International Journal of Electrical and Computer Engineering (IJECE)
Vol. 10, No. 4, August 2020, pp. 3576~3587
ISSN: 2088-8708, DOI: 10.11591/ijece.v10i4.pp3576-3587

Deep-learning based single object tracker for night surveillance

Zulaikha Kadim[1], Mohd Asyraf Zulkifley[2], Nabilah Hamzah[3]
[1,2]Department of Electrical, Electronic and Systems Engineering, Faculty of Engineering and Built Environment, Universiti Kebangsaan Malaysia, Malaysia
[1]MIMOS Berhad, Technology Park Malaysia, Malaysia
[3]Faculty of Electrical Engineering, Universiti Teknologi Mara (UiTM), Malaysia

Article Info

Article history:

Received Aug 23, 2019
Revised Jan 29, 2020
Accepted Feb 7, 2020

Keywords:

Deep-learning object tracker
Night surveillance video
Visual object tracking

ABSTRACT

Tracking an object in night surveillance video is a challenging task as the quality of the captured image is normally poor with low brightness and contrast. The task becomes harder for a small object as fewer features are apparent. Traditional approach is based on improving the image quality before tracking is performed. In this paper, a single object tracking algorithm based on deep-learning approach is proposed to exploit its outstanding capability of modelling object's appearance even during night. The algorithm uses pre-trained convolutional neural networks coupled with fully connected layers, which are trained online during the tracking so that it is able to cater for appearance changes as the object moves around. Various learning hyperparameters for the optimization function, learning rate and ratio of training samples are tested to find optimal setup for tracking in night scenarios. Fourteen night surveillance videos are collected for validation purpose, which are captured from three viewing angles. The results show that the best accuracy is obtained by using Adam optimizer with learning rate of 0.00075 and sampling ratio of 2:1 for positive and negative training data. This algorithm is suitable to be implemented in higher level surveillance applications such as abnormal behavioral recognition.

Corresponding Author:

Zulaikha Kadim,
Department of Electrical, Electronic and Systems Engineering,
Faculty of Engineering and Built Environment,
Universiti Kebangsaan Malaysia,
Bangi 43650, Malaysia.
Email: zulaikha.kadim@mimos.my

1. INTRODUCTION

The role of video surveillance is to provide a protective mean through monitoring and analyzing any abnormality in the scenes. Nowadays, it is becoming more important with the ever increasing number of crimes. Crime can take place anytime all over the daybut it is more prevalent during night time, especially after the midnight. With the application of automated video surveillance system, it can provide continuous monitoring service for 24/7 with minimal dependency on the security officer.

In the past decades, research in automated video surveillance applications has evolved tremendously and many significant progresses can be observed through availabity of many commercial products in the market. Thanks to the new breakthroughs in software technology, it has became more effective and affordable. The key technologyin the effectiveness of these systems is the ability to detect and track the moving object even in the dark environments, especially during the night.

Both object detection and object tracking are the fundamental components in an automated video surveillance application. Object detection task is to detect the presence of object of interest in the video frame. While, object tracking connects and analyses the object movements for the successive video frames. The information derived from the tracker can be used to further analyze and deduce object activities

in the video. There are many researches has been done on these two topics, however, most of them focuses on the bright environment, with little emphasize on dark environment.

Object tracking for night surveillance is a very challenging task mainly due to low information captured by the normal RGB cameras. The captured images have low brightness, low contrast and nearly no distinguishable color information. It is worst if the object is small in size, caused by the far distance from the camera [1, 2]. Although, most of the recent cameras are equipped with night vision technology to improve image quality in low-light condition, yet, the image quality is still no match as compared to the day time image. In some cases, thermal infrared camera is used for night surveillance [3, 4], but the cost of this type of camera is relatively too costly. Hence, night surveillance is normally performed using day/night CCTV camera with addition of IR filter and IR illuminator for better night vision.

These days, deep-learning study has become a center of attention among researchers in diverse fields that include object detection, classification, facial and speech recognition, rehabilitation, machine translation and etc. [5-11]. Deep-learning is a subfield of machine learning that was inspired by the human brain's structure called neuron, which can be adapted to learn complex relationship [5] and can be extended to multi-layer networks for non-linear problems. There are many types of deep-learning architecture, i.e. Convolutional Neural Network (CNN), Generative Advesarial Network (GAN), Recurrent Neural Networks (RNN) and etc. Among all of them, CNN is the most widely used architecture, especially in computer vision field for object detection, recognition and tracking. CNN architecture was devised by Yann LeCun in 1998 [7], where the feature extractor is also trained instead of hand-crafted. Figure 1 shows an example of basic CNN structure [12] that consists of two convolutional layers, two pooling layers, one fully connected layer and one output layer that defines the final classification according to the number of classes. The convolutional layers in CNN acts as the detection filters to extract specific features or patterns that are presence in the image. An addition of a new layer will increase the complexity, thus allows it to capture more abstract features.

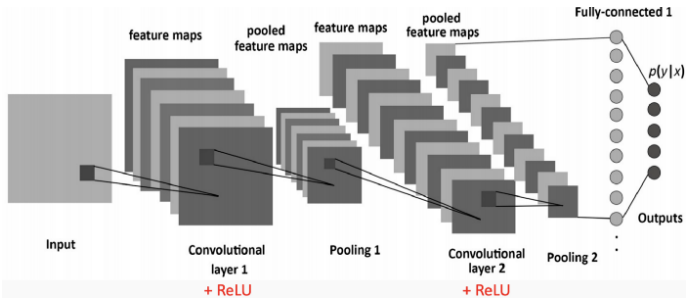

Figure 1. An example of basic CNN network [12]

Due to the CNN capability, this paper proposes a method of online tracking of object of interest for night surveillance application through deep-learning approach. A network with 3 convolutional layers and 3 fully connected layers is used to model the object appearance as proposed in [13]. The fully-connected layers will be updated online to cater the changes in target object appearance as it moves around the scene under different lighting conditions. Various hyperparameters for online learning are experimented, which include the selection of optimization algorithms, online learning rates and training sample ratio to find the optimal tracker setup. The main contributions of this work are:

- Online target tracking framework for night surveillance video that utilizes deep-learning approach to dynamically represent target appearance model.
- Research on the impact of optimal online learning hyperparameters for the best overall tracking accuracy.

The remainder of this paper is organized as follows: Section 2 discusses some related works on visual object tracking. Section 3 describes the proposed method, followed by experimental results and discussion in Section 4. Finally, Section 5 concludes all the research findings.

2. RELATED WORKS

This section will discuss general approach to visual object tracking, followed by specialized tracker for night surveillance applications and the evolution of object tracking algorithm towards deep-learning approach. A good object tracker is defined as an algorithm that is capable of providing accurate object localization with consistent object's tracking label across successive frames. Object tracking studies have been an active research field for the past several decades, and have demonstrated good progress in different

scenarios and applications. Most of the tracking algorithms are based on tracking-by-detection paradigm, whereby object of interest is detected in every frame, which will beused to update the tracking states of the object. This approach is heavily dependent on the detection accuracy. Thus, an improvment in the detection algorithm will lead to better tracking accuracy accordingly. Among others, some good tracking-by-detection algorithms are presented in [14-20]. Some of these tracking approaches are able to function wellunder good lighting conditions, however, their performance deteriorate as the environment becomes darker such as in night surveillance application.

Previously, one of the common approaches to improve tracking performance for night surveillance is by introducing preprocessing module to enhance image quality for the case of underexposed and low contrast environments. Some examples of the preprocessing step are histogram equalization, histogram specification and intensity mapping. Another approach is through analyzing the contrast level so that object detection will be improved before tracking is performed. This is based on the assumption that the human visual system is dependent on the neighbourhood spatial relation to its background. Huang et al. [1] used contrast changes information between successive frames to improve object detection accuracy in the night video application. Local contrast is computed by dividing the local standard deviation of image intensity with local mean intensity. Then, the object is detected by thresholding the contrast change between the successive frames. The computation is quite fast, but the local contrast information to indicate the presence of object of interest might be misleading as the background information itself may contrain high local contrast. On the other hand, the object might have almost similar appearance that produces low local contrast. Later in [21], Huang et al. proposed motion prediction and spatial nearest neighbour data association to further suppress the false detection. In [2], Wang et al. improve Huang's CC model by introducing salient contrast change (SCC), which involve two more steps; online learning and analyzing the detected object trajectories. By applying a threshold on the contrast change output, it is more sensitive to slight changes in the lighting level. Thus Nazib et al. [22] multiplyed Shahnon's entropy estimation with their own contrast estimation to produce illumination invariant representation. In [23], vehicles in night surveillance videos are detected by computing HOG features as input to support vector machine (SVM) to classify the detected object either as a vehicle or not, before Kalman filter is applied to track the vehicles.

Apart from previously mentioned approaches, there are also a few researches that has exploited camera technology to increase the detection and tracking accuracy in night environment. In [24], the researchers has used far-infrared cameras to obtain the foreground information through background subtraction technique. In [25], the researchers has used a near infrared camera to detect pedestrians using adaptive preprocessing technique for the night environment. Another research in [26] has used a fusion of two different types of camera, which are light visible camera and FIR camera mounted on a car to detect pedestrians during the day and night times. Even with the help from improved camera technology, the total cost of the sytems has risen because of more complex sensing hardwares.

Deep learning has been popularized by the introduction of AlexNet in 2012, when it has won ImageNet competition for image classification task [27]. Ever since, deep learning has been widely applied in many applications, overshadowing the other traditional machine learning approaches such as SVM and artificial neural network (ANN). In [28], CNN is used to detect human presence in night surveillance videos as an input to object tracker. Their proposed network consists of five convolutional layers and 3 fully connected layers. The input image is resized to 183x119 first, before histogram equalization is applied for human detection task. The proposed method is closely related to human/background classification in night scenes rather than tracking problem. Another early effort in applying CNN in object tracking is proposed in [29], where an online tracking framework based on multi-domain representations is proposed. Its architecture consists of multiple shared layers that they refer as domain independent layers, where only the classification layer is defined as the domain-specific ones. The shared layers are trained using multiple annotated video sequences offline, while classification layer is trained separately based on each domain. When a new sequence or domain is given, a new classification layer will be constructed to compute the target score based on the new input. Then, the fully-connected layers within the shared layers and the new classification layer will be updated peridiocally so that it is adapted to the new domain. In [30], multiple CNNs in TCNN is maintained in a tree structure to represent multi-modal target appearance. It will update the CNN models in the branches which has most similar appearance with the current target estimation. In [3, 13], a general tracking framework for thermal infrared videos has been proposed. Thermal images exhibit similar properties to night surveillance images where the target object usually consists of low contrast information and negligable textures. In [3], multiple models are maintained to represent the target appearance in different cases such as for the case of temporary occlusion. During network updates, parent nodes will be replaced by the new node so that there is no redundancy in the pool of target object appearance models. In [13], a Siamese approach is utilized in which pair of patches are compared to find the most likely location of the target object in the current frame.

3. METHODOLOGY

3.1. Tracker workflow

Figure 2 illustrates the overall workflow of the proposed tracking methodology. In the first frame, the tracker is initialized using a single ground truth bounding box that encloses the object. Positive and negative candidates are then generated using the given bounding box. Positive samples correspond to the patches or subimages that represent the object of interest, while negative samples correspond to subimages that belongs to the background. Let n_t and m_t be the number of positive and negative training samples, respectively. Positive training data are generated by randomly shifting the initial bounding box within a small distance (the shifted patch should at least consists of 80% overlap area with respect to the original bounding box) and negative samples are generated by randomly shifting the initial bounding box such that they will have minimal overlap area (overlap area with atmost 10% with respect to the initial bounding box). After generating all the training samples, appearance features will be extracted using the CNN networks to produce a feature vector with length of 512. Both sets of positive and negative feature vectors are then used to train the rest of the fully connected layers, which will result in the trained model.

During online tracking, the process starts by generating the possible candidate samples location pivoted on the last known location of the object. Total number of samples extracted is lesser compared to training samples to speed up the tracking process. The features are then extracted and tested using the trained network. The network output are the probabilities that the patch belongs to foreground object and background data. The locations of *n* highest foreground probalities samples will then be used to update estimated location of the tracked object in current input frame. Finally, the network is retrained or updated periodically to capture the changes in object's appearance as it moves around the scenes under different lighting exposure and background.

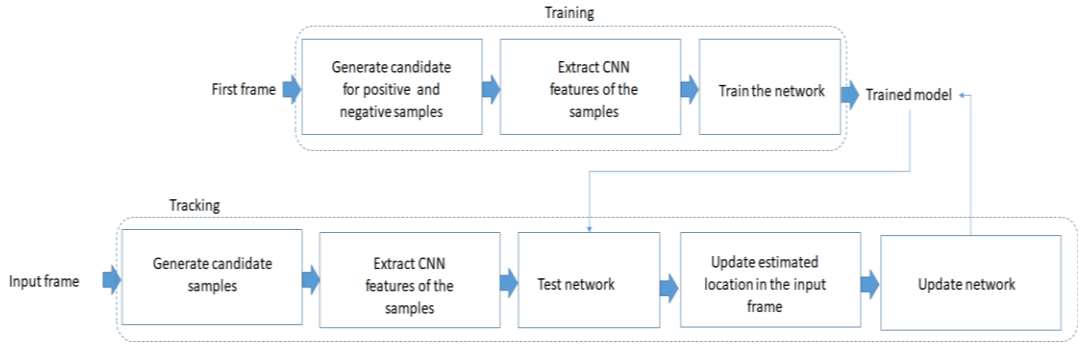

Figure 2. Overall tracking flow

3.2. Network architecture

The network architecture consists of three convolutional layers and three fully connected layers (FC). The first three convolutional layers weights and biases are obtained from VGG-M [31], which has been trained on ImageNet dataset [32]. VGG-M is an eight layers network where the first five layers are the convolutional layers, which function as feature extractor and the last three layers are the dense FC layers. The original input size of VGG-M is 224x224. However, the proposed network uses only the first three convolutional layers with input size of 75x75. Thus, all training and testing samples are resized to match the corresponding input size. The full network architecture used in this work is illustrated in Figure 3.

The first CNN layer consists of 96 filters of 7x7 kernel. The stride step is 2 in *x* and *y* directions, followed by ReLU activation function, local response normalization and 3x3 maximum pooling to produce feature maps of size 17x17x96. The second convolution layer consists of 256 different filters of kernel size 5x5, which is then followed by ReLU activation function, local response normalization and 3x3 maximum pooling to produce 3x3x256 feature maps. Finally, the third layer consists of 512 filters of kernel size 3x3, which will produce feature maps of 1x1x512.

Both positive and negative extracted feature vectors are then used to train the three FC layers. Final output from the last softmax layer are the two probabilities that represent the likelihood of the input image patch belongs to the tracked object and the likelihood that the input image patch belongs to the background. Initially, all FC parameters are randomly initialized. In this work, three different optimization algorithms are experimented to train the FC layers: Gradient Descent [33], Adam [34] and Adagrad [35] with four different learning rates: 0.00125, 0.001, 0.00075 and 0.0005.

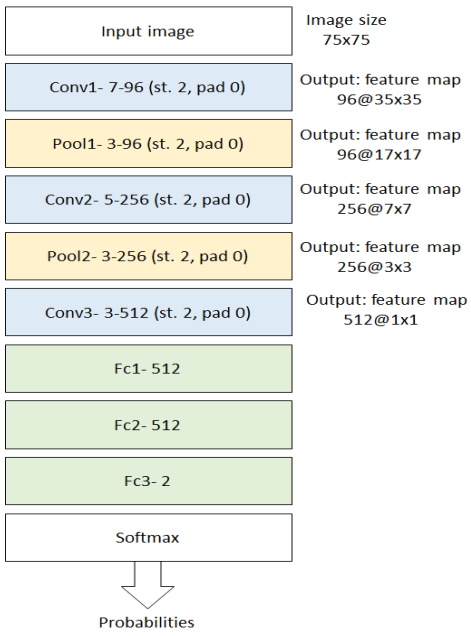

Figure 3. Network architecture of the proposed tracking algorithm

3.3. Network learning parameters

In this work, only the last three FC layers will undergo retraining so that the network is adapted to the changes in the object appearance. In the first frame, the weights of these layers are randomly initialized, while the biases are fixed to 0.05. Learning parameter values for positive samples, negative samples, initial learning rate and number of epoch are set to 500, 1000, 0.0005 and 150 respectively. Cross entropy (1) loss function is used to train the network, where p is the true label, q is the predicted probability and x is the number of output class. Since the network outputs are a set of two probabilities; probability that the sample is foreground and background, thus x value is two where the summation of each sample probabilities is equal to 1. Now, let the true label be $p_{x=0} = y$ and $p_{x=1} = 1 - y$, and the predicted probability be $q_{x=0} = \hat{y}$ and $q_{x=1} = (1 - \hat{y})$. The loss function is then computed by taking the average cross entropy of all N input samples (3).
Cross entropy,

$$H(p,q) = -\sum_x p_x \log q_x \quad (1)$$

$$H(p,q) = -y \log \hat{y} - (1-y)\log(1-\hat{y}) \quad (2)$$

Loss function,

$$J(w) = \frac{1}{N}\sum_{i=1}^{N} H(p_i, q_i) = -\frac{1}{N}\sum_{i=1}^{N}[y_i \log \hat{y}_i + (1-y_i)\log(1-\hat{y}_i)] \quad (3)$$

During online learning, number of training epoch is reduced to 75, while the other two parameters; learning rate and number of positive and negative samples varies according to the best setup. Three different optimizers; stochastic gradient descent, Adagrad and Adam (adaptive moment estimation) are compared to find the optimal values of the model parameters (weights and biases) by minimizing the loss function.

3.3.1. Optimizer #1: Stochastic gradient descent (SGD)

Gradient descent [33] is a popular optimization technique and it has been widely used in network learning [28, 29, 36]. At a time step t, gradient descent algorithm computes the gradient of loss function with respect to the model parameters, where the resultant value is used to update the network. Gradient is a vector of partial derivative of the loss function with respect to every weight and bias for the training samples. Then, each of the weight and bias are updated by subtracting previous value with the multiplication of the learning rate with the calculated gradient (5), (6). The process will be repeated until the loss function is minimized (converge) or the maximum number of epoch is reached. One iteration of a gradient descent on

one parameter is summarized as follows. Gradient of loss function with respect to parameter *i* for time step *t* is calculated as:

$$g_{i,t} = \frac{1}{N}\sum_{j=1}^{N} \frac{\partial J}{\partial w_{i,t,j}} \quad (4)$$

where *N* is the number of training samples.Then the weight and bias of parameter *i* for time step *t* is calculated as in gradient step below:

$$w_{i,t} = w_{i,t-1} - \eta g_{i,t} \quad (5)$$

$$b_{i,t} = b_{i,t-1} - \eta g_{i,t} \quad (6)$$

where η is the learning rate. Note that the same learning rate is applied to all parameters updates.

One gradient descent operation consists of one iteratation over all training samples. This is different from stochastic gradient descent, whereby instead of taking the whole training samples, it randomly selects a few training samples in each iteration to optimize the model parameters. This makes SGD computationally effective and makes it popular for online network training. Nevertheless, since SGD uses only a few training samples, the path to convergence will be noisy.

3.3.2. Optimizer #2: Adam (adaptive moment estimation)

Adam [34] optimizer stands for adaptive moment estimation. It computes different learning rate for different parameters by using the estimates of first and second order moments of gradient. The first and second order moments are the moving average and uncentered moving variance as shown in (4) and (5). It introduces three more hyperparameters compared to gradient step in SGD, which are β_1, β_2 and ε; which correspond to exponential decay rate for first order moment, exponentially decay rate for second order moment and very small constant to prevent the case zero division, respectively. 1st order moment (moving average) of parameter *i* for time step *t*,

$$m_{i,t} = \beta_1 * m_{i,t-1} + (1 - \beta_1) * g_{i,t} \quad (7)$$

2nd order moment (uncentered variance) of parameter *i* for time step *t*,

$$v_{i,t} = \beta_2 * v_{i,t-1} + (1 - \beta_2) * {g_{i,t}}^2 \quad (8)$$

Estimation of these moments will be bias-corrected before they are used to update the model parameters. This step is important to ensure that the first and second order moments are not biased towards zero as the initial values of m_0 and v_0 are set to zero. Bias-corrected first and second order moments are calculated as below. Bias-corrected 1st order moment of parameter *i* for time step *t*,

$$\widehat{m}_{i,t} = \frac{m_{i,t}}{(1-{\beta_1}^t)} \quad (9)$$

Bias-corrected 2nd order moment of parameter *i* for time step *t*,

$$\hat{v}_{i,t} = \frac{v_{i,t}}{(1-{\beta_2}^t)} \quad (10)$$

After estimating the moments, model parameter is updated as in (8). Note that the learning rate is now multiplied by the ratio of first and second order moments of the gradients. η is the learning rate and ε is a very small number to prevent division by zero. Updated weight and biases of parameter *i* for time step *t*,

$$w_{i,t} = w_{i,t-1} - \eta \frac{\widehat{m_{i,t}}}{\sqrt{\widehat{v_{i,t}}}+\varepsilon} \quad (11)$$

$$b_{i,t} = b_{i,t-1} - \eta \frac{\widehat{m_{i,t}}}{\sqrt{\widehat{v_{i,t}}}+\varepsilon} \quad (12)$$

Since its first introduction in 2015, Adam optimizer has been widely used in network learning [37]. It has fast convergence rate and thus practical for training a large model with large training samples.

3.3.3. Optimizer #3: Adagrad

AdaGrad [35] optimizer is a gradient-based learning algorithm, but it computes different learning rates for different parameters. AdaGrad performs smal update on the parameters that are associated with frequently occurring features, while it performs big update on the parameters that are associated with infrequent occurring features. This is achived by AdaGrad through modifying the general learning rate in (5), based on the past gradient of the parameter *i*. The gradient step in AdaGrad becomes:

$$w_{i,t} = w_{i,t-1} - \frac{\eta}{\sqrt{G_{i,t}+\varepsilon}} g_{i,t} \quad (13)$$

where $G_{i,t}$ is the accumulated sum of the squares of the previous gradient with respect to the parameter *i* up to time step *t*.

$$G_{i,t} = \sum_{j=1}^{t} {g_{i,j}}^2 \quad (14)$$

Note that since the gradient values are all positive, the accumulated sum $G_{i,t}$ will keep increasing during the training process which will cause the learning rate in (13) to shrink and eventually become infinitesimally small. At this point, the optimizer is not able to learn any new knowledge. Despite of this weakness, AdaGrad still performs better compared to the SGD as the learning rate is not manually fine-tuned. AdaGrad has been used at Google [38] to train large neural networks to recognize cats in youTube videos. It is also used in [39] to train GloVe word embeddings, as infrequent words require much larger updates compared to the frequent ones.

3.3.4. Learning rate

Choosing a learning rate can be a difficult task. A too small learning rate leads to a slow convergence, while a too large learning rate can hinder convergence and causes loss function to fluctuate or even cause training divergence. In this work, learning rates of 0.00125, 0.001, 0.00075 and 0.0005 are experimented to find an optimal setup.

3.4. Object location estimation

Given an input frame during online tracking, the system will estimate the object location by analyzing the output probabilities from the network. The network outputs two probabilities; (1) probabilities that the input sample belongs to the foreground object and (2) probabilities that the input sample belongs to the background. The final object's location is estimated by computing the weighted average of the top five samples with the highest foreground probabilities whereby the weight is based on their probability values.

4. RESULTS AND DISCUSSION

For validation purpose, 14 night scene videos of size 352x288 has been collected. In each video, the tracked object size is about 30x70 pixels and the total number of accumulated frames of all video is 3646. The chosen videos contain the challenge of various lighting condition, occlusion and move-stop-move problem. Snapshot of the three camera views of the videos are shown in Figure 4. The groundtruth is generated manually by drawing the object bounding box in each frame by an expert in computer vision.

(a) (b) (c)

Figure 4. Three camera views for fourteen testing videos (a) Cm01, (b) Cam02, (c) Cam03

4.1. Implementation details

The tracking code is implemented in Python with tensorflow library. Original location of the tracked object is given in the form of bounding box ([x_0, y_0, $width_0$, $height_0$]). In the first frame, hyperparameters for learning rate, number of epoch, number of positive sample and number of negative sample are initialized to 0.0005, 150, 500 and 1000 respectively. Samples extracted from first frame is the most important step as it is the only known groundtruth by the tracker. Figure 5 shows examples of positive and negative samples extracted in the first frame of three different test sequences. Then, the tracker will be updated online periodically through weak supervision as the consequent frames groundtruth data is not known.

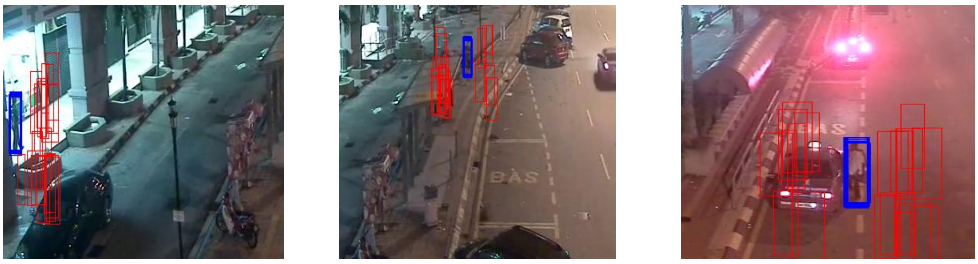

Figure 5. Examples of positive and negative samples that has been extracted from the current frame (first 20 samples), which are represented by blue and red boxes, respectively

4.2. Performance metric

To evaluate the performance of our night tracker algorithm, we use one of the VOT evaluation metrics, which is accuracy (Ac) as defined in (4). Accuracy measures how well the tracked bounding box relative to ground truth box by computing the intersection over union (IoU) area. A higher overlap area represents a better tracking accuracy. The tracker is not re-initialized in the event of track failure (where the IoU is zero).

$$\text{Accuracy, } Ac = \frac{1}{\varphi}\sum_{i=1}^{\varphi}\frac{s_{i,output} \cap s_{i,gt}}{s_{i,output} \cup s_{i,gt}} \quad (4)$$

where φ denotes the number of frames in the test video, while $s_{i,output}$ and $s_{i,gt}$ are the bounding boxes of object in frame i from the tracker output and ground truth, respectively.

Table 1 shows the accuracy comparison between the three optimizer algorithms: SGD, Adam and Adagrad. For a fair comparison, learning rate, number of positive sample and number of negative sample are fixed to 0.001, 50 and 100, respectively. Default values for Adam's hyperparameters β_1, β_2 and ε are set to 0.9, 0.999 and $1e^{-08}$, respectively. In average, Adam optimizer produces the best accuracy as compared to the other two optimizers, followed by AdaGrad. Adagrad performs significantly better in Cam01-video08 compared to the other two optimizers. While, it is noted that SGD performs the worst in most of the test videos. This indicates that the performance of adaptive learning rate method is better compared to a fixed value. As the number of iterations for each training is set to minimum, SGD may not be able to converge and contributes to its bad performance. Some samples of frame with overlaid tracking output for Cam01-video08, Cam02–video02 and Cam03-video02 are shown in Figure 6. Green, blue and magenta bounding boxescorrespond to the output of SGD, Adam and AdaGrad optimizers, respectively. In Figure 6, the first row images correspond to Cam01-video08, in which AdaGrad optimizer gives the highest accuracy. Initially, all three optimizers produce good results as shown in frame #2, then eventually SGD optimizer model has drifted to mix with the background (frame #27) followed by Adam optimizer (frame #71). The second row shows the images for Cam02-video02 sequences, in which Adam gives the best accuracy while the others give almost 0% accuracy (the bounding boxes are stucked at the background area as it contain more textures compared to thetracked object). The third row images correspond to the output for Cam03-video02, in which all three optimizers produce poor accuracy results. This might be caused by similiarity between the foreground appearance and the background.

Table 2 shows the accuracy comparison between four different values of learning rate. In this experiment, Adam optimizer has been chosen as the basis optimizer, while the number of positive and negative samples are set to 50 and 100, respectively. In average, learning rate of 0.00075 gives the best accuracy performance compared to the others, followed by 0.0005 learning rate. The results also indicate that an increase in learning rate value, the average tracker accuracy will be lower.

Table 3 shows the accuracy comparison between four different combinations of total number of positive and negative training samples used during online update. Adam optimizer with learning rate of 0.00075 will be the basic setup for the training samples comparison. In average, a combination of 50 and 100 for positive and negative traiing samples, respectively returns the best accuracy compared to other combinations. Total number of negative samples are twice of the positive samples, such that it caters for larger background area compared to concentrated foreground samples.

Table 1. Accuracy comparison between three optimizer algorithms: SGD, Adam and Adagrad, with online learning parameters; learning rate, number of positive and negative samples are fixed to 0.001, 50 and 100 respectively

No.	Datasets	Number of frames	Accuracy Learning rate= 0.001 # positive samples=50 # negative samples=100		
			Optimizer: Adam	Optimizer: SGD	Optimizer: Adagrad
1	Cam01 – video01	146	85.02	15.71	69.81
2	Cam01 – video02	184	64.82	44.92	45.81
3	Cam01 – video03	71	96.89	0.44	57.35
4	Cam01 – video04	22	91.46	14.85	74.28
5	Cam01 – video05	34	89.51	73.42	25.17
6	Cam01 – video06	150	88.96	74.95	81.64
7	Cam01 – video07	86	55.79	6.88	20.96
8	Cam01 – video08	125	59.26	21.53	94.89
9	Cam02 – video01	257	67.56	0.86	35.71
10	Cam02 – video02	1083	62.27	0	3.8
11	Cam03 – video01	344	95.83	89.66	79.93
12	Cam03 – video02	227	13.58	12.15	2.97
13	Cam03 – video03	317	62.96	55.78	74.03
14	Cam03 – video04	600	36.97	48.32	45.54
Average accuracy			**69.35**	32.82	50.85

Figure 6. Sample of frames with overlaid tracking output for (a) Cam01-video08, (b) Cam03-video02 and (c) Cam02-video02. Boxes color: green (SGD), blue (Adam) and magenta (AdaGrad)

Table 2. Accuracy comparison between four online learning rates: 0.00125, 0.001, 0.00075 and 0.0005, with online learning parameters; optimer algorithm, number of positive and negative samples fixed to Adam, 50 and 100 respectively

No.	Datasets	Number of frames	Accuracy Optimizer: Adam # positive samples=50 # negative samples=100			
			Learning rate= 0.00125	Learning rate= 0.001	Learning rate= 0.00075	Learning rate= 0.0005
1	Cam01 – video01	270	68.23	85.02	78.91	83.28
2	Cam01 – video02	448	62.82	64.82	65.59	70.56
3	Cam01 – video03	71	98.23	96.89	89.20	94.10
4	Cam01 – video04	128	87.66	91.46	94.06	88.76
5	Cam01 – video05	34	89.24	89.51	76.20	89.02
6	Cam01 – video06	224	95.19	88.96	93.76	97.77
7	Cam01 – video07	460	48.22	55.79	59.81	54.06
8	Cam01 – video08	125	27.53	59.26	91.29	95.54
9	Cam02 – video01	1137	45.26	67.56	76.11	60.43
10	Cam02 – video02	1083	66.10	62.27	88.16	76.16
11	Cam03 – video01	344	88.21	95.83	91.63	80.73
12	Cam03 – video02	700	11.76	13.58	7.18	35.29
13	Cam03 – video03	317	64.93	62.96	64.32	76.41
14	Cam03 – video04	689	36.28	36.97	43.04	16.91
Average accuracy			63.55	69.35	72.80	72.79

Table 3. Accuracy comparison between four different combination of positive and negative samples (50,100), (50,50), (100,100) and (150,150), with online learning parameters; optimizer algorithm and learning rate is fixed as Adam and 0.00075 respectively

No.	Datasets	Number of frames	**Accuracy** Optimizer: Adam Learning rate=0.00075			
			n= 100 p= 50	n= 50 p= 50	n= 100 p= 100	n= 150 p= 150
1	Cam01 – video01	270	78.91	12.84	74.49	83.84
2	Cam01 – video02	448	65.59	55.00	54.78	54.58
3	Cam01 – video03	71	89.20	96.19	93.97	96.40
4	Cam01 – video04	128	94.06	91.05	95.27	92.85
5	Cam01 – video05	34	76.20	77.65	92.40	75.07
6	Cam01 – video06	224	93.76	92.22	92.99	97.57
7	Cam01 – video07	460	59.81	64.52	49.81	10.37
8	Cam01 – video08	125	91.29	91.01	90.41	34.12
9	Cam02 – video01	1137	76.11	65.09	6.43	5.27
10	Cam02 – video02	1083	88.16	73.88	55.29	80.56
11	Cam03 – video01	344	91.63	93.92	85.35	93.74
12	Cam03 – video02	700	7.18	11.99	10.82	10.24
13	Cam03 – video03	317	64.32	67.63	69.17	75.61
14	Cam03 – video04	689	43.04	34.35	14.22	43.14
Average accuracy			72.80	66.24	63.24	60.95

5. CONCLUSION

In conclusion, the proposed tracking scheme is able to track object of interest in the night surveillance videos. Adam optimizer shows superior accuracy performance as compared to SGD and AdaGrad in most of the testing videos. The best learning rate is found to be 0.00075 that are achieved by using sample training ratio of 2:1 between negative and positive samples. Hence, this tracker can be implemented in the higher level application of night surveillance system.

ACKNOWLEDGEMENTS

This work was supported by the Nvidia Corporation through the Titan V Grant (KK-2019-005) and Ministry of Education through FRGS/1/2019/ICT02/UKM/02/1.

REFERENCES

[1] K. Huang, L. Wang, and T. Tan, "Detecting and tracking distant objects at night based on human visual system," *Asian Conference on Computer Vision*, pp. 822–831, 2006.

[2] Liangsheng Wang, Kaiqi Huang, Yongzhen Huang and Tieniu Tan, "Object detection and tracking for night surveillance based on salient contrast analysis," *16th IEEE International Conference on Image Processing (ICIP)*, Cairo, pp. 1113-1116, 2009.
[3] M. A. Zulkifley and N. Trigoni, "Multiple-Model Fully Convolutional Neural Networks for Single Object Tracking on Thermal Infrared Video," *IEEE Access*, vol. 6, pp. 42790–42799, 2018.
[4] S. K. Sharma, R. Agrawal, S. Srivastava, and D. K. Singh, "Review of Human Detection Techniques in Night Vision*," International Conference on Wireless Communications, Signal Processing and Networking (WiSPNET)*, pp. 2216–2220, 2017.
[5] C. Szegedy, A. Toshev, and D. Erhan, "Deep Neural Network for Object Detection," *Neural Inf. Process. Syst.*, vol. 7, no. 5, pp. 200–205, 2013.
[6] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 6, pp. 1137–1149, 2017.
[7] Yann LeCun, Patrick Haffner, Léon Bottou, and Yoshua Bengio, "Object Recognition with Gradient-Based Learning," In *Shape, Contour and Grouping in Computer Vision*, pp. 319-.345, 1999.
[8] H. Wang, Y. Cai, X. Chen, and L. Chen, "Night-Time Vehicle Sensing in Far Infrared Image with Deep Learning," *Journal of Sensors*, vol. 4, pp. 1-8, 2016.
[9] J. Gu, *et al*., "Recent advances in convolutional neural networks," *Pattern Recognition.*, vol. 77, pp. 354–377, 2018.
[10] M. A. Zulkifley, S. R. Abdani and N. H. Zulkifley, "Pterygium-Net: a deep learning approach to pterygium detection and localization," *Multimedia Tools and Applications*, vol. 78, no. 24, pp. 34563–34584, 2019.
[11] M. A. Zulkifley, S. R. Abdani and N. H. Zulkifley, "Squat angle assessment through tracking body movements," *IEEE Access*, vol. 7, pp. 48635-32392, 2019.
[12] Matthew Stewart, "Simple Introduction to Convolutional Neural Networks," *Towards Data Science*, 2018, Online. [Available]: https://towardsdatascience.com/simple-introduction-to-convolutional-neural-networks-cdf8d3077bac, [Accessed May. 23, 2019].
[13] M. A. Zulkifley, "Two Streams Multiple-Model Object Tracker for Thermal Infrared Video," *IEEE Access*, vol. 7, pp. 32383-32392, 2019.
[14] Özuysal M., Lepetit V., Fleuret F. and Fua P., "Feature Harvesting for Tracking-by-Detection", *European Conference on computer vision*, pp. 592-605, 2006.
[15] M. Andriluka, S. Roth and B. Schiele, "People-tracking-by-detection and people-detection-by-tracking," *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1-8, 2018.
[16] Tang, S., Andriluka, M., Milan, A., Schindler, K., Roth, S., and Schiele, B., "Learning People Detectors for Tracking in Crowded Scenes," *IEEE International Conference on Computer Vision*, vol. 1049-1056, 2013.
[17] Y. Xiang, A. Alahi and S. Savarese, "Learning to Track: Online Multi-object Tracking by Decision Making," *2015 IEEE International Conference on Computer Vision (ICCV)*, Santiago, pp. 4705-4713, 2015.
[18] Christoph Feichtenhofer, Axel Pinz, Andrew Zisserman, "Detect to Track and Track to Detect," *IEEE International Conference on Computer Vision (ICCV)*, pp. 3057-3065, 2017.
[19] M. A. Zulkifley, D. Rawlinson and B. Moran, "Robust observation detection for single object tracking: Deterministic and probabilistic path-based approaches," *Sensors*, vol. 12, no. 11, pp. 15638-15670, 2012.
[20] M. A. Zulkifley, "Robust single object tracker based on kernelled patch of a fixed RGB camera," *Optik-International Journal of Light and Electron Optics*, vol. 127, no. 3, pp. 1100-1110, 2016.
[21] Kaiqi Huang, Liangsheng Wang, Tieniu Tan, Steve Maybank, "A real-time object detecting and tracking system for outdoor night surveillance," *Pattern Recognition*, vol. 41, no. 1, pp. 432-444, 2008.
[22] Nazib, Abdullah, Oh, Chi-min and Lee Chil-Woo, "Object Detection and Tracking in Night Time Video Surveillance," *10th International Conference on Ubiquitous Robot and Ambient Intelligence*, At Jeju Island, 2013. Doi: 10.1109/URAI.2013.6677410
[23] Qing Tian, Long Zhang, Yun Wei, Wenhua Zhao, Weiwei Fei, "Vehicle Detection and Tracking at Night in Video Surveillance," *International Journal of Online and Biomedical Engineering (iJOE)*, vol. 9, pp. 60-64, 2013.
[24] E. S. Jeon, *et al*., "Human Detection Based on the Generation of a Background Image by Using a Far-Infrared Light Camera," *Sensors (Basel)*, pp. 6763–6788, 2015.
[25] T. Y. Han and B. C. Song, "Night Vision Pedestrian Detection based on Adaptive Preprocessing using Near Infrared Camera," *IEEE International Conference on Consumer Electronics Asia (ICCE-Asia)*, pp. 9–11, 2016.
[26] A. González, *et al*., "Pedestrian Detection at Day/Night Time with Visible and FIR Cameras : A Comparison," *Sensors (Basel)*, pp. 1–11, 2016.
[27] Krizhevsky, Alex et al. "ImageNet Classification with Deep Convolutional Neural Networks," *Communication ACM*, vol. 60, no. 6, pp. 84-90, 2012.
[28] Hyun Kim, Jong, Hong, Hyung and Ryoung Park, Kang, "Convolutional Neural Network-Based Human Detection in Nighttime Images Using Visible Light Camera Sensors," *Sensors*, vol. 17, no. 5, 2017.
[29] H. Nam and B. Han, "Learning multi-domain convolutional neural networks for visual tracking," *Proc. IEEE Conf. Computer Vision Pattern Recognition*, pp. 4293–4302, Jun. 2016.
[30] H. Nam, M. Baek, and B. Han, "Modeling and propagating CNNs in a tree structure for visual tracking," Aug. 2016, [Online], Available: https://arxiv.org/abs/1608.07242, [Accessed May 31, 2019].
[31] Ken Chatfield, Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman, "Return of the Devil in the Details: Delving Deep into Convolutional Networks," *BMVC*, 2014.

[32] O. Russakovsky, *et al*., "ImageNet large scale visual recognition challenge," *International J. Computer Vision*, vol. 115, no. 3, pp. 211–252, Dec. 2015.
[33] Sebastian Ruder, "An overview of gradient descent optimisation algorithms," *arXiv preprint arXiv:1609.04747*, 2017.
[34] Diederik P. Kingma and Jimmy Lei Ba Adam, "A method for stochastic optimization," *arXiv:1412.6980v9*, 2014.
[35] John Duchi, Elad Hazan, and Yoram Singer, "Adaptive Subgradient Methods for Online Learning and Stochastic Optimization," *Journal of Machine Learning Research*, vol. 12, pp. 2121–2159, 2011.
[36] MD Zeiler, "Visualizing and understanding convolutional networks," *Springer Verlag*, vol. 8689, pp. 818-833, 2014.
[37] Y Wu, "Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation," 2016, [Online], Available: http://arxiv.org/abs/1609.08144, CoRR,
[38] Dean, J., Corrado, G. S., Monga, R., Chen, K., Devin, M., Le, Q. V, Ng, A. Y., "Large Scale Distributed Deep Networks," *Neural Information Processing Systems*, 2012.
[39] Pennington, J., Socher, R., and Manning, C. D., "Glove: Global Vectors for Word Representation," *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pp. 1532–1543, 2014.

BIOGRAPHIES OF AUTHORS

Zulaikha Kadim received her B.Eng degree in electronics engineering majoring in telecommunication from Multimedia University, Malaysia in 2000 and currently persuing her Ph.D degree in Universiti Kebangsaan Malaysia. She is also a researcher in national R&D institution. Her current research interests are in video analytics, behavioral analysis and visual object tracking

Mohd Asyraf Zulkifley (M'18) received the B.Eng. degree in mechatronics from International Islamic University Malaysia in 2008 and the Ph.D. degree in electrical and electronic engineering from The University of Melbourne in 2012. He was a sponsored Researcher at the Department of Computer Science, University of Oxford from 2016 to 2018. He is currently an Associate Professor at the Faculty of Engineering and Built Environment, Universiti Kebangsaan Malaysia. His current research interests are visual object tracking and medical image analysis.

Nabilah Hamzah received her B.Scs (Electrical Eng.) degree from Universiti Teknologi Mara (UiTM) in 2019 and currently persuing her ph.D degree at Universiti Teknologi Mara (UiTM). Currently work as reseach assistant (RA) at UiTM under Trans-disciplinary Research Grant Scheme (TRGS). Her current research interests are in object tracking, image translation and facial recognition.