Building Occupancy Detection and Localization Using CCTV Camera and Deep Learning

Shushan Hu, Peng Wang, Cathal Hoare, and James O'Donnell

***Abstract*—Occupancy information plays a key role in analyzing and improving building energy performance. The advances of Internet of Things (IoT) technologies have engendered a shift in measuring building occupancy with IoT sensors, in which cameras in closed-circuit television (CCTV) systems can provide richer measurements. However, existing camera-based occupancy detection approaches cannot function well when scanning videos with a number of occupants and determining occupants' locations. This article aims to develop a novel deep-learning-based approach for better building occupancy detection based on CCTV cameras. To do so, this research proposes a deep-learning model to detect the number of occupants and determine their locations in videos. This model consists of two main modules, namely, feature extraction and three-stage occupancy detection. The first module presents a deep convolutional neural network to perform residual and multibranch convolutional calculation to extract shallow and semantic features, and constructs feature pyramids through a bidirectional feature network. The second module performs a three-stage detection procedure with three sequential and homogeneous detectors which have increasing Intersection over Union (IoU) thresholds. Empirical experiments evaluate the detection performance of the approach with CCTV videos from a university building. Experimental results show that the approach achieves the superior detection performance when compared with baseline models.**

***Index Terms*—Building occupancy detection, deep learning, Internet of Things (IoT) sensor.**

I. Introduction

INTERNET of Things (IoT) technologies have advanced the research of smart buildings in recent years [1]. As the single largest portion of global energy consumption (i.e., around 40%) and greenhouse gas emission (i.e., around 35%) [2], energy efficiency in smart buildings has attracted great attention [2]. Against this backdrop, occupancy detection plays an essential role in achieving more efficient building energy management [3]. First of all, occupancy information can assist building managers in comprehensively assessing building energy performance and identifying building operation faults [4]. In addition, this information can help to optimize operations of equipment (e.g., heating, ventilation, and air conditioning (HVAC) systems) while maintaining a comfortable indoor environment [5]. Moreover, long-term occupancy information can accomplish better demand responses by identifying energy peak demand periods of buildings and smoothing energy requirement peaks of buildings in smart grid [6].

A number of developed IoT sensors offer exciting opportunities for measuring occupancy status in buildings, but some of these sensors might be limited by inherent characteristics [7]. The approaches based on passive infrared (PIR) sensors can detect infrared light radiating from occupants which leads to the low-accuracy performance for multiple and static occupants. Environmental and electricity sensors are able to measure indoor environmental variations related to the presence of occupants, resulting in time delay and poor detection performance in complex environments (e.g., opening rooms). Radio frequency identification (RFID) and WiFi sensors require additional devices to receive signal from users and can be significantly affected by noise from obstacles.

An accurate and feasible solution for occupancy detection in buildings lies in leveraging closed-circuit television (CCTV) cameras to capture optical streams of the indoor environment. Device availability can improve the applicability and cost efficiency due to the widespread deployment of CCTV cameras in many buildings for safety and security issues. A key challenge in camera-based occupancy detection is eliminating interference caused by background and object obfuscation. Some powerful techniques from the computer vision domain have opened up potential of obtaining occupancy information from CCTV videos [8]. Some early efforts applied pattern recognition technologies (e.g., filtering algorithm, classification, and clustering methods) to subtract background information from videos, but these background subtraction-based approaches can fail if occupants remain static for extended periods.

Recent research has intended to apply machine learning to approximate the complex relationships between sensor measurements and occupancy information [9]. Machine learning performs classification and regression operations to filter image patches with learnable weights and biases. Shallow learning algorithms [e.g., support vector machine (SVM)] provide limited capacity to finish occupant localization. Deep learning significantly improves the learning performance through deeper and more complex neural network architectures and has been applied to a wild range of fields. For example, a significant body of research aimed to develop

Manuscript received 1 February 2022; revised 15 March 2022; accepted 18 August 2022. Date of publication 26 August 2022; date of current version 22 December 2022. This work was supported in part by the Natural Science Foundation of China under Grant 62106069, and in part by the Science Foundation Ireland (SFI) under Grant SFI 20/US/3695. *(Corresponding author: Shushan Hu.)*

Shushan Hu and Peng Wang are with the School of Computer Science and Information Engineering, Hubei University, Wuhan 430062, China (e-mail: shushan.hu@hubu.edu.cn; wangpeng@student.hubu.edu.cn).

Cathal Hoare and James O'Donnell are with the School of Mechanical and Materials Engineering and UCD Energy Institute, University College Dublin, Dublin 4, D04 V1W8 Ireland (e-mail: cathal.hoare@ucd.ie; james.odonnell@ucd.ie).

Digital Object Identifier 10.1109/JIOT.2022.3201877

deep-learning-based image segmentation solutions that play a key role in medical image analysis, robotic perception, and augmented reality [10]. Deep learning has engendered a shift for the communication and networking field, such as jointly optimizing transmitter and receiver components for the physical layer [11], and accurately classifying mobile encrypted traffic to obtain highly-valuable profiling information [12]. Advances in deep learning also give rise to the proliferation of industrial IoT with potential for various applications, such as smart assembling and smart manufacturing [13]. However, existing deep-learning-based occupancy detection approaches cannot function well for complex videos with overlaps among occupants [14].

This research intends to develop a novel deep-learning-based approach to obtain accurate occupancy information for building energy management. The approach needs to eliminate interference from overlaps among occupants and complex background in CCTV videos. Several technical challenges inhibit the pathway to achieve this goal. First, to inference multiple and varisized occupants leads to issues associated with relevant feature extraction from videos. Second, to accurately localize occupants in videos results in a complex regression problem adjusting the position and size of predicted boundaries of occupants. To address the above challenges, this research proposes to design a novel deep learning model to scan videos for high-quality occupancy information, including the number of occupants and their locations. In doing so, this work refines the detection problem into two main procedures: 1) extracting fine-grained features related to occupants from videos and 2) detecting the number of occupants and calculating their locations.

The main contributions are summarized as follows.

1) We propose a deep learning model for building occupancy detection and localization based on CCTV cameras. This model is capable of delivering the better detection performance when facing videos with many occupants.
2) We design a deep neural architecture to construct feature pyramids from CCTV videos. The architecture uses a deep convolutional neural network (DCNN) with residual and multibranch convolutional calculation to extract shallow and semantic features, and constructs feature pyramids through a top-down and a bottom-up pathway integration operations.
3) We perform a three-stage detection procedure to calculate the number of occupants and regress their locations. This procedure employs three sequential and homogeneous detectors to conduct iterative detection on feature pyramids. Increasing Intersection over Union (IoU) thresholds are applied to these detectors so as to improve the detection performance.
4) We conduct empirical experiments to evaluate the detection performance. Experimental results show that the approach can achieve superior performance on detecting the number of occupants and determining the locations of these occupants.

The remainder of this article is organized as follows: Section II presents the case of measuring occupancy status and filtering occupancy information. A detailed description of the approach is illustrated in Section III. Section IV demonstrates the effectiveness and superior performance with empirical experiments. Section V details the conclusions from the development and testing of this approach.

TABLE I
COMPARATIVE ANALYSIS OF SENSORS FOR OCCUPANCY DETECTION

Sensors	Strength	Weakness
PIR[24]	Less privacy concerns and localisation of occupants	Low accuracy for multiple occupants and failures for static occupants.
Environmental sensors [25]	Non-intrusive detection at no additional cost.	Time delay and low accuracy in opening rooms.
Electricity meters [8]	Non-intrusive detection and no additional installation cost.	Hard to model interactions between occupancy and power.
RFID [17]	Occupancy localisation and multiple occupant detection.	Attachments of tags and deployments of readers.
WiFi [26]	Passive activity recognition and wide availability of WiFi.	Need for signal receivers and low accuracy in complex environment.
Smartphone [20]	Passive activity recognition with no additional cost.	Low performance when placing smartphones aside.
Camera [21]	High detection accuracy and multiple occupant detection.	Computation complexity and occlusion problems.

II. RELATED WORKS

A. Occupancy Measurement With Sensors

The advances of IoT technologies have promoted a wide range of sensors available to measure building occupancy status (Table I). To be specific, PIR sensors can detect the motion of occupants by measuring infrared light radiating from objects. Yun and Woo [15] demonstrated a quantitative performance analysis on human movement direction detection based on PIR sensors, but PIR sensors may fail to measure static occupants. Environmental sensors aim to detect occupancy through environmental variations caused by the presence of occupants. Zimmermann *et al.* [16] presented an approach to detect occupants using a fusion of environmental sensors from an indoor air quality measurement system. Two limitations of environmental sensors are a time delay and low detection accuracy in open cases.

RFID sensors are capable of identifying and tracking tags attached to objects. Li *et al.* [17] developed an RFID-based system to track stationary and mobile occupants for demand-driven operations. RFID sensors may fall short in requiring tag attachments on objects and reader deployments. WiFi becomes a primary signal for occupancy detection with channel state information [18]. Sheng *et al.* [19] presented a deep learning framework for action recognition by integrating spatial features from CNN into a temporal bidirectional long short-term memory (LSTM) model. Nonetheless, WiFi approaches need additional signal receivers and have low performance in complex indoor environment. Smartphones provide an alternative for occupancy detection with embedded sensors. Chen *et al.* [20] leveraged smartphones to recognize human activities through extreme learning machine (ELM), coordinate transformation and principal component analysis (CT-PCA), and online SVM. The placing aside of smartphones significantly reduces the detection performance.

Cameras provide a straightforward optical instrument to capture visible images for occupancy detection. Zulcaffle *et al.* [21] designed a four-part method for frontal view gait recognition based on Time-of-Flight cameras. However, limitations include computation complexity and probability with occlusion. Electricity meters are used to detect interactions between occupants and appliances [22]. Feng *et al.* [8] tried to detect building occupancy from advanced metering infrastructure (AMI) data with a CNN and an LSTM network. Jin *et al.* [23] presented another smart meter-based solution which applies a multiview-based iteration and surrogate loss to learn refined results. The interactions between occupancy and power are not straightforward, and sometimes difficult to model. Among these sensors, cameras can provide richer measurements on occupancy status than other sensors. Growing deployments of CCTV cameras also promoted this relevant topic for building occupancy detection. However, a major barrier obstructing camera-based detection approaches is eliminating interference caused by background and object obfuscation.

TABLE II
COMPARISON OF CAMERA-BASED OCCUPANCY DETECTION ALGORITHMS

Algorithm	Characteristic	Weakness
Background subtraction[27]	Using Gaussian model without training data.	Significant affection from lumination variations
SVM [30]	Regressing clear margin in high dimensional spaces.	Low performance in noisy data set.
CNN [31]	Learning fine-grained features from image pixels.	falling short in occupant localisation.
Faster R-CNN[34]	Enabling region proposals and sharing features.	Unbalanced detection with single IoU threshold
YOLO [35]	Performing one stage regression for detection.	Falling short in complex scenarios.

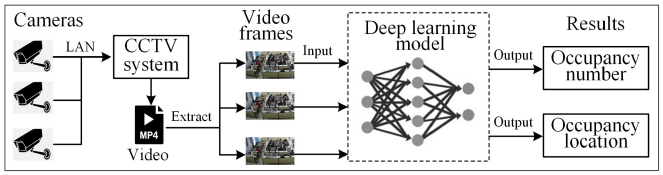

Fig. 1. System architecture of occupancy detection with CCTV video.

B. Occupancy Detection With Algorithms

The development of computer vision technologies has orchestrated a paradigm shift in the way that obtains fine-grained occupancy information from videos (Table II). An early study is background subtraction using a Gaussian mixture model. Tamgade and Bora [27] developed an optical flow motion vector estimation through iterative Lucas–Kanade pyramidal implementation. Histogram of oriented gradient (HOG) provides another solution by counting occurrences of gradient orientation in localized portions of an image. Cao *et al.* [28] leveraged an artificial neural network to classify occupancy information with HOG features.

The technology boom of machine learning enriched occupant detection within videos by learning abundant hidden rules between inputs to outputs based on a set of training data. With a clear margin in high dimensional spaces, Shih [29] proposed an SVM-based system with an image-based depth sensor and a pan-tilt-zoom camera. Yang *et al.* [30] used SVM to analyze whether a person is entering or exiting rooms based on the difference between two pixel coordinates.

CNN emerged as a deep learning algorithm to analyze visual imagery through assigning learnable weights and biases to various objects in an image. Fine-grained features learned from CNN provide confident and relevant context for occupant detection. Thys *et al.* [31] presented a CNN-based approach to generate adversarial patches of occupants with lots of intraclass variety. Zou *et al.* [32] developed a CNN approach to detect people head windows with high recall and precision.

Faster R-CNN introduces a novel detection idea by enabling nearly cost-free region proposals and sharing convolutional features with the detection network [33]. Ahmed *et al.* [34] demonstrated the applications and effectiveness of Faster R-CNN through facilitating video analysis for overhead view detection and segmentation. YOLO presents an alternative solution that aims for high speed and adequate accuracy by performing one-stage regression analysis for object detection with spatially separated bounding boxes and associated class probabilities. Kajabad and Ivanov [35] presented a YOLO-based approach to detect people in a closed space and identify density areas from CCTV cameras in a museum.

Several weaknesses arise from these algorithms. To be specific, lumination variations significantly affect the performance of background subtraction. SVM may demonstrate a lower performance when more noise exists in data sets or the feature number exceeds the sample number. CNN-based approaches fall short in occupant localization. Faster R-CNN and YOLO use a single IoU threshold which may result in an unbalanced issue for the detection performance. Low thresholds usually produce noisy results while higher thresholds tend to degrade the performance.

III. METHODOLOGY

This research intends to develop a novel deep-learning-based approach to obtain better building occupancy detection. In doing so, the approach applies CCTV cameras as sensors to measure occupants' status and designs a deep learning model to calculate the number of occupants and their locations as output (Fig. 1). Measurements from cameras are stored as video files and sequential frames are extracted as input images for the deep learning model.

A. Problem Formulation and Model

The core problem of the detection approach is designing a deep learning model to eliminate abundant and complex interference in images for enhanced detection performance. In accordance with the principle of deep learning, we formulate the detection problem into two main procedures called feature extraction and occupancy detection. First, a convolutional calculation is performed on all pixels of input images to obtain

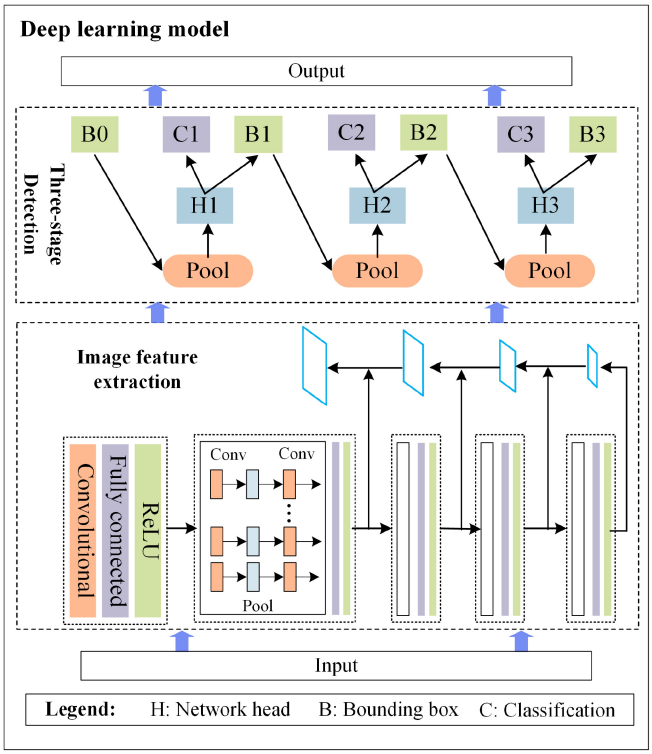

Fig. 2. Structure of the deep learning model for occupancy detection and localization.

shallow and semantic features

$$\Omega = f\left(\sum_{i=1,j=1}^{m,n}\left(w_{ij} * x_{ij}\right) + b\right) \tag{1}$$

where Ω,x_{ij}, w_{ij}, and b stand for the output features, input value at position (i, j) in images, weight value for position (i, j), and bias, respectively. m and n indicate the size of input images. The size of features $\Omega \in R^{H\times W\times C}$ consists of a height value H, a width value W, and a channel number C.

Second, the occupancy detection procedure is split into a classification subproblem and a regression subproblem. A classification function $\mathcal{O} = f_{cl}(\Omega^{H\times W\times C})$ is used to identify if the potential object is a occupant or not ($\mathcal{O} \in \{1, 0\}$). A regression function $\mathcal{B} = f_{re}(\Omega^{H\times W\times C})$ is used to determine the closest bounding boxes for target occupants. $\mathcal{B}$ represents a predicted bounding box with four parameters $\{b_x, b_y, b_w, b_h\}$, where b_x and b_x stand for the position of the left corner of the box, b_w and b_h indicate the width and height of the box.

In practice, there are multiple occupants contained in CCTV videos. The occupancy detection is extended as a multidimensional solution. Let N be the number of occupants and the output Φ indicates the N-dimensional occupancy vector $\Phi = (\mathcal{O}_1, \mathcal{B}_1), (\mathcal{O}_2, \mathcal{B}_2), \ldots, (\mathcal{O}_N, \mathcal{B}_N)$.

Following the problem formulation, this research designs a novel deep learning model to solve the detection problem (Fig. 2). Algorithm 1 illustrates a pseudocode of the deep learning model. Based on input images, this model leverages two main modules, namely, feature extraction and three-stage occupancy detection, to finish the occupancy detection mission. The first module learns multiscale features from input images with a DCNN and constructs feature pyramids

Algorithm 1 Pseudo Code of the Occupancy Detection Model

```
# construct feature pyramid from images
feature extraction function:
    extract feature as dcnn(img)
    obtain feature pyramids as
    bifpn_downchannel(feature), bifpn_convup(up_feature)
    return feature_pyramid
end function

# perform three-stage detection
three-stage detection function:
    obtain region proposals as
    rois = RPN(feature_pyramid)
    for each stage i do
        extract ROI feature as roi_pooling(rois)
        map ROI feature to sample space as
        fc_forward(pool_result)
        perform classification as fc_classifier(fc)
        perform regression as fc_bbox_reg(fc)
        obtain classification loss as
        loss_cls = cross_entropy(cls_score, label)
        obtain regression loss as
        loss_reg = L2_loss(bbox_pred, gt_bbox)
        obtain final loss as
        loss = loss_cls + loss_reg
        optimise model by backprogration
    end for
    return objects and bounding boxes
end function
```

through a bidirectional feature pyramid network consisting of a top-down and a bottom-up feature integration pathway. The second module performs a three-stage detection with sequential and homogeneous detectors which have increasing IoU thresholds. Two loss functions, including a classification loss and a bounding box regression loss, are used to conduct backpropagation for optimization of model parameters.

B. Feature Extraction

This approach designs a DCNN as the backbone network to learn multiscale and high-dimensional features from input images. The architecture of a building block in the DCNN follows a strategy of splitting, transforming, and aggregating [36] [Fig. 3(a)]. This strategy divides one single training path into a group of convolution paths. Features from these paths are depth aggregated to final output.

The splitting phase first leverages a base layer to separate the input into two parts (Parts 1 and 2) with channel data $(x_l = [x'_l, x''_l])$. x''_l will go through a dense convolutional block and a transition layer. x'_l is then combined with transmitted features to the next stage. The dense block is defined as a homogeneous and multibranch building block with a set of transformations with the same topology. A dimension called "cardinality" is defined to represent the size of the set of transformations. Cardinality is an essential factor in addition to the dimensions of depth and width. A more effective way to

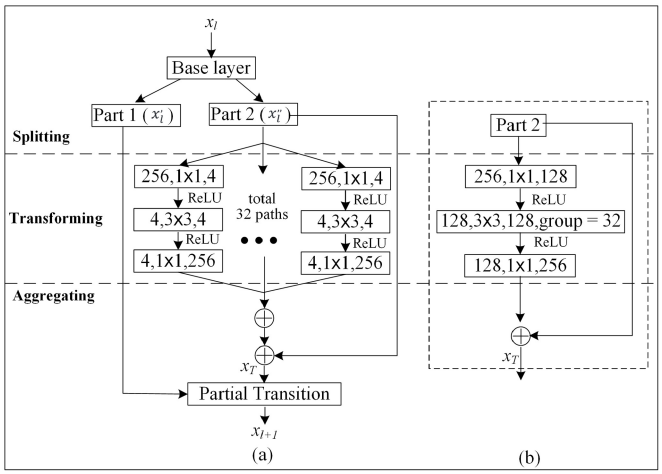

Fig. 3. (a) Building block of DCNN with cardinality = 32. (b) Building block of DCNN with grouped convolutions.

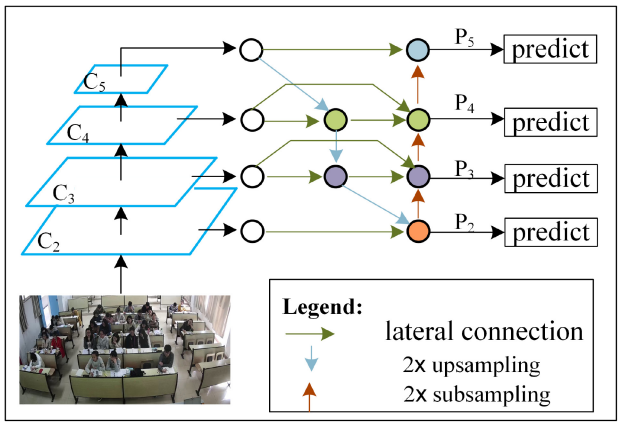

Fig. 4. Feature network of constructing feature pyramids with bidirectional cross-scale connections.

improve learning capacity lies in increasing cardinality, rather than constructing a deeper or wider architecture.

In the transforming phase, each convolution path consists of three convolution layers: 1) 1×1 4-d layer; 2) 3×3 4-d layer; and 3) 1×1 256-d layer. A novel transformation function (2) replaces the elementary transformation in a simple neuron

$$\mathcal{F}(x) = \sum_{i=1}^{C} \mathcal{T}_i(x) \tag{2}$$

where $\mathcal{T}_i(x)$ and C stand for an arbitrary function and the size of the set of transformations (i.e., cardinality), respectively. $\mathcal{T}_i(x)$ projects the input vector x into an embedding code and then, transforms the code. C is a newly introduced hyperparameter for a new pathway of adjusting the model capacity.

The aggregating phase deeply integrates intermediate transformed features into final features. An aggregated transformation function ($y = \mathcal{T}_i(x) + x$) is used to construct the residual function for the building block. y is the output of a block. This structure reformulates the network layers with reference to the layer inputs instead of learning unreferenced functions. A residual learning reformulation solves the degradation problem as the network depth increases. The reformulation approximates the network as a residual function $\mathcal{F}(x) := \mathcal{H}(x) - x$ and defines the building block as residual learning in the network layers

$$x_T = \mathcal{F}\left(x_l'', W_l\right) + W_l' x_l'' \tag{3}$$

where x_T, x_l'', $\mathcal{F}$, W_l, and W_l' stand for the output and input vectors of the network layers, the residual mapping function, and weight vectors, respectively. $\mathcal{F}(x_l'', W_l)$ needs to be learned during the training process. If the dimensions of $\mathcal{F}$ and x_l'' are not equal, the model conducts a linear projection W_l' on x_l'' to match the dimensions.

In addition, a partial transition layer is designed to generate the output feature x_{l+1} by maximizing the difference of gradient combination

$$x_{l+1} = W_l'' * \left[x_l', x_T\right]. \tag{4}$$

The partial transition layer conducts a hierarchical feature fusion strategy by truncating the gradient flow to prevent distinct layers from learning duplicate gradient information. This truncation combines the output from the dense block with the features coming from part 1. The strategy is able to significantly reduce the computation complexity since the gradient flow is truncated and the gradient information will not be reused.

In order to simplify the structure of the dense block and improve training efficiency, an equivalent structure is reformulated [Fig. 3(b)] for the dense block shown in Fig. 3(a). This reformulation replaces the low-dimensional embedding layers [i.e., the first 1×1 4-d layers in Fig. 3(a)] with a single and wider layer [i.e., the first 1×1 128-d layers in Fig. 3(b)]. The grouped convolution layer divides its input channels into 32 groups of convolutions whose input and output channels are 4-D. The third layer leverages a 1×1 filter to match the input dimensions to output vectors.

In order to enable multiscale occupancy detection with proportionally sized features, the approach applies a feature network to construct feature pyramids using intermediate features from the DCNN (Fig. 4). With the pyramidal hierarchy of DCNN, the features from shallow layers (e.g., C_2, C_3) contain high-level resolutions but low-level semantic information. The feature from deep layers (e.g., C_4, C_5) have high-level semantic information but low-level resolutions.

This feature network integrates both the bidirectional cross-scale connections and the fast normalized fusion. In detail, the network consists of a bottom-up pathway, a top-down pathway, and lateral connections to integrate features with high-level resolutions and semantic information without much calculation load [37].

With hierarchical features (C_2, C_3, C_4, C_5) from a typical feed-forward convolutional calculation, the top-down pathway constructs upsampled features by generating higher resolution features through an upsampling calculation. These upsampled features are enriched through lateral connections sourced from features at the same level. The fast normalized fusion is used to integrate different features: $O = \sum_i ([w_i]/[\epsilon + \sum_j w_j]) \cdot I_i$, where w_i is weight vector for input feature I_i, $\epsilon = 0.0001$ is a small value to avoid numerical instability. For example, the

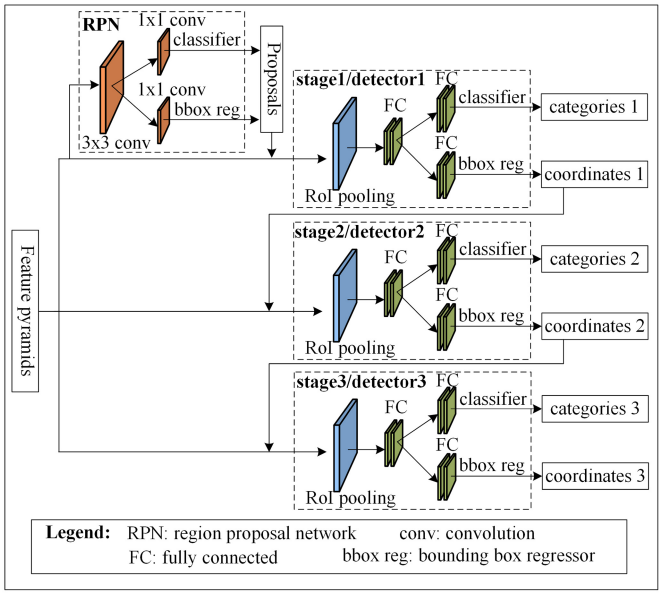

Fig. 5. Procedure of three-stage detection with three sequential and homogeneous detectors.

upsampled feature at level 4 is calculated as

$$P_4^{td} = \text{Conv}\left(\frac{w_1 \cdot C_4 + w_2 \cdot Resise(C_5)}{w_1 + w_2 + \epsilon}\right) \tag{5}$$

where the *Resise* function is an upsampling with the nearest neighbor strategy on the spatial information by a factor of $2\times$. A lateral connection is used to enhance the unsampled results with previous features at the same level (C_4). The connection indicates an element-wise addition operation and a 1×1 conv for a reduction of channel dimensions.

To obtain fine-grained features, the bottom-up pathway performs another integration upon upsampled features of the top-down pathway and features of feed-forward computation. This pathway applies subsampling functions to construct an opposite integration direction compared to the top-down pathway. For example, the final features at level 4 are calculated as

$$P_4 = \text{Conv}\left(\frac{w_1' \cdot C_4 + w_2' \cdot P_4^{td} + w_3' \cdot \text{Resise}(P_3)}{w_1' + w_2' + w_3' + \epsilon}\right) \tag{6}$$

where the *Resise* function is a subsampling with the nearest neighbor strategy on the spatial information by a factor of $2\times$. Beside subsampled features from higher resolution features (P_3), two lateral connections integrate two previous features (C_4 and P^{td}) for the final integration.

C. Three-Stage Occupancy Detection

With feature pyramids, the approach performs a three-stage detection to obtain high quality occupancy information (Fig. 5). Three homogeneous detectors (i.e., *detector*1, *detector*2, and *detector*3) perform a subtraining operation in a sequential order and have the same topology, which can generate high quality hypotheses at inference time and achieve better match results with increasing quality of detectors. Increasing IoU thresholds are used for these detectors in order to smooth the unbalanced detection arising from a single IoU threshold, as low thresholds usually produce noisy results while higher thresholds tend to degrade the performance due to overfitting and mismatch problems [38]. The three-stage detection is represented as an iterative detection process to filter occupancy information from images. A region proposal network (RPN) generates initial region proposals for the detector of stage 1 (*detector*1), the output of which provides additional and better region proposals to train the detector of stage 2 (*detector*2). The subsequent detector at stage 3 (*detector*3) applies the output of *detector*2 as region proposals to improve the detection performance.

RPN conducts a resampling operation on feature pyramids and predicts region proposals for objects at each pixel of the input images. The resampling applies a convolution layer (3×3 conv) to transform the input features as 256-d. Two parallel convolution layers (1×1 conv) are used to generate enhanced features for subsequent classification and regression missions. A classifier and a bounding box regressor are responsible for predicting k region proposals, which are parameterized relative to k reference boxes (i.e., anchors). Finally, a loss function (7) minimizes the difference between the reference and ground truth boxes

$$L(\{p_i\}, \{t_i\}) = \frac{1}{N_{\text{cls}}} \sum_i L_{\text{cls}}\left(p_i, P_i^*\right) + \lambda \frac{1}{N_{\text{reg}}} \sum_i p_i^* L_{\text{reg}}\left(t_i, t_i^*\right) \tag{7}$$

where i, p_i, and p_i^* stand for the index of an anchor, the probability of a predicted anchor being an object, and the ground-truth label valuing 1 with a positive anchor, respectively.

With initial region proposals, the three detectors perform an iterative detection with increasing IoU thresholds. The detector integrates feature pyramids and region proposals from RPN (or detectors of previous stages) as input. An RoI pooling layer is used to resample the input features as fixed size. Following a retraining process of two fully connected layers, a classifier checks if the anchors contain occupants or not. A bounding box regressor tries to precisely localize bounding box coordinates by a mount of translating and scaling operations.

The classifier is defined as a function $h(x)$ aiming to categorize candidate objects into one of two classes. Class 0 indicates image background and class 1 is the occupant category. The function $h(x)$ applies the posterior distribution upon an patch x of images and predicts a class label y for objects in this patch ($h_k(x) = p(y = k|x)$). A risk function assesses the classification performance of the function and improves the performance through a training process

$$\mathcal{R}_{\text{cls}}[h] = \sum_{i=1} L_{\text{cls}}(h(x_i), y_i) \tag{8}$$

where L_{cls} stands for a cross-entropy loss function.

The bounding box regressor represents another function $f_t(x, p)$ to precisely confirm bounding box coordinates for occupants. The function regresses a predicted box p in an image patch x to a ground-truth box g in the stage t.

The bounding box p is represented with four parameters (p_x, p_y, p_w, p_h), where p_x and p_y are coordinates of the top left corner, p_w and p_h stand for the width and height of the box. The ground-truth box g is also represented with four parameters $g = (g_x, g_y, g_w, g_h)$. The regressor tries to minimize the bounding box risk by an indicator

$$\mathcal{R}_{\text{loc}}[f] = \sum_i L_{\text{loc}}(f(x_i, p_i), g_i) \tag{9}$$

where L_{loc} stands for the loss function calculating a distance vector Δ with four variables $(\delta_x, \delta_y, \delta_w, \delta_h)$ from

$$\begin{aligned} \delta_x &= (g_x - p_x)/p_w, \quad \delta_y = \left(g_y - p_y\right)/p_h \\ \delta_w &= \log(g_w/p_w), \quad \delta_h = \log(g_h/p_h). \end{aligned} \tag{10}$$

This approach applies a cascade regression strategy by adopting a resampling procedure to adjust the distribution of hypotheses for different stages. The strategy defines an overall regression function $f(x, p)$ as a cascade of specialized regressors $f_t(x, p)$ for the three detection stages

$$f(x, p) = f_1(x, p) \circ f_2(x, p) \circ f_3(x, p) \tag{11}$$

where f_1, f_2, and f_3 stand for the regressor for stage 1, 2, and 3, respectively. The operator $\circ$ is cascaded linkage meaning each regressor f_t is optimized for the box p_t generated by the previous regressor f_{t-1}.

A loss function is defined to optimize the detection performance of each stage t with an IoU threshold u^t

$$\begin{aligned} L\left(x^t, g\right) = L_{\text{cls}}\left(h_t\left(x^t\right), y^t\right) \\ + \lambda\left[y^t \geq 1\right] L_{\text{loc}}\left(f_t\left(x^t, p^t\right), g\right) \end{aligned} \tag{12}$$

where p^t, λ, and y^t stand for the regressor of stage $t-1$ [i.e., $f_{t-1}(x^{t-1}, p^{t-1})$], the tradeoff coefficient, and the label of the patch x_t under the threshold u^t, respectively.

IV. Performance Evaluation

In order to demonstrate the effectiveness and superior performance of the approach, this work documents a case study of occupancy detection when applying some other popular solutions as the baseline.

A. Experimental Setup

The case study begins by preparing a data set with labeled images to enable verification of the approach. With videos (resolution: 1920 × 1080) collected from CCTV cameras in a university building over a period of five weeks in 2019, this study conducts a preprocessing procedure with several steps to obtain labeled images. The procedure begins by trimming video segments recorded when the room is not occupied according to the teaching schedule of the target classrooms. The second step takes images from the remaining videos with an interval of 8 s during lectures and the interval becomes 5 s during breaks between lectures. The reason is occupants normally have more movements during breaks leading to frequent variations in images. Finally, the authors manually draw bounding boxes for occupants to generate labeled images. In summary, this study prepares nearly 12 000 labeled images for the data set. The demonstration uses 10 000 images of the data set to train the occupancy detection algorithms, while 2000 images are used to test the performance of these algorithms.

TABLE III
Detection Performance Under Different Model Parameters

Anchor area	Anchor ratio	Stage IoU	AP_{50}	MAE	MSE
32,64,128,256,512	0.5,1.0,2.0	0.5,0.6,0.7	83.1%	6.13	7.71
16,32,64,128,128	0.5,1.0,2.0	0.4,0.5,0.6	83.8%	3.45	3.95
16,32,64,128,128	0.75,1.0,1.5,2.0,3.0	0.5,0.6,0.7	84.4%	1.04	1.61
16,32,64,128,128	0.75,1.0,1.5,2.0,3.0	0.4,0.5,0.6	85.1%	0.94	1.28

This research implements the proposed deep learning model as an executable program for the performance test upon the data set. The implementation uses the Python programming language for coding and applies Pytorch as the basic platform. The program is run on a server with the following specifications: AMD 3960X CPU at 4.0GHz 24 cores, 128 GB DDR4 RAM, 2 × GeForce 2080Ti GPU, 512GB SSD, and Ubuntu ×64 OS.

B. Deep Learning Training

This study conducts a training process to improve the detection performance of the model. This training process starts from a learning rate of 0.0025 × 0.001 and follows the warmup strategy which keeps linear growth until the 10th epoch. We try to optimize the model parameters using the stochastic gradient descent (SGD) optimizer, which aims to minimize the cross entropy and smooth L1 objective functions following the opposite direction of gradient descent. In order to reach the convergence point, we train the developed model by increasing the epoch value from 1 to 30 (Fig. 6). Four metrics called Accuracy, mean absolute error (MAE), mean square error (MSE), and mean average precision (mAP) are used to evaluate the detection performance. It can be found that the model tends to convergence with the epoch value of 12. In particular, the accuracy value reaches the peak point at the 12th and the 21st epoch [Fig. 6(a)]. The MAE results show three troughs when the epoch values are 12, 17, and 21 [Fig. 6(b)]. The MSE results contain four minimal values at the 10th, 12th, 14th, and 21st epoch. The mAP curve reaches the peak point with the epoch value of 12 [Fig. 6(c)].

Regarding the three-stage detection architecture, we improve the detection performance by analyzing three main parameters called anchor area, anchor ratio, and stage IoU (Table III) with MAE, MSE, and mAP metrics. The model obtains relatively good performance with initial parameters (first row). We quickly confirm the anchor area to be [16, 32, 64, 128, 128] by analyzing the area of ground truth boxes. Another stage IoU ([0.4, 0.5, 0.6]) is tested to reach the first milestone (second row). We identify that the ratio of [0.75, 1.0, 1.5, 2.0, 3.0] can achieve better detection performance (third row). The integration of the anchor ratio and the stage IoU assists the model to achieve the best performance (fourth row).

C. Baseline Models

For the purpose of evaluating the performance of the approach, the case study selects five state-of-the-art detection

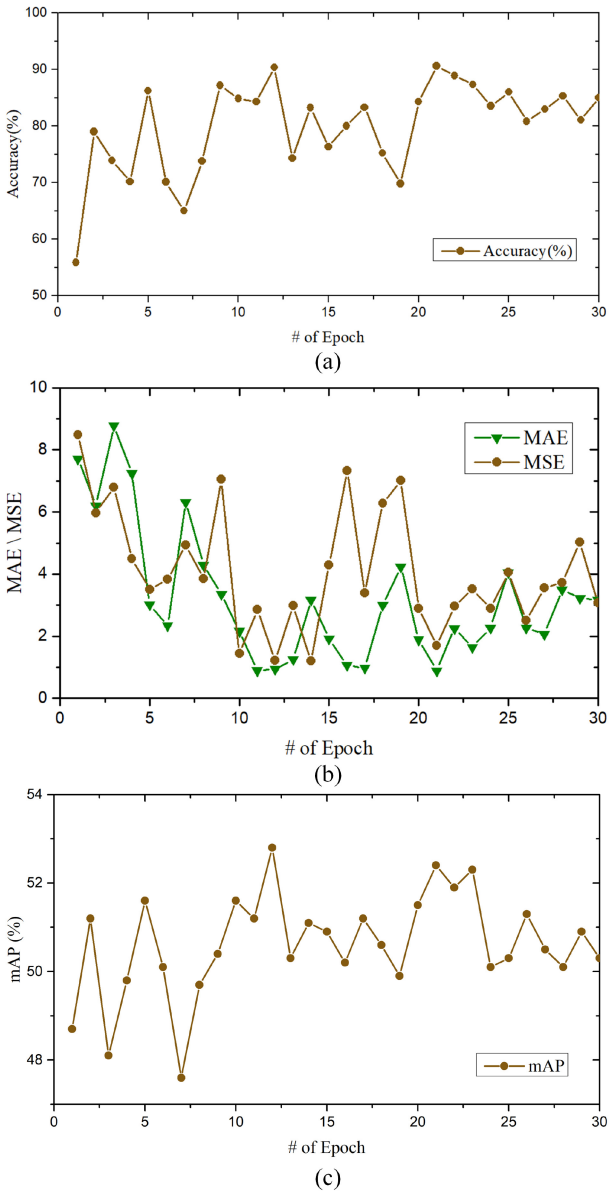

Fig. 6. Convergence curves for the deep learning model training. (a) Accuracy results. (b) MAE and MSE results. (c) mAP results.

models to establish a baseline for comparison. Faster R-CNN uses RPN to achieve good detection performance by predicting object bounding boxes and objective scores at each location. YOLOv4 presents a one-stage object detection solution with high speed and adequate accuracy and performs regression analysis for object detection with spatially separated bounding boxes and associated class probabilities. EfficientDet follows the one-stage paradigm for object detection by integrating EfficientNet with a set of fixed scaling coefficients and a bidirectional feature pyramid network. RetinaNet is another popular one-stage detector, which applies a feature pyramid network for detecting objects at multiple scales and defines the focal loss function to facilitate the extreme foreground-background class imbalance. An anchor-free extension of RetinaNet (AF RetinaNet) defines a simple and effective building block for single-shot object detectors by addressing issues such as heuristic-guided feature selection and overlap-based anchor sampling.

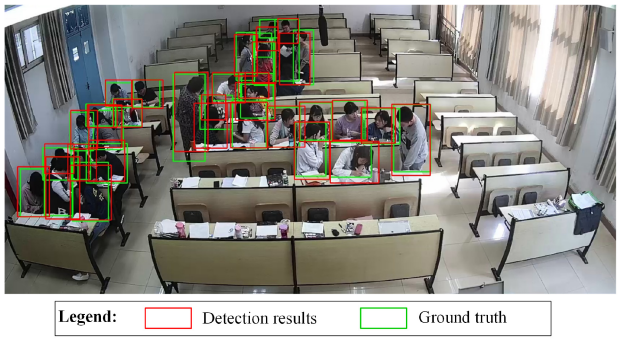

Fig. 7. Visual results of occupancy detection and localization in images.

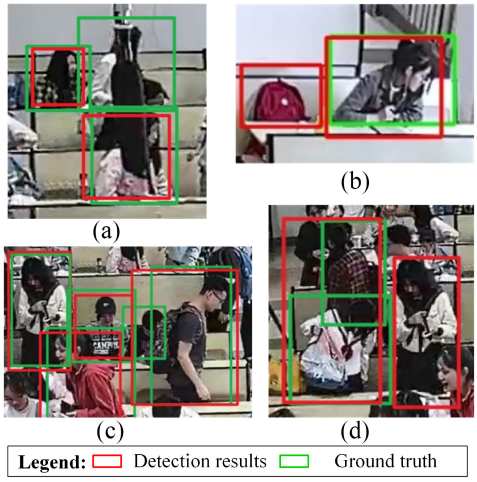

Fig. 8. Visual results of failed occupancy detection: (a) failed detection due to sheltering from a microphone; (b) incorrect detection of a schoolbag; (c) failed detection due to overlaps between occupants; and (d) failed detection due to overlaps between occupants.

D. Experimental Results

1) Detection Results Visualization: This study represents an example (Fig. 7) that visualizes the occupancy detection results of the approach. There are 32 occupants in this image where nearly one quarter of occupants are standing, while other occupants are sitting in fixed seats. Some overlaps exist among occupants due to the layout of seats and shielding from standing occupants. The approach successfully detects all 32 occupants contained in the image. Regarding occupancy localization, it is obvious that the predicted bounding boxes for occupants (i.e., red box) have high-degree overlaps with bounding boxes of ground truth (i.e., green box).

This section shows some failure cases generated by the proposed model (Fig. 8). The first two cases are caused by interference from irrelevant objects. Results in Fig. 8(a) present a failure when detecting the occupant behind the microphone and a schoolbag is recognized as an occupant in Fig. 8(b). The second two cases are due to occlusions from other occupants in crowded scenarios. One occupant is missed in Fig. 8(c) due to occlusion from another occupant in front. Two occupants in Fig. 8(d) are detected as only one occupant due to a vertically overlapping layout.

2) Occupancy Detection Performance: This study evaluates the performance when detecting the number of occupants.

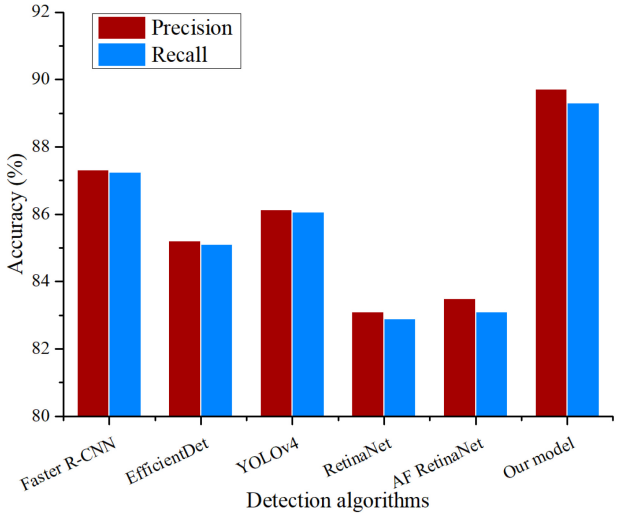

Fig. 9. Precision performance evaluating the number of occupants being correctly detected.

Two metrics called Precision and Recall are used to assess the detection accuracy (Fig. 9)

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}, \quad \text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{13}$$

where TP, FP, and FN stand for the true positive number, the false negative number, and the false positive number, respectively.

All detection models obtain relatively good performance upon detecting the number of occupants. To be specific, our model can detect most occupants with a Precision value of 89.7% and a Recall value of 89.2%, while RetinaNet has the worst performance with values (83.1%, 82.9%). The anchor free structure leads to a slight increase in the detection performance when compared with RetinaNet. Faster R-CNN has the second best performance with values (87.3%, 87.2%). YOLOv4 presents good accuracy results with the improvement based on bag of freebies and bag of specials. In general, two-stage detection models have better performance than one-stage models since RPN generates initial and valuable proposals to improve the detection performance.

The case study uses another metric called F1 score to represent the harmonic mean of Precision and Recall values when evaluating the prediction performance of models

$$F1 = \frac{2 \text{ Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}. \tag{14}$$

The F1 scores of all six models are shown in Fig. 10. These models have values between 83% to 90%, in which our model has the highest value (89.5%) and RetinaNet has the lowest value (83%). Faster R-CNN has the second best performance (87.2%). The difference between Faster R-CNN and our model is due to a single IoU threshold can not balance noisy results and good performance. YOLOv4 and EfficientDet have relatively good results (86.0% and 85.2%, respectively) due to the one-stage architecture that is a bit weak for detecting multiple occupants from complex images.

This assessment applies MAE and MSE metrics to evaluate the detection performance (Fig. 11). Our model has the lowest

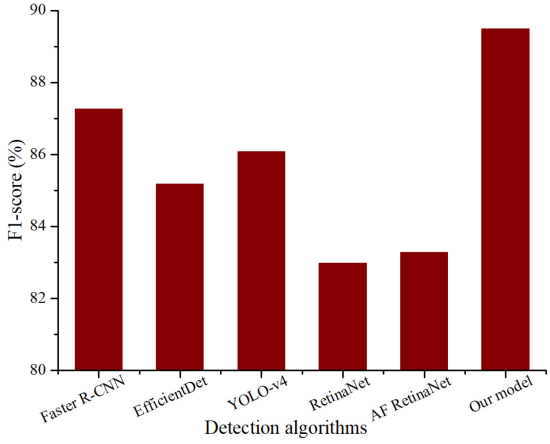

Fig. 10. Detection performance of all detection models in F1 score.

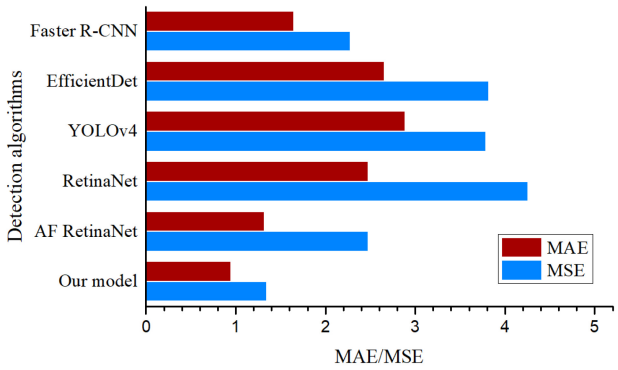

Fig. 11. Detection performance in terms of MAE and MSE, evaluating differences between predicted and observed values.

value (MAE:0.94, MSE:1.35) while YOLOv4 has the highest MAE value (2.89) and RetinaNet has the potential for the greatest improvement in terms of MSE (4.26). The reason is YOLOv4, RetinaNet and EfficientDet perform one-stage detection which predicts a fixed number of possible objects on a grid. AF RetinaNet presents good performance on both metrics (i.e., MAE:1.31, MSE:2.48) since it applies a feature selective anchor-free module to enhance limitations brought up by the conventional anchor-based detection.

3) Occupancy Localization Performance: This research leverages two popular metrics called average precision (AP) and mAP to evaluate the localization performance (Table IV). AP is a metric for calculating the overlap ratio between predicted and true boxes. mAP evaluates the overall performance by computing a mean value for all of the AP values at different IoUs.

mAP results of all models are illustrated in Fig. 12. These models obtain values of more than 40% in mAP indicating good performance when predicting bounding boxes for occupants. Our approach achieves a significant improvement in mAP (about 2%) when compared with other models. RetinaNet has the worst performance with an mAP value of 40.9%. YOLOV4 achieves better performance than Faster R-CNN. This is because mAP only relates to bounding boxes of correctly predicted occupants and YOLOV4 has fewer correctly predicted occupants than Faster R-CNN. The undetected

TABLE IV
PERFORMANCE COMPARISON OF DIFFERENT DETECTION MODELS

Detection model	AP_{50}	AP_{60}	AP_{70}	AP_{80}
Faster R-CNN	82.4%	75.1%	55.4%	18.7%
YOLOv4	82.9%	77.1%	55.3%	18.0%
RetinaNet	81.3%	73.5%	52.4%	16.2%
AF RetinaNet	81.4%	74.6%	54.2%	18.6%
EfficientDet	81.5%	73.4%	54.1%	20.5%
Our model	85.1%	77.8%	58.3%	20.8%

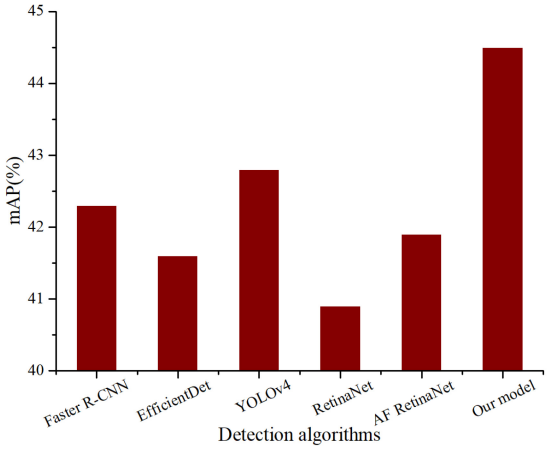

Fig. 12. Detection performance of all detection models in mAP.

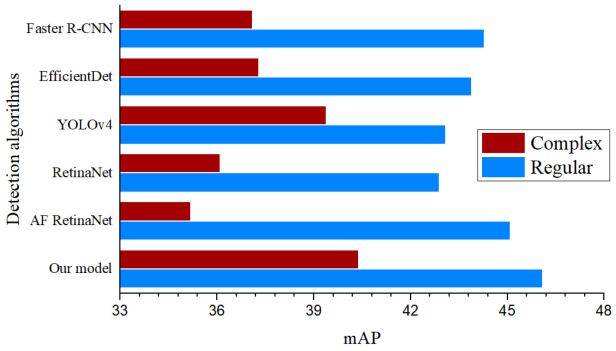

Fig. 13. Detection performance for images with different scenarios.

TABLE V
COMPARISON ON THE COMPUTATION PERFORMANCE

Detection model	Trainable parameters (Million)	Memory requirement (MegaByte)	Inference time (Second)
Faster R-CNN	4.1	1,540	0.058
YOLOv4	2.8	1,390	0.039
RetinaNet	3.6	1,450	0.051
AF RetinaNet	3.6	1,450	0.051
EfficientDet	2.3	1,330	0.045
Our model	4.8	1610	0.063

occupants are often surrounded by complex background and it is difficult to calculate their bounding boxes.

Regarding the AP metric, in accordance with the mAP metric, the proposed approach has the best performance for all IoU thresholds from AP_{50} to AP_{80}. RetinaNet has the lowest values at different IoU thresholds. It is worth noting that these models exhibit decreasing performance when increasing IoU thresholds. The reason is overlaps among occupants in images may cause detectors to treat crowded occupants as a single detection, or to mistakenly shift the target bounding box to another person with higher IoU thresholds.

Furthermore, this experiment finds that considerable differences exist in occupancy detection for different scenarios in images (Fig. 13). This research defines a complex scenario where an image contains more than 30 occupants and more than 10 occupants are standing, with more overlaps between occupants. The regular scenario indicates images outside the complex range. Although our approach achieves the best performance on both scenarios, a significant difference of nearly 5% shows in mAP when evaluating detection performance for the two scenarios. The biggest difference (around 10% in mAP) belongs to AF RetinaNet. This arises due to additional overlaps caused by more occupants and increased standing in the complex scenario, as such this poses a more difficult scenario for occupancy localization.

4) Computation Performance: This section documents the computation performance evaluation for all models (Table V). This performance evaluation selects three indicators, called trainable parameters, memory requirements, and inference time, to assess the computation and storage cost of these models. The inference time is the most important indicator representing the time of scanning one image. It is obvious that all models have super performance with a time less than 0.1 s. Although our model needs a bit more inference time due to a complex architecture for better occupancy detection, it is efficient to meet time requirements for future applications deployed in buildings. Regarding the other two metrics, though more trainable parameters in our model lead to more memory requirement, it is worth noting that memory requirements of all models (from 1390 to 1610 MB) only take up a small proportion of the total memory of popular GPUs, which normally is larger than 10 000 MB.

E. Discussion

This work enhances the performance evaluation by discussing the performance of our approach and some existing occupancy detection solutions. Due to the inaccessible source code or data set, this discussion is finished with results from references (Table VI). A head detection solution [32] applies HOG features and a 7-layer CNN to obtain good performance, but a low occupancy number (0–3) significantly reduces the detection difficulty when compared with our model. Two solutions [31], [35] apply the YOLO model to finish occupancy detection missions. It can be concluded that our model is better than them since one baseline model (YOLOv4) has better performance than YOLOv3 and YOLOv2 [39]. Although an occupancy presence detection solution [28] achieves a high accuracy value, our solution can have 100% accuracy when finishing presence detection. An overhead detection approach [34] leverages Faster R-CNN to detect occupants from an overhead angle of view and has a good true positive rate (TPR) value (94%). Though our model has a lower TPR value (89.2%), it can be inferred that our approach has superior performance than Faster R-CNN-based analysis in Section IV-D2.

TABLE VI
ANALYSIS WITH EXISTING OCCUPANCY DETECTION SOLUTIONS

No	Solution	Core model	Occupant range	Performance result
1	Head detection [32]	HOG & CNN	0-3	Precision: 92.7%
2	Occupancy localisation [35]	YOLOv3	0-10	Visuable boxes
3	Occupancy localisation [31]	YOLOv2	0-2	PR curve
4	Occupancy presence [28]	HOG & ANN	0-1	Accuracy: 96%
5	Overhead detection [34]	Faster R-CNN	0-10	TPR: 94%
6	Our model	DCNN & Detector	0-80	Precision: 89.7% TPR:89.2%

V. CONCLUSION

This research proposes a novel approach for building occupancy detection and localization using CCTV cameras and deep learning. In doing so, the approach designs a deep learning model consisting of two main modules to achieve better occupancy detection. The first module leverages a DCNN to learn shallow and semantic features from input images and constructs feature pyramids through a two-pathway integration and lateral connections. The second module represents a three-stage detection procedure that uses three sequential and homogeneous detectors to conduct iterative detection with increasing IoU thresholds. Experimental results show that the proposed model can achieve superior performance on occupancy detection and localization when compared with baseline models, including 0.94 in MAE and 1.35 in MSE for the detection performance, and 44.5% in mAP for the localization performance.

With this method, building managers and engineers can obtain the number of occupants and their locations in videos. Then, these managers have reliable information when formulating energy-efficient action plans with other stakeholders. Better energy-related strategies can be decided to improve building energy efficiency while maintaining environmental comfort for occupants. In addition, engineers are able to develop accurate building performance simulation and smooth energy consumption peaks for buildings in smart grids.

Future research will further investigate the benefits of occupancy detection and deep learning for building energy management. In particular, we plan to improve occupancy detection performance in the complex scenario by updating the deep learning model with possible improvements on the RPN architecture and the nonmaximum suppression mechanism. With precise and reliable occupancy information, a potential research opportunity lies with improving building energy performance by using deep reinforcement learning to enable better control of devices [40].

REFERENCES

[1] L. Yu, S. Qin, M. Zhang, C. Shen, T. Jiang, and X. Guan, "A review of deep reinforcement learning for smart building energy management," *IEEE Internet Things J.*, vol. 8, no. 15, pp. 12046–12063, Aug. 2021.

[2] S. J. Smith, "Cleaning cars, grid and air," *Nat. Energy*, vol. 6, no. 1, pp. 19–20, 2021.

[3] J. Jiang *et al.*, "Residential house occupancy detection: Trust-based scheme using economic and privacy-aware sensors," *IEEE Internet Things J.*, vol. 9, no. 3, pp. 1938–1950, Feb. 2022.

[4] H. Saberi, C. Zhang, and Z. Y. Dong, "Data-driven distributionally robust hierarchical coordination for home energy management," *IEEE Trans. Smart Grid*, vol. 12, no. 5, pp. 4090–4101, Sep. 2021.

[5] L. Yu *et al.*, "Multi-agent deep reinforcement learning for HVAC control in commercial buildings," *IEEE Trans. Smart Grid*, vol. 12, no. 1, pp. 407–419, Jan. 2021.

[6] K. Baek, E. Lee, and J. Kim, "Resident behavior detection model for environment responsive demand response," *IEEE Trans. Smart Grid*, vol. 12, no. 5, pp. 3980–3989, Sep. 2021.

[7] Y. Zhu, H. Luo, F. Zhao, and R. Chen, "Indoor/outdoor switching detection using multisensor DenseNet and LSTM," *IEEE Internet Things J.*, vol. 8, no. 3, pp. 1544–1556, Feb. 2021.

[8] C. Feng, A. Mehmani, and J. Zhang, "Deep learning-based real-time building occupancy detection using AMI data," *IEEE Trans. Smart Grid*, vol. 11, no. 5, pp. 4490–4501, Sep. 2020.

[9] K. Sun, Q. Zhao, and J. Zou, "A review of building occupancy measurement systems," *Energy Build.*, vol. 216, Jun. 2020, Art. no. 109965.

[10] S. Minaee, Y. Y. Boykov, F. Porikli, A. J. Plaza, N. Kehtarnavaz, and D. Terzopoulos, "Image segmentation using deep learning: A survey," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 7, pp. 3523–3542, Jul. 2022.

[11] T. O'Shea and J. Hoydis, "An introduction to deep learning for the physical layer," *IEEE Trans. Cogn. Commun. Netw.*, vol. 3, no. 4, pp. 563–575, Dec. 2017.

[12] G. Aceto, D. Ciuonzo, A. Montieri, and A. Pescapé, "Mobile encrypted traffic classification using deep learning: Experimental evaluation, lessons learned, and challenges," *IEEE Trans. Netw. Service Manag.*, vol. 16, no. 2, pp. 445–458, Jun. 2019.

[13] R. A. Khalil, N. Saeed, M. Masood, Y. M. Fard, M.-S. Alouini, and T. Y. Al-Naffouri, "Deep learning in the Industrial Internet of Things: Potentials, challenges, and emerging applications," *IEEE Internet Things J.*, vol. 8, no. 14, pp. 11016–11040, Jul. 2021.

[14] L. Rueda, K. Agbossou, A. Cardenas, N. Henao, and S. Kelouwani, "A comprehensive review of approaches to building occupancy detection," *Build. Environ.*, vol. 180, Aug. 2020, Art. no. 106966.

[15] J. Yun and J. Woo, "A comparative analysis of deep learning and machine learning on detecting movement directions using PIR sensors," *IEEE Internet Things J.*, vol. 7, no. 4, pp. 2855–2868, Apr. 2020.

[16] L. Zimmermann, R. Weigel, and G. Fischer, "Fusion of nonintrusive environmental sensors for occupancy detection in smart homes," *IEEE Internet Things J.*, vol. 5, no. 4, pp. 2343–2352, Aug. 2018.

[17] N. Li, G. Calis, and B. Becerik-Gerber, "Measuring and monitoring occupancy with an RFID based system for demand-driven HVAC operations," *Autom. Construct.*, vol. 24, pp. 89–99, Jul. 2012.

[18] X. Ma, Y. Zhao, L. Zhang, Q. Gao, M. Pan, and J. Wang, "Practical device-free gesture recognition using WiFi signals based on metalearning," *IEEE Trans. Ind. Informat.*, vol. 16, no. 1, pp. 228–237, Jan. 2020.

[19] B. Sheng, F. Xiao, L. Sha, and L. Sun, "Deep spatial–temporal model based cross-scene action recognition using commodity WiFi," *IEEE Internet Things J.*, vol. 7, no. 4, pp. 3592–3601, Apr. 2020.

[20] Z. Chen, C. Jiang, and L. Xie, "A novel ensemble ELM for human activity recognition using smartphone sensors," *IEEE Trans. Ind. Informat.*, vol. 15, no. 5, pp. 2691–2699, May 2019.

[21] T. M. A. Zulcaffle, F. Kurugollu, D. Crookes, A. Bouridane, and M. Farid, "Frontal view gait recognition with fusion of depth features from a time of flight camera," *IEEE Trans. Inf. Forensics Security*, vol. 14, pp. 1067–1082, 2019.

[22] D. H. Green, S. R. Shaw, P. Lindahl, T. J. Kane, J. S. Donnal, and S. B. Leeb, "A multiscale framework for nonintrusive load identification," *IEEE Trans. Ind. Informat.*, vol. 16, no. 2, pp. 992–1002, Feb. 2020.

[23] M. Jin, R. Jia, and C. J. Spanos, "Virtual occupancy sensing: Using smart meters to indicate your presence," *IEEE Trans. Mobile Comput.*, vol. 16, no. 11, pp. 3264–3277, Nov. 2017.

[24] C. Duarte, K. Van Den Wymelenberg, and C. Rieger, "Revealing occupancy patterns in an office building through the use of occupancy sensor data," *Energy Build.*, vol. 67, pp. 587–595, Dec. 2013.

[25] C. Jiang, Z. Chen, R. Su, M. K. Masood, and Y. C. Soh, "Bayesian filtering for building occupancy estimation from carbon dioxide concentration," *Energy Build.*, vol. 206, Jan. 2020, Art. no. 109566.

[26] J. Wang, N. C. F. Tse, and J. Y. C. Chan, "Wi-Fi based occupancy detection in a complex indoor space under discontinuous wireless communication: A robust filtering based on event-triggered updating," *Build. Environ.*, vol. 151, pp. 228–239, Mar. 2019.
[27] S. N. Tamgade and V. R. Bora, "Motion vector estimation of video image by pyramidal implementation of Lucas Kanade optical flow," in *Proc. Conf. Emerg. Trend. Eng. Technol.*, 2009, pp. 914–917.
[28] N. Cao, J. Ting, S. Sen, and A. Raychowdhury, "Smart sensing for HVAC control: Collaborative intelligence in optical and IR cameras," *IEEE Trans. Ind. Electron.*, vol. 65, no. 12, pp. 9785–9794, Dec. 2018.
[29] H.-C. Shih, "A robust occupancy detection and tracking algorithm for the automatic monitoring and commissioning of a building," *Energy Build.*, vol. 77, pp. 270–280, Jul. 2014.
[30] J. Yang *et al.*, "Comparison of different occupancy counting methods for single system-single zone applications," *Energy Build.*, vol. 172, pp. 221–234, Aug. 2018.
[31] S. Thys, W. Van Ranst, and T. Goedeme, "Fooling automated surveillance cameras: Adversarial patches to attack person detection," in *Proc. CVPR Workshops*, Jun. 2019, pp. 1–7.
[32] J. Zou, Q. Zhao, W. Yang, and F. Wang, "Occupancy detection in the office by analyzing surveillance videos and its application to building energy conservation," *Energy Build.*, vol. 152, pp. 385–398, Oct. 2017.
[33] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2015, pp. 91–99.
[34] I. Ahmed, S. Din, G. Jeon, and F. Piccialli, "Exploring deep learning models for overhead view multiple object detection," *IEEE Internet Things J.*, vol. 7, no. 7, pp. 5737–5744, Jul. 2020.
[35] E. N. Kajabad and S. V. Ivanov, "People detection and finding attractive areas by the use of movement detection analysis and deep learning approach," *Procedia Comput. Sci.*, vol. 156, pp. 327–337, Jan. 2019.
[36] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He, "Aggregated residual transformations for deep neural networks," in *Proc. CVPR*, 2017, pp. 1492–1500.
[37] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *Proc. CVPR*, 2017, pp. 2117–2125.
[38] Z. Cai and N. Vasconcelos, "Cascade R-CNN: High quality object detection and instance segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 43, no. 5, pp. 1483–1498, May 2021.
[39] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "Yolov4: Optimal speed and accuracy of object detection," 2020, *arXiv:2004.10934*.
[40] L. Yu *et al.*, "Deep reinforcement learning for smart home energy management," *IEEE Internet Things J.*, vol. 7, no. 4, pp. 2751–2762, Apr. 2020.

Shushan Hu received the B.S., M.S., and Ph.D. degrees in software engineering from Wuhan University, Wuhan, China, in 2009, 2011, and 2014, respectively.

He is currently an Associate Professor with the School of Computer Science and Information Engineering, Hubei University, Wuhan. His research interests include machine learning, deep learning, and data analytics and energy performance optimization in smart buildings.

Peng Wang received the B.S. and M.S. degrees in software engineering from Hubei University, Wuhan, China, in 2019 and 2022, respectively.

He is currently a Research Assistant with the School of Computer Science and Information Engineering, Hubei University. His research interests include deep learning, object detection, machine learning, and model optimization.

Cathal Hoare received the B.E. degree in computer science from Computer Science Department, University College Cork, Cork, Ireland, in 1998.

He joined as a Senior Energy Systems Researcher with University College Dublin, Dublin, Ireland, in 2016. He has contributed to projects through the development of software frameworks for data access and the semantic integration of disparate database schema.

James O'Donnell received the B.E. degree in civil and environmental engineering and the Ph.D. degree in holistic building performance analysis from University College Cork, Cork, Ireland, in 2001 and 2009, respectively.

He joined University College Dublin, Dublin, Ireland, in June 2013. His current work focuses on the development and deployment of interoperable and BIM based solutions to support multiscale energy modeling, from individual buildings to national building stocks.