Object Detection and Person Re-Identification in Video

Shivani Nandani
DA-IICT
Gandhinagar, India
201801076@daiict.ac.in

Rohin Nanavati
DA-IICT
Gandhinagar, India
201801108@daiict.ac.in

Manish Khare
DA-IICT
Gandhinagar, India
manish_khare@daiict.ac.in

***Abstract*—This project focuses on the problem of detection and re-identification in surveillance videos. Object detection and person re-identification involves tracking of multiple objects throughout a video. This is a two step process – detection of object and re-identification to track the object. Both these steps are fundamental computer vision problems with application in domains such as autonomous driving and pedestrian tracking. We create a pipeline that divides the task into multiple sub-tasks. We explore several existing models for detection and re-identification in this project.**

***Index Terms*—object detection, person re-identification**

I. Introduction

In today's world of increased surveillance there comes a need for an automated system to detect and identify objects appearing in the video footage. This challenging problem has gained much interest from the research community in recent years.

In this project, we intended to work on this problem by dividing it into sub-tasks – object detection and re-identification. For each of these sub-tasks, we use existing architecture to extract features, followed by similarity detection using cosine similarity and euclidean distance.

II. Literature Review

A. Object Detection

Object detection is one of the primary problems in the field of computer vision. It is the basis of many other computer vision tasks such as image segmentation, object tracking and image captioning. Unlike humans, the computer needs to be trained to be able to identify the class or category of the object presented to it. To tackle this task, various methods have been proposed in the past 20 years[30].

In 2001, Viola and Jones proposed a detector based on the sliding windows method[18]. Since 2012, convolution networks started being used instead of traditional feature extraction methods since deep convolution networks were able to learn robust and higher level feature representations of a given image. RCNN[7] extracted object proposals using selective search, which was then fed to a CNN model. Successors of RCNN, namely Fast RCNN[6] and Faster RCNN[17], enabled simultaneous training of a detector and a bounding box regressor under the same network conditions. Faster RCNN[17] also became the first near-realtime deep learning detector. However, these convolution network-based architectures used a two stage method – proposal detection and verification. You Only Look Once (YOLO)[14] was introduced as a one-stage detector that applies a neural network to the whole image. It divides the image into regions and predicts bounding boxes and their probabilities simultaneously. Several versions of YOLO have been published since it was first introduced.

The original version of YOLO[14] (YOLOv1) uses 24 convolution layers followed by two fully connected layers. YOLOv2[15] added batch normalization and anchor box architecture to the previous model. YOLOv3[16] used a combination of Darknet-53 (53 convolution layers) and Residual Networks (ResNet). YOLOv4[2] replaces the residual blocks with dense blocks, which helps preserve fine-grained features. YOLOv5[9] uses an architecture similar to YOLOv4[2] with adaptive anchor boxes so that the network learns the best anchor boxes and uses them during training.

B. Re-identification

Re-identification is the task of identifying objects that have been detected in the past as the very same object. For example, if object 10 left the frame in the 100^{th} frame, and re-enters in the 250^{th} frame the model should recognise it as object 10 and not a new object. Alternatively, re-identification also comes into play when a scene is being seen through multiple camera views and objects need to be identified consistently across all the views.

Performing re-identification by modelling spatial visual appearance (shape, texture and colour) is extremely challenging. Change in illumination, the person's pose, the angle of view and the background, or even occlusion of the person can lead to failure in re-identification of said person.[12]

Research in re-identification has accelerated since 2019[22] owing to the application of deep learning based systems[28]. There are five kinds of different deep learning based architectures – classification models, verification models, triplet-based models, part-based models and attention-based models. As the name suggests, classification models treat the problem as a multi-class classification problem[29]. Verification models treat the problem as a binary classification problem, taking a pair of input and classifying them as either the same or different[23]. Triplet models take three inputs - an anchor, a positive and a negative image. Triplet loss aims to decrease

the distance between the positive image and the anchor, and increase the distance between the negative and the anchor in the feature space[5]. Part-based models aim to differentiate between people using small inter-class variations by focusing on sub-regions within the input[24]. Attention models highlight regions of high interest holding highly discriminative features within the input[1].

III. Proposed Method

To tackle the given problem, we propose the architecture provided in Fig. 1. Here, we have divided the problem into multiple sub-tasks that are individually implemented.

As seen in Fig. 1, we first divide the input video into multiple frames, which are stored in a directory for future use. These frames are then individually passed through the object detection mechanism. The output of this step gives the annotated version of the frames (detected objects are marked by bounding boxes, class name and confidence) and a *.csv* file that contains information about the detected objects. These detected objects' frames are combined into a video and saved as a file named *detection.avi*. The detection results are used to create a dataset of all persons detected in the video which is fed to the re-identification mechanism. The re-identification task creates embeddings for each of these images and uses a similarity matrix to find matches. IDs are assigned to each unique person and the results are used to create bounding boxes on the frames. These frames are finally converted back to a video for better visualization.

A. Object Detection

Object detection was performed using YOLOv5[9]. We fed the frames created from the video to the detection algorithm and the results were saved in a *.csv* file containing the coordinates of the bounding boxes for each detected object.

We also experimented with YOLOv3[16] and YOLOv4[2] for the same task. However, YOLOv5[9] works better[8] as compared to YOLOv3[16] and YOLOv4[2].

The model loaded from `ultralytics/yolov5` is trained using the Common Objects in Context (COCO) dataset[11]. COCO[11] is a standard dataset for comparison of tasks such as object detection, segmentation, key-point detection, and captioning containing 328K images. It contains 80 object categories for object detection.

B. Re-identification

We have performed re-identification using two methods – Torchreid[26] and Centroids-reid[21].

We use the pre-trained models of the aforementioned methods to extract features from previously detected objects and use these features to re-identify objects using cosine similarity and euclidean distance.

Our approaches included limiting our search of previously identified objects to the previous frame in order to improve re-identification on a frame-to-frame basis. An obvious flaw in this system is that any occlusions and/or exits and re-entry

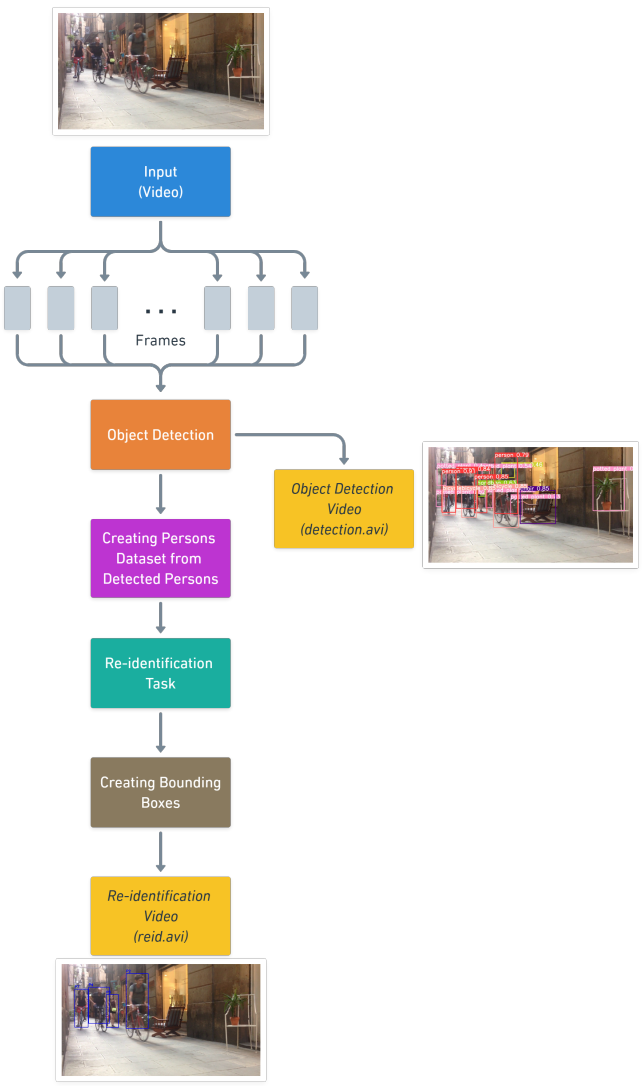

Fig. 1. Pipeline used for object detection and person re-identification

(from and to the frame) would lead to the object being branded with a new identity.

Another approach explored was to compare the detected object first with only the identified objects of the previous frame to leverage the continuity of a video. However, this resulted in poor results.

Finally, we also tried eliminating the possibility of duplicates (same identity) in a frame by finding the next best match for the object if the identity it wanted to assume had already been claimed. However, this too lead to poor results.

Our final approach, Algorithm 1, gave the best results, but it is not perfect by a long shot. While it allows for occlusions and/or exits and re-entry (from and to the frame), it disregards any continuity of the video. Objects in frame n have the same importance as objects in frame n-1 and frame 0, to the objects in frame n+1, whereas there should be a high correlation between objects in frame n and frame n+1. Secondly, there

is no elimination of duplicates, a frame can have two objects with the same identity which is physically impossible. These problems are addressed in future work.

Algorithm 1 Algorithm used for re-identification

```
for each frame do
    for each detected object in frame do
        if frame is the first frame then
            assign a new identity to object
        else
            extract features using pre-trained model
            for all previously identified objects do
                compute cosine similarity
            end for
            if similarity > certain threshold then
                assign the identity of the most similar object
            end if
            if not similar to any previously identified object then
                assign a new identity
            end if
        end if
    end for
end for
```

1) Torchreid: We used the pre-trained OSNet[27] model from the Torchreid[26] model zoo. OSNet stands for Omni-Scale Network, a ReID CNN, which learns features that not only capture different spatial scales but also encapsulate a synergistic combination of multiple scales, namely omni-scale features. The basic building block consists of multiple convolutional streams, each detecting features at a certain scale as seen in Fig. 2. What this means is that the model is trained

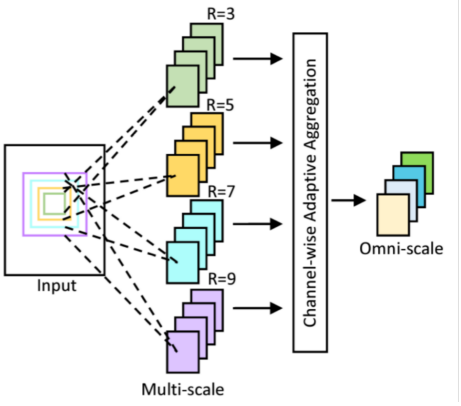

Fig. 2. A schematic of OSNet building block (R: receptive field size)

to look at local as well as global features. In Fig. 3(a) while the global features (white T-shirt and grey shorts) might be similar in all three pictures, local features such as the glasses (versus no glasses) and the sandals (versus the shoes) should help the model discriminate between classes for the query image (centre), and choose the correct class (left) as opposed to deceivingly similar classes (right). Similarly, in Fig. 3(b) the discriminative feature is the logo on the T-shirt - correct class does not have a logo (left) and the wrong class has a logo (right) - which is a feature on the smaller scale and would not be identified at the larger scale.

(a) (b)

Fig. 3. Example images for Torchreid

(a) (b)

Fig. 4. (a) Centroid based retrieval (b) Instance based retrieval

2) Centroids-reid: Deep learning models are trained to create embeddings of images such that images from the same class are closer together in the feature space. Upon receiving the query image, the model then returns the images that are most close to the embedding of the query image. Centroids-reid[21], as the name suggests, computes centroids of the images that are close together and thus represents the entire class of images with a single embedding. This changes the nature of the problem from point-to-point to point-to-centroid. This has several advantages. This approach is not only computationally efficient but also improves results compared to non-centroid-based approaches since it is less susceptible to a single-image false match. As seen in Fig. 4(a), the centroids are calculated as the mean of all the samples in a given class and the query image is assigned the nearest class (distance calculated using cosine similarity or euclidean distance). On the other hand, instance based retrieval assigns classes after calculating distance with all the images, which can lead to incorrect assignments as seen in Fig. 4(b). In this project we have used the cropped images dataset created after object detection for calculation of the centroids. These are then compared with images from the dataset to find the top k best matches. Finally, we put these matches through the re-identification algorithm explained in Algorithm 1 to assign unique IDs to each object.

IV. Dataset

We used the MOT20 benchmark[4] to test the algorithm. The MOT20 dataset contains 8 challenging video sequences (4 train, 4 test) in unconstrained environments. The dataset focuses on crowded places. It was presented at the 4th BMTT MOT Challenge Workshop at the Computer Vision and Pattern

Recognition Conference (CVPR) 2019 and has been used to test detection and re-identification algorithms in crowded places. The density in these sequences can reach up to 246 pedestrians per frame. The dataset includes indoor as well as outdoor conditions, along with day and night scenarios. Fig. 5 represents some samples from the MOT20 benchmark. Table I gives a detailed description of each sequence.

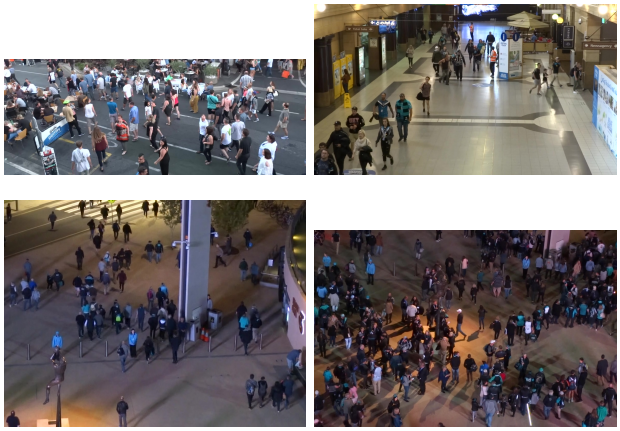

Fig. 5. Sample frames from the dataset

Name	FPS	Length	Description
MOT20-01	25	429 (00:17)	indoor
MOT20-02	25	2,782 (01:51)	indoor
MOT20-03	25	2,405 (01:36)	outdoor, night
MOT20-05	25	3,315 (02:13)	outdoor, night
MOT20-04	25	2,080 (01:23)	outdoor, night
MOT20-06	25	1,008 (00:40)	outdoor, day
MOT20-07	25	585 (00:23)	indoor
MOT20-08	25	806 (00:32)	outdoor, day

TABLE I
DETAILS OF MOT20[4] DATASET

V. OTHER EXPERIMENTS

Apart from the experiments explained above, we tested some other frameworks to better understand the problem statement and find the best possible direction for our project.

A. DG-Net

DG-Net[25] is a unified network of a learning framework that jointly couples discriminative and generative learning. We used a pre-trained model of DG-Net to create feature embeddings for re-identification. The results were below par and thus DG-Net was not further worked upon.

B. Towards-Realtime-MOT

Towards-Realtime-MOT[20] is a repository for the Joint Detection and Embedding (JDE) model – a fast and high-performance multiple-object tracker that learns the object detection task and appearance embedding task simultaneously in a shared neural network. We tried to train the model on a variety of datasets however we consistently ran out of GPU memory. With limited datasets we found little success with this model.

C. PixelLib

PixelLib[13] is a library which implements the PointRend object segmentation architecture[10] in PyTorch. We used this to perform object detection but YOLOv5[9] performed better.

D. OpenCV

OpenCV[3] is an open-source library for computer vision, machine learning and image processing. OpenCV uses Haar Cascades[19] for object detection, which uses positive and negative images to train the classifier. The results we obtained using this methods, however, were not satisfactory.

VI. RESULTS

The results for each part of the pipeline are shown in Fig. 6, Fig. 7, Fig. 8 and Fig. 9.

Fig. 6. Frame 0 of MOT-01 sequence

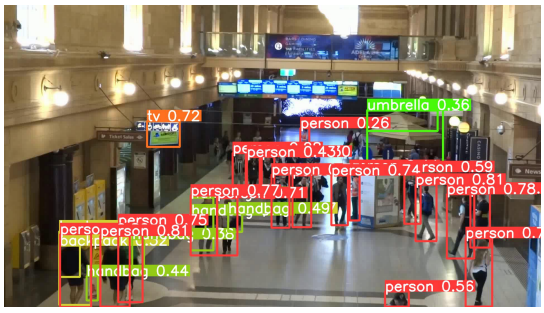

Fig. 7. Detection results for Frame 0 of MOT-01 sequence

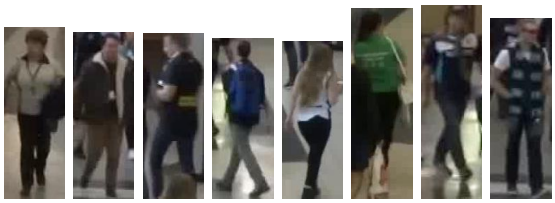

Fig. 8. Cropped individuals from Frame 0 of MOT-01 sequence

Fig. 6 shows the first frame of the MOT-01 sequence after the video has been converted to the frames. Fig. 7 shows the next step of the proposed method (Fig. 1). Here, detection has been done for all the 80 classes provided by pre-trained model of YOLOv5[9]. A filtered version of these results is

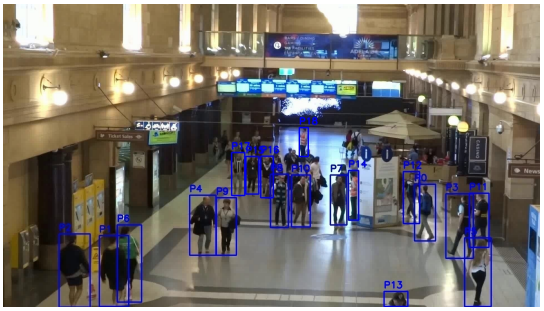

Fig. 9. Re-identification results for Frame 0 of MOT-01 sequence

used to obtain the persons dataset as seen in Fig. 8. Finally, this dataset is used to create extract features and is passed through re-identification algorithm 1. The output of this step is shown by Fig. 9.

VII. Conclusion and Future Work

The object detection algorithm we currently use fails to detect objects (persons) when they are at a very low resolution or when more than 50% of the object is not visible. These problems will have to be addressed in order to improve the current model. One way of working around this issue is to train the model with a dataset that has low resolution images and high occlusion ratio.

The re-identification algorithm should take into account the continuity of a video. A proposed solution for this is to attach a weight to the similarity metric. A high similarity with an object detected in the previous frame should take precedence over a high similarity with an object detected in the very first frame. This could be achieved by multiplying the frame number with the similarity metric. This solution comes with its own problems. A weighted similarity would greatly (negatively) affect the re-identification of objects re-entering the frame at a later time. The algorithm should not allow duplicate identities in the same frame without a significant drop in performance. The models used in our project fail to re-identify people when they change clothes. This is an issue in long-term re-identification when the footage of several days is being analysed.

Our project is limited to the view from a single camera. This is not always the case when it comes to surveillance videos. This project can be expanded to include object detection and re-identification for multiple camera views.

Finally, the ultimate goal is to create an online system which performs object detection and re-identification in real-time.

VIII. Acknowledgement

We thank Professor Manish Khare for giving us this opportunity to work on a relevant problem in the field of computer vision. We also thank him for providing us with constant guidance as well as access to the necessary technology required for the project.

References

[1] T. Bao, B. Wang, S. Karmoshi, C. Liu, and M. Zhu, "Learning discriminative features through an individual's entire body and the visual attentional parts for person re-identification," *International journal of innovative computing, information & control: IJICIC*, vol. 15, pp. 1037–1048, Jun. 2019. DOI: 10.24507/ijicic.15.03.1037.

[2] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "Yolov4: Optimal speed and accuracy of object detection," *arXiv*, 2020. DOI: 10.48550/ARXIV.2004.10934.

[3] G. Bradski, "The OpenCV Library," *Dr. Dobb's Journal of Software Tools*, 2000.

[4] P. Dendorfer, H. Rezatofighi, A. Milan, *et al.*, "Mot20: A benchmark for multi object tracking in crowded scenes," *arXiv*, 2020. DOI: 10.48550/ARXIV.2003.09003.

[5] S. Ding, L. Lin, G. Wang, and H. Chao, "Deep feature learning with relative distance comparison for person re-identification," *Pattern Recognition*, vol. 48, no. 10, pp. 2993–3003, Oct. 2015, ISSN: 0031-3203. DOI: 10.1016/j.patcog.2015.04.005.

[6] R. Girshick, "Fast r-cnn," in *2015 IEEE International Conference on Computer Vision (ICCV)*, 2015, pp. 1440–1448. DOI: 10.1109/ICCV.2015.169.

[7] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *2014 IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 580–587. DOI: 10.1109/CVPR.2014.81.

[8] P. Jiang, D. Ergu, F. Liu, Y. Cai, and B. Ma, "A review of yolo algorithm developments," *Procedia Computer Science*, vol. 199, pp. 1066–1073, 2022, The 8th International Conference on Information Technology and Quantitative Management (ITQM 2020 & 2021): Developing Global Digital Economy after COVID-19, ISSN: 1877-0509. DOI: https://doi.org/10.1016/j.procs.2022.01.135.

[9] G. Jocher, A. Chaurasia, A. Stoken, *et al.*, *ultralytics/yolov5: v6.1 - TensorRT, TensorFlow Edge TPU and OpenVINO Export and Inference*, version v6.1, Feb. 2022. DOI: 10.5281/zenodo.6222936.

[10] A. Kirillov, Y. Wu, K. He, and R. Girshick, "Pointrend: Image segmentation as rendering," in *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Los Alamitos, CA, USA: IEEE Computer Society, Jun. 2020, pp. 9796–9805. DOI: 10.1109/CVPR42600.2020.00982.

[11] T.-Y. Lin, M. Maire, S. Belongie, *et al.*, "Microsoft coco: Common objects in context," in *Computer Vision – ECCV 2014*, Cham: Springer International Publishing, 2014, pp. 740–755, ISBN: 978-3-319-10602-1.

[12] X. Ma, X. Zhu, S. Gong, *et al.*, "Person re-identification by unsupervised video matching," *Pattern Recognition*, vol. 65, Nov. 2016. DOI: 10.1016/j.patcog.2016.11.018.

[13] A. Olafenwa, "Simplifying object segmentation with pixellib library," *viXra*, 2021.

[14] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Los Alamitos, CA, USA: IEEE Computer Society, Jun. 2016, pp. 779–788. DOI: 10.1109/CVPR.2016.91.

[15] J. Redmon and A. Farhadi, "Yolo9000: Better, faster, stronger," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 6517–6525. DOI: 10.1109/CVPR.2017.690.

[16] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement," *arXiv*, 2018. DOI: 10.48550/ARXIV.1804.02767.

[17] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," in *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1*, ser. NIPS'15, Montreal, Canada: MIT Press, 2015, pp. 91–99.

[18] P. Viola and M. Jones, "Robust real-time object detection," *International journal of computer vision*, vol. 4, no. 34-47, p. 4, 2001.

[19] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*, vol. 1, 2001, pp. I–I. DOI: 10.1109/CVPR.2001.990517.

[20] Z. Wang, L. Zheng, Y. Liu, Y. Li, and S. Wang, "Towards real-time multi-object tracking," in *Computer Vision – ECCV 2020*, Cham: Springer International Publishing, 2020, pp. 107–122, ISBN: 978-3-030-58621-8. DOI: 10.1007/978-3-030-58621-8_7.

[21] M. Wieczorek, B. Rychalska, and J. Dabrowski, "On the unreasonable effectiveness of centroids in image retrieval," in *Neural Information Processing*, Cham: Springer International Publishing, 2021, pp. 212–223, ISBN: 978-3-030-92273-3. DOI: 10.1007/978-3-030-92273-3_18.

[22] A. Yadav and D. K. Vishwakarma, "Person re-identification using deep learning networks: A systematic review," *arXiv*, 2020. DOI: 10.48550/ARXIV.2012.13318.

[23] Z. Zhang and T. Si, "Learning deep features from body and parts for person re-identification in camera networks," *EURASIP Journal on Wireless Communications and Networking*, vol. 2018, Mar. 2018. DOI: 10.1186/s13638-018-1060-2.

[24] Y. Zheng, H. Sheng, Y. Liu, l. Kai, W. Ke, and Z. Xiong, "Learning irregular space transformation for person re-identification," *IEEE Access*, vol. PP, pp. 1–1, Sep. 2018. DOI: 10.1109/ACCESS.2018.2871149.

[25] Z. Zheng, X. Yang, Z. Yu, L. Zheng, Y. Yang, and J. Kautz, "Joint discriminative and generative learning for person re-identification," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[26] K. Zhou and T. Xiang, "Torchreid: A library for deep learning person re-identification in pytorch," *arXiv*, 2019. DOI: 10.48550/ARXIV.1910.10093.

[27] K. Zhou, Y. Yang, A. Cavallaro, and T. Xiang, "Learning generalisable omni-scale representations for person re-identification," *TPAMI*, 2021.

[28] S. Zhou, J. Wang, D. Meng, Y. Liang, Y. Gong, and N. Zheng, "Discriminative feature learning with foreground attention for person re-identification," *IEEE Transactions on Image Processing*, vol. 28, no. 9, pp. 4671–4684, Sep. 2019. DOI: 10.1109/tip.2019.2908065.

[29] F. Zhu, X. Kong, Q. Wu, H. Fu, and M. Li, "A loss combination based deep model for person re-identification," *Multimedia Tools and Applications*, vol. 77, Feb. 2018. DOI: 10.1007/s11042-017-5009-y.

[30] Z. Zou, Z. Shi, Y. Guo, and J. Ye, "Object detection in 20 years: A survey," *arXiv*, 2019. DOI: 10.48550/ARXIV.1905.05055.