International Journal of Electrical and Computer Engineering (IJECE)
Vol. 11, No. 2, April 2021, pp. 1510~1520
ISSN: 2088-8708, DOI: 10.11591/ijece.v11i2.pp1510-1520

Real-time human detection for electricity conservation using pruned-SSD and arduino

Ushasukhanya S.[1], Jothilakshmi S.[2]
[1]Department of Computer Science and Engineering, Annamalai University, Tamilnadu, India
[2]Department of Information Technology, Annamalai University, Tamilnadu, India

Article Info

Article history:

Received Jul 1, 2020
Revised Aug 2, 2020
Accepted Sep 24, 2020

Keywords:

Arduino
Electricity management
Human detection
Pruning
Single shot detection

ABSTRACT

Electricity conservation techniques have gained more importance in recent years. Many smart techniques are invented to save electricity with the help of assisted devices like sensors. Though it saves electricity, it adds an additional sensor cost to the system. This work aims to develop a system that manages the electric power supply, only when it is actually needed i.e., the system enables the power supply when a human is present in the location and disables it otherwise. The system avoids any additional costs by using the closed circuit television, which is installed in most of the places for security reasons. Human detection is done by a modified-single shot detection with a specific hyperparameter tuning method. Further the model is pruned to reduce the computational cost of the framework which in turn reduces the processing speed of the network drastically. The model yields the output to the Arduino micro-controller to enable the power supply in and around the location only when a human is detected and disables it when the human exits. The model is evaluated on CHOKEPOINT dataset and real-time video surveillance footage. Experimental results have shown an average accuracy of 85.82% with 2.1 seconds of processing time per frame.

This is an open access article under the CC BY-SA license.

Corresponding Author:

Ushasukhanya S.
Department of Computer Science and Engineering
Annamalai University
Annamalai nagar, Tamilnadu, India
Email: ushasukhanya@gmail.com

1. INTRODUCTION

Unmanned electric power management system (UEPMS) is one of the challenging tasks that are required to save electrical resources in most of the countries. Several researches are going on for UEPMS with the help of sensors to detect the presence/absence of humans and to manage the power supply accordingly. Usage of sensors incurs an additional cost and hence this work aims to develop a system that manages the electrical resource using the existing closed circuit television (CCTV) surveillance camera. The surveillance video captured from CCTV cameras is used to detect the human's presence/absence to enable/disable the power supply thereby avoiding additional cost. Human detection in surveillance cameras footage has been an interesting [1] and challenging [2] topic in the recent years. The traditional hand-crafted methods like local binary pattern (LBP), histogram of oriented gradients (HOG) etc., are time consuming and proved to be comparatively inefficient to the recent convolutional neural network (CNN) based algorithms [3]. Various object detection algorithms in deep learning (DL) have shown promising results in classifying and detecting the location of the objects [4]. The first category of DNN is a two stage approach like RCNN, faster-regional CNN (faster R-CNN), region based fully convolutional neural network (R-FCN) [5, 6] etc.,

which proposes regions by a separate network called region proposal network and the classifier processes these regions for classification [7]. The second one is a one stage approach like you only look once (YOLO) and single shot multibox detector (SSD) in which both the class probabilities and the bounding boxes are produced by the CNN itself [8].

In this work a system using modified-SSD which is based on visual geometry group (VGG16) is used for human detection in surveillance cameras [9]. The CNN based network takes the input from CHOKEPOINT dataset, which has frames of a surveillance video. The model is initially trained with the set of hyper-parameters obtained from orthogonal array tuning method (OATM). The optimal factors are derived from the factor table of the OATM method. Once the model is converged to the minimal loss function, the network is pruned to remove the less important parameters of the network. Pruning is a method done to reduce the complexity of the network thereby maintaining the accuracy of the model. The identification of these less important weights are done by the H-ranking algorithm proposed by Lin *et al*. After pruning, the network is retrained with the set of hyperparameters obtained from the OATM method. The process is iterated until the convergence or the error loss function is similar to the one obtained by the model before pruning is done. The output of the classifier is fed to an Arduino microcontroller to manage the power supply. Arduino enables the power supply only in the location (2.1 metres) in and around where a human is detected and disables the power when undetected. The system is also validated on a real-time dataset of a surveillance video in an indoor environment of a living room where the footage is converted to frames at the rate of 5 per second. The intensely compressed model shows promising results in prediction accuracy with reduced training time.

2. RELATED WORKS

Object detection has gained a lot of attraction by the researchers in various applications. From small crack detection to human detection, lesion detection [10] to detection from satellite images [11] etc. Overcoming the shortfalls of traditional hand-crafted methods, DL has achieved enormous growth in these detections. Object detection in surveillance videos is one of the most challenging tasks due to the lack of clarity in frames. It all started with the regional convolutional neural network (R-CNN) which uses selective search to detect the location of the object with a bounding-box [12]. Approximately 2000 candidate regions are extracted in this process which is extensively time consuming. This is overcome by spatial pyramid pooling net (SPP-Net) on the feature maps [13] and further by faster-RCNN [14] which uses a separate network called region proposal network (RPN) to generate candidate regions. A system for facial expression recognition was developed for an emotional audio and video data [15] using faster R-CNN.

Several modified versions of faster R-CNN were developed to improve the prediction accuracy at minimal cost [16]. Though all these techniques have improved the accuracy, it still serves as a time consuming task as it requires two stages for prediction. This was overcome by the YOLO model where the entire process is carried out by a single neural network that makes optimization quite easier [17]. An advanced version of one stage approach is a single SSD system, which is achieving promising results in real-time surveillance videos [18]. One such application was developed to detect small objects using contextual information in SSD at increased speed [19]. When compared to all the above listed detection algorithms, SSD has achieved relatively promising results at increased speed by applying prediction filters on every feature map produced. Though SSD achieves good results, one of the major challenging tasks of DL is the high training time required by the model to learn [20, 21]. Hence the proposed work uses SSD architecture with a specific hyper-parameter tuning method to reduce the training time of the network [22]. As the deep network designed for any real-time application involves high computational cost, pruning is done in many applications to keep the model simple. Data-dependent and data-independent methods are the two techniques adopted to evaluate the importance of the weights among which optimal brain damage [23], optimal brain surgeon [24], absolute value method [25], LASSO regression method [26] are certain renowned techniques. An interesting method by Mingbao Lin *et al*. has been developed which uses a H-Rank filter pruning method to prune the model by calculating the rank of each and every parameter. The technique then re-arranges the ranked parameters in decreasing order and finally eliminates the least important parameters.

One of the important tools in literature to control the power supply is the Arduino micro-controller which has its usage in a wide range of applications. A system to quantify the energy of the given load and plan appropriate energy conservation policies was also designed [27]. Another application was developed for smart home energy management systems (SHEMS) [28] to enable/disable the power supply when a human is detected/undetected respectively. All these systems use sensors which incurs additional cost. Hence our system uses a modified-SSD to detect humans in existing CCTV surveillance footage with an Arduino micro-controller to manage the power supply accordingly.

3. PROPOSED FRAMEWORK

A modified SSD is developed in this segment with an orthogonal array based tuning method to reduce the training time with the set of obtained factor values. The model is further pruned to reduce the complexity of the DNN model which in turn reduces the computation cost intensely. The working model of the proposed architecture is given in Figure 1. The output of the model is embedded with an Arduino micro-controller to manage the power supply of the environment.

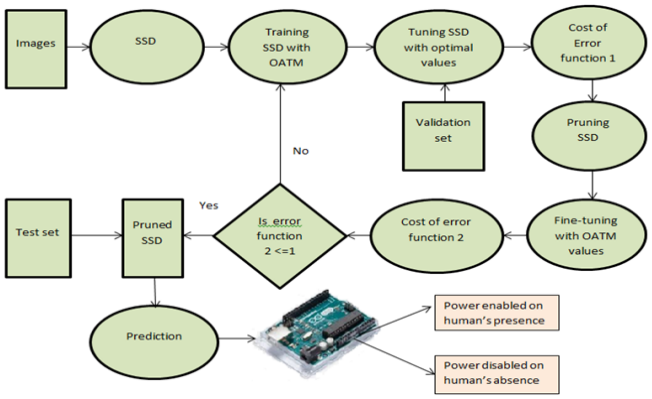


Figure 1. Working model of pruned OATM based SSD

3.1. Datasets and data augmentation

The proposed system is evaluated using standard CHOKEPOINT dataset, consisting 64.204 frames of a surveillance video in an indoor environment. The system is also evaluated on a real-time surveillance video of a living room which is converted at the rate of 5 frames per second (fps). It consists of 9000 frames totally out of which 7420 frames (including augmented frames) are considered for training and the remaining 3180 are considered for testing. Validation set consists of 371 images to tune the hyper-parameters. Augmentation is also done to make the network more robust. Shrinking and cropping of the original training images are done. Gamma correction is applied to create variations in intensities and brightness of the frames using two sets of values (0.5 and 3.0) and (0.5 and 2.0) for all the three channels individually [29]. Hide and seek [30] is finally used to divide the entire image into patches with a division number of 4 to make the network learn the fine entities of the patches. The sub patches are hidden with a probability of 0.25. In both the cases the positive bounding boxes are the ones with the match score greater than 0.5 with that of its ground truth and the remaining are considered negative. All the experiments are conducted in GPU-NVIDIA TITAN X 12 GB, RAM-32 GB DDR4.

3.2. Model construction

The proposed framework uses a modified SSD technique which is a feed-forward convolutional neural network based on VGG16. The structure has a truncated part of VGG 16 with an additional convolutional structure attached to its end. It eliminates the need for RPN in faster R-CNN thereby increasing the detection speed drastically [31]. Instead of RPN, it uses multi-scale features and default boxes to make its prediction accuracy be in par with the average accuracy obtained by faster R-CNN [32]. The entire architecture of the proposed system is given in Figure 1. The input is a colour image I which is fed into the initial base network consisting of five convolutional layers ($L = 5$) as in VGG-16 which detects the edges, blobs, texture and object parts. The sixth and seventh dense layers of VGG-16 are replaced with convolutional layers while removing the eighth and the other dropout layers. ReLU is the activation function used at every convolutional layer except the output layer.

$$Y = W_1 I_1 + W_2 I_2 + \cdots + W_n I_n \quad (1)$$

where Y is the feature map, W is the weight associated with input I. Each of these convolutional layers with size $x \times y$ and z channels produces a feature map on which the prediction layer (classification layer) of size $3 \times 3 \times z$ is applied. The values produced are with respect to each location of the feature map thereby yielding $(c + 4)xyz$ outputs where c represents the class scores with four offsets for z filters. The feature map is then subsequently processed by other convolutional layers from till the final layer where for each

feature map $(c + 4)xyz$ values are produced [33]. Appropriate default boxes are to be selected for better prediction and hence the ground truth box is matched with the default box using Jaccard overlap method with a threshold value set to 0.5. This yields a large number of negative samples out of which the samples with the highest confidence loss is alone selected, thereby making a ratio of 1:3 for positive to negative samples. Finally, sigmoid function is used at the output layer to make a binary classification of the frames and the non-maximum suppressions (NMS) makes object detections. The weights of the network are initialized using the "HE" initialization technique based on the formula

$$Var(W_i) = \frac{2}{fan\_in} \quad (2)$$

where fan_in represents the number of input units. It provides a controlled initialization, thereby increasing the convergence rate. Learning rate, momentum, dropout and weight decay are the four hyper-parameters considered for tuning and are done by orthogonal array tuning method (OATM) based on Taguchi's factor table. This method yields a certain set of factor values which makes the model's convergence easier and faster. The hyper-parameters are represented as factors, while the corresponding values are defined as levels in the table. The orthogonal array table is generated using Weibull++ software with 4 factors that has 9 rows. The model is trained with the frames of the training datasets which is divided into 10 batches. The model is iterated for 90 epochs with the hyperparameters obtained by the OATM method and the accuracy for each set of hyperparameters is estimated. The row with the highest accuracy is considered and the corresponding values are the optimal values with which the hyperparameters are tuned. Each and every batch is iterated for 4 epochs and the average of these four experiments is considered as the accuracy of a selected single set of hyperparameters.

The accuracies and the corresponding values for these selected hyperparameters are presented in Table 1. Level 5 in Table 1 obtains the highest accuracy rate and hence the corresponding values are the optimal values for this phase of SSD. The specifications of the constructed model like the kernel size, input and output dimensions of the feature maps of every layer, the total parameters used in every layer etc are given in Table 2.

Table 1. Orthogonal array table

Exp.No	Factor 1	Factor 2	Factor 3	Factor 4	Accuracy (%)
1	0.01	0.999	0.3	0.001	85.21
2	0.001	0.99	0.25	0.003	84.32
3	0.01	0.9	0.20	0.005	86.12
4	0.001	0.999	0.25	0.005	85.72
5	**0.001**	**0.99**	**0.20**	**0.001**	**87.24**
6	0.001	0.9	0.3	0.003	86.88
7	0.0001	0.999	0.20	0.003	85.32
8	0.0001	0.99	0.3	0.005	86.91
9	0.0001	0.9	0.25	0.001	85.47

Table 2. Specifications of the constructed model

SSD LAYER	X	Y	Kernel size	C_{in}	C_{out}	Parameters
Conv1_1	2400	4000	3	3	64	1.728
Conv1_2	2400	4000	3	64	64	36.864
Conv2_1	1200	2000	3	64	128	73.728
Conv2_2	1200	2000	3	128	128	147.456
Conv3_1	600	1000	3	128	256	294.912
Conv3_2	600	1000	3	256	256	589.824
Conv3_3	600	1000	3	256	256	589.824
Conv4_1	300	500	3	256	512	1.179.648
Conv4_2	300	500	3	512	512	2.359.296
Conv4_3	300	500	3	512	512	2.359.296
Conv5_3	150	250	3	512	1.024	4.718.592
Conv5_3	150	250	3	1.024	1.024	9.437.184
Conv5_3	150	250	3	1.024	1.024	9.437.184
SSD_Conv6	75	125	1	1.024	2.048	2.097.152
SSD_Conv7	75	125	1	2.048	4.096	8.388.608
SSD_Conv8	38	63	2	4.096	8.192	134.217.728
SSD_Conv9	38	63	2	8.192	16.384	536.870.912
SSD_Conv10	19	32	2	16.384	32.768	2.147.483.648
SSD_Conv11	19	32	2	32.768	32.768	4.294.967.296
Detections	1	1	1	32.768	21	688.128
NMS	1	1	1	32.768	84	2.752.512

The parameters are calculated using the formula;

$$parameter = kernel\_size^2 \times channel_{in} \times channel_{out} \quad (3)$$

where the kernel_size, channel^{in}, and channel^{out} are defined as the kernel size of the weight filter, number of input channels, and number of output channels in each convolution layer. *X* and *Y* represent the horizontal and vertical dimensions of the feature maps in each convolution layer. The amount of computation is directly proportional to the number of parameters of the network in each and every layer [34]. Therefore, reducing the number of parameters in both convolution and FC layers helps achieve the goal. There are various strategies to decrease the model size, such as pruning [35] and quantization [36] etc., out of which pruning is selected for our work as it has yielded promising results in various applications [37].

3.3. Pruning

The SSD network exists with a set of *N* convolutional layers where i^{th} convolutional layer is represented by N_i. Pruning is defined as the removal of filters or parameters which are considered to be less important. Hence, the entire set of filters is divided into I_{Ni} and U_{Ni}which represents the important filter and less important filter set respectively. The total number of important and less important filters is depicted by *K* and *L* respectively where K+L=Q representing the total filters. Inspired by [38], we perform pruning in the same way where filter pruning in general is formulated as;

$$\delta_{ij} \ \sum_{i=1}^{N} \sum_{j=1}^{Q} \delta_{ij} X(w_j^i) \quad (4)$$

such that

$$\sum_{j=1}^{Q} \delta_{ij} = L \quad (5)$$

In (4) where $\boldsymbol{\delta}_{ij}$ represents 1 if the weights are labelled as *K* and 0 otherwise. The importance of the filter is measured by $X(w_j^i)$. Hence the objective is to minimize the equation to remove *L*. As each and every feature map of the N_i th layer plays different roles, equation 4 has been reformulated as;

$$\delta ij \ \sum_{i=1}^{N} \sum_{j=1}^{Q} \delta_{ij} E_{I-D(I)} \left[\dot{\mathrm{X}} \left(Y_i^j(I,:,:) \right) \right] \quad (6)$$

where *Y* represents the feature maps, *I* is the input image sampled from distribution D(I) such that;

$$\sum_{j=1}^{Q} \delta_{ij} = L \quad (7)$$

also the evaluation of the filter is defined as;

$$\left[\dot{\mathrm{X}} \left(Y_i^j(I,:,:) \right) \right] = Rank \ (Y_i^j(I,:,:)) \quad (8)$$

single value decomposition is applied where;

$$Y_j^i(I,:,:) = \sum_{i=1}^{r} \sigma_i m_i n_i T \quad (9)$$

such that;

$$\sum_{i=1}^{\acute{r}} \sigma_i m_i n_i T + \sum_{i=\acute{r}+1}^{r} \sigma_i m_i n_i T \quad (10)$$

when $r' < r$, $_i$ m_i and n_i are top, left and right singular values respectively. Thus, the rank of each and every parameter is calculated and the technique re-arranges the ranked parameters in decreasing order which finally eliminates the least important parameters (bottom most). The model is trained again with the hyperparameters obtained by OATM method for all the factor values. The optimal set of hyperparameters obtained in this phase is different as the network is actually pruned. The H-Rank algorithm is again implemented and the process is iterated until the error function is similar to the one obtained by the model without pruning. The entire working model of the pruning methodology is given in Figure 2.

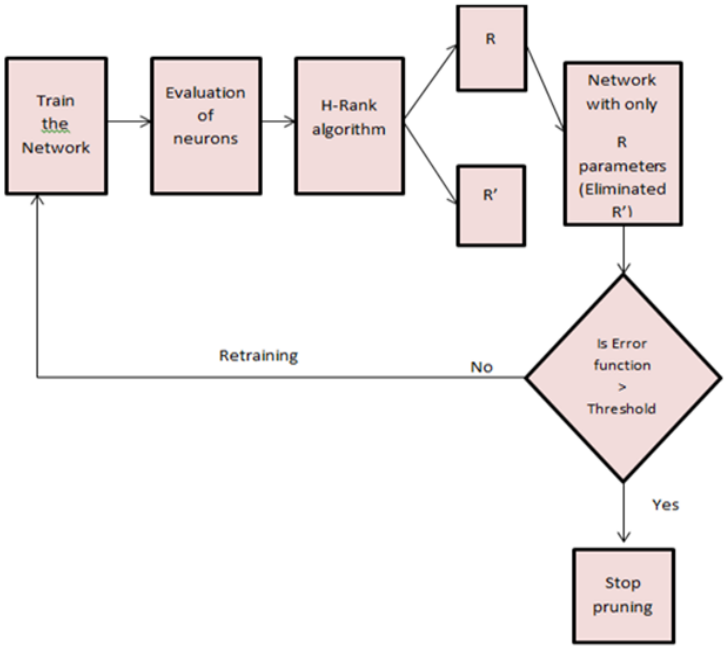


Figure 2. Pruning SSD model

3.4. Training and testing process

The frames are resized to 300×300 to feed it into the modified-SSD network. The dataset is divided into a training set and testing set in the ratio of 70:30. The training set consists of 47.299 frames in which the validation set is a sub class containing 5% of the training frames (2364 frames) to tune the hyper-parameters of the network. This is followed by processing of the testing set (20.271 frames). Initializing the weights of the network is done using "HE" initialization technique. The frames of the training set are fed into the network and the experiments are run for all the levels of Table 1 (OATM). The model is then pruned by removing the less important parameters of the network using the H-rank algorithm. The network is retrained with the set of hyperparameters obtained by the OATM method. The process is iterated until the error loss function is similar to the one obtained by the model before pruning. A validation set and test set is passed after training the network and the experiment is iterated 90 epochs, and the mean average precision (mAP) is taken for each level. The highest mAP obtained for the test set is 87.21% before pruning and 85.82% after pruning. Compression rate of 42% is achieved by pruning which reduces the testing time to 2.1 seconds from 4.5 seconds of an un-pruned model.

3.5. Evaluation

The predicted values and the ground truth values are represented as $p = \{p_{cx}, p_{cy}, p_w, p_h\}$ and $g = \{g_{cx}, g_{cy}, g_w, g_h\}$ respectively. The loss function is the total loss calculated by summing the classification and the regression loss which is denoted by:

$$L = (x, c, l, g) = \frac{1}{N}(N_{cls}(x, c) + \lambda x L_{reg}(p, g)) \quad (11)$$

where c represents the centre of the bounding box, p is the predicted value, g is the ground truth of the bounding boxes, N is the number of matched default boxes and the regression loss is calculated using the formula given by [39, 40]:

$$L_{reg}(p, g) = \sum_{q\epsilon\{cx,cy,w,h\}} smooth_{L1}(p_m - g_m) \quad (12)$$

where $smoothL1(z) = \{ 0.5z^2, if\ |z|<1$ and z - 0.5 otherwise. The classification loss based on cross entropy is given by

$$L_{cls}(x, c) = -xlog(c) - (1 - x)log\ (1 - c) \quad (13)$$

mAP and mean F1 index (mF1) are the metrics used to evaluate the detection accuracy of the entire model and sub layers respectively. The average precision (AP) is given by the equation

$$AP = \int_0^1 p(r)dr \tag{14}$$

where p and r denotes precision and recall respectively. mF1 is given by the equation;

$$mF1 = \frac{1}{n}\sum_{i=1}^{n} F1_I \tag{15}$$

where F1 is given by the equation;

$$F1 = \frac{2\times Precision\times Recall}{Precision+Recall} \tag{16}$$

and precision and recall are given by the equations;

$$Precision = \frac{TP}{TP+FP} \; and \; Recall = \frac{TP}{TP+FN} \tag{17}$$

TP, FP and FN denotes true positive, false positive and false negative respectively.

3.6. Power supply management

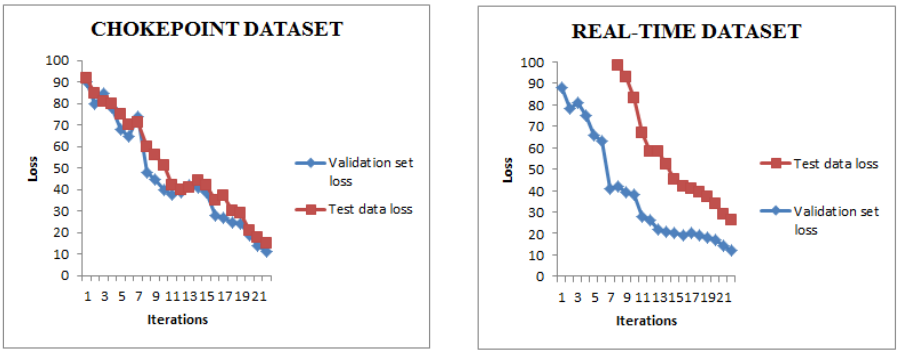
The output of the pruned SSD is fed to the Arduino microcontroller which is connected to the electrical supply of a room or any indoor environment. The controller enables the power supply in and around the location (2 mts) where a human is detected and disables it, if the human is undetected. Therefore, the electric resource is utilized only when and where it is actually needed and saves the resource efficiently. Using the proposed framework, the average monthly consumption of electricity for a residential environment is reduced to 72 from 90 units (kWh), which is nearly one quarter of the total electricity consumption.

4. RESULTS AND DISCUSSIONS

The prediction accuracies of the modified-SSD model which is based on OATM technique for both the datasets are given in Table 3. The validation and test data accuracies are measured and it is found that the prediction accuracy achieved by modified-SSD is very close to that of the original SSD but the training time is extensively reduced in modified-SSD, thereby increasing the processing speed drastically. The average loss of validation and test data sets by modified-SSD (without pruning) is 13.475% for both the datasets. The graphical representation of it is given in Figure 3. SSD with other pruning methods like Sparse structure selection and generative adversarial learning (GAL) have also been implemented for our prediction but the results of SSD with H-rank pruning tops other techniques. The results of various pruning techniques in terms of accuracy, compression rate and floating-point operations (FLOPs) are represented in Table 4. Our model out-performs the training speed of the model created by multi-layer pruning framework [41], as the latter implements three stages of pruning, only after completely training the network from the scratch. Hence the training time increases many folds in this method, whereas the former (our model) trains the network initially based on OATM method which reduces the training time drastically. Secondly, our model uses H-rank algorithm to focus on low-rank feature maps rather than eliminating the zero weights based on sparsity statistics with a negligible loss of 0.83% at 42.01% compression.

Table 3. Results of baseline SSD and modified-SSD

Models	Datasets	Classes	Precision (%)	Recall (%)	F1 (%)	Training time (sec)
SSD (baseline)	CHOKEPOINT	Human	89.1	86.7	87.8	58346
		Non-Human	88.9	85.3	87.0	
	Real-time	Human	87.1	85.6	86.3	21236
		Non-Human	86.9	84.3	85.5	
Modified-SSD	CHOKEPOINT	Human	88.6	86.4	87.4	5621
		Non-Human	88.8	85.1	87.3	
	Real-time	Human	87.1	85.1	86.1	1721
		Non-Human	86.5	84.3	85.3	



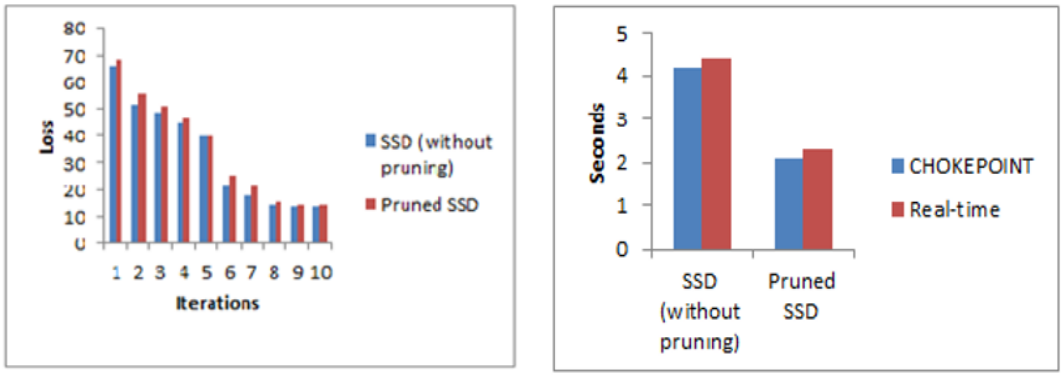
(a) (b)

Figure 3. (a) Loss of CHOKEPOINT dataset, b) loss of real-time dataset

As our framework uses, low resolution indoor CCTV images, due to dullness, the accuracy is affected on further pruning and hence we stop pruning at this level. Though the results of Pruned-SSD seem slightly lesser than the traditional SSD, this difference can be ignored when compared to the processing speed of the pruned-SSD which is drastically improved. Pruning reduces the model's complexity, which eventually increases the prediction speed of the test data from 4.4 seconds on an average by un-pruned SSD to 2.1 seconds by pruned-SSD. The prediction speed and the loss function of both SSD with and without pruning are given in Figure 4.

Table 4. Comparison of H-rank pruning with other techniques

Techniques	Map	Compression	Computation	FLOPs
SSD (Baseline)	86.65	Nil	100%	343.60 M
SSD with SSS	83.21%	38.71%	22%	210.16 M
SSD with GAL	84.91%	40.20%	21.2%	208.02 M
SSD with H-Rank	85.82%	42.01%	18.9%	144.31 M



(a) (b)

Figure 4. (a) Error rate of SSD without and with pruning, (b) detection speed of test data

The processing speed of the test data set is compared with various preliminary models and the results are presented in Figure 5. Among all the other techniques, the proposed modified-pruned SSD excels by yielding the lowest processing speed of 2.2 seconds on an average. The model detects the human along with the location represented by a bounding box which is given in Figure 6. This is then processed by the controller and the results of the controller are given in Figure 7. As the rotation of the fan (when power enabled) will not be clearly visible in an image, we have taken two bulbs, one representing the light and the other representing the fan in Figure 7.

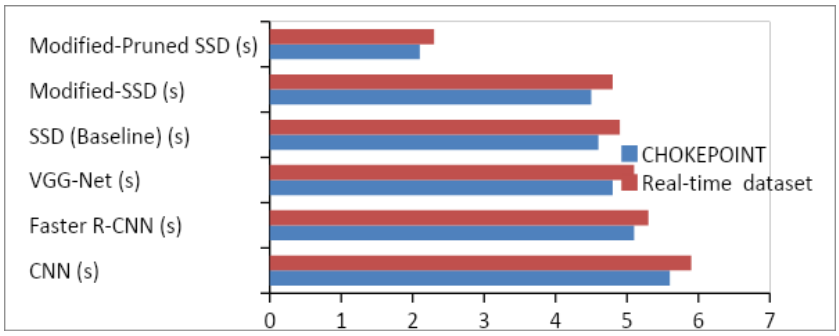


Figure 5. Processing speeds of various techniques

(a) (b)

Figure 6. (a) Location of human detected by pruned SSD, (b) human undetected by pruned SSD

(a) (b)

Figure 7. (a) Enabling the resource on human's presence, (b) disabling the resource when human exits

5. CONCLUSION

UEPMS is one of the most essential services in day to day life due to the depletion of resources. Among various existing methods, this system uses the existing CCTV footage to detect humans and enable the power supply only in the location (2 mts) where humans are detected. The system uses modified-SSD for human detection with a specific hyperparameter tuning to decrease the training time of the model. The model is further pruned by the H-rank algorithm to decrease the computational cost thereby increasing the processing speed of the network. An Arduino micro-controller is used to manage the power supply of the system. The proposed architecture achieves an average prediction accuracy of 85.82% with a much reduced compression rate of 42% of the original network. It is evident that the proposed system saves nearly one third of the total electricity consumption. As the system is developed only for indoor environments, considering larger places like malls or outdoor environments, occlusion handling in these places could be taken as one of the future directions.

REFERENCES

[1] A. Raghunandan, Mohana, P. Raghav and H. V. R. Aradhya, "Object Detection Algorithms for Video Surveillance Applications," *2018 International Conference on Communication and Signal Processing (ICCSP)*, Chennai, 2018, pp. 0563-0568.

[2] Xiongwei, Wu, Sahoo, Doyen and Hoi, Steven, "Recent Advances in Deep Learning for Object Detection," *Neurocomputing*, vol. 396, pp. 39-64, 2019.

[3] Ding, S., et al., "SurvSurf: human retrieval on large surveillance video data," *Multimedia Tools and Applications*, vol. 76, pp. 6521-6549, 2017.

[4] J. Dai, Y. Li, K. He, and J. Sun, "R-FCN: object detection via region-based fully convolutional networks," *NIPS'16: Proceedings of the 30th International Conference on Neural Information Processing Systems*, 2016, pp. 379-387.

[5] Mahmoud, Hanan and Mengash, Hanan, "A novel technique for automated concealed face detection in surveillance videos," *Personal and Ubiquitous Computing*, pp. 1-12, 2020.

[6] Z. Cai, Q. Fan, R. S. Feris, and N. Vasconcelos, "A unified multi-scale deep convolutional neural network for fast object detection," *Conference: European Conference on Computer Vision*, 2016, pp. 354-370.

[7] Alberto Castillo, Siham Tabik, Francisco Pérez, Roberto Olmos, Francisco Herrera, "Brightness guided preprocessing for automatic cold steel weapon detection in surveillance videos with deep learning," *Neurocomputing*, vol. 330, pp. 151-161, 2019.

[8] J. Redmon and A. Farhadi, "YOLO9000: better, faster, stronger," *CVPR*, 2016.

[9] Yundong Li, et al., "Multi-block SSD based on small object detection for UAV railway scene surveillance," *Chinese Journal of Aeronautics*, vol. 33, no. 6, pp. 1747-1755, 2020.

[10] Cruz-Roa A. A., Arevalo Ovalle J. E., Madabhushi A., González Osorio F. A., "A Deep Learning Architecture for Image Representation," *Medical Image computing and computer-assisted intervention*, vol. 16, pp. 403-410, 2013.

[11] Freitas S., Almeida C., Silva H., et al., "Supervised classification for hyperspectral imaging in UAV maritime target detection," *2018 IEEE international conference on autonomous robot systems and competitions*, Torres Vedras, 2018, pp. 84-90.

[12] Rafique, M. A., Pedrycz, W. and Jeon, M., "Vehicle license plate detection using region-based convolutional neural networks," *Soft Computing*, vol. 22, pp. 6429-6440, 2018.

[13] He K. M., Zhang X. Y., Ren S. Q., et al., "Spatial pyramid pooling in deep convolutional networks for visual recognition," in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, no. 9, pp. 1904-1916, 2015.

[14] Ren, Yun, Zhu, Changren and Xiao, Shunping, "Object Detection Based on Fast/Faster RCNN Employing Fully Convolutional Architectures," *Mathematical Problems in Engineering*, vol. 2018, pp. 1-7, 2018.

[15] Li, Jiaxing, et al., "Facial Expression Recognition with Faster R-CNN," *Procedia Computer Science*, vol. 107, pp. 135-140, 2017.

[16] B. Liu, W. Zhao and Q. Sun, "Study of object detection based on Faster R-CNN," *2017 Chinese Automation Congress (CAC)*, Jinan, 2017, pp. 6233-6236.

[17] J. Redmon, S. Divvala, R. Girshick and A. Farhadi, "You Only Look Once: Unified, Real-Time Object Detection," *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Las Vegas, NV, 2016, pp. 779-788.

[18] C. Fu, W. Liu, A. Ranga, A. Tyagi, and A. C. Berg, "DSSD: Deconvolutional single shot detector," *CoRR*, 2017.

[19] Guimei Cao, Xuemei Xie, Wenzhe Yang, Quan Liao, Guangming Shi, and Jinjian Wu, "Feature-fused SSD: fast detection for small objects," *Ninth International Conference on Graphic and Image Processing (ICGIP 2017)*, 2018, pp. 1-8.

[20] A. G. Howard, "Some improvements on deep convolutional neural network based image classification," *CoRR*, 2013.

[21] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *ICLR*, 2015.

[22] Zhang, Xiang, Chen, Xiaocong, Yao, Lina, Ge, Chang, Dong, Manqing, "Deep Neural Network Hyperparameter Optimization with Orthogonal Array Tuning," *Neural Information Processing*, pp. 287-295, 2019.

[23] Y. LeCun, J. S. Denker, and S. A. Solla, "Optimal brain damage," *Advances in Neural Information Processing Systems 2 (NIPS 1989)*, pp. 598-605, 1990.

[24] B. Hassibi and D. G. Stork, "Second order derivatives for network pruning: Optimal brain surgeon," *Advances in Neural Information Processing Systems 5 (NIPS 1992)*, pp. 164-171, 1993.

[25] H. Li, A. Kadav, I. Durdanovic, H. Samet, and H. P. Graf, "Pruning filters for efficient convNets," *ICLR 2017 conference submission*, 2016.

[26] R. Tibshirani, "Regression shrinkage and selection via the lasso," *Journal of the Royal Statistical Society. Series B (Methodological)*, vol. 58, no. 1, pp. 267-288, 1996.

[27] X. Ren, H. Guo, G. He, X. Xu, C. Di and S. Li, "Convolutional Neural Network Based on Principal Component Analysis Initialization for Image Classification," *2016 IEEE International Conference on Data Science in Cyberspace (DSC)*, Changsha, 2016, pp. 329-334.

[28] Hu Q., Li F., "Hardware design of smart home energy management system with dynamic price response," *IEEE Transactions on Smart Grid*, vol. 4, no. 4, pp. 1878-1887, 2013.

[29] Jang, Youngkyoon, Gunes, Hatice and Patras, Ioannis, "Registration-free Face-SSD: Single shot analysis of smiles, facial attributes, and affect in the wild," *Computer Vision and Image Understanding*, pp. 1-14, 2019.

[30] Singh, K. K. and Lee, Y. J., "Hide-and-seek: Forcing a network to be meticulous for weakly supervised object and action localization," *2017 IEEE International Conference on Computer Vision (ICCV)*, Venice, 2017, pp. 3544-3553.

[31] Zhou Z., Sanders J. W., Johnson J. M., et al., "Computer-aided Detection of Brain Metastases in T1-weighted MRI for Stereotactic Radiosurgery Using Deep Learning Single-Shot Detectors," *Radiology*, vol. 295, no. 2, pp. 407-415, 2020.

[32] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. E. Reed, C. Fu, and A. C. Berg, "SSD: single shot multibox detector," *European Conference on Computer Vision-ECCV 2016*, vol. 9905, 2016, pp. 21-37.

[33] Shi, Wenxu, Bao, Shengli and Tan, Dailun, "FFESSD: An Accurate and Efficient Single-Shot Detector for Target Detection," *Applied Sciences*, vol. 9, no. 20, pp. 4276-4288, 2019.
[34] K. Shih, C. Chiu, J. Lin and Y. Bu, "Real-Time Object Detection With Reduced Region Proposal Network via Multi-Feature Concatenation," in *IEEE Transactions on Neural Networks and Learning Systems*, vol. 31, no. 6, pp. 2164-2173, 2020.
[35] Y. He, X. Zhang, and J. Sun, "Channel pruning for accelerating very deep neural networks," in *Proc. Int. Conf. Comput. Vis. (ICCV)*, vol. 2, no. 6, 2017, pp. 1389-1397.
[36] J. Wu, C. Leng, Y. Wang, Q. Hu, and J. Cheng, "Quantized convolutional neural networks for mobile devices," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Las Vegas, NV, 2016, pp. 4820-4828.
[37] Lin, Mingbao, et al., "HRank: Filter Pruning using High-Rank Feature Map," *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Seattle, WA, USA, 2020, pp. 1526-1535.
[38] Rui Wang, Jingwen Xu, Tony X. Han, "Object instance detection with pruned Alexnet and extended training data, Signal Processing," *Singnal Processing: Image Communication*, vol. 70, pp. 145-156, 2019.
[39] Girshick, R., "Fast R-CNN," *2015 IEEE International Conference on Computer Vision (ICCV)*, Santiago, pp. 1440-1448, 2015.
[40] Arora Adwitiya, Grover Atul, Chugh Raksha, Reka S. Sofana, "Real Time Multi Object Detection for Blind Using Single Shot Multibox Detector," *2019 Wireless Personal Communications, 651 EP, 661 VL, 107, IS-1AB*, 2019.
[41] Singh, Pravendra, Ravikiran, Manikandan, Matiyali, Neeraj, Namboodiri, Vinay, "Multi-Layer Pruning Framework for Compressing Single Shot MultiBox Detector," *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*, Waikoloa Village, HI, USA, 2018, pp. 1318-1327.

BIOGRAPHIES OF AUTHORS

I., S. Ushasukhanya pursuing doctorate in Annamalai University, Faculty of Engineering and Technology, Tamilnadu, India. Research interests include Machine learning, Deep learning and Optimization Techniques.

S. Jothilakshmi, Associate professor, Department of Information and Technology, Annamalai University, Tamilnadu, India. Research interests include Speech processing, Image processing and Machine learning.