Automation in Construction 112 (2020) 103085

Contents lists available at ScienceDirect

Automation in Construction

journal homepage: www.elsevier.com/locate/autcon

Deep learning for site safety: Real-time detection of personal protective equipment

Nipun D. Nath[a], Amir H. Behzadan[b,*], Stephanie G. Paal[a]

[a] Zachry Department of Civil Engineering, Texas A&M University, 3136 TAMU, College Station, TX 77843, USA
[b] Department of Construction Science, Texas A&M University, 3137 TAMU, College Station, TX 77843, USA

ARTICLE INFO

Keywords:
Personal protective equipment (PPE)
Construction safety
Deep learning
Transfer learning
Image dataset
Real-time object detection

ABSTRACT

The leading causes of construction fatalities include traumatic brain injuries (resulted from fall and electrocution) and collisions (resulted from struck by objects). As a preventive step, the U.S. Occupational Safety and Health Administration (OSHA) requires that contractors enforce and monitor appropriate usage of personal protective equipment (PPE) of workers (e.g., hard hat and vest) at all times. This paper presents three deep learning (DL) models built on You-Only-Look-Once (YOLO) architecture to verify PPE compliance of workers; i.e., if a worker is wearing hard hat, vest, or both, from image/video in real-time. In the first approach, the algorithm detects workers, hats, and vests and then, a machine learning model (e.g., neural network and decision tree) verifies if each detected worker is properly wearing hat or vest. In the second approach, the algorithm simultaneously detects individual workers and verifies PPE compliance with a single convolutional neural network (CNN) framework. In the third approach, the algorithm first detects only the workers in the input image which are then cropped and classified by CNN-based classifiers (i.e., VGG-16, ResNet-50, and Xception) according to the presence of PPE attire. All models are trained on an in-house image dataset that is created using crowd-sourcing and web-mining. The dataset, named *Pictor-v3*, contains ~1,500 annotated images and ~4,700 instances of workers wearing various combinations of PPE components. It is found that the second approach achieves the best performance, i.e., 72.3% mean average precision (mAP), in real-world settings, and can process 11 frames per second (FPS) on a laptop computer which makes it suitable for real-time detection, as well as a good candidate for running on light-weight mobile devices. The closest alternative in terms of performance (67.93% mAP) is the third approach where VGG-16, ResNet-50, and Xception classifiers are assembled in a Bayesian framework. However, the first approach is the fastest among all and can process 13 FPS with 63.1% mAP. The crowed-sourced *Pictor-v3* dataset and all trained models are publicly available to support the design and testing of other innovative applications for monitoring safety compliance, and advancing future research in automation in construction.

1. Introduction

With $1.3 trillion annual expenditure (~6.3% of the GDP) [1] and 7.2 million employees (~5% of the total workforce) [2], construction is one of the largest sectors of the U.S. economy. However, the high number of workplace accidents and worker injuries has also made this industry one of the most hazardous among all industries. In 2016–17 alone, the total number of fatal occupational injuries was the highest in construction compared to all other industries in the U.S. According to the Bureau of Labor Statistics (BLS), during this period, 991 fatal incidents (~19% of all the fatalities) were recorded [3]. In addition, in 2017, the number of non-fatal occupational injuries and illnesses in construction jobs was 79,810 (~9% of total cases) which was also exorbitant [3]. The leading causes of construction fatalities include fall, struck by an object, electrocutions, and caught-in/between – together comprising the "fatal four" – which were responsible for nearly 60% of construction worker deaths in 2017 (Fig. 1) [4].

The majority of these injuries/fatalities could be prevented if workers wore appropriate personal protective equipment (PPE), e.g., hard hat, safety vest, gloves, safety goggles, and steel toe shoes [5]. The U.S. Occupational Safety and Health Administration (OSHA) and similar agencies in other countries require that all personnel, working in close proximity of site hazards, wear proper PPE to minimize the risk of being exposed to or injured by hazards [5]. According to a report by the National Institute for Occupational Safety and Health (NIOSH), between 2003 and 2010, a total of 2,210 construction fatalities occurred

* Corresponding author.
E-mail addresses: nipundebnath@tamu.edu (N.D. Nath), abehzadan@tamu.edu (A.H. Behzadan), spaal@tamu.edu (S.G. Paal).

https://doi.org/10.1016/j.autcon.2020.103085
Received 7 July 2019; Received in revised form 6 January 2020; Accepted 8 January 2020

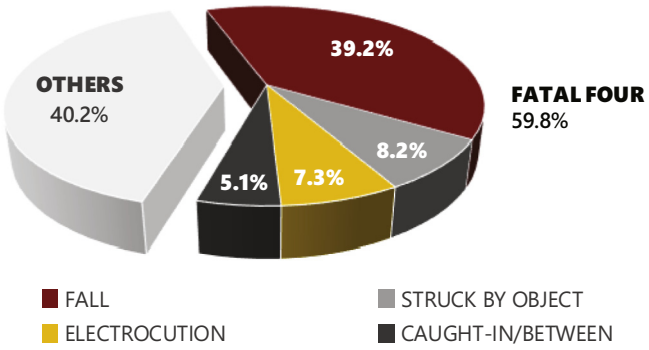

Fig. 1. Percentage of fatal injuries caused by the "fatal four" in construction industry in 2017.

because of traumatic brain injury (TBI) which represented 25% of all construction fatalities during that period [6]. The most common causes of TBIs in construction sites are fall from height, fall of objects on head, and electrocution by overhanging wires [5,6]. OSHA tries to minimize head injuries by mandating workers to wear hard hats when there are overhead objects, or there is a possibility of workers bumping their heads against objects, or having accidental contact with overhanging electrical hazards [5]. Similarly, in many construction sites, struck by objects can occur due to site traffic or heavy equipment (e.g., trucks, bulldozers, graders) operating in close proximity to workers [4]. To ensure high visibility in all lighting and weather conditions, OSHA requires that workers wear safety vests (generally, red or orange in color, and reflective if employee works at night) [7].

While governing laws and safety regulations (OSHA-1926.28a) almost always hold employers responsible for enforcing, monitoring, and maintaining appropriate PPE on the job site [8], employees often do not comply with this regulation due to the lack of safety awareness, discomfort of wearing PPE, and the feeling that PPE interferes with their work [9–11]. According to OSHA, not using proper PPE (e.g., eye and face protection) was one of the most violated regulations in 2017–18 [12]. To incentivize compliance, OSHA fines the employer by imposing a penalty for each employee (up to $13,260) who fails to comply with PPE requirements [13]. For example, in a 12-month period ending in September 2018, OSHA issued 895 citations for not wearing protective helmet (total penalty of $1,916,511) and 78 citations for violating other PPE requirements (total penalty of $309,474) in construction sites [14]. To note, the burden of full compensation falls onto employers and/or employees since insurance companies do not cover any damages caused by improper PPE practice. At the same time, OSHA also encourages employers to train workers about the necessity of proper PPE [4]. However, from a practical perspective, manual monitoring of large number of workers for PPE compliance is expensive, time-consuming, and resource-intensive [15]. Therefore, recent research has looked into the possibility of automating PPE detection. In this paper, the authors explore current trends in PPE detection and introduce deep learning (DL) methods for fast and accurate PPE detection. In particular, three approaches are presented to identify, in real-time, if a construction worker is wearing PPE components (e.g., hard hat and vest). Moreover, the approach to collecting relevant data and preparing the annotations is also discussed.

2. Literature review

Current techniques of automated monitoring of PPE compliance can be broadly categorized into two types: sensor-based and vision-based. Sensor-based techniques comprise of installing a sensor and analyzing its signals. An example includes using Radio Frequency Identification (RFID) tags installed on each PPE component, and checking the tags with a scanner at the entrance of a job site to control if workers are wearing proper PPE [16]. Another example includes using a local area network (LAN) to check RFIDs, installed on PPE components, that continuously monitor PPE compliance while the employees are working [17]. A similar approach is investigated by Naticchia et al. [18] where each PPE item is equipped with short-range transponders, and a wireless system checks the types of PPE each worker is wearing to verify if it complies with the regulations. However, the sensor-based approach requires a significant investment in purchasing, installing, and maintaining complicated sensor networks that might deter its implementation in practice.

On the contrary, vision-based approaches use cameras to record images or videos of the job site, which are then analyzed to verify PPE compliance. This approach provides richer information about the scene that can be utilized to understand complex construction sites more promptly, precisely, and comprehensively [19]. Although many researchers have used depth-based cameras, e.g., Kinect and VICON, to analyze unsafe behaviors of construction workers [20], due to the RGB-D camera's various shortcomings (e.g., limited range of view), regular RGB cameras are more feasible for practical uses [10]. Examples include but are not limited to video-based surveillance methods that use facial features, and motion and color information [21], image-based edge detection [22], and Histogram of Oriented Gradient (HOG)-based method [23], to detect hard hats. Other examples are using HOG-based feature and machine-learning algorithms, e.g., Support Vector Machine (SVM) [24] and k-Nearest Neighbor (kNN) to detect if a person is wearing a safety vest [25].

In recent years, deep learning methods have drawn significant attention in computer vision due to their ability to self-learn useful features from large-scale, annotated training data [26]. Particularly, Convolutional Neural Network (CNN) is being widely used for image classification and object detection. For example, LeCun et al. [27] used CNNs to recognize hand-written digits, and Krizhevsky et al. [28] and Simonyan and Zisserman [29] studied deep CNNs to classify 1.2 million images (ImageNet dataset) into 1000 different classes of various everyday objects and animals. Within the construction domain, Kolar et al. [26] used CNNs to detect safety guardrails. Siddula et al. [30] coupled a Gaussian Mixture Model (GMM) [31] with CNNs to detect objects on roof construction sites. Ding et al. [32] combined a Long Short-Term Memory (LSTM) [33] model with CNNs to recognize workers' potentially unsafe behaviors (e.g., climbing a ladder). More recently, Nath et al. [34,35] used CNNs to identify common construction-related objects (e.g., building, equipment, and workers).

3. State-of-the-art techniques and problem statement

Most existing vision-based methods for monitoring PPE compliance merely focus on identifying hard hats. For example, Fang et al. [10] used Region-based CNNs (R-CNNs) to detect if a worker is not wearing hard hat. Wu et al. [36] proposed a Single Shot Detector (SSD)-based algorithm to detect hard hat, and Mneymneh et al. [15] isolated moving worker (by detecting motion) in videos and identified if any hard hat was located in or around the top area of a worker's detection box. Similarly, Xie et al. [37] used fully convolution-based algorithms to detect workers' hard hats. However, to date, few studies have been directed at identifying multi-class PPE harnessing the power of DL algorithms. Among the scarce examples of multiple PPE detection, a commercially available software, named *smartvid.io*, applies an AI-driven algorithm to detect multiple PPE components (e.g., hard hat, safety vest, gloves, safety goggles, and steel toe shoes) [38]. Nonetheless, methods for "real time" detection [36,37] of PPE are scanter, albeit significantly important for ensuring safety. Previous studies [39] have defined "real-time" as processing at least 5 video frames per second (FPS) although more recent studies [40,41] prefer higher FPS to be considered as real-time. However, in this study, processing speed ≥ 5 FPS is considered as "real-time" and failing that, at least ≥ 1 FPS is considered as "near real-time". At such a fast rate, detection of PPE

(e.g., hard hat, and vest) as a standalone application or in conjunction with other construction-related objects (e.g., building, equipment) enables the identification of more complex and subtle spatiotemporal relationships (including those that may put workers in life-threatening situations) that might not be possible to detect otherwise. For example, an employee, not wearing a hard hat and/or safety vest while working in close proximity to heavy construction equipment (e.g., excavator), is exposed to a high risk of being struck by the equipment. Therefore, to precisely track object movements in a live video feed and predict imminent collisions, it is necessary to have an extremely fast algorithm that can repeatedly process videos frame-by-frame and provide feedback in real-time. Moreover, since a fast algorithm is less computationally expensive, it will significantly reduce the upfront cost for computational resources and can be launched on mobile devices (e.g., smartphones, tablets) and even on light-weight drones [42].

Since safety incidents occur in specific contexts (i.e., when certain spatial or temporal rules are not properly followed), identifying only the workers and PPE components (e.g., hard hat, and vest) in an image or video frame, without recognizing the contextual relationship between them, does not effectively verify PPE compliance or the lack thereof. Even after detecting the presence of individual objects (e.g., worker, hard hat, and vest), verifying whether a worker is properly using the PPE components that comply with safety regulations still remains a question that needs to be answered by analyzing the spatiotemporal relationships between the detected objects considering context. For example, while verifying PPE compliance for hard hat, one needs to know if a worker is actually wearing it on his/her head, just holding it, or simply laying it somewhere. Therefore, the task of monitoring PPE compliance involves two subtasks in itself, namely *detection*, i.e., detecting workers and PPE components, and *verification*, i.e., checking if a worker is properly using PPE components. The majority of previous research in this area has attempted at identifying workers (i.e., detection) first and then checking if a hard hat is present in or around a worker's detection region (i.e., verification) [15,21,22]. For instance, Fang et al. [10] identify candidate regions in the image (that may or may not include a worker) and classify if the region is a "worker without hat". On the other hand, Xie et al. [37] detect workers and hard hats individually but simultaneously and then verify if a worker is or is not wearing hard hat based on the amount of overlap between the worker's head and the detected hard hats. However, the level of complexity for the verification task exponentially increases with an increase in the number and types of PPE components. Particularly, there are 2^n possible combinations of PPE attire for n different types of PPE components. For example, for one type of PPE ($n = 1$), suppose hard hat, there are two ($2^1 = 2$) combinations, i.e., worker not wearing hard hat and worker wearing hard hat. However, for two components ($n = 2$), suppose hard hat and vest, there are four ($2^2 = 4$) combinations, i.e., worker not wearing hard hat and vest (referred to as W), wearing just hard hat (referred to as WH), wearing just vest (referred to as WV), and wearing both hard hat and vest (referred to as WHV). Therefore, a comprehensive model should be able to identify all workers in a video frame, and then determine if individual workers are wearing each (or a combination) of considered PPE components.

Our literature review has revealed that previously proposed methods of verifying PPE compliance either depend on obtrusive and expensive sensor networks or can identify only one type of PPE component (e.g., hard hat), or require significant amount of processing time (i.e., not real-time) to complete the analyses. This research offers a scientifically robust and comprehensive method capable of overcoming these limitations while achieving reliable performance. In particular, this study aims to develop DL-assisted vision-based frameworks to monitor workers' safety compliance, associated with multiple PPE components, in real-time for more feasible, comprehensive, and prompt assessment of workplace safety.

4. Overview of relevant deep learning methods

The building blocks of the developed methodology comprise of several DL-based methods. These methods are briefly discussed in this section, followed by a detailed demonstration of the overall methodology.

4.1. Real-time object detection

In computer vision, the object detection problem is defined as identifying an object (a.k.a., classification) in an image and precisely estimating its location (a.k.a., localization) within the image [43]. One of the most prevalent examples of state-of-the-art object detection algorithms is Region-based CNN (R-CNN) [44]. An R-CNN first identifies several regions of interest (a.k.a., candidate regions), and then uses a CNN to classify those regions to detect objects in them [44,45]. Since the original R-CNN is slow [45], faster variants of it, e.g., Fast R-CNN [45], Faster R-CNN [39], and Mask R-CNN [46] have been proposed. Yet, all of these algorithms fall short of achieving real-time detection of objects from continuous video feeds. At present, the most promising algorithms capable of doing real-time object detection are SSD (Single Shot Detector) [47], YOLO (You-Only-Look-Once) [40], R-FCN (region-based fully convolutional network), and RetinaNet [48]. However, the challenge in achieving real-time computation is to sustain decent accuracy. While most often, fast algorithms significantly compromise accuracy for achieving real-time computation, to date, only YOLO (particularly, YOLO-v3) is faster yet more accurate than other alternatives [49]. For example, previous research [37] has found that YOLO significantly performs better than SSD and R-FCN in detecting hard hat, while still performing the detection at higher frame rate (i.e., faster) than those algorithms.

Unlike R-CNN-based approaches, YOLO reduces the computational burden and, thus, allows much faster detection of objects, by combining the classification and localization tasks into a single CNN framework [40]. Particularly, a variant of YOLO model, YOLO-v3, takes a 416 × 416 RGB image as input and contains three output layers, each dividing the input image into 13 × 13 grids (Output-1), 26 × 26 grids (Output-2), and 52 × 52 grids (Output-3), respectively (Fig. 2). Each of the three output layers is associated with three anchor boxes, resulting in a total of nine anchor boxes. Output-1 layer is associated with the three largest anchor boxes (to detect large objects), Output-2 layer is associated with the next three larger anchor boxes (to detect medium-sized objects), and Output-3 layer is associated with the three smallest anchor boxes (to detect small objects). In the training phase, each grid cell in the output layers takes corresponding anchor boxes and learns how to shift and/or scale these anchor boxes so that the modified boxes (a.k.a., bounding boxes) perfectly fit the objects of interest, as illustrated in Fig. 2. As shown in this Figure, each predicted bounding box is associated with a $(N + 5)$-dimensional vector $[\, t_x \;\; t_y \;\; t_w \;\; t_h \;\; p_0 \;\; p_1 \;\cdots\; p_N]^{\mathrm{T}}$. In this vector, values t_x, t_y, t_w, and t_h represent x- and y-coordinates of the center, and the width and height of the box, respectively. The value p_0 (a.k.a., objectness score) is the probability of an object located inside the bounding box. The remaining values are N conditional probabilities, $P(C_i|\, object)$, each indicating that given an object presents inside the box what is the probability that it belongs to class C_i, where $i = 1, \ldots, N$. In total, YOLO-v3 outputs $(13 \times 13 \times 3) + (26 \times 26 \times 3) + (52 \times 52 \times 3) = 10{,}647$ boxes for a single image. However, intuitively, most of these output boxes are either false positive or represent the same object in the input image. Therefore, to discard redundancy and duplication in the inference phase, YOLO uses the non-maximum suppression (NMS) technique [40,41,44] to eliminate boxes with lower confidence levels but higher percentage of overlapping, thus preserving a single bounding box for a single corresponding object.

A common metric to measure the performance of object detection algorithm is intersection over union (a.k.a. IoU). As shown in Fig. 3, IoU

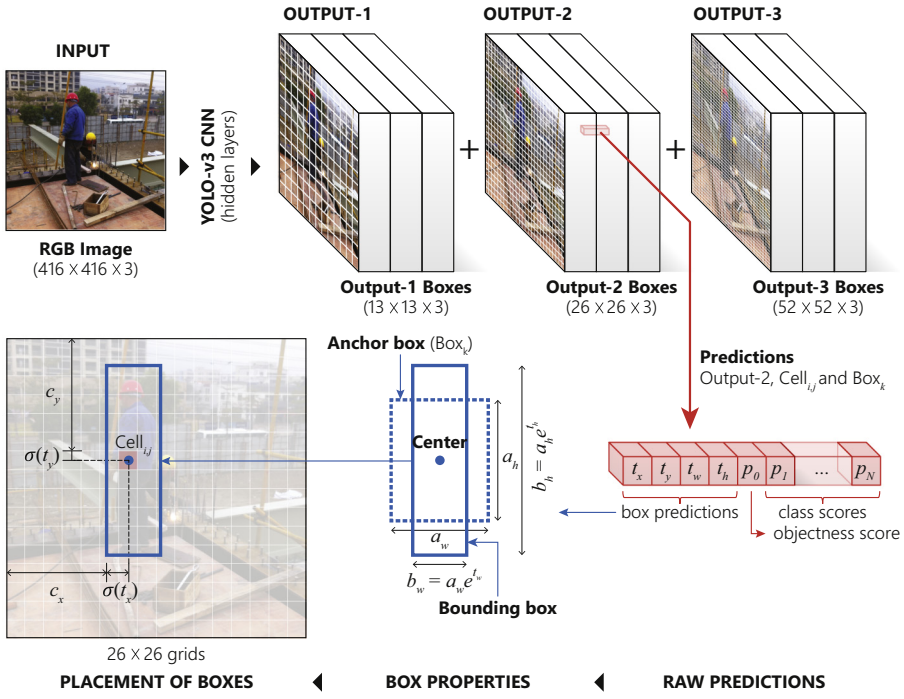

Fig. 2. Schematic diagram of the YOLO-v3 algorithm.

represents the percentage of overlap between two boxes, e.g., the ground-truth box (G) and the predicted box (P), and is calculated using Eq. (1) [37].

$$IoU = \frac{intersection}{union} = \frac{G \cap P}{G \cup P} \tag{1}$$

4.2. Transfer learning and image dataset

A CNN model can be trained from scratch on a particular dataset. However, achieving optimal results in this approach requires large training data along with hyperparameter tuning which can take a substantial processing time [26]. To overcome these challenges and achieve significantly better and consistent performance, transfer learning is performed [50,51]. Transfer learning is a process where a DL model is pre-trained on a different, but related dataset (a.k.a., source dataset) which is generally larger in size. Next, the model is re-trained on the desired dataset (a.k.a., target dataset) which could be smaller in size. Through this process, the model learns high- and mid-level features (e.g., edges, shapes, colors) from the source dataset to the extent that is relevant and useful for the classes in the target dataset [50].

Publicly available large-scale datasets generally contain annotated images of common daily objects. Examples include but are not limited to ImageNet [52] dataset containing 3.2 million images of 5,247 categories of objects, PASCAL (pattern analysis, statistical modelling and computational learning) VOC (visual object class) 2012 dataset [53] containing 20 categories of objects in ~21,000 images, and Microsoft's COCO (common objects in context) [54] dataset containing 328,000 images with 2.5 million instances of 91 different types of objects. In this study, YOLO-v3 models are pre-trained on the COCO dataset [54] since it contains classes (e.g., person) and contexts (e.g., outdoor locations) that are visually relevant to the target application (e.g., workers in a construction site).

5. Methodology

This section presents generic frameworks for verifying safety

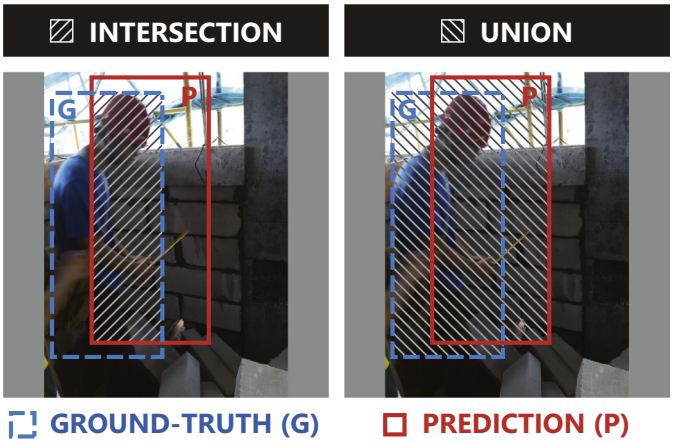

Fig. 3. Calculation of intersection over union (IoU) between ground-truth and predicted boxes.

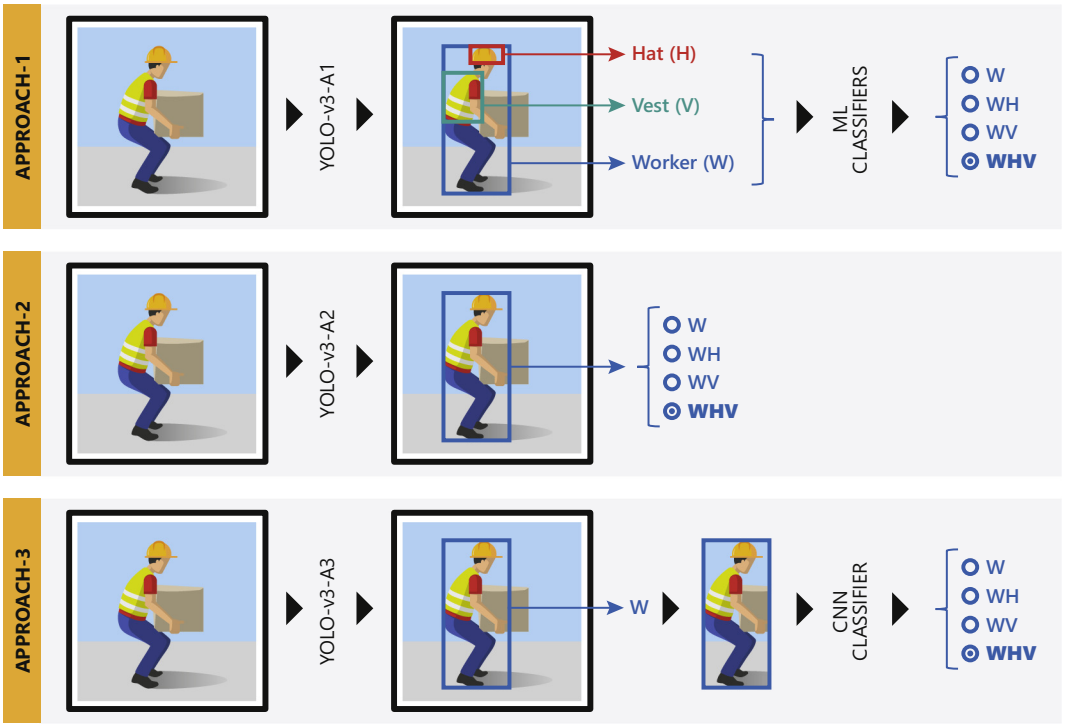

Fig. 4. Three different approaches.

compliance for multiple PPE from visual data (image or video). In particular, three different DL-based approaches are proposed to perform the verification in real-time. Although the proposed techniques are designed to work for any number (≥ 1) and type of PPE (e.g., hard hat, safety vest, gloves, safety goggles, and steel toe shoes), for the scope of this study, the technical discussions, data analysis, and validation are particularly conducted for two types of PPE, i.e., hard hat (referred to as *hat*) and safety vest (referred to as *vest*).

5.1. Overview of the three different approaches

Fig. 4 illustrates a schematic diagram of the three proposed approaches for automatically verifying PPE compliance of workers from visual data. In Approach-1, a YOLO-v3-based model is trained and tested to detect worker (W) and different PPE types, e.g., hat (H), and vest (V), individually. Next, a verification algorithm is applied to check if each worker is wearing the PPE properly. This algorithm, particularly, checks all the workers and classifies each worker into one of the four classes: worker wearing no hat and vest (W), worker wearing only hat (WH), worker wearing only vest (WV), and worker wearing both hat and vest (WHV). In contrast, in Approach-2, a YOLO-v3-based model is designed to localize workers and directly classify them based on their PPE attire, i.e., W, WH, WV, and WHV. Finally, Approach-3 first uses a YOLO-v3-based model to detect only workers (regardless of the PPE attire), then crops parts of the image that contain workers, and finally, applies a CNN-based classifier to each cropped image to classify it into one of the possible classes, i.e., W, WH, WV, and WHV.

5.2. Common steps in all three approaches

The common denominator of all three approaches is that they use adopted YOLO-v3 models to detect targeted object classes. Therefore, the preliminary steps, e.g., preparation of the data and formation of the YOLO-v3 models follow similar procedures in all three approaches. Considering the generality in these procedures, the common steps in all three approaches are discussed in this subsection and more technical details on each approach are discussed individually in the following subsections.

5.2.1. Preparation of the dataset and the pre-trained models

In general, a YOLO-v3 model takes 416 × 416 images as input. Therefore, all images are initially resized to a size of 416 × 416 using bi-cubic interpolation [55]. The original aspect ratio is maintained during resizing by padding each image equally on both sides along the shorter dimension. Next, the entire dataset is randomly split into training (64%), validation (16%), and testing (20%) subsets. Then, k-means clustering [41] is performed on all the rectangular boxes in the training dataset to obtain nine anchor boxes for the models. All YOLO-v3 models are subsequently pre-trained on the COCO dataset [54].

5.2.2. Architecture of the YOLO-v3 models

In all three approaches, the architecture of the pre-trained YOLO-v3 models (particularly, the last three output layers) are modified based on the number of considered classes in each approach. For example, in Approach-1, the YOLO-v3 model detects worker and n number of different PPE types, a total of $n + 1$ classes, individually. Therefore, each bounding box in each grid cell of the output layers of the YOLO-v3 architecture (Fig. 2) should be a $(n + 6)$-dimensional vector (Section 4.1). For example, since there are 3 bounding boxes in each 13 × 13 grid of Output-1 layer, the dimension of the predicted tensor should be $13 \times 13 \times 3(n + 6)$ (Table 1). Similarly, for Output-2 and Output-3 layers, the dimensions of the predicted tensors should be $26 \times 26 \times 3(n + 6)$ and $52 \times 52 \times 3(n + 6)$, respectively (Table 1). However, since the dimensions of output layers and the number and types of classes in the pre-trained YOLO-v3 models may not match with the problem at hand, all three output layers are replaced with new layers with required dimensions, and the weights in the newly added layers are randomly initialized. The modified model is referred to as YOLO-v3-A1.

In Approach-2, the YOLO-v3 model directly detects worker's 2^n different combinations of PPE attire for n different PPE types. Therefore, the dimensions of the output layers are $13 \times 13 \times 3(2^n + 5)$, $26 \times 26 \times 3(2^n + 5)$, and $52 \times 52 \times 3(2^n + 5)$, respectively (Table 1). With the same token as in

Table 1
Dimensions of the output layers in modified YOLO-v3 models in three approaches.

PPE types	Layers	Approach-1	Approach-2	Approach-3
n types of PPE	Output-1	$13 \times 13 \times 3(n + 6)$	$13 \times 13 \times 3(2^n + 5)$	$13 \times 13 \times (3 \times 6)$
	Output-2	$26 \times 26 \times 3(n + 6)$	$26 \times 26 \times 3(2^n + 5)$	$26 \times 26 \times (3 \times 6)$
	Output-3	$52 \times 52 \times 3(n + 6)$	$52 \times 52 \times 3(2^n + 5)$	$52 \times 52 \times (3 \times 6)$
hat and vest ($n = 2$)	Output-1	$13 \times 13 \times 24$	$13 \times 13 \times 27$	$13 \times 13 \times 18$
	Output-2	$26 \times 26 \times 24$	$26 \times 26 \times 27$	$26 \times 26 \times 18$
	Output-3	$52 \times 52 \times 24$	$52 \times 52 \times 27$	$52 \times 52 \times 18$

Approach-1, the output layers of the YOLO-v3 model for Approach-2 are modified and referred to as YOLO-v3-A2 hereafter. In contrast, regardless of the number of considered PPE types, in Approach-3, the YOLO model detects only workers (i.e., one class) in the input image or video frame. Therefore, the dimensions of the output layers in the modified YOLO-v3 model for this approach (referred to as YOLO-v3-A3) are $13 \times 13 \times (3 \times 6)$, $26 \times 26 \times (3 \times 6)$, and $52 \times 52 \times (3 \times 6)$, respectively (Table 1).

5.2.3. Training of the YOLO-v3 models

In YOLO-v3-A1, -A2, and -A3 models, all layers, except the last three output layers, contain pre-determined weights obtained by pre-training the models on COCO dataset. These pre-trained weights can extract useful features (e.g., colors, edges) that can effectively distinguish the classes (e.g., person, dog, car, apple, laptop, clock) present in the COCO dataset. However, in the newly added output layers (listed in Table 1), weights are initialized with random values and updated through re-training the models with the Pictor-v3 dataset for classifying the target classes. During this re-training process, the weights in all other layers are kept frozen (i.e., unchanged). The idea is to familiarize the models with the new target classes and allow them to learn how to use the pre-learned features to distinguish the target classes. The re-training process involves training for 25 epochs with a learning rate of 10^{-3} using Adam [56] optimizer.

Next, the entire model is fine-tuned by updating the weights in all layers, however, with a slower learning rate. This allows the models to slightly modify the pre-learned features to find more effective features that work better for detecting the target classes. To prevent the model from overfitting, fine-tuning is performed by continuously monitoring the validation loss after each epoch and adjusting the learning rate accordingly. Particularly, the following criteria are maintained in this step: fine-tuning of the model is started with an initial learning rate of 10^{-4}; the learning rate is reduced by half if the validation loss does not decrease for three consecutive epochs; training is terminated if the validation loss does not decrease for 10 consecutive epochs.

5.2.4. Data augmentation

During the re-training and fine-tuning phases, real-time data augmentation is performed. In particular, at the beginning of each epoch, training images and annotations are randomly distorted to obtain more diverse data that would help prevent overfitting [57]. To do this, the geometric shape of each training image is randomly changed by scaling up or down by ± 30%, translating horizontally and vertically by ± 30%, and horizontally flipping in randomly chosen 50% of the times. Additionally, the color space, i.e., hue, saturation, and value (brightness), of the training image are also randomly (with uniform probability) changed in the range of [-10%, +10%], [-33%, +50%], and [-33%, +50%], respectively.

5.3. Overall framework of each approach

5.3.1. Approach-1

In Aprroach-1, the dataset and the YOLO-v3-A1 model are prepared following the procedures described in Section 5.2. The trained YOLO-v3-A1 model outputs individual boxes for worker and PPE components in the image. After obtaining the predicted boxes, the next task is to verify PPE compliance for each worker. One potential way could be applying an algorithm with explicit rules that are determined through empirical observations. For example, if a hat detection box is located at the upper portion of the worker detection box, and the two boxes have an overlapping area larger than a certain threshold, then the worker is marked as probably wearing a hat. However, manually establishing these rules for multiple PPEs is a complicated task and results may not be structured or comprehensive enough to consider all possible scenarios. Therefore, in this paper, traditional machine learning (ML) algorithms (e.g., neural network and decision tree) are used to implicitly learn such rules by harnessing the power of big data.

In particular, the authors suggest training n different ML classifiers, each for n different PPE types, since the rules for different types of PPE might be considerably different. However, for all classifiers, we propose to prepare the inputs according to the following steps. First, as shown in Fig. 5, each bounding box C_i for class $C \in \{W, H, V\}$ is represented by two positional vectors: $\overrightarrow{C_{i,0}} = [x_0^{C_i} \quad y_0^{C_i}]^T$ pointing to the top-left corner $(x_0^{C_i}, y_0^{C_i})$ of the box, and $\overrightarrow{C_{i,1}} = [x_1^{C_i} \quad y_1^{C_i}]^T$ pointing to the bottom-right corner $(x_1^{C_i}, y_1^{C_i})$ of the box. Here, i is the index for detected boxes of a class. Next, when considering if worker W_i is wearing any particular type of PPE, $P \in \{H, V\}$, first, the width and height of the box W_i (denoted by w^{W_i} and h^{W_i}, respectively) are calculated using Eq. (2). Then, all P_j boxes of that PPE type are normalized using Eq. (3). As shown in Fig. 5, this equation performs a coordinate transformation, i.e., a translation which sets the top corner of W_i as the origin of the transformed coordinate system, followed by a scaling that scales all the boxes such that the width and height of the transformed W_i box become unity.

$$\begin{bmatrix} w^{W_i} \\ h^{W_i} \end{bmatrix} = \begin{bmatrix} x_1^{W_i} \\ y_1^{W_i} \end{bmatrix} - \begin{bmatrix} x_0^{W_i} \\ y_0^{W_i} \end{bmatrix} = \overrightarrow{W_{i,1}} - \overrightarrow{W_{i,0}} \tag{2}$$

$$\begin{bmatrix} \tilde{x}_0^{P_j,W_i} & \tilde{x}_1^{P_j,W_i} \\ \tilde{y}_0^{P_j,W_i} & \tilde{y}_1^{P_j,W_i} \end{bmatrix} = \begin{bmatrix} \frac{1}{w^{W_i}} & 0 \\ 0 & \frac{1}{h^{W_i}} \end{bmatrix} \left(\begin{bmatrix} x_0^{P_j} & x_1^{P_j} \\ y_0^{P_j} & y_1^{P_j} \end{bmatrix} - \begin{bmatrix} x_0^{W_i} & x_0^{W_i} \\ y_0^{W_i} & y_0^{W_i} \end{bmatrix} \right) \tag{3}$$

The input of the ML classifiers is the four elements of the transformed matrix, shown on the LHS of Eq. (3), which can be also expressed as $\begin{bmatrix} \overrightarrow{\tilde{P}}_{j,0}^{W_i} & \overrightarrow{\tilde{P}}_{j,1}^{W_i} \end{bmatrix}$, and the output is a binary decision, i.e., if a worker W_i is wearing that P_j or not. An important point to consider is that in the testing phase, the W_i and P_j boxes are the outputs of YOLO-v3-A1 model that may contain inherent noise, i.e., slightly differ from the ground-truth boxes. Therefore, in the training phase, the ML classifiers should be trained with ground-truth W_i and P_j boxes, however, with added noise. In this research, it has been empirically found that for the training set of Pictor-v3 dataset, and the trained YOLO models, the noise in the elements of transformed matrix resembles a standard normal distribution with a standard deviation of 0.08, i.e., $N(0, 0.0064)$.

In this study, two types of ML classifier algorithms are investigated, namely neural network (NN) and decision tree (DT). The reason behind using NN is that it can learn a complex function to map input features to output decisions. The architecture of the NN model consists of one input

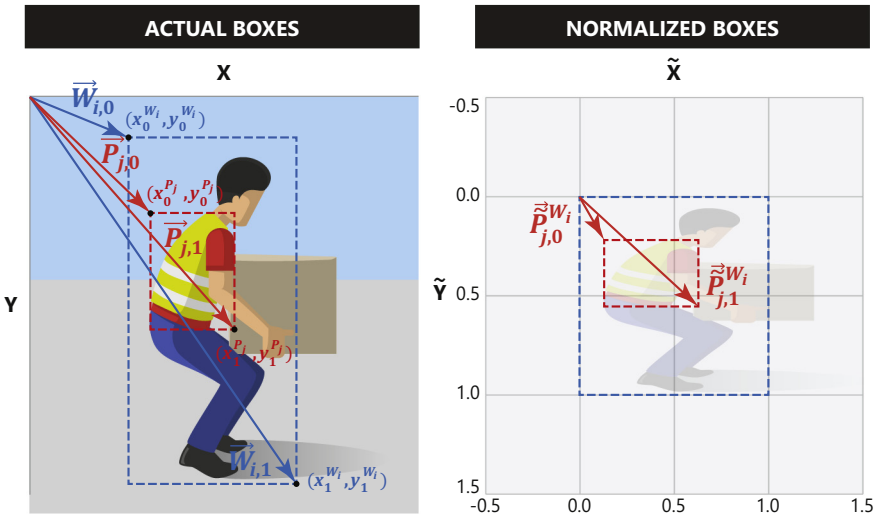

Fig. 5. Normalization of the worker and PPE boxes by vector transformation.

layer containing four nodes (since the number of features is four), two hidden layers each containing eight nodes, and one output layer containing one node (since the output is a binary decision). The model is trained for 50 epochs with Adam optimizer. However, often the function learned by NN is not human interpretable. In contrast, rules learned by DT are easy to interpret and apply by human to reach the decision by examining the input features [58]. Therefore, a DT model is trained with the Gini index [58] to be the criteria for splitting a decision branch, and by restricting the depth of the tree to be not > 4.

Suppose, an ML model M_P is trained to verify if a worker is wearing a particular type of PPE $P \in \{H, V, \ldots\}$. Then, the overall structure of the verification algorithm is as follows,

```
For each detected worker W_i:
    Let S be the set of PPE types that worker W_i is wearing.
    Set S = ∅.
    For each type of PPE P ∈ {H, V, ...}:
        For each detected box P_j of type P:
            Transform P⃗^{W_i}_{j,m} vectors to P̃^{W_i}_{j,m} (Equation 3) for m = 0, 1
            Apply model M_P to verify if worker W_i is wearing P_j.
            If worker W_i is wearing P_j:
                Insert P in the set S, i.e., P ∈ S.
                Break and go to the next type of PPE.
    Return S.
```

5.3.2. Approach-2

It must be noted that there is an important distinction between the object classes in Approach-2 and the object classes for which the original YOLO-v3 is designed. For example, in the original YOLO-v3 model, multiple classes may coexist inside a particular grid cell of an image. As shown in Fig. 6, this coexistence may arise from different sizes and shapes of objects (e.g., table, ball, and pencil) or from the semantic definition of the classes (e.g., person and female). Therefore, the original YOLO-v3 implementation considers each class individually and calculates class probabilities by applying the Sigmoid activation function [59] in the last output layers, as expressed in Eq. (4). This function treats each class independently and normalizes the predicted probability z_i for each class i to the range [0,1]. A normalized value $\sigma_{\text{sigmoid}}(z_i)$ (i.e., probability) greater than a certain threshold (e.g., 0.5) indicates the presence of a particular class in a grid cell.

$$\sigma_{\text{sigmoid}}(z_i) = \frac{e^{z_i}}{e^{z_i} + 1} \tag{4}$$

However, as shown in Fig. 6, when applying Approach-2 to the particular problem of multiple PPE detection (scope of this paper), the following observations are made. First, since the size and shape of the workers are very similar, there is a considerably fewer number of grids that may contain multiple classes, especially when the grid sizes are very small. Second, given the semantic definition of the classes (i.e., W, WH, WV, WHV), one worker can belong to only one class. For instance, if a worker is detected as WHV, this detection supersedes any WV, WH, or W detections of the same worker.

In light of this fundamental implementation difference, to calculate class probabilities in YOLO-v3-A2, the SoftMax activation function [60] is used, as expressed in Eq. (5). This function assumes that all classes are disjoint and, therefore, normalizes the class probabilities altogether such that their sum equals 1.

$$\sigma_{\text{SoftMax}}(\mathbf{z})_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}} \tag{5}$$

Furthermore, in the original YOLO-v3 algorithm, non-max-suppression is applied by considering each class individually. This means that for each object class, only the detection box with the highest confidence level is kept and all other overlapping detection boxes (of the same class) are considered redundant and therefore, removed. However, since classes in Approach-2 are disjoint, we propose to use non-max-suppression across all classes after applying the regular non-max-suppression, to remove duplications where the same worker is detected as multiple classes. After performing brute force search, it is found in this research that the best result can be achieved by removing all boxes (regardless of the class) with confidence scores lower than 80% if they have significant overlap (IoU > 90%) with another box with a higher confidence level. Besides these modifications, the re-training and fine-tuning of the YOLO-v3-A2 model, and the data augmentation during training are similar to the protocols used in Approach-1 (YOLO-v3-A1 model).

5.3.3. Approach-3

After modifying the model architecture, the remaining steps for training the YOLO-v3-A3 model, i.e., retraining and fine-tuning, and data augmentation during training, are similar to the protocols used in Approach-1 (YOLO-v3-A1). However, Approach-3 involves additional training of the classifier models, e.g., VGG-16 [29], ResNet-50 [61], and Xception [62]. In the testing phase, the CNN models process only the segments of the actual image that contain workers. However, the bounding boxes of workers are determined by YOLO-v3-A3 and, thus, the geometric shape of the boxes may slightly differ from the ground-

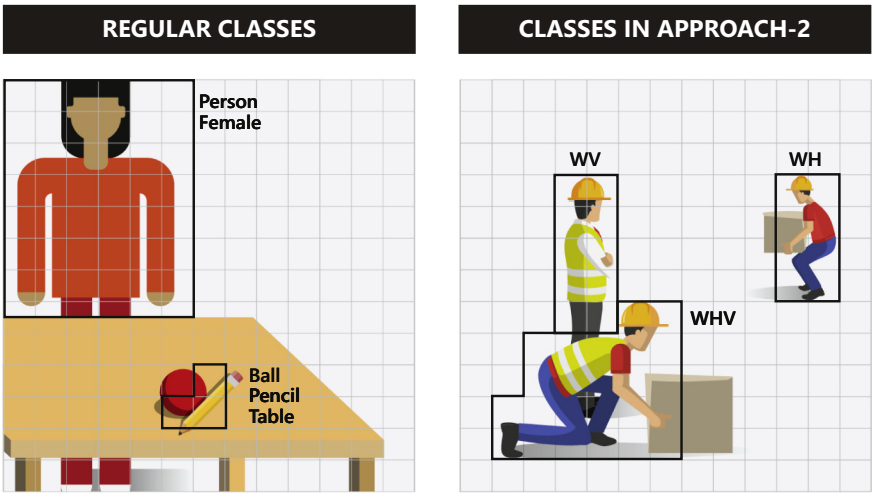

Fig. 6. Classes in regular object detection (original YOLO-v3 implementation) vs. classes in Approach-2 (multiple PPE detection).

truth boxes. Therefore, for the reasons described in Section 5.3.1, random noises are added to ground-truth boxes similar to Approach-1 while cropping the images of workers. Next, the cropped images are resized to square-sized images (e.g., 150 × 150 for VGG-16, 224 × 224 for ResNet-50, and 229 × 229 for Xception) using the method described in Section 5.2.1. Next, from each pre-trained classifier (VGG-16, ResNet-50, and Xception), only the convolutional layers (referred to as the base model) are taken. Additionally, two fully connected layers, containing 64 nodes and 16 nodes, respectively, and each followed by a Rectified Linear Unit (ReLU) activation function, is appended to the base model. For each classifier model, the output layer contains 4 nodes (each for one of the four classes: W, WH, WV, and WHV) and is followed by a SoftMax (Eq. (5)) activation function (because of the similar arguments posed in Section 5.3.2). Each model is pre-trained on the ImageNet dataset and, then, only the last two layers are re-trained and, finally, the entire model is fine-tuned using Adam [56] optimizer.

Additionally, Bayesian inference models are investigated where the prior and posterior probabilities are calculated from the predictions of CNN classifiers (e.g., VGG-16, ResNet-50, and Xception) on trained data, and the posterior probability is used to classify the test images. Particularly, given a set of n classes $C = \{c_1, \ldots c_n\}$ and CNN models M_1, …, M_m, a sample that is classified to class $c_i^{M_j} \in C$ by the CNN model M_j, the posterior probability $P(c_k^B | c_i^{M_1}, \ldots, c_j^{M_m})$ represents the probability that the sample belongs to class $c_k^B \in C$. The posterior probabilities for all combination of c_k^B, $c_i^{M_1}$, …, $c_j^{M_m} \in C$ are calculated from the training dataset using the Bayesian formula shown in Eq. (6).

$$P(c_k^B \mid c_i^{M_1},\ldots,c_j^{M_m}) = \frac{P(c_k^B, c_i^{M_1},\ldots, c_j^{M_m})}{P(c_i^{M_1},\ldots, c_j^{M_m})} \tag{6}$$

During the inference phase, a test sample is classified by each CNN model as $c_i^{M_j}$ with probability $p^{M_j}(c_i)$. The final class (c^B) is determined by checking the posterior probabilities for all classes $c_k^B \in C$ and choosing the class (c^B) that yields the maximum probability, as shown in Eq. (7). Also, the final probability is the maximum of the probabilities of the final class, i.e., $\max_j p^{M_j}(c^B)$.

$$c^B = \underset{c_k^B \in C}{\operatorname{argmax}} P(c_k^B \mid c_i^{M_1},\ldots,c_j^{M_m}) \tag{7}$$

In this approach, the final confidence score of a detection is determined by the product of the confident score provided by the YOLO-v3-A3 model and the probability yielded by the classifier model.

5.4. Quantifying the performance of the YOLO-v3 models

The performance of each model is evaluated considering the mean average precision (mAP) metric, commonly used in object detection and information retrieval domains. Using mAP, the usefulness of the entire system can be presented by a single numerical value [39,63]. To calculate mAP, first, the IoU (Eq. (1)) is calculated. Next, all detections are sorted in descending order of their corresponding confidence level. Then, starting from the detection with the highest confidence level, precision, and recall for a particular class are calculated at each position. To calculate precision (Eq. (8)) and recall (Eq. (9)) [37], TP, FP, and FN are determined based on IoU (Fig. 7). TP, FP, and FN refer to true positive (correctly classified to the class), false positive (incorrectly classified to the class), and false negative (incorrectly classified to another class), respectively.

$$Precision = \frac{TP}{TP + FP} \tag{8}$$

$$Recall = \frac{TP}{TP + FN} \tag{9}$$

For each class, the average precision (AP) is calculated using Eq. (10) [37] where n is the total number of detections, i is the rank of a particular detection in the list of sorted detections, $p(i)$ is the precision of the sub-list ranged from the first to ith detection, and $\Delta r(i)$ is the change in recall from $(i-1)$th to ith detection. Finally, mAP is estimated by calculating the mean of APs of all possible classes.

$$AP = \sum_{i=1}^{n} p(i)\Delta r(i) \tag{10}$$

6. Dataset description

This study expands the annotated, in-house image dataset, *Pictor-v2* (containing buildings, equipment, and workers) by including two new classes: hat and vest. The new dataset is named *Pictor-v3*. In this paper, only the relevant classes (i.e., worker, hat, and vest) from this dataset are used. Images in this dataset are collected through crowd-sourcing [64] and web-mining [65]. Crowd-sourced images are obtained from three different construction projects, while web-mined images are retrieved from publicly available image search engines (i.e., Google) using keyword search [66]. To achieve the best annotation results, a human annotator initially used a web-based annotation toolbox, LabelBox [67], to annotate all images. Next, similar to VOC's annotation protocol [53], all annotations were reviewed by a second annotator in LabelMe [68], an offline annotation toolbox, to ensure completeness and correctness of the labels. Using two independent human annotators also helped minimize the subjective bias in annotation. It must be noted that the choice of toolbox (LabelBox and LabelMe) was based on

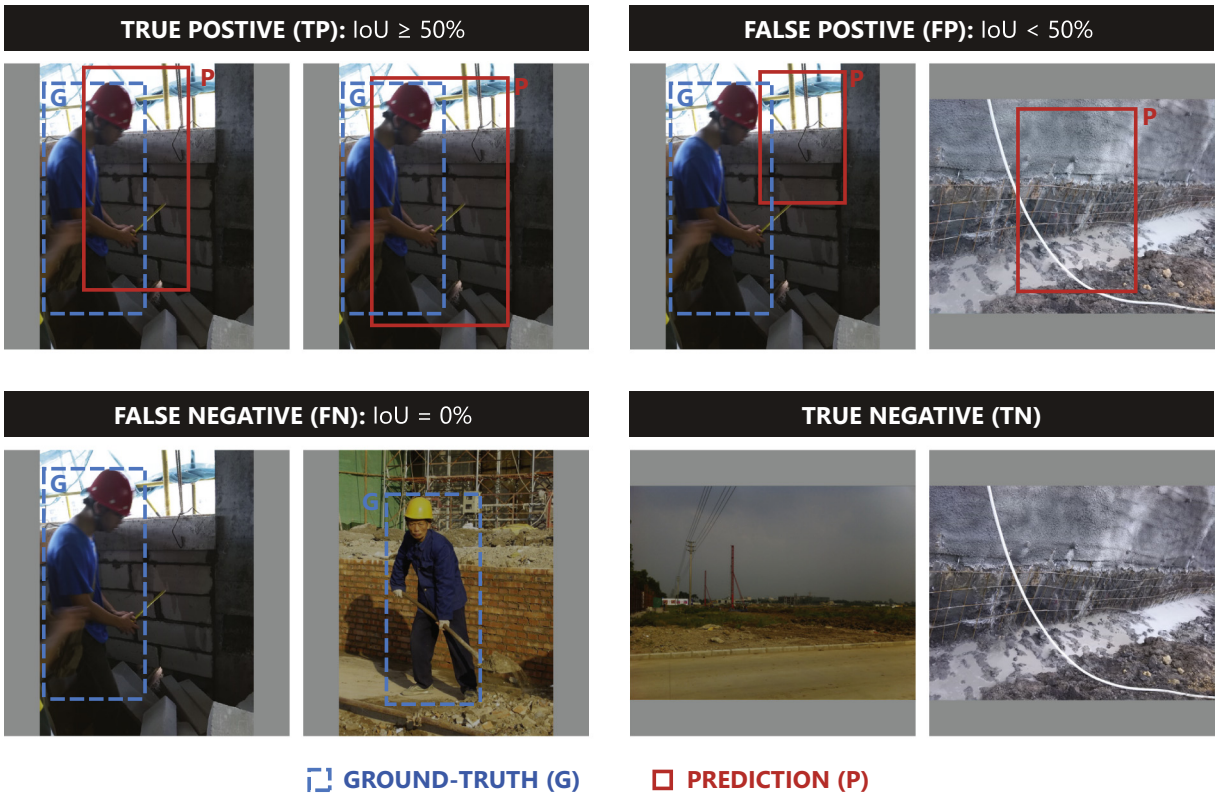

Fig. 7. Examples of true positive (TP), false positive (FP), false negative (FN), and true negative (TN).

annotators' personal preferences and had no effect on the annotation outcome.

The number of images and the number of instances per class label (worker, hat, and vest) in the crowd-sourced and web-mined subsets of the *Pictor-v3* dataset are shown in Fig. 8. As this Figure shows, there are 774 crowd-sourced images containing workers in *Pictor-v3* dataset. However, among them, 240 images contain only worker (W), 517 images contain worker and hat (W + H), and 17 images contain worker, hat, and vest (W + H + V). There is no image containing worker and vest (W + V) in the crowd-sourced subset of *Pictor-v3* dataset. Evidently, a single image can contain multiple worker instances. As shown in Fig. 8, there are a total of 2,496 instances of worker in the crowd-sourced subset of *Pictor-v3* dataset. However, 873 workers do not wear any hat or vest (W), 1,583 workers wear hat (WH), 40 workers wear both hat and vest (WHV), and no worker wears only vest without hat (WV) in the crowd-sourced images.

Examples of annotations for the three detection approaches (Section 5.1) in the *Pictor-v3* dataset are shown in Fig. 9. As explained earlier, this Figure shows that in Approach-1, workers, hats, and vests are annotated individually. However, in Approach-2, only workers are annotated with boxes but assigned to four different classes (W, WH, WV, and WHV) based on their PPE attire. On the other hand, in Approach-3, each worker's portion is cropped into a separate image, and the entire cropped image is labeled as one of the four classes (W, WH, WV, and WHV), similar to Approach-2.

The number of images and number of instances of hat, vest, worker, W, WH, WV, and WHV in the randomly-split training, validation, and testing subsets of the *Pictor-v3* dataset are shown in Fig. 10. For example, there are 944 training images, 240 validation images, and 288 test images. In the training images, there are 2,115 instances of hat, 916 instances of vest, and 3,109 instances of worker. Moreover, among all the workers in the training images, 778 of them are W (wearing no hat or vest), 1,436 are WH (wearing only hat), 248 are WV (wearing only vest), and the rest (647) are WHV (wearing both hat and vest).

7. Results

7.1. Clustering

Since YOLO-v3 requires nine anchor boxes, for each approach, all boxes in the training subset (regardless of class labels) of the *Pictor-v3* dataset are clustered into nine groups using k-means clustering ($k = 9$) [41]. For example, in Fig. 11, points represent geometric shapes of the training boxes (X- and Y-axis are the width and height in pixels, respectively) and colors represent different clusters. Next, a representative (centroid) from each group is selected as the anchor box, and anchor boxes (Fig. 11) are assigned to the output layers according to their sizes, as mentioned in Section 4.1.

It is worth noting that the number and geometric shapes of the bounding boxes for Approach-2 and Approach-3 are identical; both containing boxes for workers. However, in Approach-2, boxes are labeled as either W, WH, WV, or WHV, while in Approach-3 all boxes are labeled only as worker. Therefore, the same clusters and representative anchor boxes are used in Approach-2 and Approach-3. Moreover, the set of bounding boxes in Approach-2 and Approach-3 is a subset of the bounding boxes in Approach-1. That is, the boxes of workers in Approach-2 and Approach-3 are also present in Approach-1. Moreover, Approach-1 also considers boxes for hat and vest, which are relatively smaller in size compared to worker boxes. Therefore, compared to Approach-2 and Approach-3, in Approach-1 there are more boxes but the sizes are smaller. Consequently, the sizes of the anchor boxes in Approach-1 are slightly smaller (Fig. 11). Furthermore, anchor boxes in Approach-2 and Approach-3 are slenderer indicating the presence of slender objects (i.e., standing workers) in the dataset. On the other hand, wider anchor boxes in Approach-1 mostly represent the presence of hats.

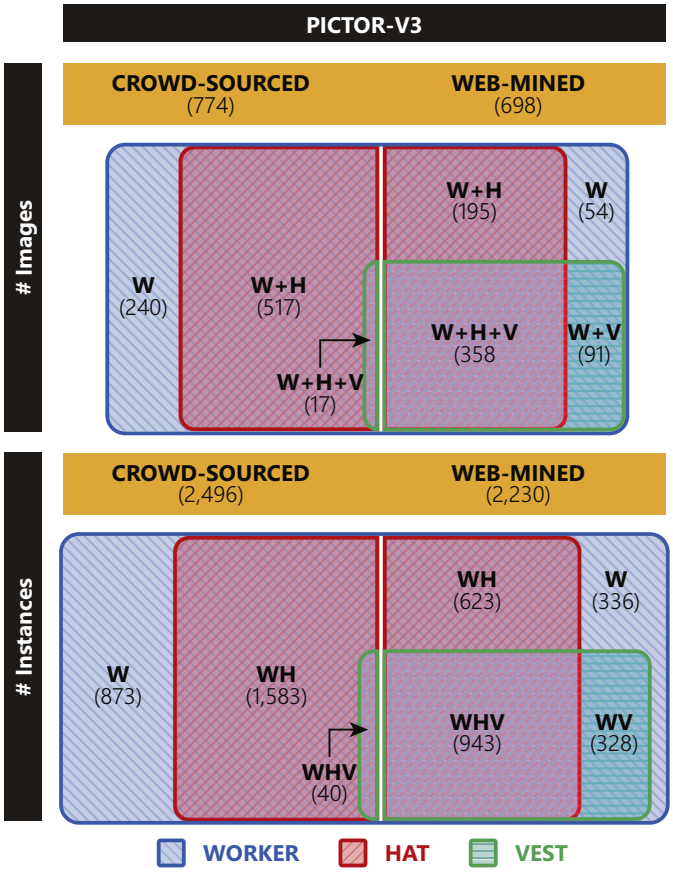

Fig. 8. Number of images and number of instances per class label in the crowd-sourced and web-mined subsets of *Pictor-v3* dataset.

7.2. Training of the YOLO-v3 models

All YOLO-v3 models are trained following the method described in Section 5.2.3. During training, data augmentation is performed, as laid out in Section 5.2.4. Examples of actual data and generated data through random data augmentation are shown in Fig. 12.

NN-based DL models are often criticized for being "black box" since the underlying processes and criteria for feature creation and the complex non-linear functions learned by the model to make decisions (e.g., feature selection, transforming the feature space) are rather difficult to explain [69]. However, researchers generally suggest visualizing the features (particularly, in the upstream layers) to get an idea of what features (e.g., edges, colors) the model finds important [70]. Although a thorough investigation of this topic is not within the scope of this paper, to facilitate the discussion, a visual example of an input image and extracted features by YOLO-v3-A1 model in its first convolutional layer is presented in Fig. 13. This Figure shows that features A8 (row A, column 8), C7, and D1 highlight red- and yellow-colored portions of the input image. These colors are generally associated with hard hat, and therefore, highlighted regions could potentially contain *hat* objects. Similarly, features B2, C8, and D7 focus on the upper-body of the workers, which are potential regions of *vest* objects. Furthermore, features B3 and B4 outline the silhouette of the workers. Continuing from there, in the next layers, the model refines these features, adds extra new features, and/or eliminates irrelevant features so that it can ultimately conclude where and what types of objects are present in the image.

7.3. Performance of the YOLO-v3 models

The performance of the implemented YOLO-v3 models in all three approaches is illustrated in Fig. 14, which shows that Approach-1 achieves 81.2% mAP, with the AP of the worker class (W) reaching 85% (highest among all three classes). This can be attributed to the use of transfer learning where the model "remembers" the learned features of a similar class (i.e., person) presented in the pre-trained dataset (COCO), and effectively "transfers" the learning to the target class, i.e., worker. This observation is consistent with a previous study [50] that concluded that the presence of overlapping classes between the source and target datasets improves the transfer task. In contrast to the worker class, the AP of the class hat (H) is the lowest among all classes potentially because of its small size.

The mAP of the YOLO-v3 model is the lowest in Approach-2 (72.3%). This result is not surprising since the target classes (i.e., W, WH, WV, and WHV) are visually very similar (i.e., all contain humans). Moreover, target classes can only be differentiated by subtle attributes (e.g., based on the presence of hat and/or vest) that are not directly fed to the model in the training phase (in contrast to Approach-1). However, in this approach, the AP of class WHV is disproportionately higher (80%) than other classes. This can be explained considering that when a worker is wearing hat and vest (i.e., WHV), it is easier for the model to distinguish the worker from the background.

As expected, the mAP of Approach-3 is the highest (85.6%) of all three approaches, mainly due to the fact that since there is only one class to learn (i.e., W), there is no chance of inter-class confusion (in contrast to Approach-2). Moreover, similar to Approach-1, the transfer learning framework provides the model with a better starting point to effectively learn the worker class.

7.4. Performance of the ML classifiers in Approach-1

As mentioned in Section 5.3.1, detection boxes of workers, hats, and

Fig. 9. Examples of annotations in the crowd-sourced and web-mined subsets of *Pictor-v3* dataset.

vests are normalized using Eq. (3). Examples of normalized boxes in the randomly selected training samples are shown in Fig. 15. According to this Figure, the hat worn by a worker generally resides in the upper-middle portion of that worker's body. Also, a hat that is significantly outside the detected worker's box is most likely not worn by that worker (e.g., could be worn by another worker). On the other hand, in general, the vest that is worn by a worker covers the entire width of that worker and resides mostly within the upper half of the body.

To demonstrate the importance of normalizing the boxes and training the models with added noise (as described in Section 5.3.1), the results of different models with or without these steps are summarized in Table 2. In particular, each model receives detected boxes of workers, hats, and vests from YOLO-v3-A1 model, and classifies each detected worker as W, WH, WV or WHV. Results shown in Table 2 indicate that normalization and training noise, individually, improves the model's mAP by 5.4% to 25.5%, and by 0.7% to 20.6%,

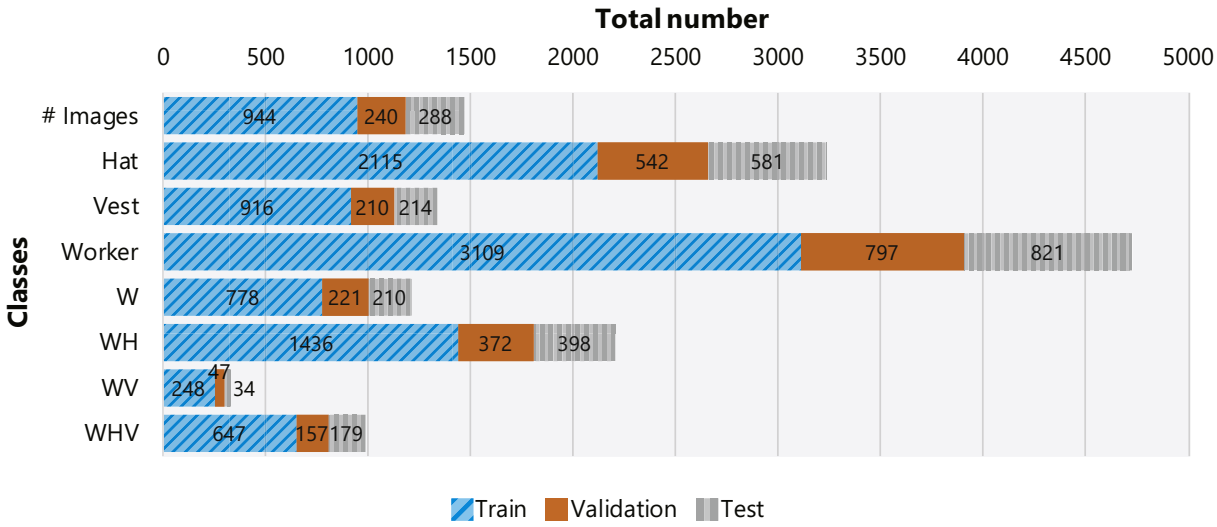

Fig. 10. Number of images and instances in the training, validation and testing subsets.

respectively. Thus, when normalization and training noise are applied together, the models achieve the best mAP and mean F1 score. Moreover, to contrast the performance of an ML model that has implicitly learned the PPE verification rules (Section 5.3.1), a baseline model with explicit rules is also tested. The baseline model simply checks the IoU between a worker box and hat/vest box, and if the IoU $>$ 45% (empirically-found threshold), it is assumed that the worker is wearing that hat/vest. Table 2 shows that ML models with normalization and training noise outperform the baseline model, particularly, the DT model, by achieving ~3% higher mAP and ~5% higher mean F1 score. This finding indicates that ML models can learn more efficient rules.

Furthermore, the best model, i.e., DT model, also provides the learned rules in the form of a decision tree that is human-interpretable. An example of the DT-generated rules for verification of hat in Approach-1 is illustrated in Fig. 16. This Figure shows that based on the input features, i.e., coordinates of top-left (x_0, y_0) and bottom-right (x_1, y_1) corners of the normalized box of hat, the model can verify PPE compliance by performing at most 4 Boolean comparisons.

7.5. Effect of the proposed modifications in Approach-2

In Approach-2, we proposed two major modifications over the original YOLO-v3 implementation (apart from the modification to the architecture), i.e., using SoftMax activation in the output layer, and performing NMS across all classes in the post-processing step. To verify the effectiveness of these proposed modifications, a baseline model is trained and tested similar to the YOLO-v3-A2 model, however, without those modifications. The comparison between the two models is demonstrated in Fig. 17, which shows that the modified model achieves ~2.4% higher mAP than the baseline model. However, the

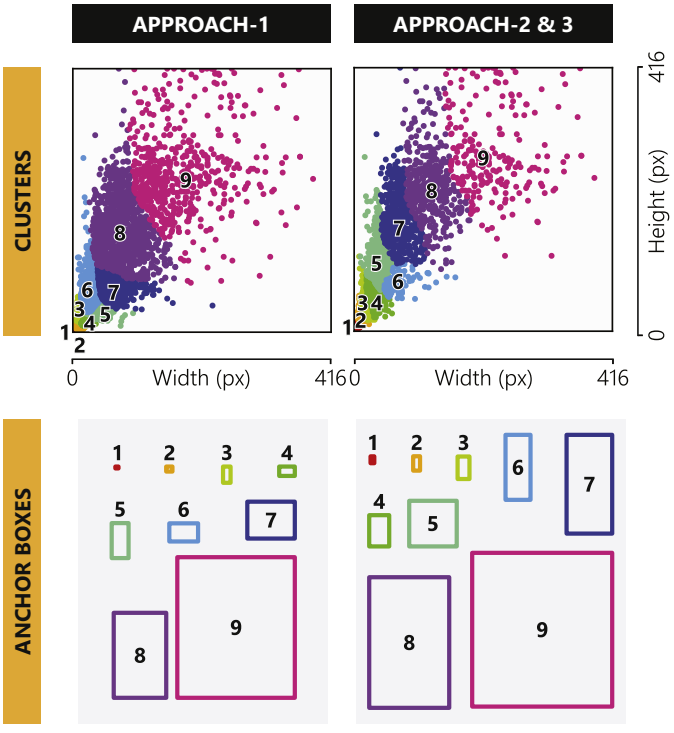

Fig. 11. Nine clusters and corresponding anchor boxes for each approach.

Fig. 12. Examples of actual and randomly augmented data in three approaches.

improvement in the average precision of classes W (4.2%) and WV (4.4%) is significant. The underlying reasons can be better understood from the given examples in Fig. 17, which show that the baseline model sometimes detects a single worker as multiple classes, e.g., worker in example B1 (row B, column 1) in Fig. 17 is detected as both W and WH. However, the modified model effectively eliminates such redundant detections (e.g., example B3 in Fig. 17) and generally outputs only one detection for each worker. These visual examples support the arguments posed in Section 5.3.2 and justify the proposed modifications in Approach-2.

7.6. Performance of the CNN classifierss in Approach-3

Given cropped test images of workers, the accuracy of the VGG-16, ResNet-50, and Xception models in classifying that image into W, WH, WV, and WHV classes are 78.2%, 77.8%, and 76.8%, respectively. The confusion matrices for these models are illustrated in Fig. 18, which shows that in general, all three models are better at detecting classes that contain hat (i.e., WH and WHV) compared to those that do not contain hat (i.e., W and WV). For example, all three models detect classes WH and WHV with > 81% accuracy, while the accuracy of W and WH detection is capped under 78%. Furthermore, classes without hat are mostly confused with their counterpart classes with hat. For

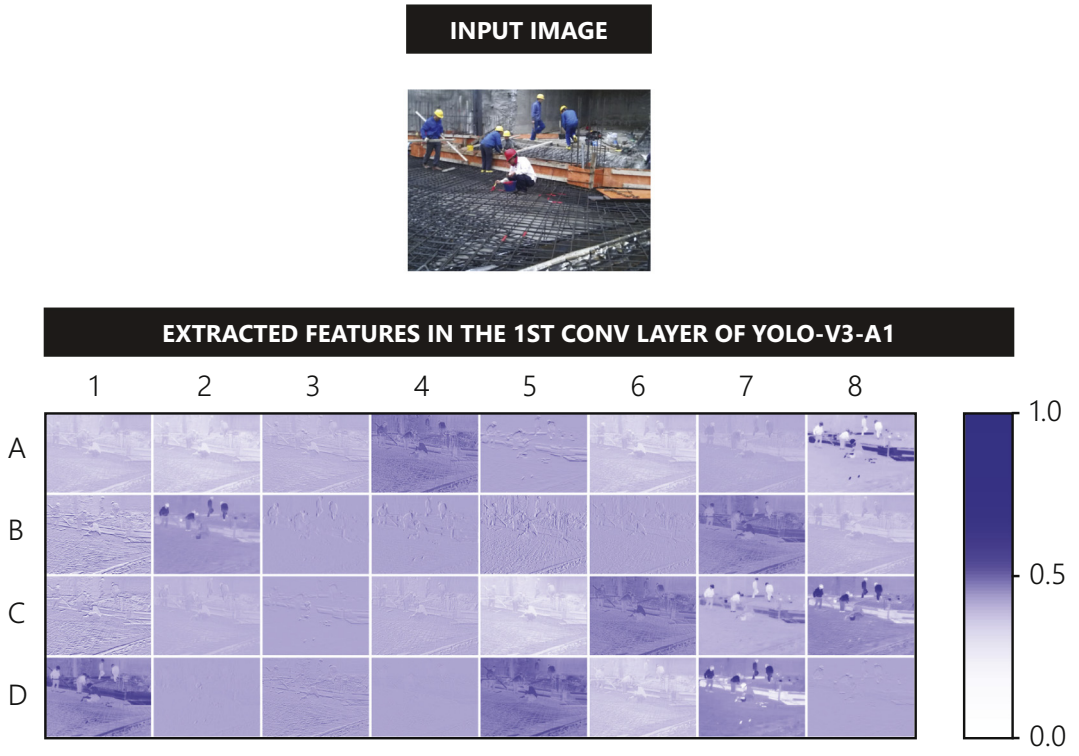

Fig. 13. Example of extracted features in the first convolutional layer of YOLO-v3-A1 model.

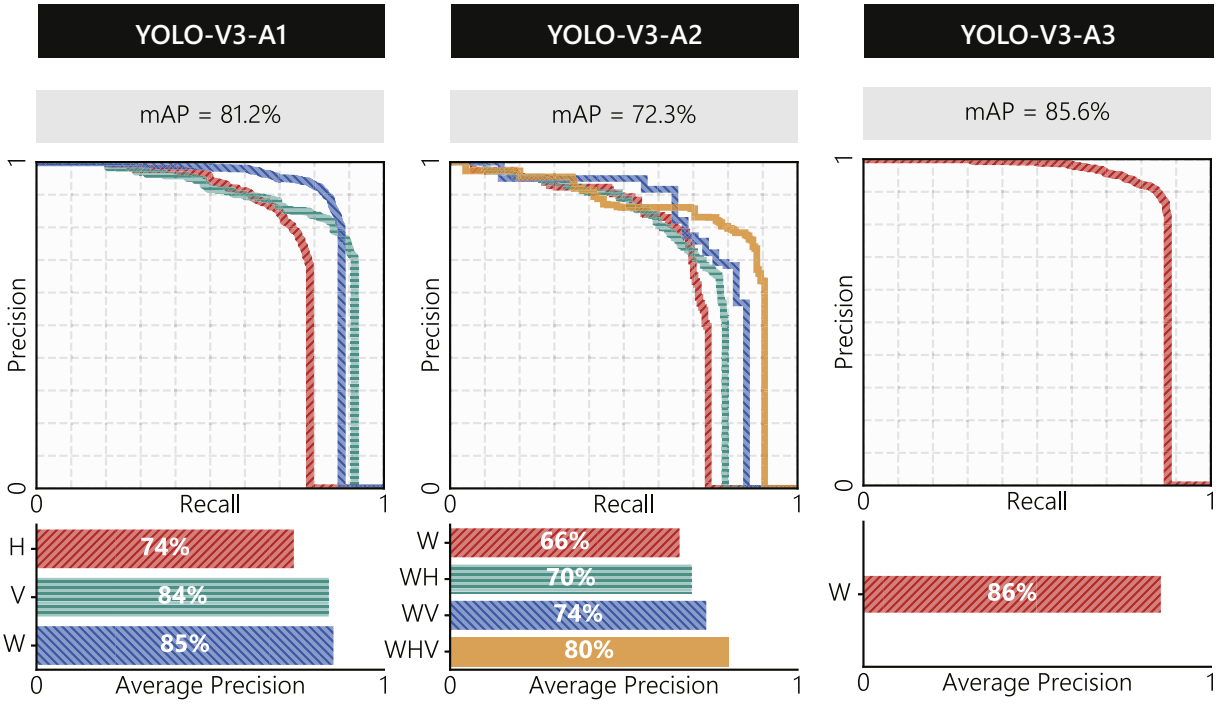

Fig. 14. Performance of the YOLO-v3 models in three approaches.

example, in VGG-16, W is confused with WH in 21% of cases, while WV is confused with WHV in 35% of the time. This indicates that the models false-positively detect hats in those images. A potential reason could be the unbalance of the dataset, where the number of samples in W and WV is significantly lower than WH and WHV, respectively (Fig. 8). This can bias the models toward more frequently predicting the classes with higher number of samples (i.e., WH and WHV). Additionally, similar to the YOLO-v3 models, the classifier models might find it difficult to determine the presence or absence of hat in an image since hats are relatively smaller in size.

7.7. Comparison of the overall performance of three approaches

In previous subsections, the performance of various components in each approach was reported individually. In this subsection, the overall performance of each approach is investigated. Fig. 19 indicates that

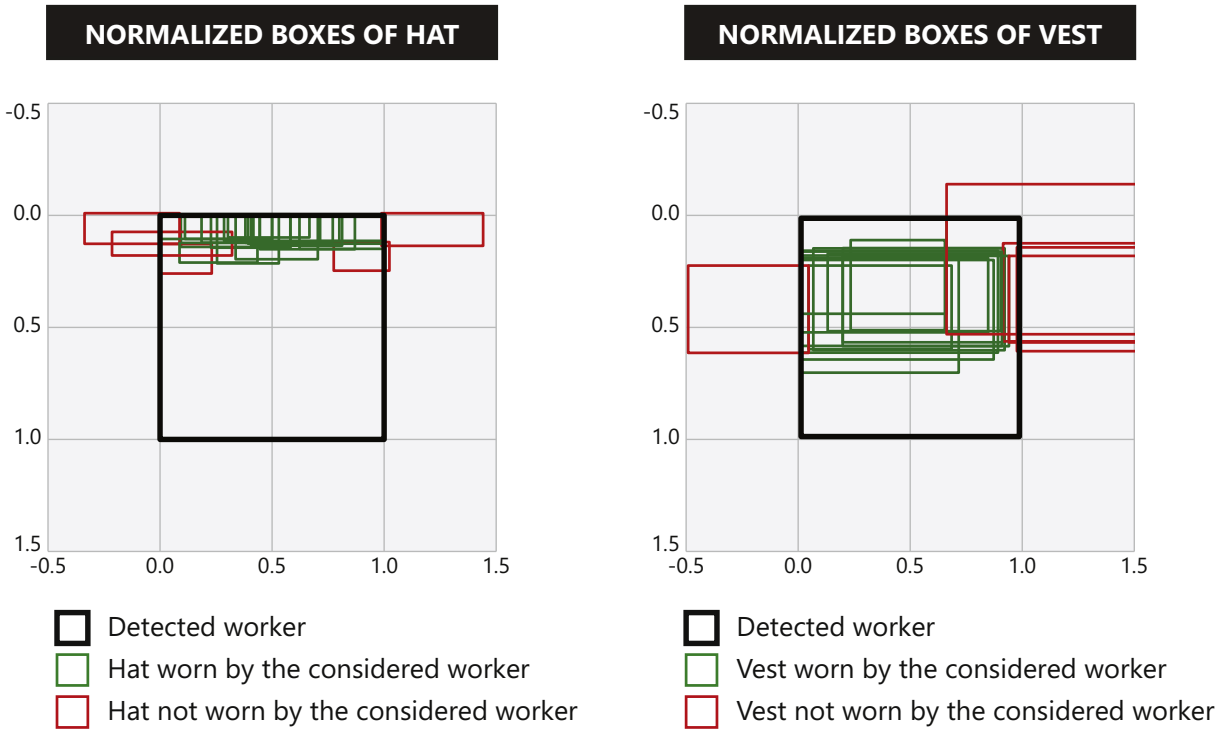

Fig. 15. Randomly selected examples of normalized boxes of hat and vest.

Table 2
Summary of the results from ML classifiers in Approach-1.

	Baseline	Not normalized, and no training noise		Not normalized, but with training noise		Normalized, but no training noise		Normalized, and with training noise	
		DT	NN	DT	NN	DT	NN	DT	NN
mAP	60.3%	36.9%	42.6%	37.6%	56.9%	42.5%	59.4%	63.1%	62.3%
Mean F1	61.3%	45.3%	49.0%	46.2%	60.6%	49.2%	63.3%	66.3%	65.5%

Approach-2 yields the best mAP (72.3%) among all approaches. Although the mAP of YOLO-v3 models in Approach-1 (81.2%) and Approach-3 (85.6%) is higher than the one in Approach-2 (72.3%), the accumulated errors in the post-processing steps reduce the final mAP in these approaches. To note, in Approach-3, while using a single CNN classifier, ResNet-50 provides the best performance (64.2%). However, almost all the Bayesian models tend perform better than the single CNN classifiers with one exception of Bayesian with VGG-16 and Xception (61.5%). The best model in this approach is the Bayesian with all three classifiers (67.9%).

To compare the processing time in each approach, all models are run on the same device, a Dell Precision-7530 laptop with Intel Core i7 8850H (6 cores, each at 2.6 Hz), 16 GB RAM, NVIDIA Quadro P2000 GPU with 4GB memory, and Windows 10 operating system. The average processing time for one test image in each approach is shown in Fig. 20, according to which, Approach-1 with DT is the fastest (74.8 ms), closely followed by Approach-1 with NN (79.2 ms). Although the processing time for YOLO-v3 models in Approach-3 is faster (69.4 ms) than in Approach-1 and Approach-2, the added time for subsequent classifier models made this approach the slowest among all.

It must be noted that the processing time for YOLO-v3 models is directly proportional to the number of classes these models predict. The number of output classes is 3, 4, and 1, in YOLO-v3-A1, YOLO-v3-A2, and YOLO-v3-A3, respectively. This explains the variations in the processing time of YOLO-v3 models in Fig. 20. Also, in Approach-3, VGG-16 (51.5 ms) is faster than ResNet-50 (89.7 ms) and Xception (118.8 ms) because it has fewer convolutional layers and thus, requires less time for processing. On the other hand, the Bayesian models are the slowest among all since they require the prediction from CNN classifiers before applying the Bayesian theorem to infer the final output. Nonetheless, Approach-1, Approach-2, and Approach-3 with single classifier models can process > 5 images per second and, therefore, are suitable for real-time detection of workers' PPE attire. Moreover, the Bayesian models can process multiple images (> 3) per second and, thus, can be considered as near real-time.

8. Discussion

8.1. Benchmarking the results

To benchmark the performance of the developed PPE detection approaches, results are compared with the reported findings in the literature [10,15,36,37], and summarized in Table 3. To ensure a fair comparison with previous studies that mostly focused on verifying hard hat usage, the performance of the developed methods in this study in detecting only hard hat is also measured and reported. To do this, the final output classes, i.e., W, WH, WV, and WHV, are merged into two classes: *hat* (WH ∪ WHV) and *no hat* (W ∪ WV). Similarly, to measure the performance of detecting only vest, final output classes are merged into *vest* (WV ∪ WHV) and *no vest* (W ∪ WH).

Results listed in Table 3 shows that Fang et al. [10] achieves the highest precision, but can only detect workers who are not wearing hard hat. Besides this limitation, training and testing images were randomly drawn from consecutive video frames which makes it highly

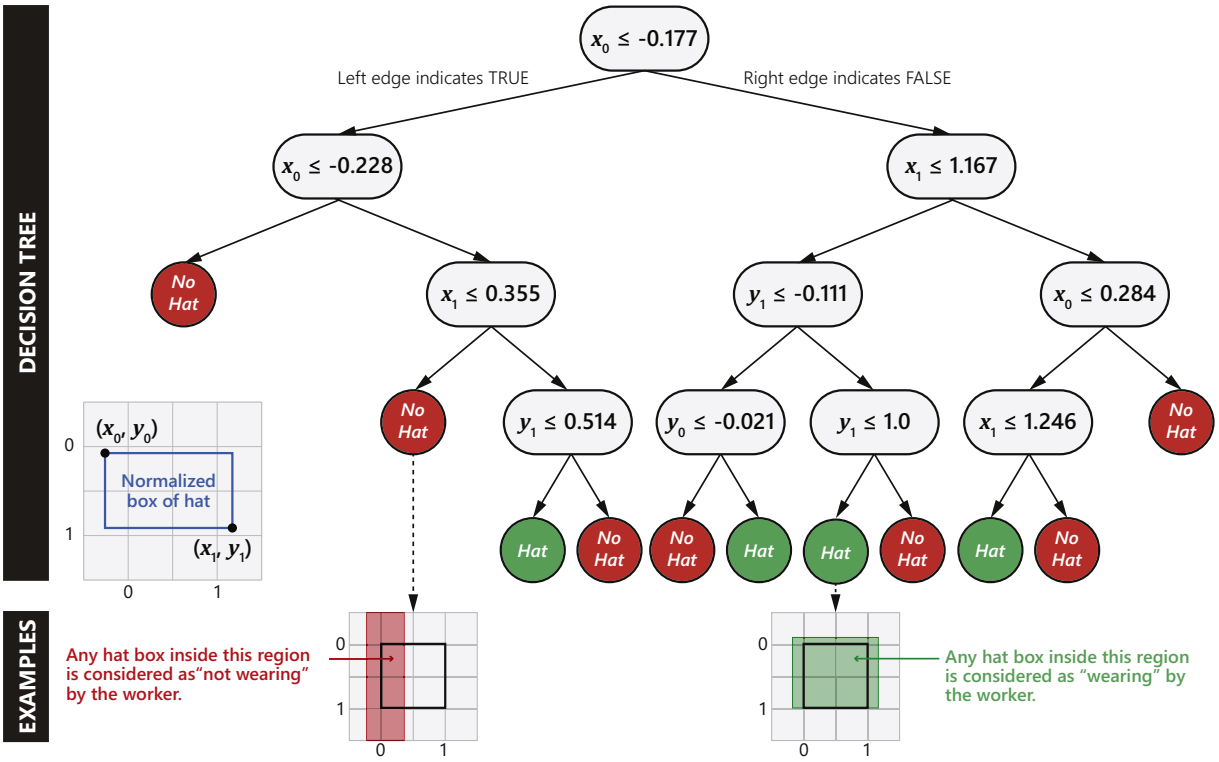

Fig. 16. Decision tree for verifying PPE compliance of wearing hat.

likely that testing images are visually very similar (i.e., taken by the same device, at the same location, from the same view angle, and almost at the same time). In contrast, in the research presented in this paper, images are taken individually from a variety of devices, locations, times, perspectives, and projects. Therefore, testing images are more challenging and thus, the performances of the proposed methods reflect their most probable performance when tested on new (unseen) images, making these methods highly scalable and more reliable for practical use.

Table 3 also shows that the method proposed by Mneymneh et al. [15] can achieve a high precision (86.79%) in detecting hat. However, the method can process only 2 FPS, and it requires video frames as input to identify workers based on their movements in the scene. In contrast, the proposed methods in this paper are applicable to both still images and video frames and do not rely on any kind of object movement, making them more practical for general applications.

As listed in Table 3, the performance of the proposed methods are on par with or higher than the methods developed by Xie et al. [37] and Wu et al. [36] in terms of average precision and detection speed. Moreover, the methods introduced in this study have an added capability to verify PPE compliance with vest. However, the method developed by Wu et al. [36] can verify different colors of hat as well.

The performance of the YOLO-v3-AI model in detecting hat is further assessed by testing it on the GDUT-HWD dataset published by Wu et al. [36]. To note, the ground-truth boxes of hat objects in GDUT-HWD include not only the hat but also the face of the worker who is wearing that hat, while the output hat boxes generated by the YOLO-v3-A1 model only cover hat objects (and not workers' faces). To account for this difference, output hat boxes generated by the YOLO-v3-AI model are elongated 65% from the bottom to approximately include the worker's face. Detection results indicate that the YOLO-v3-A1 model can detect hat with 72.73% average precision, compared to the mean average precision of 74.76% in Wu et al. [36] (including all colors).

8.2. Qualitative comparison of the three approaches

A comparison of the three proposed approaches is summarized in Table 4. This qualitative analysis confirms that all three approaches are scalable, i.e., they can be scaled up to detect more than two PPE types, or scaled down to detect only one PPE type (hat or vest). However, scaling up may require collecting and annotating more training images and re-training some or all models in each of the approaches. As such, the modularity of Approach-1 and Approach-3 can be significantly beneficial. For example, in Approach-1, ML classifiers (each verifying individual PPE compliance) can be applied independently. Therefore, when adding a new type of PPE (e.g., steel toe shoe), the ML classifiers that are trained to verify existing PPE types (hat and vest) do not require any further re-training. The same is true in Approach-3 since the YOLO-v3-A3 model does not require any re-training either as it only detects workers (regardless of the PPE attire). Moreover, the modularity of these approaches facilitates the investigation of each module independently for further improvement. However, Approach-2 sacrifices modularity for simplicity, i.e., only one model (YOLO-v3-A2) directly provides end-to-end verification results. Nonetheless, Approach-1 and Approach-2 are more robust as they can perform the verification task in real-time, while Approach-3 can be considered as "near" real-time. Finally, it can be inferred that in a broader sense, Approach-1 can identify the contextual relationship between a person and objects (e.g., if a person is "wearing" a hat or not) without explicitly defining the rules. This capability makes Approach-1 adoptable by other domains beyond checking PPE compliance, e.g., if a person is "holding" a weapon (e.g., in safety and security applications).

8.3. Lessons learned and limitations

One of the most important lessons learned from this study is that using transfer learning from larger datasets that contain person, can significantly benefit object detection models intended for identifying construction workers in visual data. It is also found that models

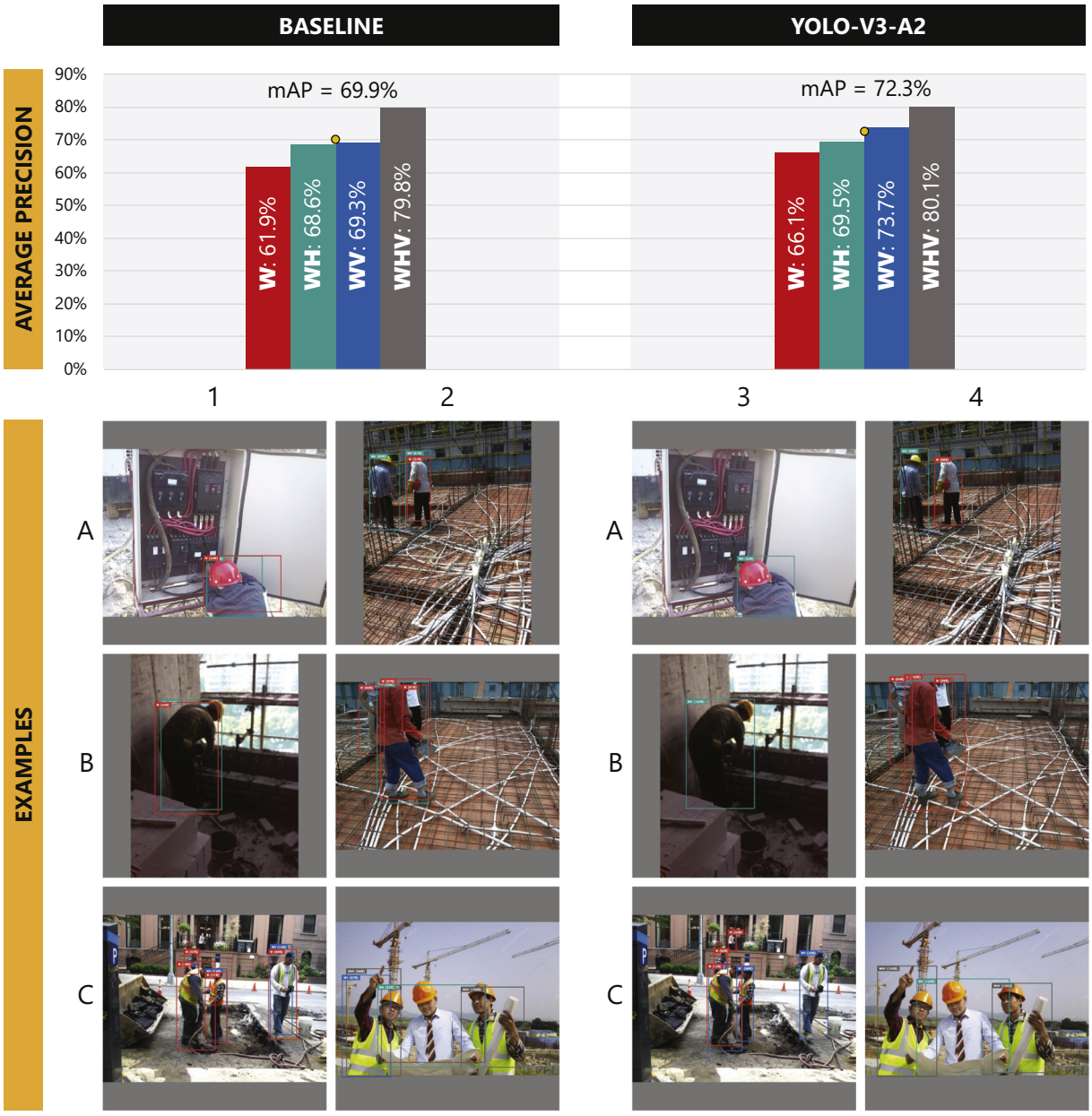

Fig. 17. Comparison between baseline YOLO-v3 model and modified YOLO-v3-A2 model in Approach-2.

generally struggle to detect smaller objects, e.g., hat. Therefore, it is anticipated that even a sufficiently-trained model may find it difficult to detect smaller PPE components (e.g., gloves). In Approach-1, it was found that ML classifiers can explicitly learn the rules to verify if a worker is properly wearing any PPE component. In particular, a DT model can express the rules in a more structured form that might be often unmanageable by humans, particularly, when the rules are too complicated. Moreover, in this process, normalizing the boxes and training the model with added noise improve the accuracy of the models. Furthermore, in Approach-2, it was observed that since classes are disjoint, applying the SoftMax activation function (instead of Sigmoid) in the last layers of YOLO-v3 models improves detection

		VGG-16				ResNet-50				Xception			
		Predicted				Predicted				Predicted			
		W	WH	WV	WHV	W	WH	WV	WHV	W	WH	WV	WHV
Actual	W	**78%**	21%	0%	0%	**72%**	26%	2%	0%	**74%**	23%	0%	3%
	WH	6%	**92%**	0%	2%	6%	**88%**	0%	6%	9%	**87%**	0%	4%
	WV	3%	0%	**62%**	35%	0%	3%	**65%**	32%	3%	3%	**62%**	32%
	WHV	1%	15%	3%	**81%**	0%	10%	3%	**87%**	1%	11%	4%	**84%**

Fig. 18. Confusion matrices for VGG-16, ResNet-50, and Xception classifier models.

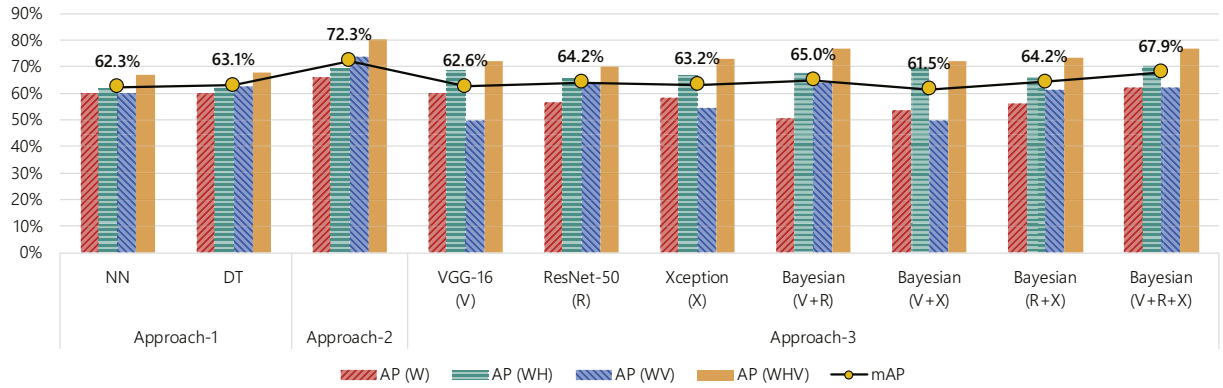

Fig. 19. Comparison of the overall performance of each approach.

accuracy. Finally, in Approach-3, the YOLO-v3-A3 model showed the most promising result in detecting workers since the entire model architecture was dedicated to learning one object class.

One of the main limitations of vision-based detection methods is that they are susceptible to occlusion, poor illumination, and blurriness. Moreover, the speed of the implementation may vary depending on the computer hardware used for testing. Furthermore, although the theoretical framework introduced in this study is designed to be scalable to any number and type of PPE, proposed methods were only tested on hat and vest classes, necessitating future experimentation with more data to draw conclusions about the generalizability of the results. Moreover, the proposed methods do not identify the color of the PPE components, neither do they associate any personal identification with the output of the verification (i.e., results do not indicate which worker is not in compliance). At the same time, this latter limitation guarantees the privacy of individual workers on the job site.

9. Summary and conclusion

This paper introduced, tested, and evaluated three DL-based approaches for detecting PPE attire (i.e., if a worker is wearing hat, vest, or both). The proposed methods are designed to be scalable to any number and type of PPE (e.g., not only hard hat, and safety vest but also gloves, safety goggles, and steel toe shoes). This can be achieved by modifying the last layer of the YOLO-v3 models to accommodate object classes of interest in each application (Table 1).

In Aproach-1, worker, hat, and vest (three object classes) were detected by YOLO-v3-A1 individually. Next, ML classifiers, e.g., NN and DT, were used to check if a detected hat or vest was in fact worn by a detected worker. Based on the worker's PPE attire, s/he was classified as W (wearing no hat or vest), WH (wearing only hat), WV (wearing only vest), or WHV (wearing both hat and vest). Furthermore, the authors proposed novel techniques to normalize the detected boxes and add random noise (i.e., randomly shifting and scaling the boxes) while training the ML models (Section 5.3.1). It was found that the proposed normalization and addition of noise significantly improved the performance of the ML models (Table 2).

In Approach-2, the YOLO-v3-A2 model localized workers in the input image and directly classified each detected worker as W, WH, WV, or WHV. Since the classes in this approach are disjoint (Section 5.3.2), the authors proposed to apply SoftMax activation function in the output layer of YOLO-v3-A2 as opposed to Sigmoid activation function used in the original YOLO-v3 algorithm. Moreover, the original YOLO algorithm uses non-max-suppression (NMS) to discard duplicated detections for each class individually. However, in this research, NMS was used across all classes since experimental observation indicated that one worker might be detected as multiple classes. Results confirm that the proposed modifications to YOLO algorithms improved the performance of the model in this approach.

In Approach-3, the model first detected all workers in the input image and then, a CNN-based classifier model (e.g., VGG-16, ResNet-50, Xception, or Bayesian) was applied to the cropped worker images to classify the detected worker as W, WH, WV, or WHV. To achieve better results by combining the output from multiple CNN classifiers, the authors proposed to apply Bayesian framework, which resulted in generally better results compared to using a single CNN classifier.

All in all, the Approach-2 provided the most accurate detection of PPE attire (72.3% mAP) among all. While the YOLO models in Approach-1 and -3 achieved 81.2% and 85.6% mAP, respectively, the errors generated by the classifiers in the subsequent parts lowered the mAP to 63.1% (with DT) and 67.9% (Bayesian with all three CNN classifiers), respectively. It indicates that potentially better performances may be achieved by improving the classifiers in these approaches. In terms of processing speed, Approach-1 with DT was the

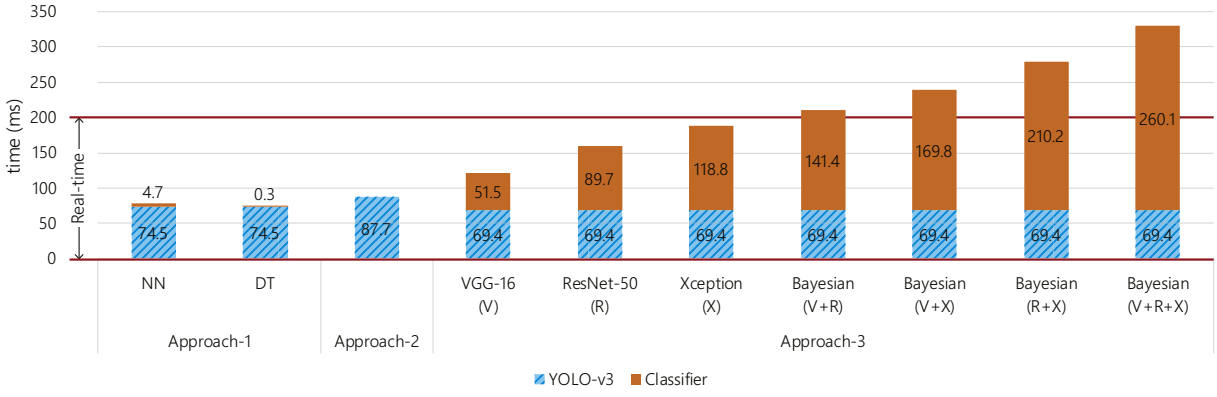

Fig. 20. Average time for processing one test image in each approach.

Table 3
Performance of different PPE detection models.

Criteria	Detection	Previous studies				This study		
		Fang et al. [10]	Mneymneh et al. [15]	Xie et al. [37] (CAHD)	Wu et al. [36] (Pelee-RPA)	Approach-1 (DT)	Approach-2	Approach-3 (Bayesian: V + R + X)
Hat	Hat	×	–	–	74.76%	74.29%	79.81%	79.33%
	No hat	> 90%[a]	–	–	62.67%	63.84%	68.12%	63.13%
	Overall	> 90%[a]	86.79%[a]	54.6%	72.34%	69.06%	73.97%	71.23%
Vest	Vest	×	×	×	×	74.32%	84.96%	83.01%
	No Vest	×	×	×	×	73.18%	78.71%	77.98%
	Overall	×	×	×	×	73.75%	81.84%	80.49%
Hat + Vest	Overall	×	×	×	×	63.1%	72.3%	67.93%
FPS		10	2	11	3.22	13	11	3

– Not available.
× Not applicable.
[a] Precision.

Table 4
Qualitative comparison of the three proposed approaches.

Criteria	Approach-1	Approach-2	Approach-3
Scalability	✓	✓	✓
Modularity	✓		✓
Robustness	✓	✓	
Adoptability	✓		

fastest (74.8 ms) among all while Approach-2 being a close alternative (87.7 ms). In Approach-3, assembling multiple CNN classifiers in Bayesian frameworks improved the detection performance, however, it lowered the detection speed by significant margin.

Approach-1 can adapt to other domains since it has the potential to understand contextual relationship between different objects (e.g., if a worker is “wearing” hat or not). On the other hand, Approach-2 provides a simplified framework with single DL-based network for end-to-end PPE verification. Last but not least, Approach-3 has the highest potential for achieving the best performance (e.g., by assembling multiple CNN classifiers in a Bayesian framework) since the YOLO algorithm adopted in this approach detects workers with the highest precision. Therefore, all three proposed methods are deemed practical for detecting PPE attire of workers. Particularly, Approach-1, Approach-2, and Approach-3 with single CNN classifier can be implemented in real-time. Also, these methods can be scaled to detect more than two types of PPE. Furthermore, the modularity of Approach-1 and -3 allows their individual components to be improved independently to obtain a better overall performance. From a practical point of view, when fast PPE detection or running on a mobile device is important, Approach-1 is recommended since it requires the least amount of computation. On the other hand, for best prediction, Approach-2 is recommended since it outperforms the other two methods in terms of PPE detection accuracy.

The trained YOLO-v3 models can be applied to construction images/videos to generate results in real-time. These models have effectively learned the features of workers and the most common PPE components, i.e., hard hat and vest, and therefore, can be used in a transfer learning framework to build other solutions for detecting PPE. To foster future innovative solutions and assist construction practitioners and researchers to apply or improve DL-based applications to ensure workplace safety, all models, as well as the crowd-sourced Pictor-v3 dataset described in this paper, are publicly available on authors' GitHub page at https://github.com/ciber-lab/pictor-ppe. In the future, the authors will expand the dataset to detect other common PPE components, e.g., safety glass and gloves. Furthermore, the authors are planning to develop a DL-based methodology to automatically generate safety reports in natural language based on construction site imagery.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

The authors gratefully acknowledge the U.S. National Science Foundation (NSF) for supporting this project through grant CMMI 1800957, as well as Mr. Yalong Pi for assisting in data preparation. Any opinions, findings, conclusions, and recommendations expressed in this paper are those of the authors and do not necessarily represent the views of the NSF or the individual named above.

References

[1] United States Census Bureau, Construction spending, https://www.census.gov/construction/c30/prpdf.html , Accessed date: 6 July 2019.
[2] Bureau of Labor Statistics, Industries at a glance: construction, https://www.bls.gov/iag/tgs/iag23.htm , Accessed date: 6 July 2019.
[3] Bureau of Labor Statistics, Fatal occupational injuries counts and rates by selected industries, https://www.bls.gov/news.release/cfoi.t04.htm , Accessed date: 6 July 2019.
[4] OSHA, Commonly used statistics, https://www.osha.gov/oshstats/commonstats.html , Accessed date: 6 July 2019.
[5] OSHA, Worker safety series: construction, https://www.osha.gov/Publications/OSHA3252/3252.html , Accessed date: 6 July 2019.
[6] Centers for Disease Control and Prevention, Traumatic brain injuries in construction, https://blogs.cdc.gov/niosh-science-blog/2016/03/21/constructiontbi/ , Accessed date: 6 July 2019.
[7] OSHA, Safety vest requirements to protect flaggers from traffic hazards during construction work, https://www.osha.gov/laws-regs/standardinterpretations/2002-03-11 , Accessed date: 6 July 2019.
[8] OSHA, Safety and health regulations for construction, https://www.osha.gov/laws-regs/regulations/standardnumber/1926/1926.28 , Accessed date: 6 July 2019.
[9] F. Akbar-Khanzadeh, Factors contributing to discomfort or dissatisfaction as a result of wearing personal protective equipment, J. Hum. Ergol. 27 (1–2) (1998) 70–75.
[10] Q. Fang, H. Li, X. Luo, L. Ding, H. Luo, T.M. Rose, W. An, Detecting non-hardhat-use by a deep learning method from far-field surveillance videos, Autom. Constr. 85 (2018) 1–9, https://doi.org/10.1016/j.autcon.2017.09.018.
[11] X. Huang, J. Hinze, Analysis of construction worker fall accidents, J. Constr. Eng. Manag. 129 (3) (2003) 262–271, https://doi.org/10.1061/(asce)0733-9364(2003)129:3(262).
[12] OSHA, Top 10 most frequently cited standards, https://www.osha.gov/Top_Ten_Standards.html , Accessed date: 6 July 2019.
[13] OSHA, Employer payment for personal protective equipment; final rule, https://www.osha.gov/laws-regs/federalregister/2007-11-15-0 , Accessed date: 6 July 2019.
[14] OSHA, Cited standard in construction, https://www.osha.gov/pls/imis/citedstandard.naics?p_naics=23&p_esize=&p_state=FEFederal , Accessed date: 20 January 2019.
[15] B.E. Mneymneh, M. Abbas, H. Khoury, Vision-based framework for intelligent monitoring of hardhat wearing on construction sites, J. Comput. Civ. Eng. 33 (2) (2018) 4018066, https://doi.org/10.1061/(asce)cp.1943-5487.0000813.
[16] A. Kelm, L. Laußat, A. Meins-Becker, D. Platz, M.J. Khazaee, A.M. Costin, M. Helmus, J. Teizer, Mobile passive radio frequency identification (RFID) portal

for automated and rapid control of personal protective equipment (PPE) on construction sites, Autom. Constr. 36 (2013) 38–52, https://doi.org/10.1016/j.autcon.2013.08.009.
[17] S. Barro-Torres, T.M. Fernández-Caramés, H.J. Pérez-Iglesias, C.J. Escudero, Real-time personal protective equipment monitoring system, Comput. Commun. 36 (1) (2012) 42–50, https://doi.org/10.1016/j.comcom.2012.01.005.
[18] B. Naticchia, M. Vaccarini, A. Carbonari, A monitoring system for real-time interference control on large construction sites, Autom. Constr. 29 (2013) 148–160, https://doi.org/10.1016/j.autcon.2012.09.016.
[19] J. Seo, S. Han, S. Lee, H. Kim, Computer vision techniques for construction safety and health monitoring, Adv. Eng. Inform. 29 (2) (2015) 239–251, https://doi.org/10.1016/j.aei.2015.02.001.
[20] S. Han, S. Lee, A vision-based motion capture and recognition framework for behavior-based safety management, Autom. Constr. 35 (2013) 131–141, https://doi.org/10.1016/j.autcon.2013.05.001.
[21] S. Du, M. Shehata, W. Badawy, Hard hat detection in video sequences based on face features, motion and color information, Proc. 3rd International Conference on Computer Research and Development (ICCRD), Shanghai, China, 4 2011, pp. 25–29, , https://doi.org/10.1109/iccrd.2011.5763846.
[22] K. Shrestha, P.P. Shrestha, D. Bajracharya, E.A. Yfantis, Hard-hat detection for construction safety visualization, Journal of Construction Engineering (2015) 1–8, https://doi.org/10.1155/2015/721380.
[23] M.W. Park, N. Elsafty, Z. Zhu, Hardhat-wearing detection for enhancing on-site safety of construction workers, J. Constr. Eng. Manag. 141 (9) (2015) 04015024, , https://doi.org/10.1061/(asce)co.1943-7862.0000974.
[24] J. Weston, C. Watkins, Technical Report CSD-TR-98-04, Department of Computer Science, University of London, 1998.
[25] M.W. Park, I. Brilakis, Construction worker detection in video frames for initializing vision trackers, Autom. Constr. 28 (2012) 15–25, https://doi.org/10.1016/j.autcon.2012.06.001.
[26] Z. Kolar, H. Chen, X. Luo, Transfer learning and deep convolutional neural networks for safety guardrail detection in 2D images, Autom. Constr. 89 (2018) 58–70, https://doi.org/10.1016/j.autcon.2018.01.003.
[27] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner, Gradient-based learning applied to document recognition, Proc. IEEE 86 (11) (1998) 2278–2324, https://doi.org/10.1109/5.726791.
[28] A. Krizhevsky, I. Sutskever, G.E. Hinton, ImageNet classification with deep convolutional neural networks, Commun. ACM 60 (6) (2017), https://doi.org/10.1145/3065386.
[29] K. Simonyan, A. Zisserman, Very deep convolutional networks for large-scale image recognition, arXiv preprint arXiv:1409.1556, 2014.
[30] M. Siddula, F. Dai, Y. Ye, J. Fan, Unsupervised feature learning for objects of interest detection in cluttered construction roof site images, Procedia Engineering 145 (2016) 428–435, https://doi.org/10.1016/j.proeng.2016.04.010.
[31] Z. Zivkovic, Improved adaptive Gaussian mixture model for background subtraction, Proc. ICPR, Cambridge, UK, 2004, pp. 28–31, , https://doi.org/10.1109/icpr.2004.1333992.
[32] L. Ding, W. Fang, H. Luo, P.E.D. Love, B. Zhong, X. Ouyang, A deep hybrid learning model to detect unsafe behavior: integrating convolution neural networks and long short-term memory, Autom. Constr. 86 (2018) 118–124, https://doi.org/10.1016/j.autcon.2017.11.002.
[33] S. Hochreiter, J. Schmidhuber, Long short-term memory, Neural Comput. 9 (8) (1997) 1735–1780, https://doi.org/10.1162/neco.1997.9.8.1735.
[34] N.D. Nath, T. Chaspari, A.H. Behzadan, A transfer learning method for deep neural network annotation of construction site imagery, Proc. Conference on Construction Applications of Virtual Reality (CONVR), Auckland, New Zealand, 2018, pp. 1–10.
[35] N.D. Nath, T. Chaspari, A.H. Behzadan, Single- and multi-label classification of construction objects using deep transfer learning methods, Journal of Information Technology in Construction 24 (28) (2019) 511–526, https://doi.org/10.36680/j.itcon.2019.028.
[36] J. Wu, N. Cai, W. Chen, H. Wang, G. Wang, Automatic detection of hardhats worn by construction personnel: a deep learning approach and benchmark dataset, Autom. Constr. 106 (2019) 102894, , https://doi.org/10.1016/j.autcon.2019.102894.
[37] Z. Xie, H. Liu, Z. Li, Y. He, A convolutional neural network based approach towards real-time hard hat detection, Proc. IEEE International Conference on Progress in Informatics and Computing (PIC), IEEE, 2018, pp. 430–434, , https://doi.org/10.1109/pic.2018.8706269.
[38] Smartvid.io, Inc, Smartvid.io, https://www.smartvid.io , Accessed date: 6 July 2019.
[39] S. Ren, K. He, R. Girshick, J. Sun, Faster R-CNN: towards real-time object detection with region proposal networks, IEEE Trans. Pattern Anal. Mach. Intell. 39 (6) (2017) 1137–1149, https://doi.org/10.1109/tpami.2016.2577031.
[40] J. Redmon, S. Divvala, R. Girshick, Ali Farhadi, You only look once: Unified, real-time object detection, Proc. IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, 2016, pp. 779–788 https://doi.org/10.1109/cvpr.2016.91.
[41] J. Redmon, A. Farhadi, YOLO9000: better, faster, stronger, Proc. IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, 2017, pp. 7263–7271, , https://doi.org/10.1109/cvpr.2017.690.
[42] C. Kyrkou, G. Plastiras, T. Theocharides, S.I. Venieris, C.S. Bouganis, DroNet: efficient convolutional neural network detector for real-time UAV applications, Proc. 2018 Design, Automation and Test in Europe Conference and Exhibition (DATE), Dresden, Germany, 2018, pp. 967–972, , https://doi.org/10.23919/date.2018.8342149.
[43] C. Szegedy, A. Toshev, D. Erhan, Deep neural networks for object detection, Adv. Neural Inf. Proces. Syst. 26 (2013) 2553–2561.
[44] R. Girshick, J. Donahue, T. Darrell, J. Malik, Rich feature hierarchies for accurate object detection and semantic segmentation, Proc. IEEE Conference on Computer Vision and Pattern Recognition, Columbus, OH, 2014, pp. 580–587, , https://doi.org/10.1109/cvpr.2014.81.
[45] R. Girshick, Fast R-CNN, Proc. IEEE International Conference on Computer Vision, Santiago, Chile, 2015, pp. 1440–1448, , https://doi.org/10.1109/iccv.2015.169.
[46] K. He, G. Gkioxari, P. Dollar, R. Girshick, Mask R-CNN, Proc. IEEE International Conference on Computer Vision (ICCV), Venice, Italy, 2017, pp. 2961–2969, , https://doi.org/10.1109/iccv.2017.322.
[47] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.Y. Fu, A.C. Berg, SSD: Single shot multibox detector, Proc. European Conference on Computer Vision, Amsterdam, the Netherlands, 2016, pp. 21–37, , https://doi.org/10.1007/978-3-319-46448-0_2.
[48] T.Y. Lin, P. Goyal, R. Girshick, K. He, P. Dollár, Focal loss for dense object detection, Proc. IEEE International Conference on Computer Vision, Venice, Italy, 2017, pp. 2980–2988, , https://doi.org/10.1109/iccv.2017.324.
[49] J. Redmon, A. Farhadi, YOLOv3: An incremental improvement, arXiv preprint, arXiv:1804.02767, 2018.
[50] M. Oquab, L. Bottou, I. Laptev, J. Sivic, Learning and transferring mid-level image representations using convolutional neural networks, Proc. IEEE Conference on Computer Vision and Pattern Recognition, Columbus, OH, 2014, pp. 1717–1724, , https://doi.org/10.1109/cvpr.2014.222.
[51] H.C. Shin, H.R. Roth, M. Gao, L. Lu, Z. Xu, I. Nogues, J. Yao, D. Mollura, R.M. Summers, Deep convolutional neural networks for computer-aided detection: CNN architectures, dataset characteristics and transfer learning, IEEE Trans. Med. Imaging 35 (5) (2016) 1285–1298, https://doi.org/10.1109/tmi.2016.2528162.
[52] J. Deng, W. Dong, R. Socher, L.J. Li, K. Li, L. Fei-Fei, ImageNet: a large-scale hierarchical image database, Proc. IEEE Conference on Computer Vision and Pattern Recognition, Miami, FL, 2009, pp. 248–255, , https://doi.org/10.1109/cvpr.2009.5206848.
[53] M. Everingham, L.V. Gool, C.K.I. Williams, J. Winn, A. Zisserman, The pascal visual object classes (VOC) challenge, Int. J. Comput. Vis. 88 (2) (2010) 303–338, https://doi.org/10.1007/s11263-009-0275-4.
[54] X. Chen, H. Fang, T.Y. Lin, R. Vedantam, S. Gupta, P. Dollár, C.L. Zitnick, Microsoft COCO captions: data collection and evaluation server, arXiv preprint arXiv:1504.00325, 2015.
[55] Y. Zhang, D. Zhao, J. Zhang, R. Xiong, W. Gao, Interpolation-dependent image downsampling, IEEE Trans. Image Process. 20 (11) (2011) 3291–3296, https://doi.org/10.1109/tip.2011.2158226.
[56] D.P. Kingma, J. Ba, Adam: a method for stochastic optimization, arXiv preprint arXiv:1412.6980, 2014.
[57] L. Perez, J. Wang, The effectiveness of data augmentation in image classification using deep learning, arXiv preprint arXiv:1712.04621, 2017.
[58] C.M. Bishop, Pattern Recognition and Machine Learning, Springer, 2006.
[59] J. Friedman, T. Hastie, R. Tibshirani, The Elements of Statistical Learning, Springer Series in Statistics vol. 1, Springer, 2001.
[60] K.P. Murphy, Machine Learning: A Probabilistic Perspective, MIT press, 2012.
[61] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, Proc. IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, 2016, pp. 770–778, , https://doi.org/10.1109/cvpr.2016.90.
[62] F. Chollet, Xception: Deep learning with depthwise separable convolutions, Proc. IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, 2017, pp. 1251–1258, , https://doi.org/10.1109/cvpr.2017.195.
[63] A. Turpin, F. Scholer, User performance versus precision measures for simple search tasks, Proc. 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Seattle, Washington, 2006, pp. 11–18, , https://doi.org/10.1145/1148170.1148176.
[64] M.C. Yuen, I. King and K.S. Leung, A survey of crowdsourcing systems, in: Proc. 3rd IEEE International Conference on Privacy, Security, Risk and Trust, Boston, MA, 2011, pp. 766–773, https://doi.org/10.1109/passat/socialcom.2011.203.
[65] R. Kosala, H. Blockeel, Web mining research: a survey, ACM SIGKDD Explorations Newsletter 2 (1) (2000) 1–15, https://doi.org/10.1145/360402.360406.
[66] R. Fergus, L. Fei-Fei, P. Perona, A. Zisserman, Learning object categories from Google's image search, Proc. 10th IEEE International Conference on Computer Vision, Beijing, China, 2005, pp. 1816–1823, , https://doi.org/10.1109/iccv.2005.142.
[67] Labelbox, Inc, Labelbox, www.labelbox.com , Accessed date: 6 July 2019.
[68] B.C. Russell, A. Torralba, K.P. Murphy, W.T. Freeman, LabelMe: a database and web-based tool for image annotation, Int. J. Comput. Vis. 77 (1–3) (2008) 157–173 https://doi.org/10.1007/s11263-007-0090-8.
[69] D. Castelvecchi, Can we open the black box of AI? Nature News 538 (7623) (2016) 20, https://doi.org/10.1038/538020a.
[70] M.D. Zeiler and R. Fergus, Visualizing and understanding convolutional networks, in: Proc. European Conference on Computer Vision (2014), Zurich, Switzerland, pp. 818–833, https://doi.org/10.1007/978-3-319-10590-1_53.