Online Multi-Object Tracking in Videos Based on Features Detected by YOLO

Younis A. Al-Arbo*, Prof.Dr. Khalil I. Alsaif *

*Department of Computer Sciences, College of Computer Sciences and Mathematics, University of Mosul, Iraq
Emails: younis.bayati@uomosul.edu.iq khalil_alsaif@uomosul.edu.iq

Article History: Received: 10 November 2020; Revised 12 January 2021 Accepted: 27 January 2021; Published online: 5 April 2021

Abstract-With the rapid development of different applications that rely on multi-object detection and tracking, significant attention has been brought toward improving the performance of these methods. Recently, Artificial Neural Networks (ANNs) have shown outstanding performance in different applications, where objects detection and tracking are no exception. In this paper, we proposed a new object tracking method based on descriptors extracted using the convolutional filters of the YOLOv3 neural network. As these features are detected and processed during the detection phase, the proposed method has exploited these features to produce efficient and robust descriptors. The proposed method has shown better performance, compared to state-of-the-art methods, by producing better predictions using less computations. The evaluation results show that the proposed method has been able to process an average of 207.6 frames per second to track objects with 67.6% Multi-Object Tracking Accuracy (MOTA) and 89.1% Multi-Object Tracking Precision (MOTP).

Keywords: *Convolutional Neural Networks; Multi-Object Detection; Multi-Object Tracking; You Only Look Once (YOLO).*

1. Introduction

In recent years, different applications, such as autonomous driving, robot navigation, video surveillance and analysis, require the use of multiple object detection and tracking. According to this importance, significant attention has been brought toward detecting and tracking techniques that can provide accurate and rapid predictions [1, 2]. With the recent outbreaks in Artificial Neural Networks (ANNs) and their outstanding performance in different tasks, You Only Look Once (YOLO) has emerged as a solution for objects detection based on these networks. As the name suggests, all the objects in an input image are detected and localized in a single pass, using the YOLO method, i.e. all the objects are detected by processing the image once, which allows rapid detections [3].

The ability of ANNs to detect features that can relate each input with the required output, i.e. distinctive features, is the main advantage of these networks, compared to earlier methods that rely on hand-crafted features, such as Local Binary Patterns (LBP) [4], Haar-like features [5] and Histogram of Oriented Gradients (HOG) [6]. The elimination of non-distinctive features in the inputs, with respect to the required output, allows the neural networks to provide faster and more accurate decisions, according to the elimination of the computations required by these features and the false positives resulting by comparing them. Hence, more efficient descriptors can be collected from these neural networks.

In this paper, we propose a new multi-object tracking method, based on the features detected by the neural network employed by YOLO detection method. The main motivation behind the proposed

method is the efficiency of the descriptors collected from neural networks, as illustrated above. These descriptors allow matching the object in different frames, which allows tracking it. Combined with the accurate localization of the objects detected by the YOLO, the use of the descriptors that are already used during the detection phase can significantly reduce the complexity of the computations required for the tracking phase.

2. Background

2.1. Related Work

With the outstanding performance of different object detection methods, especially those that rely on ANNs [3, 7, 8], tracking-by-detection methods have been able to exploit such performance to achieve better tracking. Mainly, two types of tracking-by-detection methods exist, offline and online. Despite the relatively easier tracking in offline methods, as all the frames of the video, including future frames with respect to the one being processed, are available during the analysis, several applications, e.g. autonomous driving and robot navigation, can rely only on online methods. To track an object in a frame, online methods can rely only on the information available at that frame and earlier ones, which is limited information compared to offline tracking.

The main challenge in object tracking is recognizing the relations between the objects detected in the current frame with those from earlier frames. Such an object can be one of the objects from the earlier frames and its new position represents a point in the path it is following, or a new object that has just appeared in this frame. Hence, multi-object tracking problem can be considered as Data Association (DA) problem and several earlier methods rely on using DA approaches in multi-object tracking, especially tracking-by-detection approaches [9-12]. Joint Probabilistic Data Association Filter (JPDAF) is one of the DA methods that is employed in multi-object tracking, which makes the assignment decisions by considering all the possible associations in the frame, i.e. each object detected in the current frame is compared against all the objects in the previous ones. Another DA method that is applied in multi-object tracking is the Multi-Hypothesis Tracking (MHT), in which the object is assigned based on multiple associations that the object can make with previous ones. However, the consideration of multiple paths increases the complexity of the computations, which limits the applications of this method.

With the significantly better performance achieved by the use of artificial neural networks in different applications, several multi-object tracking methods have been proposed based on these networks [13, 14]. In ANNs, the neurons are distributed in layers. The complexity of the features detected by the neuron is increased as the layer that the neuron is positioned at is deeper. As this neuron detects features in its inputs, which are collected from the previous layer, the complexity of the features is increased accumulatively [15]. Hence, ANNs with more than one hidden layer are known as deep neural networks and have the ability to detect complex features[16]. Thus, a deep neural network is employed by Wang et al. [17], which uses an auxiliary number of images to learn the generic features in the visual representation of the objects that the intended application interacts with. These features are learned using a stacked denoising autoencoder. However, this method has limited ability to provide an accurate description of the temporal invariances of the deep features.

Jiang et al. [18] propose another tracking method that relies on ANNs, specifically using Long-Short-Term Memory (LSTM) neurons. These neurons feedback their output at a certain time instance into their inputs at the next time instance. This feedback allows the neuron to detect time-relevant features. Thus, by using a reinforcement learning approach, this method predicts the next shape of the object detected at this time instance, taking into consideration the shape of the object in previous frames, i.e. time instances. This prediction allows more accurate matching when the next frame arrives, as a more accurate representation of each object is available for the matching stage. Moreover, Yan et al. [19] propose a method that also employs a Convolutional Neural Network (CNN) to measure the similarity between objects from different frames. A convolutional layer uses multi-dimensional filters to detect the features in the inputs of its neurons, which allow detection of local features. The proposed method uses a Siamese approach to train the neural network to produce a 150-value descriptor per each object. The objects are extracted from each frame, based on the prediction of the detection phase, and fed to the neural network to produce the descriptor. Then, the Euclidean distance is measured between

these descriptors as an indication of their similarity. Despite the improvement in the performance imposed by the use of ANNs in these methods, the extraction of the images of the objects from the frames and use them in separate neural networks adds significantly higher computations to the method.

2.2. You Only Look Once (YOLO) Multi-Object Detection Algorithm

The YOLO algorithm uses a CNN to detect objects in an image by dividing that image into a set of subregions. Then, per each of these regions, a descriptor that indicates the existence of an object, its positioning with respect to the region and dimensions, is generated. Accordingly, each of these descriptors contains a value that represents the confidence of an object being in that region, two values that represent the *x* and *y* coordinates with respect to the region, the width and height of the object, with respect to the width and height of the region, and a number of values equal to the number of classes that the detected objects can be from [3, 20]. For instance, if a neural network is trained to detect objects from five types, it outputs ten-value descriptors. However, according to the possibility of detecting multiple objects in a single region, anchors are used to provide the neural network with such an ability. Hence, the described sample neural network must output ten times the number of anchors per values per each region descriptor.

When a prediction is required, the confidence value per each anchor per each region is compared to a threshold value. If that value exceeds the threshold value, the region defined by the remaining four value is extracted. Then, the overlaps among the regions per each class per anchor are calculated, so that, predictions provided for the same object in different subregions are eliminated [3]. Moreover, several versions of the YOLO have been proposed to improve its performance by adopting new approaches that are being proposed for CNNs, such as the use of the Darknet-53 neural network in the most recent version of YOLO, as shown in Fig. 1 [8]. This neural network detects object at three different scales and has shown significantly better and faster predictions, compared to earlier versions of the YOLO.

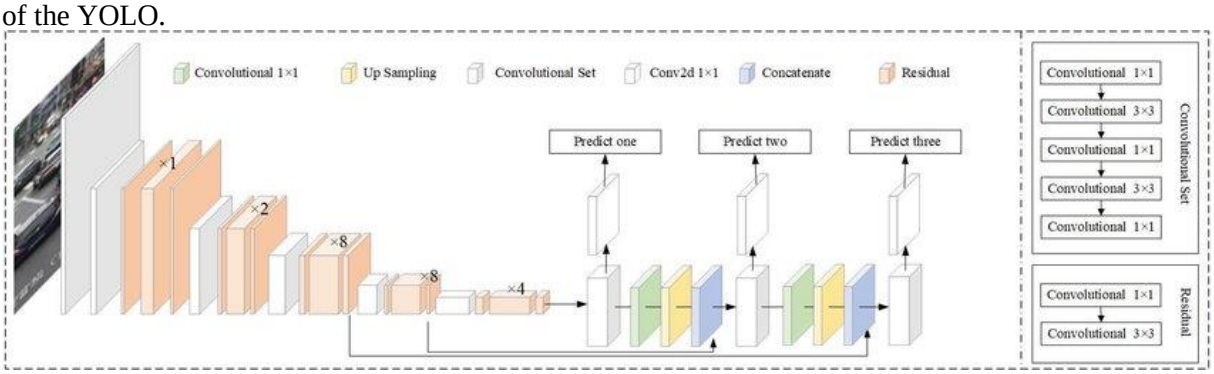

Fig. 1. Structure of the YOLOv3 [21].

3. The Proposed Online Multi Object Detection Method

The multi-object tracking method proposed in this paper relies on the features detected by the neural network of the YOLO algorithm, so that, faster and more accurate descriptors can be provided and used in the matching phase. The fully-convolutional neural network in YOLO detects objects based on their visual features in the input image, detected by the convolutional filters that learn these features during the training. However, these filters neglect the intra-class features, i.e. the features that are different in the objects of the same class, as these features are recognized to be non-distinctive toward the output during training. Such a negligence occurs in deeper layers, as these layers combine less-complex features to produce the distinctive features that they recognize. Additionally, these deeper layers contain more specific information about the object itself and its position in the region, which are also required to produce a descriptor for the detected object. Thus, both types of information must be considered to produce descriptive and distinctive descriptors. Moreover, the main challenge that the proposed method faces is the need for similar descriptors for the same object when detected at the different levels of the neural network, shown in Fig. 1. Although the layers of the neural networks are connected to each other,

the output of each level is separate from the other, as each subnetwork can find its own features to reach the required output.

3.1. Structure of the Proposed Neural Network

Per each level, a concatenation layer is placed to concatenate the output of two layers into a single layer. One of these layers is selected to be closer to the output, i.e. contains object information, while the other layer is selected from a previous layer, which contains the visual features that are being used by this layer as well as deeper ones to detect the objects, as shown in Fig2. After the concatenation, three convolutional layers are used to process the combined visual features and object information and produce a descriptor for each object. The "Target" layers reshape the inputs into three-dimensional by adding a new axis prior to the last one in the original input. This array is then also concatenated on the new axis according to the number of anchors in the trained YOLO neural network. Then, the concatenated array is processed using 128 three-dimensional convolutional filters, so that, the output of the last layer is equal to $n \times m \times a \times 128$, where $n \times m$ is the number of regions, identical to that in the corresponding output of the original neural network, a is the number of anchors that exist in the trained neural network and 128 is the number of values in each generated descriptor. The layers that are added to the first scale level in the YOLO neural network is shown in Fig. 3.

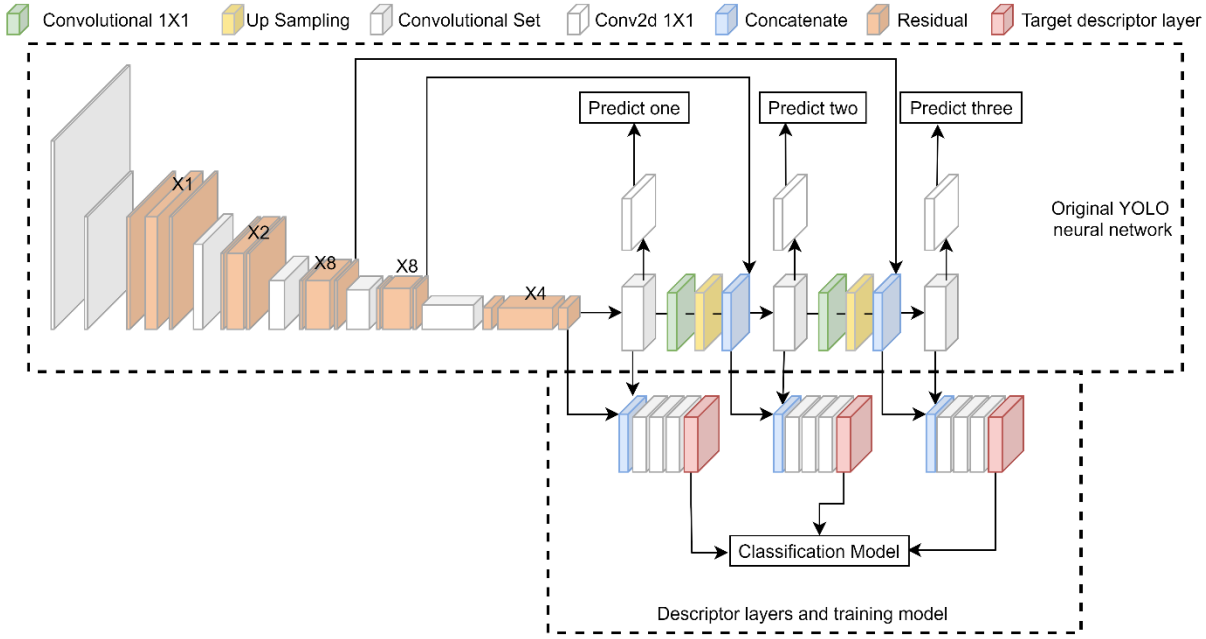

Fig. 2. Illustration of layers added to the YOLO neural network to generate objects descriptors.

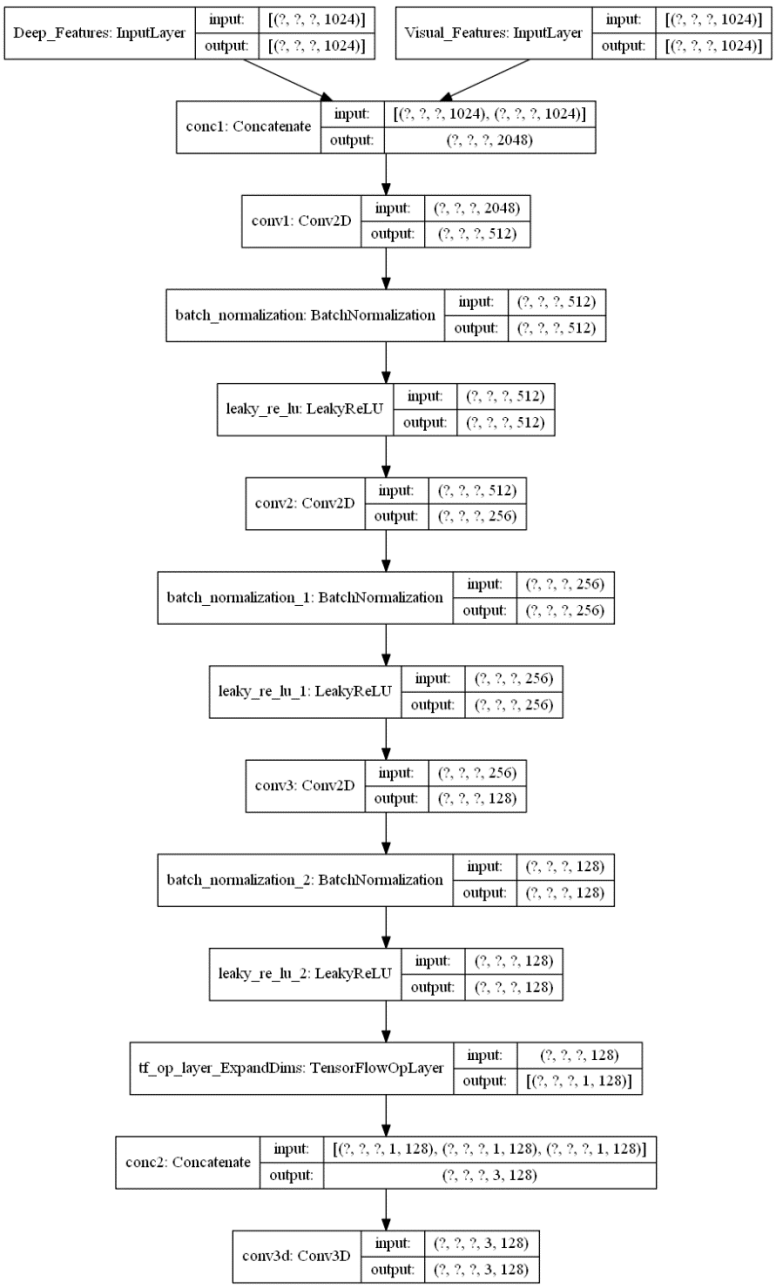

Fig. 3. Structure and output shapes of the layers added to the first level of the YOLO neural network.

To handle the challenge of producing the same descriptor values at different levels, a model is implemented and shared among the different scale levels. As shown in Fig. 4, this model contains only one three-dimensional convolutional layer that accepts the output of the implemented layers and produce $a \times C$ descriptors per each region, where C is the number of classes in the training dataset. For instance, if the proposed neural network is required to track different types of cars and there are 5953 types of cars in the training dataset, the output layer contains 5953 neurons per each anchor per region. The output of the three-dimensional convolutional layer is then passed through a soft-max activation function, as each input can only be in one class.

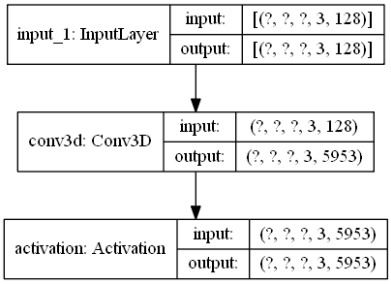

Fig. 4. Structure of the model shared by the outputs of the target layers.

3.2. Training the Neural Network

Sharing the model shown in Fig. 4 has two advantages toward training the proposed neural network: Ensures that the descriptors generated at different scale levels are similar, to reach the required output; And to allow the weights and biases of the three-dimensional convolutional layer to be updating, according to the detected relations. As the output of each layer is calculated based on its input, and as the same exact output is being required from the different levels when the same object is detected, the neural network is forced to produce similar values at the target layers to reach the same output using the same model. Hence, per each anchor per region a single value is set to one if and only if an object is detected at that anchor in the region. The position of the value of one is determined by the class of the object, i.e. one-hot-encoding, at the last dimension of the output, which is 5953 in Fig. 4. The use of the soft-max activation function proposes significant improvement to the training, as each input belongs to only one class. However, per each anchor in the region, the summation of the outputs must equal to one when this activation function is used, while several positions that do not contain any objects are labeled with zeros, i.e. the entire vector is zero. To overcome this problem, the output of the positions that do not contain objects is neglected, so that, the neural network can output any values as we do not care about these values. Hence, the formula shown below is used to calculate the loss between the predictions of the neural network and the assigned labels for each input.

$$loss = \sum \left[\sum_{r=0}^{a} \left(\sum_{i=0}^{a} \sum_{j=0}^{C} \left(y_{i,j} - \hat{y}_{i,j} \right)^2 \right)_r \times \sum_{m=0}^{C} y_{r,m} \right] \quad (1)$$

The squared error is calculated first using the $\sum_{i=0}^{a} \sum_{j=0}^{C} \left(y_{i,j} - \hat{y}_{i,j} \right)^2$ formula. Then, the summation of each row in that array is calculated and multiplied by the sum of the same row in the labels array. As the label array is one-hot-encoded when an object is detected and all values are set zero when no object is located at that anchor in the region, the summation of this row is either one or zero. Hence, by multiplying this sum with the sum of the error, the error values corresponding to anchors that contain no objects become equal to zero, i.e. neglected. The summation of the loss of all anchors and regions is then calculated and returned to train the neural network.

Additionally, to avoid affecting the performance of the neural network in the standard detection task, all the weights and biases of the layers other than those added to generate the required descriptors are frozen. Freezing these parameters indicates that they can still be used in all operations other than being updated during the training of the neural network. Hence, the training phase affect only the added layers in order to generate the required descriptors, based on the features detected by the previous layers in the neural network. Finally, after the training is complete, the model that is used to convert the descriptor values to classes is removed from the neural network, so that, the output correspondent to each vector in the original output of the YOLO has 128-value descriptor.

3.3. Multi-Object Tracking Using the Produced Descriptors

After detecting the objects in the image using YOLO algorithm, the descriptors in the positions correspondent to the positions of the outputs that are used to define the detected object are collected to represent those objects. Per each object, the last three descriptors are stored, whenever available, and used to produce a single descriptor, using median function. Then, the produced vector is used to match the objects in the new frame. Initially, when a new object is detected or at the first frame of the video, the descriptor that is generated for that object is used to match it with the objects to be detected in the

next frame. Then, when three descriptors become available, the median function is applied to the last three descriptors before matching the ones in the new frame. The use of the median function provides the proposed method with better ability to resist noise values while producing values than are more similar to dominant values in the descriptors. Hence, the proposed method adapts to the changes in the shape of the object as it is moving while eliminating any effect of noisy values that may occur in the produced descriptors.

Matching is conducted by simply measuring the Euclidean distance between each object in the new frame and the descriptors for the objects detected in previous frame. Any distance that is larger than a predefined threshold value is neglected and the corresponding objects are considered as different, i.e. new, objects. Then, the objects with the highest matches are considered as the first match and the position of the object in the new frame is considered as the new position of the object. By removing these objects from the list, in which a single object can match to more than one object from the new frame and vice versa, the next highest match, i.e. least distance, is selected for the same purpose. This procedure is repeated until all the distances that are found to satisfy the required threshold are processed. The new descriptor of the object is then appended to the list of the last three descriptors, while dropping the oldest one, to generate a new descriptor for the next frame using the median function. Moreover, any objects in the new frame that remain unpaired are considered new objects and their descriptors are stored for the next matching.

4. Experimental Results

To evaluate the performance of the proposed method, the recent University at Albany Detection and Tracking (UA-DETRAC) benchmark dataset [22] is used. This dataset contains more than 140000 frame images of 100 real-world videos that are collected using traffic sensors. These videos are split into 60 training and 40 testing videos, each is manually annotated with each vehicles ID and position, so that, the same vehicle can be tracked throughout several frames. A pretrained YOLOv3 neural network[1] is used to evaluate the proposed method, which is trained using the 60 training videos and the approach described in Section 3.2. This YOLOv3 implementation has been able to achieve 76.17% overall Average Precision (AP) using the testing set of the database. Sample of the detected and actual cars in a sample image are shown in Fig. 4.

[1] https://github.com/qqwweee/keras-yolo3

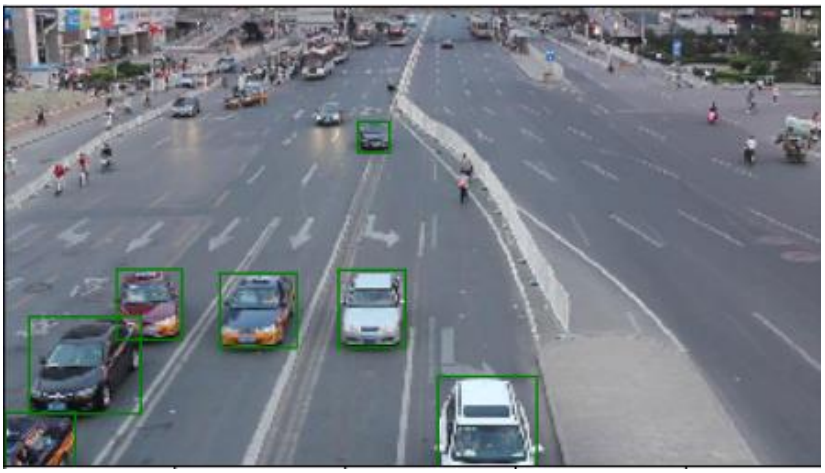

Fig. 5. Objects in a sample frame. Top: Detected by the standard YOLOv3. Bottom: Annotated by the UA-DETRAC.

The performance of the YOLOv3 in detecting the objects is beyond the interest and control of the proposed method, as a pretrained neural network is used as is. In-depth analysis of this performance is described in [22], which shows that this method has top performance in detection phase at different scenarios. However, the performance of the proposed YOLO-based tracking method is evaluated using the Multi-Object Tracking Accuracy (MOTA) and Multi-Object Tracking Precision (MOTP), as suggested by the benchmark. These performance measures for the proposed method and state-of-the-art methods are summarized in Table 1.

Table 1. Quality measures of the MOT predictions.

Detection Method	**Tracking Method**	**MOTA (%)**	**MOTP (%)**
Deformable Part-Based Model (DPB) [23]	Globally-Optimal Greedy (GOG) [24]	26.2	76.2
ACF [25]	Globally-Optimal Greedy (GOG) [24]	35.7	80.3
R-CNN [16]	Discrete Continuous Tracking (DCT) [26]	38.4	80.6
Faster R-CNN [15]	Multiple Hypothesis Tracking (MHT)	58.2	78.4
Evolving Boxes (EB) [27]	KIOU [28]	62.1	81.9
YOLOv3	Proposed	67.6	89.1

The results show that the proposed multi-object tracking method has been able to provide better predictions, which indicate better matching that is a result of more robust and distinctive descriptors. These characteristics are inherited from the YOLO neural network and the convolutional filters that are used to detect objects based on their visual features. The additional layers added to the neural network emphasize the intra-class variations, which are by default neglected by the YOLO neural network, in shallower layers as well as the object type and localization information that is detected in deeper layers. Additionally, the use of the same model in the training to force all levels to reach the same output has allowed the production of similar descriptors at different levels. Moreover, the use of multiple descriptors per each object to produce a single one that is used in the matching stage with the objects in the next frame has allowed more accurate matching. This improved accuracy is according to the moving nature of the values in the descriptors that are used with the matching and the ability of the median function to eliminate noise and extreme values.

Another important performance measure, which is the main motivation behind this work, is the execution time and is illustrated by the number of frames being processed per each second in Table 2. The significantly higher frame rate when using the proposed method is a result of using the same features detected by the YOLO neural network during the object detection in the tracking phase, in addition to the less-complex computations required by the CNN layers, compared to RNN. Moreover, instead of extracting the pixel values of the objects in the image and process them in a separate neural network, the proposed method has imposed a slight addition to the structure of the standard YOLO neural network to produce efficient descriptors that have been able to achieve good tracking performance. The number of frames that the proposed method has been able to process per each second is significantly higher than standard cameras that are normally used in different applications. Hence, the proposed method can easily be used with these cameras in online applications.

Table 2. Number of frames existing state-of-the-art methods and the proposed method have been able to process per second.

Method	**Frames per Second**
Siamese CNN [13]	2.1
RNN-LSTM [14]	166.8
JPDA [29]	35.6
LSTM-DRL [18]	108
YOLO-Based (This study)	207.6

5. Conclusion

A new online multi-object tracking method is proposed in this paper, based on descriptors extracted from the features detected by the convolutional filters of the YOLO neural network. The use of these features allows the proposed method to produce distinctive descriptors for the inputs without the need for significant additional computations, as these features are already calculated in the detection stage. With the existence of multiple levels in the YOLOv3 neural network, the production of similar values when the same object is detected at different levels was the main challenge toward the proposed method. This challenge is addressed by using a shared model that converts the values in the descriptors into classification results. Hence, the different layers are forced to produce similar values, so that, passing these values through the same model must produce the same outputs.

The evaluation results show that the proposed method has better performance, compared to the existing state-of-the-art methods, in terms of quality of the predictions. This improved quality is a result of the high-quality features detected by the YOLO neural network and the use of the median function to summarize the last three descriptors, when available, to match the object with those detected in the next frame. The use of the last three descriptors allows the values to change according to the changes in the visual features of the object, whereas the use of the median function eliminates noisy and extreme values. Additionally, the proposed method has been able to process more frames per each second, according to the low-complex computations required to produce the required descriptors and match them with the existing objects, to track them.

References

[1] M. Jiang, Z. Pan, and Z. Tang, "Visual object tracking based on cross-modality Gaussian-Bernoulli deep Boltzmann machines with RGB-D sensors," *Sensors*, vol. 17, p. 121, 2017.

[2] A. K. KC, L. Jacques, and C. De Vleeschouwer, "Discriminative and efficient label propagation on complementary graphs for multi-object tracking," *IEEE transactions on pattern analysis and machine intelligence*, vol. 39, pp. 61-74, 2016.

[3] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 779-788.

[4] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan, "Object detection with discriminatively trained part-based models," *IEEE transactions on pattern analysis and machine intelligence*, vol. 32, pp. 1627-1645, 2009.

[5] P. Viola and M. J. Jones, "Robust real-time face detection," *International journal of computer vision*, vol. 57, pp. 137-154, 2004.

[6] X. Wang, T. X. Han, and S. Yan, "An HOG-LBP human detector with partial occlusion handling," in *2009 IEEE 12th international conference on computer vision*, 2009, pp. 32-39.

[7] R. Girshick, "Fast r-cnn," in Proceedings of the IEEE international conference on computer vision, 2015, pp. 1440-1448.

[8] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement," *arXiv preprint arXiv:1804.02767*, 2018.

[9] Z. Wu, A. Thangali, S. Sclaroff, and M. Betke, "Coupling detection and data association for multiple object tracking," in *2012 IEEE Conference on Computer Vision and Pattern Recognition*, 2012, pp. 1948-1955.

[10] W. Brendel, M. Amer, and S. Todorovic, "Multiobject tracking as maximum weight independent set," in *CVPR 2011*, 2011, pp. 1273-1280.

[11] A. A. Butt and R. T. Collins, "Multi-target tracking by lagrangian relaxation to min-cost network flow," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2013, pp. 1846-1853.
[12] M. D. Breitenstein, F. Reichlin, B. Leibe, E. Koller-Meier, and L. Van Gool, "Online multiperson tracking-by-detection from a single, uncalibrated camera," *IEEE transactions on pattern analysis and machine intelligence*, vol. 33, pp. 1820-1833, 2010.
[13] L. Leal-Taixé, C. Canton-Ferrer, and K. Schindler, "Learning by tracking: Siamese CNN for robust target association," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2016, pp. 33-40.
[14] L. Leal-Taixé, A. Milan, I. Reid, S. Roth, and K. Schindler, "Motchallenge 2015: Towards a benchmark for multi-target tracking," *arXiv preprint arXiv:1504.01942*, 2015.
[15] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," in *Advances in neural information processing systems*, 2015, pp. 91-99.
[16] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014, pp. 580-587.
[17] L. Wang, T. Liu, G. Wang, K. L. Chan, and Q. Yang, "Video tracking using learned hierarchical features," *IEEE Transactions on Image Processing*, vol. 24, pp. 1424-1435, 2015.
[18] M.-x. Jiang, C. Deng, Z.-g. Pan, L.-f. Wang, and X. Sun, "Multiobject Tracking in Videos Based on LSTM and Deep Reinforcement Learning," *Complexity*, vol. 2018, 2018.
[19] E. Yang, J. Gwak, and M. Jeon, "Conditional random field (CRF)-boosting: constructing a robust online hybrid boosting multiple object tracker facilitated by CRF learning," *Sensors*, vol. 17, p. 617, 2017.
[20] L. Jing, X. Yang, and Y. Tian, "Video you only look once: Overall temporal convolutions for action recognition," *Journal of Visual Communication and Image Representation*, vol. 52, pp. 58-65, 2018.
[21] Q.-C. Mao, H.-M. Sun, Y.-B. Liu, and R.-S. Jia, "Mini-YOLOv3: real-time object detector for embedded applications," *IEEE Access*, vol. 7, pp. 133529-133538, 2019.
[22] L. Wen, D. Du, Z. Cai, Z. Lei, M.-C. Chang, H. Qi, *et al.*, "UA-DETRAC: A new benchmark and protocol for multi-object detection and tracking," *Computer Vision and Image Understanding*, vol. 193, p. 102907, 2020.
[23] J. Yan, Z. Lei, L. Wen, and S. Z. Li, "The fastest deformable part model for object detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 2497-2504.
[24] H. Pirsiavash, D. Ramanan, and C. C. Fowlkes, "Globally-optimal greedy algorithms for tracking a variable number of objects," in *CVPR 2011*, 2011, pp. 1201-1208.
[25] P. Dollar, C. Wojek, B. Schiele, and P. Perona, "Pedestrian detection: An evaluation of the state of the art," *IEEE transactions on pattern analysis and machine intelligence*, vol. 34, pp. 743-761, 2011.
[26] A. Andriyenko, K. Schindler, and S. Roth, "Discrete-continuous optimization for multi-target tracking," in *2012 IEEE Conference on Computer Vision and Pattern Recognition*, 2012, pp. 1926-1933.
[27] L. Wang, Y. Lu, H. Wang, Y. Zheng, H. Ye, and X. Xue, "Evolving boxes for fast vehicle detection," in *2017 IEEE international conference on multimedia and Expo (ICME)*, 2017, pp. 1135-1140.
[28] S. Lyu, M.-C. Chang, D. Du, W. Li, Y. Wei, M. Del Coco, *et al.*, "UA-DETRAC 2018: Report of AVSS2018 & IWT4S challenge on advanced traffic monitoring," in *2018 15th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, 2018, pp. 1-6.
[29] S. Hamid Rezatofighi, A. Milan, Z. Zhang, Q. Shi, A. Dick, and I. Reid, "Joint probabilistic data association revisited," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 3047-3055.