CNNs, only a limited number of tracking algorithms (discussed above) exploiting CNNs are proposed so far in the literature. Moreover, previous works have not integrated the approach of detection and tracking simultaneously with CNNs.

3 Methodology

The developed framework uses CNNs both to detect pedestrians within the frames and track across the frames. A state-of-the-art object detection framework, Faster R-CNN (Ren et al., 2015) is used for the detection of pedestrians. The features used for the tracking are derived from a pre-trained CNN (Fig. 1) and serve as a strong basis for object recognition. The proposed algorithm[1] for creating correspondence is closest to the appearance based kernel tracking, but a robust representation is developed by imposing weights for appearance and spatial information.

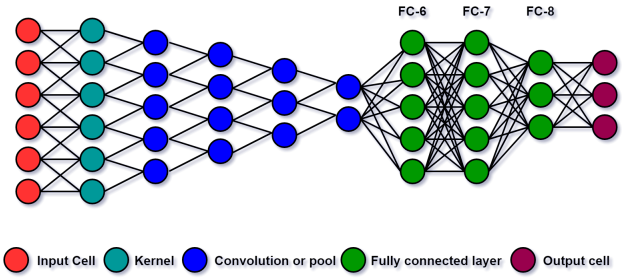

Figure 1: A simplified architecture of the deep CNN (after Krizhevsky et al. (2012)). Fully connected - 8 (FC-8) layer is the last layer before the classification, from which the features are extracted.

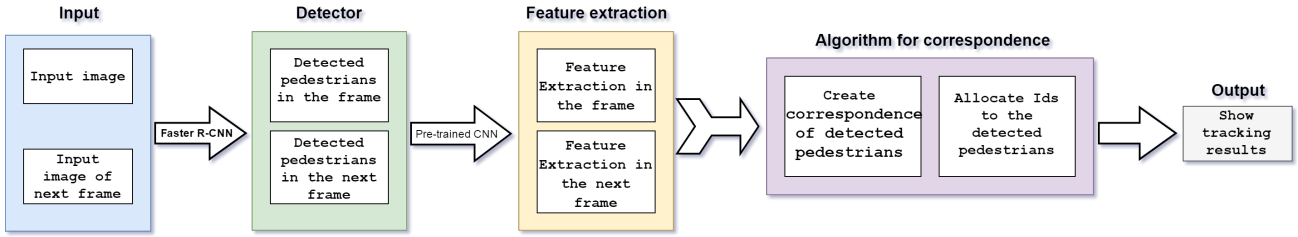

Figure 2: A simplified layout of the framework.

A simplified layout of the framework is provided in Fig. 2. The CCTV image frames are input to the detector that detects and localises individual pedestrians. Features from the cropped images of the pedestrians are extracted by a pre-trained CNN. The developed algorithm is used to make correspondences of the detections across the frames and ids are allocated to individual detections. Tracking results are shown by overlaying the ids of the detections on the respective frames.

3.1 Definitions

The last layer before the classification layer (FC-8) of the CNN generates a vector of 1000 features for each input image. Individual detections from each frame are in form of a bounding box around pedestrians. Subsequently, the detections are cropped and fed to the CNN that generates a matrix of feature vectors. Mathematically this can be represented as Eq. 1, where $\vec{FV_{i(k)}}$ denotes the matrix of feature vectors of the i detections for a single Frame k, the set $\{A_{(1,i(k))}, ..., A_{(1000,i(k))}\}$ denotes the activations of i^{th} detection in Frame k, and $i(k)$ denotes the number of detections in Frame k.

$$\vec{FV_{i(k)}} = \begin{bmatrix} A_{(1,1)} & \cdots & A_{(1,i(k))} \\ \vdots & \ddots & \vdots \\ A_{(1000,1)} & \cdots & A_{(1000,i(k))} \end{bmatrix} \quad (1)$$

$$\vec{PC_{i(k)}} = \begin{bmatrix} x_{(1,k)} & \cdots & x_{(i,k)} \\ \vdots & \ddots & \vdots \\ y_{(1,k)} & \cdots & y_{(i,k)} \end{bmatrix} \quad (2)$$

The centroids of the detections can be expressed by Eq. 2. Where $\vec{PC_{i(k)}}$ denotes the matrix of x and y coordinates of the centroids of i detections in Frame k. Correspondence is established by calculating a feature distance and a pixel distance between every pair of detections in two consecutive frames. Let $FV_{i(k)}$ and $FV_{j(k+1)}$ denote respectively the feature vectors for i and j detections in Frame k and Frame $k+1$. The normalised feature

[1]For MATLAB implementation visit *https://github.com/debaditya-unimelb/CNN pedestrian tracking/*.

distance between the two detections $F_{d(i(k),j(k+1))}$ is expressed as Eq. 3, where $|FV|$ denotes l^2-norm of a real vector FV. Let $PC_{i(k)}$ and $PC_{j(k+1)}$ denote the centroids for i and j detections in Frame k and Frame $k+1$ respectively. Similarly the normalised pixel distance between the two detections $P_{d(i(k),j(k+1))}$ is expressed as Eq. 4.

$$F_{d(i(k),j(k+1))} = \frac{|F\vec{V}_{i(k)} - F\vec{V}_{j(k+1)}|}{|F\vec{V}_{i(k)}||F\vec{V}_{j(k+1)}|} \quad (3) \qquad P_{d(i(k),j(k+1))} = \frac{|P\vec{C}_{i(k)} - P\vec{C}_{j(k+1)}|}{|P\vec{C}_{i(k)}||P\vec{C}_{j(k+1)}|} \quad (4)$$

A distance matrix $F_{d(k+1)}$ for the feature vectors is generated from the normalised pairwise feature distances and is represented by Eq. 5. A distance matrix for the pixel distances $P_{d(k+1)}$ is generated from the normalised pairwise pixel distances and is represented by Eq. 6. The matrices $F_{d(k+1)}$ and $P_{d(k+1)}$ are combined using a weight w $(0 \leq w \leq 1)$. The combination result is called a tracking matrix $T_{d(k+1)}$ and is defined by Eq. 7. Where $t_{i(k),j(k+1)}$ represents the weighted additions of $F_{d(i(k),j(k+1))}$ and $P_{d(i(k),j(k+1))}$.

$$F_{d(k+1)} = \begin{bmatrix} F_{d(1,1)} & \cdots & F_{d(1,j(k+1))} \\ \vdots & \ddots & \vdots \\ F_{d(i(k),1)} & \cdots & F_{d(i(k),j(k+1))} \end{bmatrix} \quad (5) \qquad P_{d(k+1)} = \begin{bmatrix} P_{d(1,1)} & \cdots & P_{d(1,j(k+1))} \\ \vdots & \ddots & \vdots \\ P_{d(i(k),1)} & \cdots & P_{d(i(k),j(k+1))} \end{bmatrix} \quad (6)$$

$$T_{d(k+1)} = (w)P_{d(k+1)} + (1-w)E_{d(k+1)} = \begin{bmatrix} t_{(1,1)} & \cdots & t_{(1,j(k+1))} \\ \vdots & \ddots & \vdots \\ t_{(i(k),1)} & \cdots & t_{(i(k),j(k+1))} \end{bmatrix} \quad (7)$$

3.2 Algorithm

In the first frame, the ids are generated randomly and tracked in the subsequent frames. The number of generated ids in the first frame is equal to the number of detections. For the detections in the subsequent frames either an id is assigned from the previous frame (which involves the matching based on the minimum distance criteria) or a new id is generated (which is for the case a new person enters the frame).

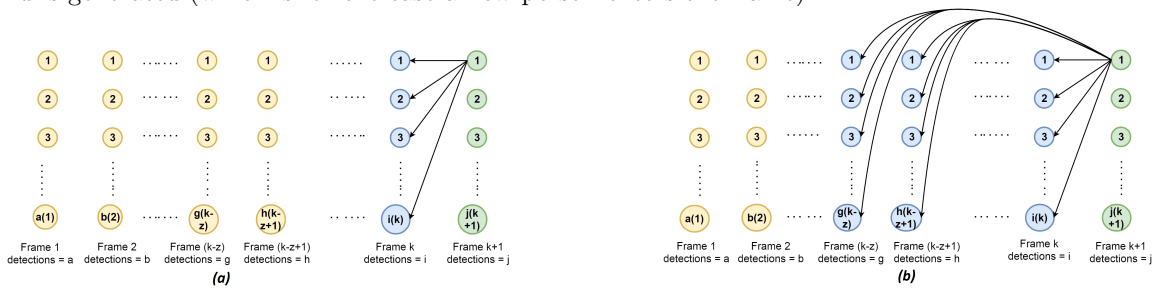

Figure 3: The algorithm used for correspondence. Green colour represents the unallocated detections. Blue colour represents the detections of the previous frame/ frames for comparing. Yellow colour represents the rest of the detections that are present in the database.

Let the set $\{t_{(1,i(k))}, ..., t_{(i(k),j(k+1))}\}$ denote the weighted distances from the detection j in Frame $k+1$ to all detections in Frame k. The minimum value of the set $\{t_{(1,i(k))}, ..., t_{(i(k),j(k+1)}\}$ is used to make correspondence of $j(k+1)^{th}$ detection in Frame $k+1$ to the $1^{st}, ..., i(k)^{th}$ detections in Frame k, only if this minimum value is below a threshold. Fig. 3(a) illustrates the process of establishing correspondences for this case, where detection 1 of frame $k+1$ is compared with $i(k)$ detections of Frame k for a correspondence. If the minimum value of the set $\{t_{(1,i(k))}, ..., t_{(i(k),j(k+1))}\}$ for a detection $j(k+1)$ in Frame $k+1$ is above the threshold, no correspondence is made to the Frame k, but the detection is compared to the detections of previous z frames for a match. This is explained in Fig. 3(b), where detection 1 of frame $k+1$ is compared with all the detections from Frame k to Frame $k-z$ and each frame can contain different number of detections (a,b,g,h,i and j).

If a match is found, a correspondence of $j(k+1)^{th}$ detection is made to the corresponding id of the detection in $(k-z)^{th}$ frame . If there is no match after comparing the previous z frames, the detection is assumed as a new pedestrian entering the frame. The new pedestrian is allocated a new id and it is tracked in the subsequent frames. If a pedestrian leaves the scene or is totally occluded in Frame $k+1$, the the corresponding detection in Frame k will not have any match in Frame $k+1$, but, that id will be stored in the database for future correspondences. However, if the algorithm is able to re-identify the pedestrian after total occlusion in the z previous frames, it is allocated the corresponding id of the detection in the $(k-z)^{th}$ frame.

Multiple correspondences from $j(k+1)$ detections to $i(k)^{th}$ detection might happen, if $j(k+1) > i(k)$. Such situations may be resolved by creating a correspondence of $j(k+1)^th$ detection to the $n(k)^{th}$ detection having

the least value of the set $\{t_{(1,i(k))}, ..., t_{(i(k),j(k+1))}\}$. The correspondence of unallocated detections is done by using the second least value of the set $\{t_{(1,i(k))}, ..., t_{(i(k),j(k+1))}\}$ if it is below the threshold. If not, then the unallocated detections are compared to the detections of previous z frames for a match.

4 Results

Town centre dataset[2] was used for evaluation in the study. First 30 seconds of the video was used at a reduced frame rate of 8 frames per second. The detection and tracking are evaluated separately using tracking matrices. The results of the detections and tracking are shown in Fig. 4.

Figure 4: (a) shows the detections in video sequence that are 10 frames apart. (b) shows the tracking results. The number denoting each pedestrian is generated randomly in the first frame.

4.1 Evaluation of the detector and the tracker

The average precision and recall of the detector for used data are 0.93 and 0.80 respectively. High precision means that most detected objects are actually pedestrians and high recall means that most pedestrians in the scene are detected. The variation of precision and recall with frames is shown in Fig. 5.

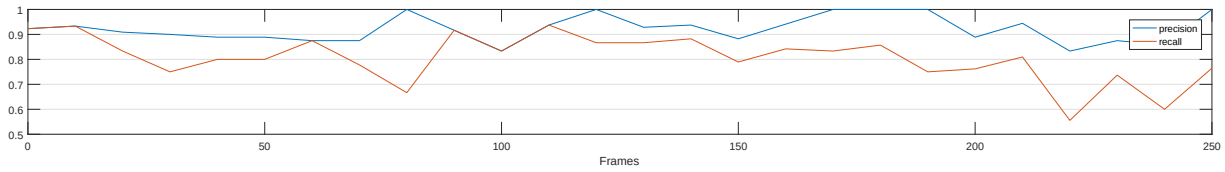

Figure 5: The variation of precision and recall with frames.

Multi-object tracking precision (MOTP) and multi-object tracking accuracy (MOTA) (Bernardin and Stiefelhagen, 2008) matrices are used for the evaluation of the tracker. The matrices are used for objective comparison of tracker characteristics on their precision in estimating object locations, their accuracy in recognising object configuration and their ability to consistently label objects over time. MOTP and MOTA are expressed mathematically as:

$$MOTP = \frac{\sum_{i,t} d_t^i}{\sum_t C_t} \quad (8) \qquad MOTA = 1 - \frac{\sum_t (m_t + fp_t + mme_t)}{\sum_t g_t} \quad (9)$$

Where d_t^i is the distance between the detection and the i^th pedestrian (from the ground truth) and C_t is the number of matches found in time t. m_t, fp_t and mme_t are the number of misses in detection (false negatives), number of false positives and the number of mismatches in the correspondence respectively, and g_t represents the number of pedestrians present at time t. MOTP is the total error in estimated position of detections over all frames, averaged by the number of correspondences made. Higher value of MOTP signifies low accuracy of the bounding boxes around the object. Higher values of MOTA signifies high accuracy in tracking. Experimental results of MOTP and MOTA for the dataset are 27.92 pixels and 71.13 % respectively.

5 Discussion

Tracking is achieved by creating a correspondence of detections of two consecutive frames only (provided that there are no multiple correspondences or correspondence to z previous frames). Hence, the appearance of pedestrians are updated over time and the framework is robust to change in appearance (pose, shape and scale). The detector misses some of the detections due to total occlusions and hence explains the low value of recall. Another contributor to lower recall values is that the detector misses pedestrians that appear smaller due to their distance to the camera. This can be alleviated in a multi-camera setting, where pedestrians that are missed in one camera are likely to be detected in another camera. On a closer observation, the low value of precision

[2]available at: *http://www.robots.ox.ac.uk/ActiveVision/Research/Projects/2009bbenfold headpose/project.html*

is due to the false detections created by the reflection of the pedestrians in a glass panel that is present in the dataset. High value of MOTP is due to the inaccuracy of the bounding boxes of the detected pedestrians. This is insignificant considering the high resolution of the dataset. Low value of MOTA is mainly due to the large number of misses in the detection and partially due to the false detections and mismatches in the correspondence.

6 Conclusion

A framework is developed for real-time detection and tracking of pedestrians in CCTV image frames using CNNs. A new algorithm is developed for making correspondence of the detections across multiple frames. The detector is able to overcome the challenges of variations in the illumination, cluttered backgrounds, partial occlusions and changes in the scale. The tracking algorithm is able to track pedestrians with 71.13 % accuracy and addresses the problem of changes in appearance (pose and shape) and total occlusions for short periods. However, total occlusions for longer periods remains a challenge to be addressed for future work. To improve the accuracy, it is proposed to perform the evaluation and estimation of pedestrians' future trajectories from past observations (e.g. Kalman filtering) for overcoming the problem of unpredictable pedestrian movements. To address the problem of total occlusions and similar persons, an average representation of individual pedestrians (for all the tracked frames) can be used.

Acknowledgements

This research was supported by a Research Engagement Grant from the Melbourne School of Engineering and the Melbourne Research Scholarship. The authors thank Active Vision Laboratory, Department of Engineering Science, University of Oxford for the publicly available dataset and ground-truth data.

References

Bernardin, K. and R. Stiefelhagen (2008). Evaluating multiple object tracking performance: The clear mot metrics. *EURASIP Journal on Image and Video Processing 2008*(1), 246309.

Chen, Y., X. Yang, B. Zhong, S. Pan, D. Chen, and H. Zhang (2016). Cnntracker: Online discriminative object tracking via deep convolutional neural network. *Applied Soft Computing 38*, 1088 – 1098.

Donahue, J., Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, and T. Darrell (2014). Decaf: A deep convolutional activation feature for generic visual recognition. In *International Conference on Machine Learning*, pp. 647–655.

Fan, J., W. Xu, Y. Wu, and Y. Gong (2010). Human tracking using convolutional neural networks. *IEEE Transactions on Neural Networks 21*(10), 1610–1623.

Feris, R., A. Datta, S. Pankanti, and M. T. Sun (2013). Boosting object detection performance in crowded surveillance videos. In *IEEE Workshop on Applications of Computer Vision*, pp. 427–432.

Hong, S., T. You, S. Kwak, and B. Han (2015). Online tracking by learning discriminative saliency map with convolutional neural network. *arXiv preprint arXiv:1502.06796*.

Ji, S., W. Xu, M. Yang, and K. Yu (2013). 3d convolutional neural networks for human action recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence 35*(1), 221–231.

Jin, J., A. Dundar, J. Bates, C. Farabet, and E. Culurciello (2013). Tracking with deep neural networks. In *Information Sciences and Systems (CISS), 2013 47th Annual Conference on*, pp. 1–5.

Krizhevsky, A., I. Sutskever, and G. E. Hinton (2012). Imagenet classification with deep convolutional neural networks. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger (Eds.), *Advances in Neural Information Processing Systems 25*, pp. 1097–1105. Curran Associates, Inc.

Lecun, Y., L. Bottou, Y. Bengio, and P. Haffner (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE 86*(11), 2278–2324.

Li, H., Y. Li, and F. Porikli (2016). Deeptrack: Learning discriminative feature representations online for robust visual tracking. *IEEE Transactions on Image Processing 25*(4), 1834–1848.

Nam, H. and B. Han (2015). Learning multi-domain convolutional neural networks for visual tracking. *Computing Research Repository abs/1510.07945*.

Ren, S., K. He, R. Girshick, and J. Sun (2015). Faster r-cnn: Towards real-time object detection with region proposal networks. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett (Eds.), *Advances in Neural Information Processing Systems 28*, pp. 91–99. Curran Associates, Inc.

Smeulders, A. W., D. M. Chu, R. Cucchiara, S. Calderara, A. Dehghan, and M. Shah (2014). Visual tracking: An experimental survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence 36*(7), 1442–1468.

Wang, N., S. Li, A. Gupta, and D. Yeung (2015). Transferring rich feature hierarchies for robust visual tracking. *Computing Research Repository abs/1501.04587*.

Wang, N. and D. Y. Yeung (2013). Learning a deep compact image representation for visual tracking. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger (Eds.), *Advances in Neural Information Processing Systems 26*, pp. 809–817. Curran Associates, Inc.

Wang, X. (2013). Intelligent multi-camera video surveillance: A review. *Pattern Recognition Letters 34*(1), 3 – 19.

Yilmaz, A., O. Javed, and M. Shah (2006). Object tracking: A survey. *ACM Computing Surveys 38*(4).