KeAi
CHINESE ROOTS
GLOBAL IMPACT

International Journal of Innovation Studies

EVALUATION OF DEEP LEARNING MODELS FOR ACCURATE SAFETY HELMET DETECTION IN CONSTRUCTION ENVIRONMENTS

Dr. Shivakumar Dalali[1], Dr. Girish Rao Salanke N S[2], Dr. Manjunath C. R[3], Dr. Manjunath T. K[4]

[1]Professor, Department of Computer Science and Engineering (CSE), Global Academy of Technology, Bengaluru, 560098, Karnataka, India
shivakumardalali@gat.ac.in

[2]Professor, Department of Artificial Intelligence and Data Science (AI&DS), Global Academy of Technology, Bengaluru, 560098, Karnataka, India
girishraosalanke@gat.ac.in

[3]Professor, Department of Artificial Intelligence and Machine Learning (AI&ML), CMR University, Bengaluru, 562149, Karnataka, India
manjunath.c@cmr.edu.in

[4]Professor, Department of Artificial Intelligence and Data Science (AI&DS), KSSEM, Bengaluru, 560109, Karnataka, India
manjunathtk@kssem.edu.in

Abstract— The safety of construction workers can be significantly enhanced by wearing safety helmets. Workers frequently remove their helmets due to a lack of knowledge and discomfort, exposing them to hidden dangers. Accidents involving falling objects from heights put workers who are not wearing safety helmets at increased risk. Therefore, there is a critical need for a fast and accurate safety helmet detector to oversee construction site safety. However, standard manual monitoring is time-consuming and labor-intensive, and solutions that include attaching sensors to helmets are not generally used. To address this issue, this study presents a Deep Learning (DL)--based technique for accurately and rapidly detecting safety helmets. The data for training the DL models is sourced from the Kaggle site and goes through preprocessing stages such as image improvement and resizing. Helmet detection utilizes DL models such as You Only Look Once (YOLO), Single Shot MultiBox Detector (SSD), and Region-based Convolutional Neural Network (RCNN). The experimental results show that the YOLO model has the highest accuracy, precision, recall, and mean average precision (mAP), at 94.7%, 93.8%, 94.2%, and 94.62%, respectively. These findings highlight the effectiveness of YOLO in real-time helmet identification, greatly contributing to accident avoidance on construction sites.

Keywords—Safety Helmet, Deep Learning, Object Detection, Computer Vision, Kaggle, Construction Site, Accuracy.

Introduction

The impact of unsafe working conditions on productivity, as well as the subsequent labor exodus, has made workplace safety a prominent concern across various industries. Every year, many Americans die due to the hazardous working conditions that much of the labor force must endure. In 2012, 4,383 people lost their lives due to occupational injuries in the United States

(US), averaging 89 deaths/week and over 12 deaths/day [1]. Construction workers are frequently injured due to the industry's dangerous nature. In 2014, the construction industry accounted for one out of every five private sector employee fatalities, making it the deadliest industry in the American economy. Between 2015 and 2018, inadequate helmet use contributed to 53 construction accidents (67.95% of all accidents) reported to the State Office of work safety. In many developing countries, the death rate is higher than in industrialized countries. For example, compared to the US, the Republic of Korea's construction industry fatality rate is more than twice as high. The greater frequency of construction deaths in developing countries is a cause for concern among construction managers. Based on the International Labour Organization report, the construction site has the maximum workplace accident rate in any industry [2]. Workers in the construction industry typically work in hazardous conditions and risk their lives to perform high-risk jobs. The workers safety must be a key consideration. Monitoring the use of protective equipment is part of the construction site safety management process. The majority of workplace falls involve workers injuring their heads on hard floors after falling from significant heights [3]. The primary function of safety helmets is to protect workers from harm in the event of a fall. Properly worn hard helmets can prevent half of all fatalities caused by falls, as well as a significant rate of deaths. According to previous research, a safety helmet can eliminate the risk of serious brain damage by up to 95% [4].

Numerous countries have implemented industrial safety measures to ensure the well-being of their workers. It is impractical to personally track individuals who do not wear safety helmets, especially on large construction sites [5]. Therefore, it is critical to have a system that can automatically detect when a safety helmet is worn. Computer Vision (CV) and DL can address the object detection challenge in automatic helmet detection [6, 7]. DL's computational approach and precision in object detection have revolutionized CV. In this study, we used images of construction workers to develop and evaluate three DL models: SSD, YOLO, and RCNN, all aimed at recognizing helmets.

Literature Survey

Several studies have been conducted to determine the effectiveness of construction workers' safety helmets. This review on safety helmet recognition covers two key approaches: Machine Learning (ML) and DL detection. A study [8] discusses one of many deep surveillance applications that monitor people who ride motorcycles without helmets or with more than one passenger. To manage overloaded traffic, an SSD was implemented, and multiscale features were extracted using transfer learning. To further enhance detection accuracy, the suggested model integrates aspect ratio awareness training during the following fine-tuning process. A real-world dataset collected from surveillance cameras was utilized to evaluate the model. This dataset encompassed diverse viewpoints, temperatures, and population densities on the streets. At 57 frames per second (FPS), the method successfully classifies items in a dataset, achieving a high average mean precision across four classes. By comparing the proposed methodology to current methods that use experimental data, it is found to be quicker and more accurate. Using a transfer learning technique, the study [9] trained two SSD models to recognize industrial safety helmets: one with ResNet50 and the other with MobileNetV2. The training was done with a dataset available on the Kaggle website. Regularization, localization, classification, and total loss were some of the metrics used to assess the models. When looking at the loss parameters, SSD ResNet50 was outperformed by SSD MobileNetV2 in classification, regularization, and localization.

The study [10] introduces Improved Boosted Random Ferns (IBRFs) as a new method for detecting the wear state of safety helmets. IBRFs are based on the Boosted Random Ferns method and use a weighted coefficient to boost performance. To build the feature domain space of an image, the Oriented Gradient Histogram is used for feature extraction. Next, the feature domain space is seeded with ferns using the random binary testing approach. The next step is to generate a subpar classifier by randomly selecting ferns. Ultimately, an improved Real AdaBoost algorithm is used to produce IBRFs from the best of these. Experimental validation on a larger public safety helmet dataset reveals that IBRFs outperform current sophisticated detection techniques. Research [11] presents a technique for optimizing the BottleneckCSP structure, which can drastically reduce model complexity without changing the network's input and output sizes. This work creates an upsampling feature enhancement element to enrich the semantic richness of the feature map while preventing information loss during upsampling. Furthermore, this research proposes a self-attention strategy to reduce the impact of duplicated information in feature fusion on detection results. New feature maps with robust semantics and accurate location information are adaptively created by fusing neighboring shallow feature maps with upsampled feature maps using the channel and location attention elements, respectively. Under the same computational restrictions as the leading algorithms, the proposed methodology surpasses them in terms of inference speed and mAP performance.

Researchers [12] created a revolutionary automated method that uses a lightweight CNN model to detect whether everyone on a site is wearing a helmet. The feature extraction network is based on GhostNet. During feature processing, a network capable of combining features and performing multi-scale segmentation was built. The feature fusion network architecture improves detection accuracy by diversifying helmet features. The LRCA-Net, a lighter and more efficient version of the attention mechanism, was also introduced. When evaluated on a dataset, the recommended lightweight safety helmet detection system achieves outstanding mAP and FPS performance. The paper [13] proposes a method for detecting helmet usage using the EfficientDet approach. The initial cluster centers were optimized using the K-Means++ clustering method. Image feature maps were obtained by employing the SeparableConv2D network from the EfficientDet model, along with the straightforward and effective Bi-directional Feature Pyramid Network (BiFPN). The Channel-wise Class Loss function was utilized to improve model detection accuracy. Experimental outcome on a publicly accessible helmet dataset shows that the enhanced EfficientDet model better fulfills the requirements for recognizing safety helmet usage compared to existing methods.

DL Models

Three DL models were employed in this research for helmet detection from images. The working and architecture of all three models are detailed in this section.

YOLO

The architecture of YOLO was inspired by Google's Neural Network [14]. The network was trained by the Darknet model and tested for item detection on the VOC Pascal Dataset. Here, (1 × 1) convolutional filters are used instead of GoogLeNet's inception modules, followed by (3 × 3) filters. The only exception is the first Convolutional Layer (CL), which uses a (7 × 7) filter. Figure 1 shows the YOLO architecture, which includes 24-CLs, 2-Fully Connected Layers (FCL), and other components. Only four out of the twenty-four CLs have max Pooling Layers (PL) after them. The technique's application of PL and (1 × 1) convolution are its standout features. The initial twenty layers, followed by an average PL and an FCL, were trained and fine-tuned using the ImageNet 2012 dataset. The authors devoted around one week to this process. Subsequently, 4-CLs and 2-FCLs with random weights were used to fine-tune the model for the helmet detection task. All convolutional and dense layers employ the

Leaky Rectified Linear Unit, except for the last layer, which employs a linear function. This final layer is responsible for predicting class probabilities as well as the placements of the bounding box (BB). Some have noted that this YOLO version is inaccurate in its localization and has lower recall than two-stage object detectors [15]. The loss function, given by Equation (1), consistently makes use of the sum of squared errors. It is clear that the underlying equation contains four terms that may be stated using the following notations:$(\hat{x}_i, \hat{y}_i)$, and (x_i, y_i) represents the estimated and the Ground Truth (GT) of the BB's center, $(\widehat{w}_i, \widehat{h}_i)$, and (w_i, h_i) indicates the estimated and GT of the BB's width and height.

The first two terms are linked with errors caused by differences between the actual and expected locations of BB. When dealing with smaller BBs, deviations have a greater impact on IoU than when working with larger ones. To address this issue, the loss function uses the square root of the BB's width and height. The third term represents the difference in confidence scores between predictions made before and after the object's existence in each BB. In the first three formulas, 1_{ij}^{obj} represents the i^{th} grid that forecasts the j^{th} BB. If the cell contains the object, it will be 1; otherwise, it will be 0. The object may be present in a specific grid cell according to GT, yet the model may falsely claim that there is no object there. The attempt is to reduce the loss not only while the object is in the grid cell but also when it is not. Objects in specific grid cells may not exist in the real world, but the model may mistakenly show that they do. Similarly, the indicator function 1_{ij}^{noobj}represents the i^{th} grid that forecasts the j^{th} BB. If the cell is empty, it will be 1; if it contains the object, it will be 0. The final term, classification loss, aims to reduce misclassification errors. The indicator function 1_{i}^{obj} indicates a grid cell with an object, returning 1 if present and 0 otherwise. To prevent gradient divergence, the hyperparameters λ_{coord} and λ_{noobj} are commonly used. If no grid cell includes the object, the confidence score and following gradients will converge to zero. The confidence score and subsequent gradients will converge to 0 if the object is not present in any grid cell. To address this issue, we increase the hyperparameter λ_{coord} in the first two terms to optimize the loss of bounding box coordinates when the grid cell contains the object. Similarly, in the fourth term, we raise the hyperparameter λ_{noobj} to reduce the loss when the grid cell does not contain the object. Common practice is to set λ_{coord} to a high value and λ_{noobj} to a low value.$\lambda_{coord}\sum_{i=0}^{S^2}\sum_{j=0}^{B}1_{ij}^{noobj}[(x_i-\hat{x}_l)^2+(y_i-\hat{y}_l)^2]+\lambda_{coord}\sum_{i=0}^{S^2}\sum_{j=0}^{B}1_{ij}^{noobj}\left[\left(\sqrt{w_i}-\sqrt{\widehat{w}_l}\right)^2+\left(\sqrt{h_i}-\sqrt{\hat{h}_l}\right)^2\right]+loss=\sum_{i=0}^{S^2}\sum_{j=0}^{B}1_{ij}^{obj}\left[\left(C_i-\hat{C}_l\right)^2\right]+$

$\lambda_{noobj}\sum_{i=0}^{S^2}\sum_{j=0}^{B}1_{ij}^{noobj}\left[\left(C_i-\hat{C}_l\right)^2\right]+\sum_{i=0}^{S^2}1_{i}^{obj}\sum_{c\epsilon classes}\left(p_i(c)-\hat{p}_i(c)\right)^2$ [1]

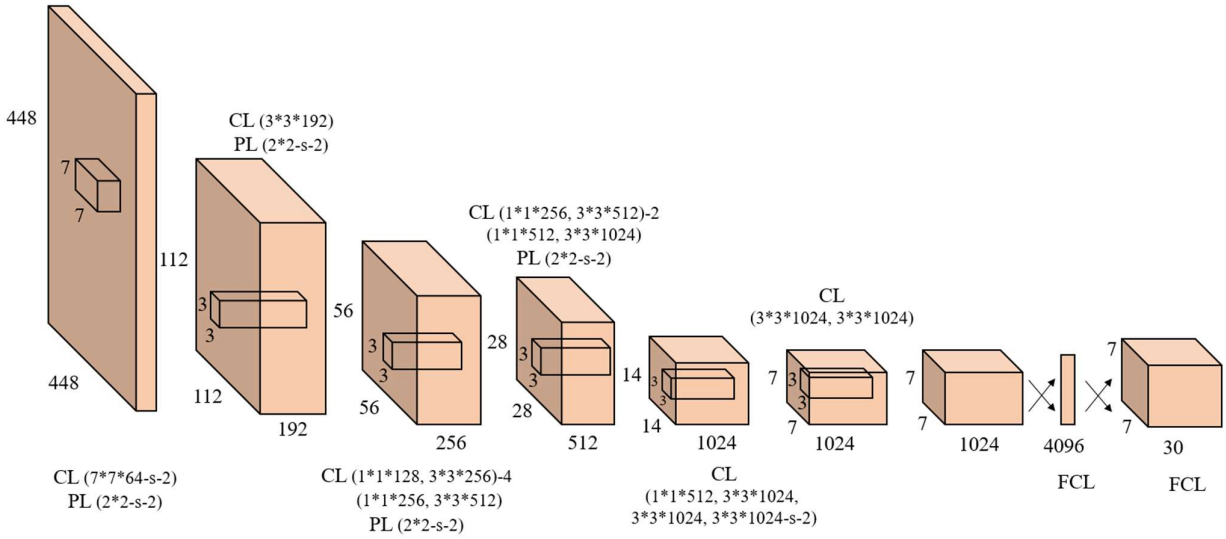


Fig. 1. YOLO architectureSSD

To begin with, the SSD method constructs a set of bounding boxes (BB) of a fixed size and assigns scores to each based on the number of instances of the given object class it detects inside [16]. This set of boxes is then inputted into a feed-forward convolutional network. A non-maximum suppression (NMS) is employed to generate the predicted outcome. The base network is commonly used for high-quality image classification. Subsequently, the network is augmented with an auxiliary structure to create detections that incorporate the following key criteria.

Convolutional feature layers are included in the terminated base network. These layers allow for detection predictions at various scales by gradually diminishing in size. For detection prediction, the convolutional model varies across feature layers. Each additional feature layer utilizes a predefined set of convolutional filters to generate detection predictions. In a feature layer with $m \times n$ channels, these predictions rely on a $3 \times 3 \times p$ small kernel. This kernel outputs either a shape offset relative to the default box coordinates or a category score at each of the $m \times n$ locations where it is applied. After determining the default box location, the BB offset output values are then added to each point on the feature map. When there are numerous feature maps, the network's top level assigns default BB to each feature map cell. Because the feature map is tiled in a convolutional fashion, the default boxes are fixedly positioned relative to their corresponding cells. Offsets are anticipated with regard to the cell's default box shapes, as well as the per-class scores, which indicates whether or not each box contains an instance of a specific class at each feature map cell.

Unlike typical detectors that rely on region proposals, SSD training involves assigning GT data to specific outputs within a predefined set of detector outputs. This distinction sets SSD apart from traditional detectors. Before training the network, it is crucial to determine which default boxes correspond to the GT detections. Default boxes vary in size, aspect ratio, and location, and one is selected for each GT box. The SSD training objective builds on the MultiBox objective by handling several object categories. To match the i^{th} default box with the j^{th} GT box of category p, use the indicator $x_{ij}^{p} = \{1,0\}$. The previously mentioned matching strategy allows for $\sum_i x_{ij}^{p} \geq 1$. The objective loss function is calculated by combining the confidence loss $(conf)$ and localization loss (loc).

$$L(x, c, l. g) = \frac{1}{N}\Big(L_{conf}(x, c) + \alpha L_{loc}(x, l, g)\Big) \quad [2]$$

Here, N represents the sum of all matching default boxes. When $N = 0$, the loss is zero. Localization loss is the Smooth $L1$ loss when comparing the features of the GT box (g) with those of the predicted box (l). Offsets for the center (cx, cy), height (h), and width (w) of the default BB are regressed.

$$L_{loc}(x, l, g) = \sum_{i\in Pos}^{N}\sum_{m\in\{cx,cy,w,h\}} x_{ij}^{k} smooth_{L1}\left(l_i^m - \hat{g}_j^m\right) \quad [3]$$

The confidence loss is estimated by applying the softmax function to the confidences (c) of several classes.

$$L_{conf}(x, c) = -\sum_{i\in Pos}^{N} x_{ij}^{p} log\left(\hat{c}_i^p\right) - \sum_{i\in Neg} log(\hat{c}_i^0) \quad [4]$$

Where,

$$\hat{c}_i^p = \frac{exp(c_i^p)}{\sum_p exp(c_i^p)} \quad [5]$$

RCNN

The Region-based CNN, or R-CNN, is a type of DL architecture used for CV tasks that require object detection. R-CNN was an early model that helped advance object detection by combining the strengths of region-based methods with CNNs [17]. Now, let's take a deeper look at R-CNN to see how it works.

Region Proposal: R-CNN begins by partitioning the input image into smaller pieces. These locations are referred to as "region proposals" and "region candidates." The region recommendation phase is responsible for generating a list of prospective image regions that may contain objects. Rather than relying on its internal algorithms, R-CNN generates region proposals using third-party methodologies such as Selective Search or Edge Boxes. For example, Selective Search uses image cues such as color, texture, and shape to merge or separate image components to provide a diverse set of region suggestions. The phases of Selective Search are demonstrated below, starting with an input image and progressing through an image with several segmented masks, a reduced collection of masks, and finally the masks that comprise the image's key features.

Extraction of Features: To train a CNN to extract features, region proposals are first generated, then around 2,000 areas are extracted and warped to a consistent input size (224x224 pixels). To get 16 pixels of context in the warped picture, the region size is increased to a new size before warping. For generic feature representation, the AlexNet is used and it is fine-tuned on a large dataset such as ImageNet. The CNN generates a high-dimensional feature vector that represents the content of the suggested region.

Classifying Objects: To process the feature vectors obtained from the region proposals, each object type of interest is assigned its own ML classifier. When it comes to classification, R-CNN typically uses ML. To determine if the proposed region contains a member of each class, a separate support vector machine (SVM) is trained for each. Positive samples during training are regions that contain at least one instance of the class. Areas that do not are considered negative samples.

Bounded Box Regression: R-CNN supports BB regression in addition to object classification. For each class, several regression models are built to fine-tune the BB position and size of the detected object. BB regression improves object localization accuracy by altering the initially suggested BB to better reflect the object's actual boundaries.

NMS: To eliminate the duplicate or very overlapping BB, R-CNN employs NMS after classifying and regressing BB for each area proposal. NMS ensures that only the highest confidence and non-overlapping BB are used for final object detections.

Results and Discussion

The experiment was conducted in a Jupyter Notebook using the Python programming language. For safety helmet detection, publicly available data from the internet was collected and processed. The processed data was split and fed into DL models like YOLO, SSD, and RCNN for detecting helmets and heads. The performance of all three models was evaluated using metrics such as accuracy, precision, recall, and mAP. Finally, the best model for helmet detection from real-time images of construction sites was identified. The entire methodology is outlined in Figure 2.

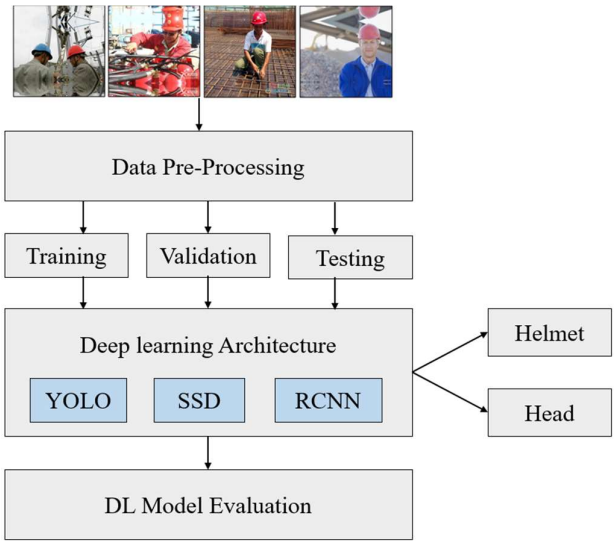


Fig. 2. Helmet detection methodology

Data Collection and Processing

For safety helmet detection, we collected data from the Kaggle site [15]. The dataset contains 5000 real-world images annotated with three classes: head, helmet, and person. We used annotations for head and helmet in our research. Sample images from the Kaggle are given in Figure 3. The dataset is further split into three sections: training, validation, and test, in the ratio of 7:2:1. Some images in the dataset are not clear, so we employed image enhancement techniques. Specifically, gamma correction was used for image enhancement. Next, we applied three different DL models for helmet detection. The input neurons of the algorithms vary, and based on each algorithm, the images are resized accordingly.

Fig. 3. Sample images from Kaggle dataset

Experimental Outcome

The effectiveness of the DL algorithm is measured using four performance measures and it can be determined through the confusion matrix parameters. True Positive (TP) denotes the right estimation of a human with a helmet, False Positive (FP) the wrong forecast of a human with a helmet, True Negative (TN) represents the right estimation of a human without a helmet, and False Negative (FN) the wrong estimation of a human without a helmet. Table 1 gives the metrics formula and the values achieved by each algorithm.

The YOLO model achieved the highest accuracy among all models, reaching 94.7%. The SSD model followed closely with an accuracy of 93.6%. In terms of precision, YOLO also performed the best with a score of 93.8%, slightly higher than SSD's 92.78%. For recall, YOLO and RCNN obtained the highest values of 94.2% and 90.5% respectively. YOLO also excelled in mAP with a score of 94.62%.

Table 1. Performance evaluation of DL model on helmet detection from construction site imagesModel	Accuracy (%)	Precision (%)	Recall (%)	mAP (%)
Formula	$\frac{TP+TN}{TP+TN+FP+FN}$	$\frac{TP}{TP+FP}$	$\frac{TP}{TP+FN}$	$\frac{\sum_{c=1}^{2} Average\ Precision\ (c)}{2}$
YOLO	94.7	93.8	94.2	94.62
RCNN	92.7	91.04	90.5	92.2
SSD	93.6	92.78	92	93.8

A comparative analysis of all DL models for helmet detection using the test data is presented in Figure 4. Different colors are used to distinguish between the outcomes of each DL model. The figure clearly illustrates YOLO's superior performance across all metrics.

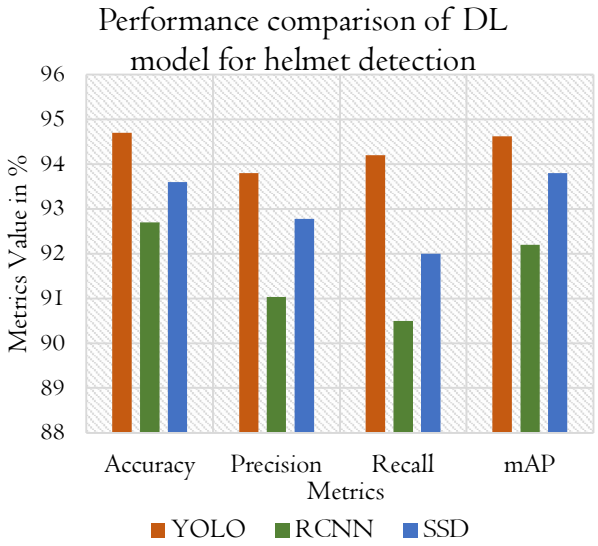
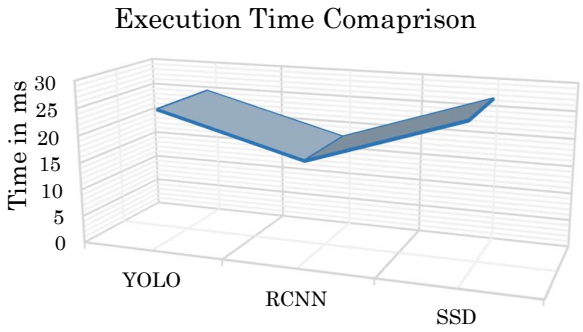


Fig. 4. DL model performance comparison on safety helmet detection

Apart from performance metrics, the execution time required for helmet identification was evaluated and it is given in Figure 5. RCNN achieved the shortest time of 18 seconds, while YOLO required a maximum of 25 seconds. RCNN performed better in terms of execution time, whereas YOLO excelled in accuracy. Considering the critical importance of detection accuracy in a construction site setting, YOLO was chosen as the preferred model.



	YOLO	RCNN	SSD
Time	25	18	27

Fig. 5. Execution time comparison of DL model

Figure 6 depicts the outcome of the YOLO model in detecting heads and helmets from images. The results demonstrate YOLO's ability to accurately identify individuals wearing helmets. Based on these results, YOLO is a good choice for applications that need accurate and reliable object recognition in the real world, such as safety helmet detection.

Fig. 6. Safety helmet detection by YOLO model

Conclusion

The present research explores helmet recognition as a CV challenge and proposes a DL-based solution. Worker safety is paramount on construction sites. Existing research has struggled with distinguishing items in poor-light conditions and smaller objects. To enhance worker safety, a DL-based architecture for automatically detecting safety helmets on construction sites was developed and compared. In this study, we employed the RCNN model, SSD, and YOLO for helmet identification. All three models were evaluated using performance measures and execution time. YOLO emerged as the top performer, achieving an excellent helmet detection accuracy of 94.7%. Following YOLO, SSD achieved the next highest accuracy of 93.6%. YOLO required 25 seconds for helmet prediction from images. These results indicate that the YOLO model is suitable for real-time implementation, enhancing worker safety without human intervention. In the future, the deployment of the designed YOLO model at construction sites is planned. The DL model will be integrated with advanced technologies such as Field-Programmable Gate Array (FPGA) and Internet of Things (IoT) to transmit data, including images and alert messages, to construction supervisors. FPGA will handle powerful computational tasks and memory management efficiently.

References

[1] Arrow, Carlos Robert. *An action research study focused on training programs to reduce injuries and fatalities in high risk industries*. Colorado Technical University, 2015.

[2] Yilmaz, Fatih, and Ugur Bugra Çelebi. "The importance of safety in construction sector: Costs of occupational accidents in construction sites." *Business and Economics Research Journal* 6, no. 2 (2015): 25.

[3] Kodithuwakku Arachchige, Sachini NK, Harish Chander, Adam C. Knight, Reuben F. Burch V, and Daniel W. Carruth. "Occupational falls: Interventions for fall detection, prevention and safety promotion." *Theoretical Issues in Ergonomics Science* 22, no. 5 (2021): 603-618.

[4] Sone, Je Yeong, Douglas Kondziolka, Jason H. Huang, and Uzma Samadani. "Helmet efficacy against concussion and traumatic brain injury: a review." *Journal of neurosurgery* 126, no. 3 (2017): 768-781.

[5] Li, Heng, Xiaoying Li, Xiaochun Luo, and Joanna Siebert. "Investigation of the causality patterns of non-helmet use behavior of construction workers." *Automation in Construction* 80 (2017): 95-103.

[6] Korkmaz, Adem, and Mehmet Tevfik Ağdaş. "Deep Learning-Based Automatic Helmet Detection System in Construction Site Cameras." *Bitlis Eren Üniversitesi Fen Bilimleri Dergisi* 12, no. 3 (2023): 773-782.

[7] Li, Jun, Yuanyuan Li, Jocelyn F. Villaverde, Xiaoji Chen, and Xiaozhi Zhang. "A safety wearing helmet detection method using deep leaning approach." *Journal of Optics* 53, no. 2 (2024): 1163-1169.

[8] Nandhini, C., and M. Brindha. "Transfer learning based SSD model for helmet and multiple rider detection." *International Journal of Information Technology* 15, no. 2 (2023): 565-576.

[9] Umair, Muhammad, and Yee-Loo Foo. "Industrial Safety Helmet Detection Using Single Shot Detectors Models and Transfer Learning." In *Multimedia University Engineering Conference (MECON 2022)*, pp. 390-400. Atlantis Press, 2022.

[10] Yue, Shiqin, Qian Zhang, Dingqin Shao, Yu Fan, and Jinhua Bai. "Safety helmet wearing status detection based on improved boosted random ferns." *Multimedia Tools and Applications* 81, no. 12 (2022): 16783-16796.

[11] Qian, YueJing, and Bo Wang. "A new method for safety helmet detection based on convolutional neural network." *PLoS one* 18, no. 10 (2023): e0292970.

[12] Liang, Han, and Suyoung Seo. "Automatic detection of construction workers' helmet wear based on lightweight deep learning." *Applied Sciences* 12, no. 20 (2022): 10369.

[13] Fan, Xinlei, Fan Wang, Shuai Pang, Jiaxing Wang, and Wenjing Wang. "Safety helmet wearing detection based on EfficientDet algorithm." In *2nd International Conference on Artificial Intelligence, Automation, and High-Performance Computing (AIAHPC 2022)*, vol. 12348, pp. 305-312. SPIE, 2022.

[14] Redmon, Joseph, Santosh Divvala, Ross Girshick, and Ali Farhadi. "You only look once: Unified, real-time object detection." In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 779-788. 2016.

[15] Liu, Chengji, Yufan Tao, Jiawei Liang, Kai Li, and Yihang Chen. "Object detection based on YOLO network." In *2018 IEEE 4th information technology and mechatronics engineering conference (ITOEC)*, pp. 799-803. IEEE, 2018.

[16] Liu, Wei, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C. Berg. "Ssd: Single shot multibox detector." In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part I 14*, pp. 21-37. Springer International Publishing, 2016.

[17] Chen, Chenyi, Ming-Yu Liu, Oncel Tuzel, and Jianxiong Xiao. "R-CNN for small object detection." In *Computer Vision–ACCV 2016: 13th Asian Conference on Computer Vision, Taipei, Taiwan, November 20-24, 2016, Revised Selected Papers, Part V 13*, pp. 214-230. Springer International Publishing, 2017.