Construction Innovation: Information, Process, Management

A deep learning-based approach to facilitate the as-built state recognition of indoor construction works

Journal:	*Construction Innovation: Information, Process, Management*
Manuscript ID	CI-05-2022-0121.R1
Manuscript Type:	Research Article
Keywords:	As-built state, Indoor construction progress monitoring, Deep learning, Google Colab, Virtual machine, YOLOv4

A deep learning-based approach to facilitate the as-built state recognition of indoor construction works

Abstract

Purpose – Recognising the as-built state of construction elements is crucial for construction progress monitoring. Construction scholars have used computer vision-based algorithms to automate this process. Robust object recognition from indoor site images has been inhibited by technical challenges related to indoor objects, lighting conditions and camera positioning. Compared to traditional machine learning algorithms, one-stage detector deep learning (DL) algorithms can prioritise the inference speed, enable real-time accurate object detection and classification. Therefore, this study presents a DL-based approach to facilitate the as-built state recognition of indoor construction works.

Design/methodology/approach - The one-stage DL-based approach was built upon YOLO version 4 (YOLOv4) algorithm using transfer learning with few hyperparameters customised and trained in the Google Colab virtual machine. The process of framing, insulation, and drywall installation of indoor partitions was selected as the as-built scenario. For training, images were captured from two indoor sites with publicly available online images.

Findings - The DL model reported a best trained weight with a mean average precision of 92% and an average loss of 0.83. Compared to previous studies, the automation level of this study is high due to the use of fixed time-lapse cameras for data collection and zero manual intervention from the pre-processing algorithms to enhance visual quality of indoor images.

Originality – This study extends the application of DL models for recognising as-built state of indoor construction works upon providing training images. Presenting a workflow on training DL models in a virtual machine platform by reducing the computational complexities associated with DL models is also materialised.

Keywords As-built state; Indoor construction progress monitoring; Deep learning; Google Colab; Virtual machine; YOLOv4

Paper type Research paper

Introduction

Recognising the as-built state is crucial for monitoring the progress of construction works (Hamledari *et al.*, 2017) for the purposes of calculating progress payments, determining deviations from the baseline program and taking remedial actions to address any budgetary and delay issues (Kropp *et al.*, 2014). Traditional methods of as-built state recognition typically involve manual site inspections by different construction personnel utilising visual subjective assessments that produce approximate rather than precise results (Golparvar-Fard *et al.*, 2015). These traditional practices are labour intensive, time-consuming, costly, and generally lacking in precise accuracy (Bosché, 2012; Golparvar-Fard *et al*., 2015; Yang *et al*., 2015). Automated visual recognition of the as-built state of construction elements entails object detection and their state classification (Ekanayake *et al*., 2021a; Guven and Ergen, 2021). By employing computer vision (CV)-based technologies of utilising cameras to capture images and machine learning (ML) algorithms to process images, automated visual recognition can provide more precise information about the as-built state (Ekanayake *et al*., 2021b).

Existing CV-based studies predominantly focus on exterior construction elements with relatively few studies on indoor construction (Deng *et al*., 2020; Hamledari and McCabe, 2016; Kropp *et al*., 2014). Kopsida et al. (2015) contend that many schedule delays and budget overruns in indoor construction projects are triggered by misunderstanding of the details and complexities of the indoor elements. Recognising the as-built state of construction elements is challenging in the indoor environment because of obstructions, cluttered indoor environments, illumination changes and the achromatic appearance of most indoor components (Deng *et al*., 2020; Hamledari and McCabe, 2016; Kropp *et al*., 2014). These challenges have been categorised as technical challenges related to indoor objects, lighting conditions and camera positioning (Ekanayake *et al*., 2021a; Ekanayake *et al*., 2021b).

Pioneering CV-based indoor construction progress monitoring studies have employed traditional ML algorithms, which use manual feature extraction to determine the as-built state of indoor construction elements (Hamledari *et al*., 2017; Kropp *et al*., 2014). The algorithms relying on manually extracting features such as edges, colour and texture for object detection and classification are sensitive to the visual quality of the input images and are difficult to be extended to new image datasets with different visual conditions (Ying and Lee, 2019). As a result of the technical challenges, the region of interest (ROI) in the images cannot be detected easily without initially performing pre-processing algorithms to remove background noise and lighting impacts (Ekanayake *et al*., 2021a).

Wang et al. (2021) note that the advances in CV have led to the use of deep learning (DL), which is a branch of ML to improve automation. DL models automatically learn features by training large amount of data under supervised learning (Nanni *et al*., 2017). This self-training ability of DL models enables the use of a single object recognition algorithm to detect and classify objects, without conducting additional steps of pre-processing (Ying and Lee, 2019). Therefore, the DL models not only improve automation but also reduce the inaccuracies caused by biases of the programmers in manual feature extraction (Slaton *et al*., 2020). Despite the efficiency and accuracy of DL, DL models have not been widely used to resolve issues related to real-time indoor elements as-built state recognition. This is mainly due to the high computing resource requirement and training configuration difficulties associated with DL models (O'Mahony *et al.*, 2019).

This paper presents a DL-based approach to facilitate the as-built state recognition of indoor construction works. It is a one-stage detector DL approach, which was built upon the YOLOv4 model. YOLOv4 is highly efficient and accurate in real-time object detection and classification (Bochkovskiy *et al.*, 2020). The framing, insulation, and drywall installation process of indoor partitions was used to demonstrate the DL model. Indoor site images from this as-built process were captured to train and test the model. The onerous process of building DL models from scratch and training them using high computational resources outweigh their anticipated benefits. To address the computational complexities of building these model from scratch, transfer learning (Pan and Yang, 2009) was employed on a pre-trained YOLOv4 model with few hyperparameters customised. Then the model was trained on a cloud enabled virtual machine (VM) runtime using Google Colab (Google Research, 2022) to reduce the computational resource requirements and to enable sharing among project stakeholders. The main objective of this study is to present an efficient, accurate and readily shareable DL-based approach to facilitate the as-built state recognition of indoor construction works. This paper commences with a literature review on CV, DL and ML approaches followed by a description of the research methods used. The development of the DL-based object recognition approach is then described in detail and discussed. The paper culminates with a summary of the key findings and recommendations on future research directions.

Literature review

The literature review section explains how construction elements recognition has advanced from using traditional ML algorithms to DL models. Followed by a discussion on the mechanism behind deep neural networks, this section further highlights the role of VM technology in reducing the training complexities associated with DL models.

Construction elements recognition using traditional ML algorithms

Traditional ML algorithms administer object detection and classification by manual feature extraction. This is also referred to as handcrafted feature extraction, which involves the programmer designing the specific features to be extracted (O'Mahony *et al.*, 2019). This can be further explained by a feature extraction algorithm such as the Canny edge detector (Canny, 1986). The programmer must manually design how to extract the edges (Nanni *et al.*, 2017). As the number of classes to detect increases, feature extraction becomes inefficient (O'Mahony *et al.*, 2019). CV-based indoor construction elements recognition studies such as those conducted by Kropp et al. (2014); Kropp et al. (2018); Hamledari and McCabe (2016); and Hamledari et al. (2017) have employed traditional ML algorithms to determine the as-built state of indoor construction elements.

A key difficulty with the traditional approach is that a significant level of algorithmic pre-processing is required to remove background noise (i.e. unnecessary data) and enhance visual quality in images (Razavi *et al.*, 2008). Lighting variation related pre-processing is usually done using low-light image enhancement (LIME) algorithms to enhance the images captured in environments with low natural lighting (Guo *et al.*, 2016). For noise smoothing due to cluttered scenes and background objects, background subtraction techniques such as frame differencing are employed (Kartika and Mohamed, 2011). As a result, instead of employing a single object recognition algorithm to detect and classify objects, handcrafted feature extraction requires conducting multiple steps of pre-processing to make the ROI easily detectable (Ying and Lee, 2019).

The mechanism behind deep neural networks

The use of DL, which is a branch of ML, for construction progress monitoring has been advancing rapidly (Martinez *et al.*, 2019). DL models incorporate deep neural networks, which leverage input-to-target mapping through a deep neural network to extract features from input data (Chollet, 2017; LeCun *et al.*, 2015). Object recognition using DL aims at locating, classifying, and detecting objects in the images and labelling them with rectangular bounding boxes to show the confidence score of existence (Chollet, 2017). The convolution neural networks (CNNs) are the widespread type of DL neural networks used for image processing (Chollet, 2017). Figure 1 illustrates the difference in mechanism behind traditional ML and DL models in detecting a framing instance in an image by using a Canny edge detector and a CNN respectively.

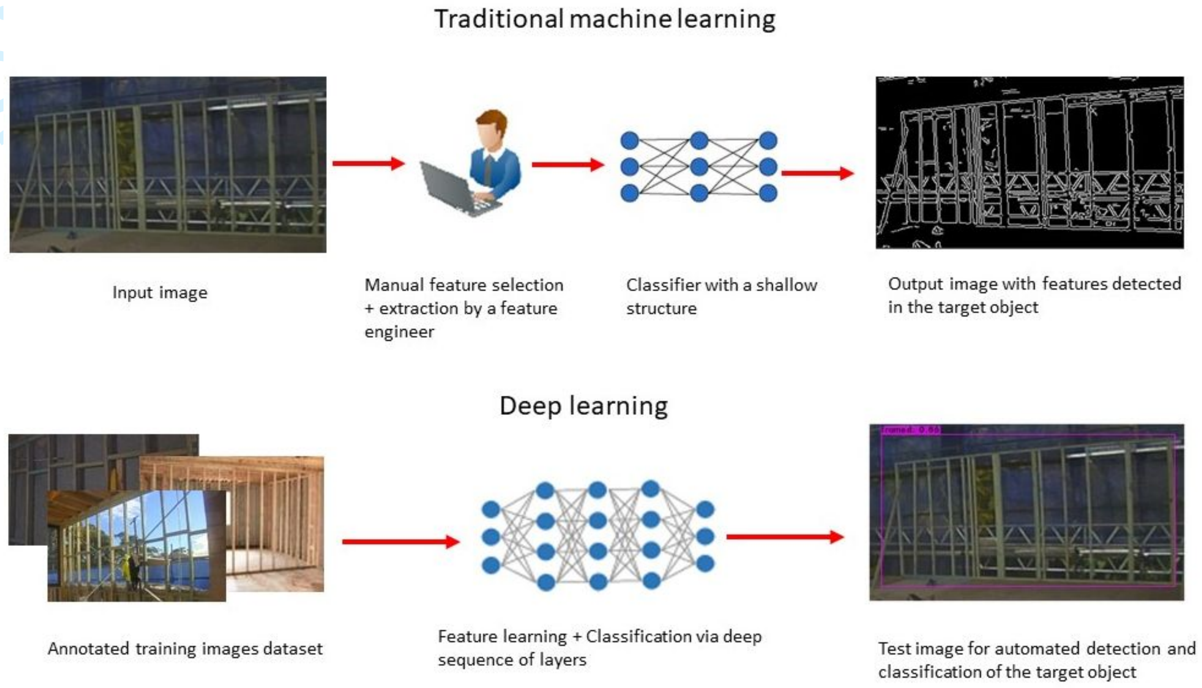

Figure 1: Mechanism behind (a) traditional machine learning (Canny edge detector); (b) deep learning (CNN)

As illustrated in Figure 1, when using a traditional ML algorithm such as Canny edge detector, the programmer selects and extracts edges as the feature. Conversely, a typical CNN structure is composed of a deep sequence of input layer, convolutional layers, pooling layers, fully connected layers, and an output layer. Compared to traditional ML algorithms, CNNs can achieve better detection and classification accuracy on large image datasets due to the ability of joint feature and classifier learning from training images (Chollet, 2017; LeCun *et al.*, 2015).

There are two types of CNN object recognition frameworks. The two-stage detectors generate region proposals initially and then classify each proposal into different object categories (Kardovskyi and Moon, 2021). Region-based convolutional neural networks (R-CNN) belong to this category (Zhao *et al.*, 2019). In CNNs such as Fast R-CNN, Mask R-CNN, object detection is complex and slow because of the initial region proposals to predict the ROI (Bochkovskiy *et al.*, 2020). In one-stage detectors, object detection is treated as a regression/classification problem. Regression predicts classes and bounding boxes for the whole image in a single run and identifies the object's position in an image. Classification establishes the object's class (Redmon *et al.*, 2016; Zhao *et al.*, 2019). Two examples are You Only Look Once (YOLO) (Redmon *et al.*, 2016) and Single Shot Multi-Box Detector (Liu *et* *al*., 2016). As a result of this neural network operation, the inference speed is high for accurate real-time object detection and classification in one-stage detectors (Bochkovskiy *et al*., 2020).

The application of DL models for construction progress monitoring has gained momentum in recent years. Examples include rebar counting using YOLO (Li *et al*., 2021) and pre-cast walls installation monitoring using Mask R-CNN (Wang *et al*., 2021). However, these models largely focus on construction works that can be viewed externally and studies incorporating indoor progress monitoring are limited. In recent CV-based indoor construction progress monitoring studies, Mask R-CNN models have been applied to recognise the building objects (walls, doors, and lifts) (Ying and Lee, 2019) and HVAC ducts (Shamsollahi *et al*., 2021). Mask R-CNN has also been employed for calculating the work-in-progress of brick layering and plastering of an indoor wall (Wei *et al*., 2022). These applications have been gaining recognition because of improved automation and reduced inaccuracies compared to traditional ML counterparts.

Virtual machines to train DL models

DL models perform far better than traditional ML algorithms, albeit with trade-offs related to computing requirements and training time (O'Mahony *et al*., 2019; Wang *et al*., 2021). It is essential to build an extensive training image database with annotations for supervised DL models implementation. DL models training requires high level hardware resources such as graphic processing units (GPUs), high performing memory, processor, and storage (O'Mahony *et al*., 2019; Wang *et al*., 2021). In addition, computing platforms such as compute unified device architecture (CUDA), and libraries including CUDA based deep neural networks (cuDNN) should be installed for GPU enabled DL execution (Jian *et al*., 2013; Jorda *et al*., 2019). Without a proper training platform, DL models training on datasets of thousands of images could take days (Carneiro *et al*., 2018). Advances in computing technology have facilitated the use of GPU-enabled gaming computers and edge computing devices for training DL models (Pal and Hsieh, 2021). Despite these advancements, the hardware requirements are still expensive, and the configurations needed for training DL models are complicated and time consuming.

With the proliferation of cloud computing and virtualisation, leading technology companies have provided dedicated development environments to overcome these hardware and configuration issues that have been impeding DL model deployment. Examples that are at the forefront of this development include Colaboratory (Colab) by Google, Azure Machine Learning by Microsoft, Watson Studio by IBM, and SageMaker by Amazon (Pal and Hsieh, 2021). Virtualisation using cloud computing is the process of creating a virtual version of a physical computer with a dedicated amount of processer, memory and GPU borrowed from a cloud provider's server. As a result of this, virtual machines (VMs) remain independent of the local physical host computer (Rahman *et al*., 2022).

For DL model training, a functional computer with the hardware requirements mentioned above currently cost approximately USD 2,000. Apart from the freely available Colab version, Google offers Colab Pro, Colab Pro+ and cloud enabled platforms for a subscription fee (Google Research, 2022). Colab Pro for DL model training is currently the cheapest option as it saves money on special hardware requirements. Employing Colab as a VM only requires a Google account and a cost of approximately USD 10. Successful developments of DL models using Colab platform are evident in the studies of Canesche et al. (2021); Carneiro et al. (2018) and Ohkawara et al. (2021). The major advantage of using VMs to develop DL models for construction applications is that they enable efficient DL models to be shared among project stakeholders through the cloud without configuration modifications (Pal and Hsieh, 2021; Rahman *et al*., 2022). Despite the avenues such as VM technology to reduce computational complexities associated with DL models development, construction elements recognition of indoor construction works using DL models is lacking.

Research methods

The overarching research process used to develop the DL-based approach to recognise as-built indoor elements during construction works is shown in Figure 2. It involves three major stages of the research process.

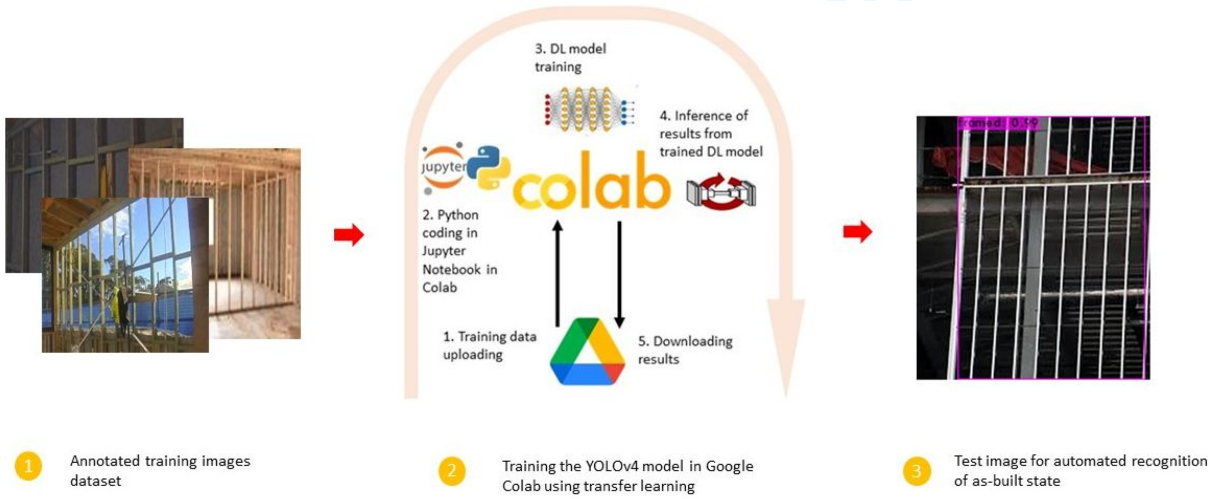

Figure 2: Process of developing the DL-based approach to recognise indoor as-built elements

The first stage involves building an annotated indoor site images dataset. Having a training dataset comprising high-quality images with different lighting conditions, material, texture, and colour is crucial for overcoming the underfitting and overfitting problems related to DL models (Wang *et al*., 2018). Underfitting is the failure to capture relevant patterns in data, which leads to inaccurate predictions (Jabbar and Khan, 2015). Overfitting occurs when the model accurately recognises objects within training images, but the model is not as accurate at recognising objects in the images that are not trained on or are not present in the training dataset (Rice *et al*., 2020). Therefore, it is essential to build an annotated image dataset by overcoming the aforementioned challenges.

The second stage of the DL-based approach is built upon YOLOv4 using transfer learning with few hyperparameters customised and then trained on Google Colab. Programmers employ transfer learning to reuse pre-trained DL neural networks because transfer learning reduces time and manual intervention (Nalini and Radhika, 2020). A DL model can either be built from scratch or a pretrained model which uses existing networks such as GoogleNet, AlexNet, ResNet, VGG-16 can be employed (Simonyan and Zisserman, 2014). The first approach involves computational complexities of building the convolutional, pooling and fully connected layers from scratch with their optimisations. The latter approach uses transfer learning to refine the pre-trained model to which the new data containing previously unknown classes is introduced only by customising certain hyperparameters of the new DL model (Pan and Yang, 2009; Torrey and Shavlik, 2010). Since the DL model has been pre-trained on large dataset of object classes, this approach is not as prolonged and manually intervened as creating a model from scratch (O'Mahony *et al*., 2019).

Google Colab's ability to run as a VM with the runtime fully configured for DL model training and free-of-charge access to GPUs, memory and processors have gained widespread recognition (Canesche *et al*., 2021; Carneiro *et al*., 2018). Colab is a web based Jupyter notebook enabled to execute Python codes. Colab notebooks are stored in Google Drive enabling Google Drive as the storage unit to be accessed from any web browser as opposed to using the hard drive in a local computer (Ohkawara *et al*., 2021). Colab enables setting up VM as runtime by connecting to GPUs and tensor processing units (TPUs) hosted by Google or through Google cloud platform hosted services. Colab users also can opt to connect to a local runtime by executing the code in local computers' hardware (Google Research, 2022). Since zero configuration is required and most of the ML libraries are already installed, DL models can be trained on Colab with a few lines of code and can be shared, stored, and accessed using Google Drive (Pal and Hsieh, 2021).

The final stage involves the DL model being tested on indoor site images to confirm that the as-built state of indoor elements during the construction process can be recognised automatically by using this model. Upon providing as-built images, this model can be extended to automated recognition of any indoor construction elements. The indoor wall element comprising the framing, insulation, and drywall installation of indoor partitions was selected as the as-built case scenario to test the model. The reason for selecting this scenario is that indoor partitions cover a significant portion of indoor construction and delays with this indoor element can typically create costly consequences (Kropp *et al*., 2012). Internal wall partition works also overlap several different trades such as framers, insulation installers and drywall installers and different levels of site supervision and management (Hamledari *et al*., 2017). Three indoor sites were used as case projects. Two projects were used to capture training images and the third was used to capture test images.

Image data collection and preparation

Training images were collected from two construction sites in Sydney, Australia. Site 1 is a residential building renovation project and Site 2 is an office building fit-out project. Two time-lapse cameras (Brinno TLC200 PRO) were used at each site. Fixed time-lapse cameras were selected due to their ubiquitous use in construction sites for inexpensive progress monitoring and surveillance (Ahmadian Fard Fini *et al*., 2022). The reason for using two cameras at the same site was to collect images under various lighting conditions in different floor layouts and to capture images from best vantage points. The resolution of the images captured was 1280×720 pixels. The cameras were checked, and videos were collected on a fortnightly basis over a 10-months period to avoid memory and battery outage and damages to the cameras in heavily cluttered indoor areas.

To create a diverse and large dataset from each category in the framing, insulation, and drywall installation scenario, publicly available online images were also sourced. When the number of diversified images is higher, DL model has sufficient features to learn, and the accuracy increases by overcoming the underfitting problem (Wang *et al*., 2018). For example, as shown in Figure 3, image (a) was captured from insulation in Site 1 and image (b) was sourced from insulation images available online.

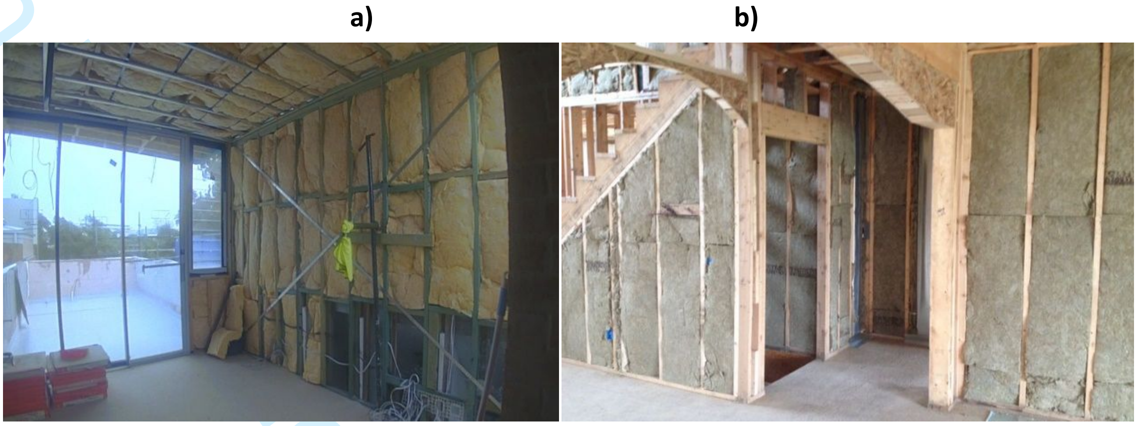

Figure 3: a) Image captured from Site 1; b) image sourced from the Internet

To generate more training images with variability, data augmentation was applied. It is a technique to transform the existing images to create new versions of the original images (Shorten and Khoshgoftaar, 2019). This helps in reducing the overfitting problem (Rice *et al*., 2020). Data augmentation can be performed through photometric distortions and geometric distortions. Adjusting the brightness, contrast, hue, saturation, and noise of an image are examples of photometric distortion. Strategies for geometric distortion are random scaling, cropping, flipping, and rotating (Bochkovskiy *et al*., 2020). Using the ML library "imgaug", which is commonly used for image augmentation, a code was developed to enable image augmentation. After preparing the dataset, the images annotated with a bounding box using the online annotation tool "Make Sense". The corresponding text files containing the coordinates of the ground truth bounding box were obtained as the labels. The labels in the dataset were "framed", "insulated", "drywall_installed". 2,250 annotated images were prepared for training.

Developing the DL-based object recognition approach using YOLOv4

YOLO is computationally faster and simpler compared to R-CNNs for object recognition (O'Mahony *et al*., 2019). YOLOv4 (Bochkovskiy *et al*., 2020) is currently the most stable, accurate and optimal speed version of YOLO. Understanding the network structure of YOLOv4 is important to determine which hyperparameters need to be customised using transfer learning. The network structure of a DL model comprises a CNN backbone for feature learning and extraction and a head to predict classes and bounding boxes of the objects

(Bochkovskiy *et al*., 2020). YOLOv4 has a backbone made of Darknet-53 (Redmon, 2013). Its head is made of YOLOv3 (Redmon and Farhadi, 2018). The original YOLOv4 model has been trained by the creators of YOLOv4, Bochkovskiy et al. (2020) on the COCO dataset which comprises of day-to-day general objects of 80 different classes. Darknet has been pre-trained for these objects and thus the network backbone of YOLOv4 is capable of feature learning and extraction (Bochkovskiy *et al*., 2020; Wang *et al*., 2020). Transfer learning was used for the current study to harness this feature learning and extraction ability of pre-trained Darknet to generate the weights for the new classes of "framed", "insulated", "drywall_installed".

The workflow of training the DL model in Colab

The steps in training YOLOv4 using transfer learning in Colab are illustrated in Figure 4 and explained forthwith. Figure 4 presents the technical algorithm for DL-based object recognition that was used for this study. This process relates to the training images collected for the classes of "framed", "insulated", "drywall_installed" in the progress monitoring scenario.

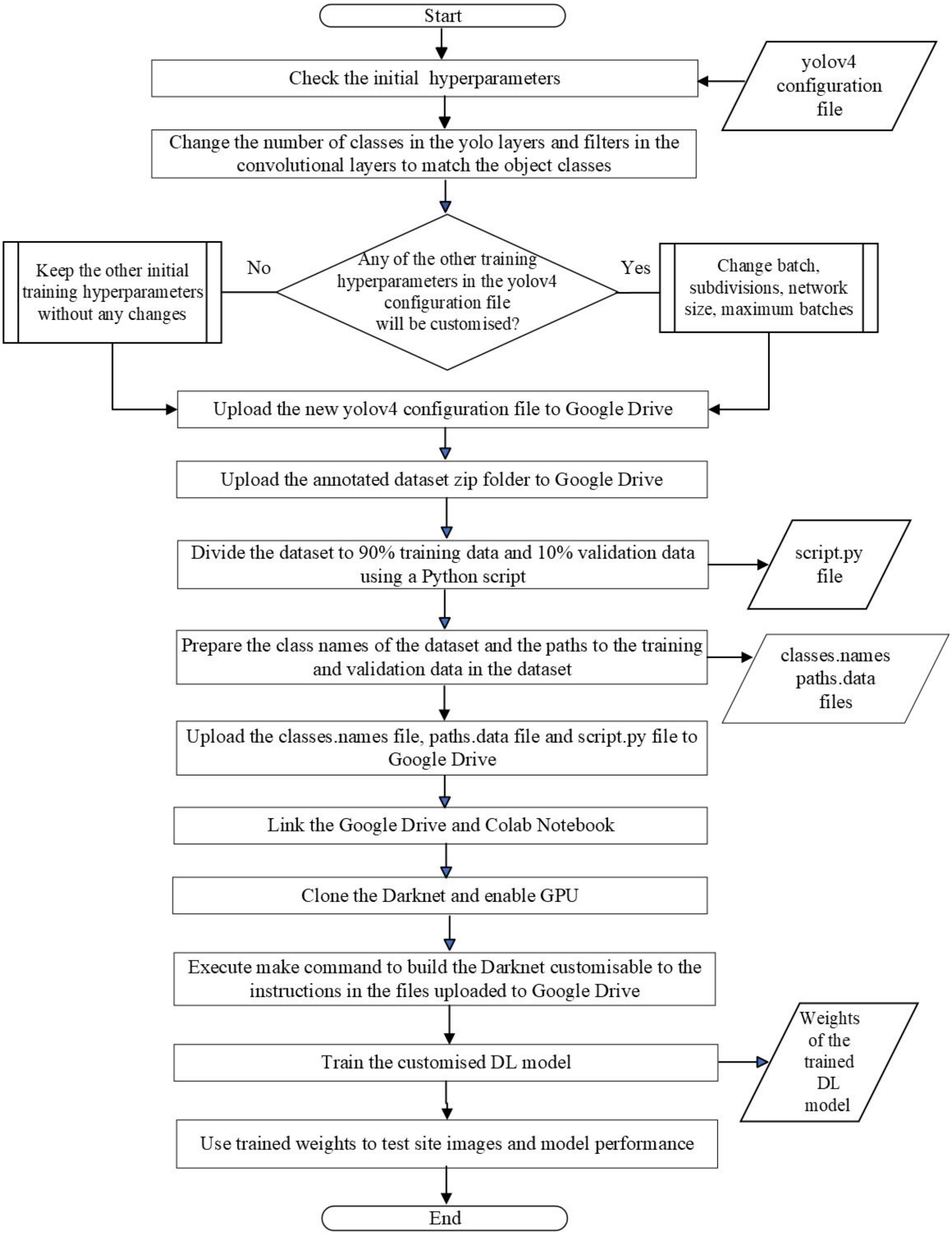

Figure 4: Technical algorithm for DL-based object recognition approach using YOLOv4

Step 1: Customising the hyperparameters in the yolov4 configuration file

As the first phase of using transfer learning, the yolov4 configuration file "*yolov4-custom.cfg*" was downloaded from the Github repository for YOLOv4, AlexeyAB (Bochkovskiy *et al.*, 2020). The changes made to some of the training hyperparameters in the yolov4 configuration file are listed below.

- Number of classes in yolo layers = 3
- Filters in the convolutional layers before each yolo layer = 24
- Batch = 64
- Subdivisions = 8
- Network size = 416x416
- Maximum batches = 6000

These hyperparameters are explained as follows.

- In the yolov4 configuration file, the YOLO layers and the convolutional layers before each YOLO layer are modified. Before each of the 3 YOLO layers, there are 3 convolutional layers. In convolutional layers, filters are used to extract features to build high-level feature map of the objects. The original Darknet used 255 filters which is dependent on the number of classes of 80. The number of filters is calculated using the formula: (number of classes + 5) x 3 (Bochkovskiy *et al.*, 2020). As there are 3 classes in the image training dataset, the number of classes in the YOLO layers and the corresponding number of filters must be adjusted. The rest of the layers are kept without making any changes since the original model has been optimally tuned for large number of classes as 80.
- Batch=64 indicates loading 64 images for one iteration.
- Subdivision=8 indicates splitting batch into 8 mini-batches, such that 64/8 = 8 images per mini-batch. These 8 images are sent to the GPU for processing. Processing in mini batches facilitates fast processing by GPU (Bochkovskiy *et al.*, 2020). The process is performed 8 times until the batch is completed, and a new iteration starts with 64 new images.
- The pixel resolution must be a multiple of 32. The resolution size chosen for the current study is 416x416. Larger image resolution may slow down the training and smaller image resolution may reduce the accuracy of training. Medium sized resolution is considered as the best practice (Redmon *et al.*, 2016).
- The maximum batches is 6000, which can be calculated using the formula (number of classes x 2000 = 3 x 2000 = 6000). For Darknet, the minimum batches should be 2000. In Darknet YOLO, the number of iterations depends on the max_batches (Redmon,

2013). A complete epoch requires 100 iterations. Since the max_batches = 6000, training ends after 60 epochs.

Step 2: Uploading files needed for training to Google Drive

Google Drive is the storage location for Colab. Therefore, before executing training in Colab, the files carrying instructions for training must be uploaded to Google Drive. In this study, a folder named "*yolov4"* was created in the Google Drive. The following files and sub folders were uploaded to this "*yolov4"* folder. The naming convention was adapted to reflect the purpose of each file.

- "data.zip"- The zip folder containing the images and their corresponding text files with annotation details.
- "training"- The sub folder to save the weights of the YOLOv4 model trained on the image dataset.
- "yolov4-custom.cfg"- The yolov4 configuration file downloaded from the Github repository, AlexeyAB.
- "script.py"-The Python script containing the instructions to split the dataset into 2 parts as 90% for training and 10% for validation.
- "classes.names"-The names file with the instructions on the 3 name classes of the objects, "framed", "insulated'', "drywall_installed".
- "paths.data"-The data file with the instructions on the paths to training and validation data.

Step 3: Linking the Google Drive and Colab notebook

The Colab notebook was created from the same Google account linked to the Google Drive for executing the Python code for training. This Colab notebook was saved to Google Drive. The mount drive command was executed to link the "*yolov4"* folder to the Colab notebook. For this study, Colab was connected to a hosted runtime and the runtime was set to GPU and high RAM capacity. At the time of executing this DL model, Colab offered NVIDIA Tesla T4 GPU of 16GB, 13GB RAM and 2.2 GHz of processor speed.

Step 4: Cloning Darknet and enabling GPU

Darknet was cloned to Colab from the Github repository AlexeyAB. Cloning was done to import the repository to Colab with the pre-trained weights. Enabling the GPU was carried out to execute the DL model using the CUDA version 11.2 and cuDNN version 7.6.5. In this study, a sub folder called *"darknet*" was created automatically inside "*yolov4"* folder after cloning.

Step 5: Building and customising the Darknet

As the second phase of transfer learning, the "make" command was executed to build the Darknet customisable to the instructions in the files uploaded to Google Drive. With the make command, the files uploaded in Step 2 were copied to the Darknet directory. This enabled Darknet to be customised according to the newly introduced training data and their class names.

Step 6: Training the customised DL model

Upon executing the train custom detector command, as per the changes made in Step 5, weights of the custom YOLOv4 model were saved to the "training" sub folder in every 1000 iteration, until 6000 iterations were achieved.

Evaluating the performance of the DL model

The study focused on the mean average precision (mAP) and average loss to capture the overall performance of the DL model at an intersection over union (IoU) of 0.5. The metric, mAP is widely used to evaluate the detection accuracy of DL models (O'Mahony *et al*., 2019). In addition to mAP, when training DL models, loss value indicates how well a DL model behaves after each iteration. The reduction of loss after each or several iterations is an indication of the higher accuracy of the DL model (Akbari *et al*., 2021). The IoU measures how much the predicted boundary detected by the DL model overlaps with the real object boundary or the ground truth (O'Mahony *et al*., 2019). Accordingly, this DL model did not detect objects, whose confidence of existence score was less than 50%. This accuracy level is usually set as the minimum threshold of detection for many DL models. The mAP of the best weight is 92% and the overall mAP of the DL model considering all the weights is 87.3%. The average loss of the model is 0.83. The performance of all the trained weights is illustrated in the chart in Figure 5.

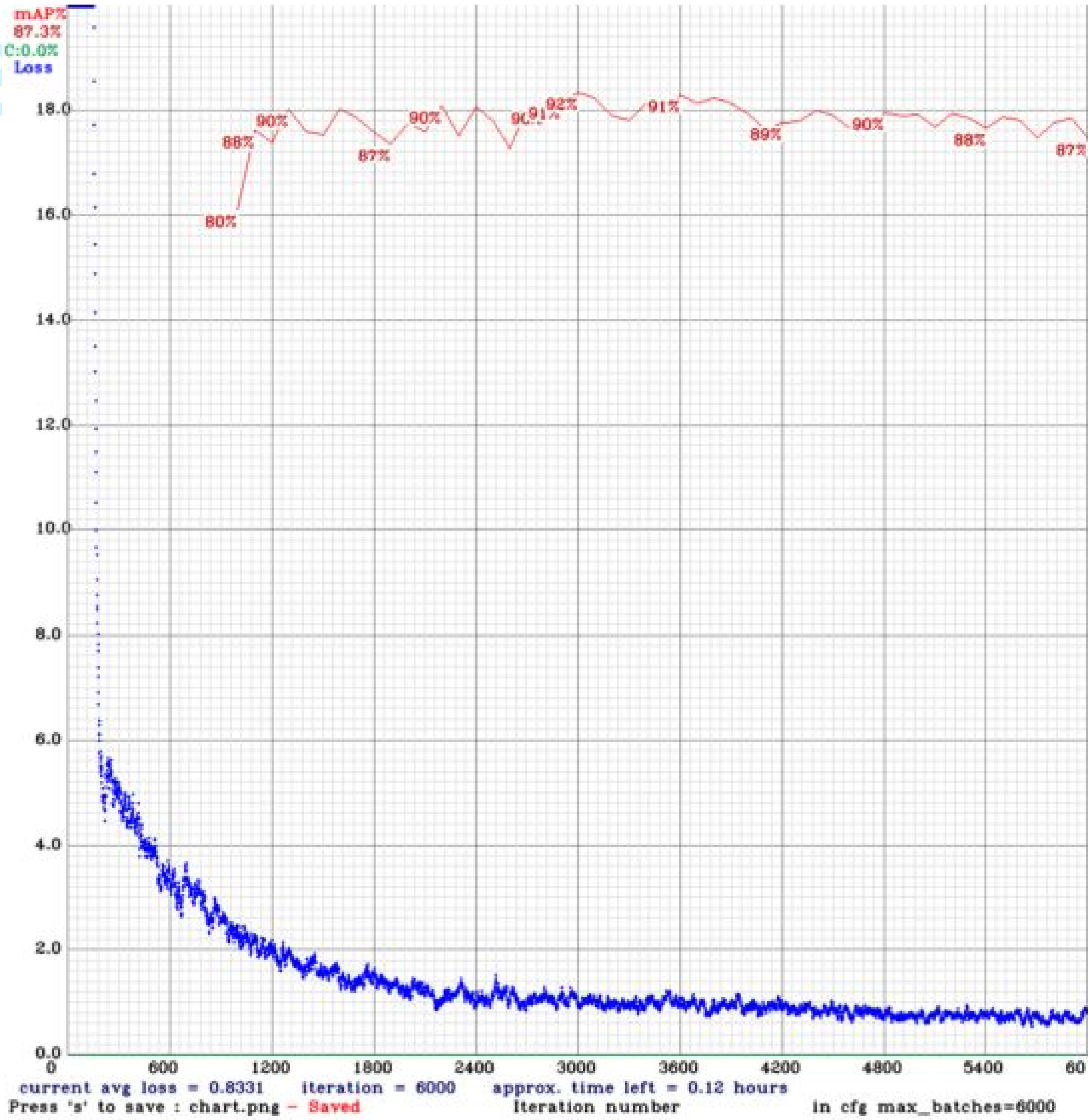

Figure 5: Performance chart of the proposed DL model

The best weight with the highest mAP of 92% was used to detect the test images uploaded to Google Drive. The test images were obtained as in Figure 6a and 6b to recognise the framing, insulation, and drywall installation states.

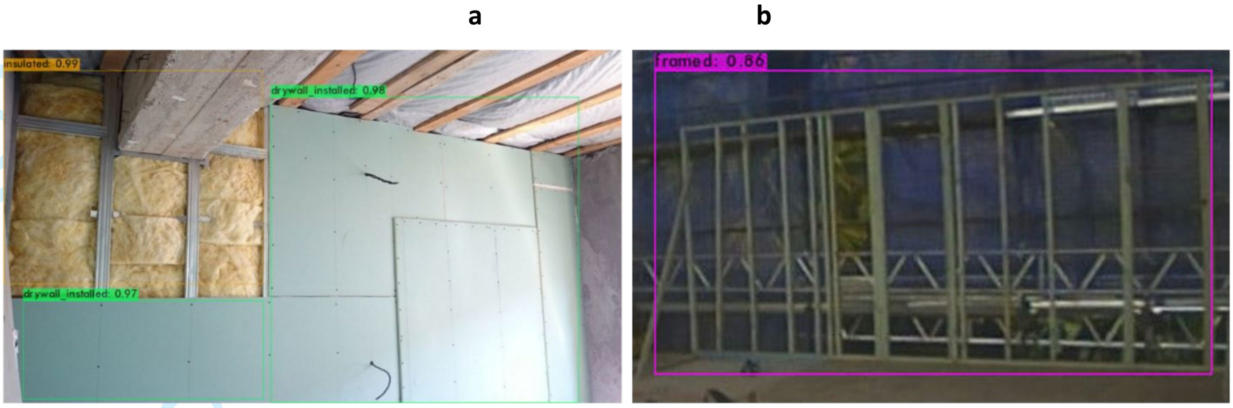

Figure 6: Test image for a) insulation and drywall installation; b) framing

Confidence of existence scores of 97% and 98% were recorded for recognising the state of drywall installation and 99% for insulation respectively in Figure 8a. Figure 8b shows how framing state was recognised with a confidence of existence score of 86%. Accordingly, the detection and classification ability of YOLOv4 was harnessed to recognise the as-built states of indoor partitions through this automated object recognition approach.

Discussion

This section discusses how the results generated by the DL model of the current study can be compared with the previous CV-based studies on indoor construction elements recognition in terms of reducing the impacts of the technical challenges related to indoor objects, lighting conditions and camera positioning. A comparison with the recent studies, which employed DL models is also provided. Challenges encountered in training the DL model in Colab environment are also discussed.

Indoor elements state recognition by overcoming the technical challenges

Previous CV-based studies on indoor construction elements recognition have used handcrafted feature extraction and employed pre-processing algorithms to enhance visual quality and remove background noise in images for recognising as-built elements. This study aimed at reducing the impacts of the technical challenges and improving the accuracy of objects recognition by harnessing the detection and classification ability of DL models for complex indoor construction environments.

Figure 7a exhibits challenges related to CV-based indoor construction elements recognition, when the framing process (shown inside the red-coloured rectangular box) was captured. The major challenge was to determine the strategic location to install the camera that provides the best viewing angle of the framing process. Previous studies by Kropp et al. (2013); Hamledari and McCabe (2016) and Ekanayake et al. (2021a) identified the limitations related to relocating fixed cameras and limited field of view and angles. Additionally, detecting the ROI was constrained by the presence of a stepladder and movements of construction personnel obstructing the framing area. The presence of temporary equipment and material and movements of construction personnel create clutter and obstructions in images (Ekanayake *et al.*, 2021b; Hamledari and McCabe, 2016; Kropp *et al.*, 2014).

Moreover, the natural light entering the indoor site from openings produced backlight and caused shadows. As stated by Kropp et al. (2013); Hamledari and McCabe (2016) and Ekanayake et al. (2021a), backlights and shadows constrain feature extraction. To obtain the ROI, the image was only cropped and resized, more accurately without subjecting to algorithmic pre-processing to enhance visual quality and remove noise (as shown in Figure 7b) The customised DL model recognised the framing state as evidenced in Figure 7b although the confidence score of existence in detecting framing state is 58% due to the background noise and poor visual quality in the indoor site image.

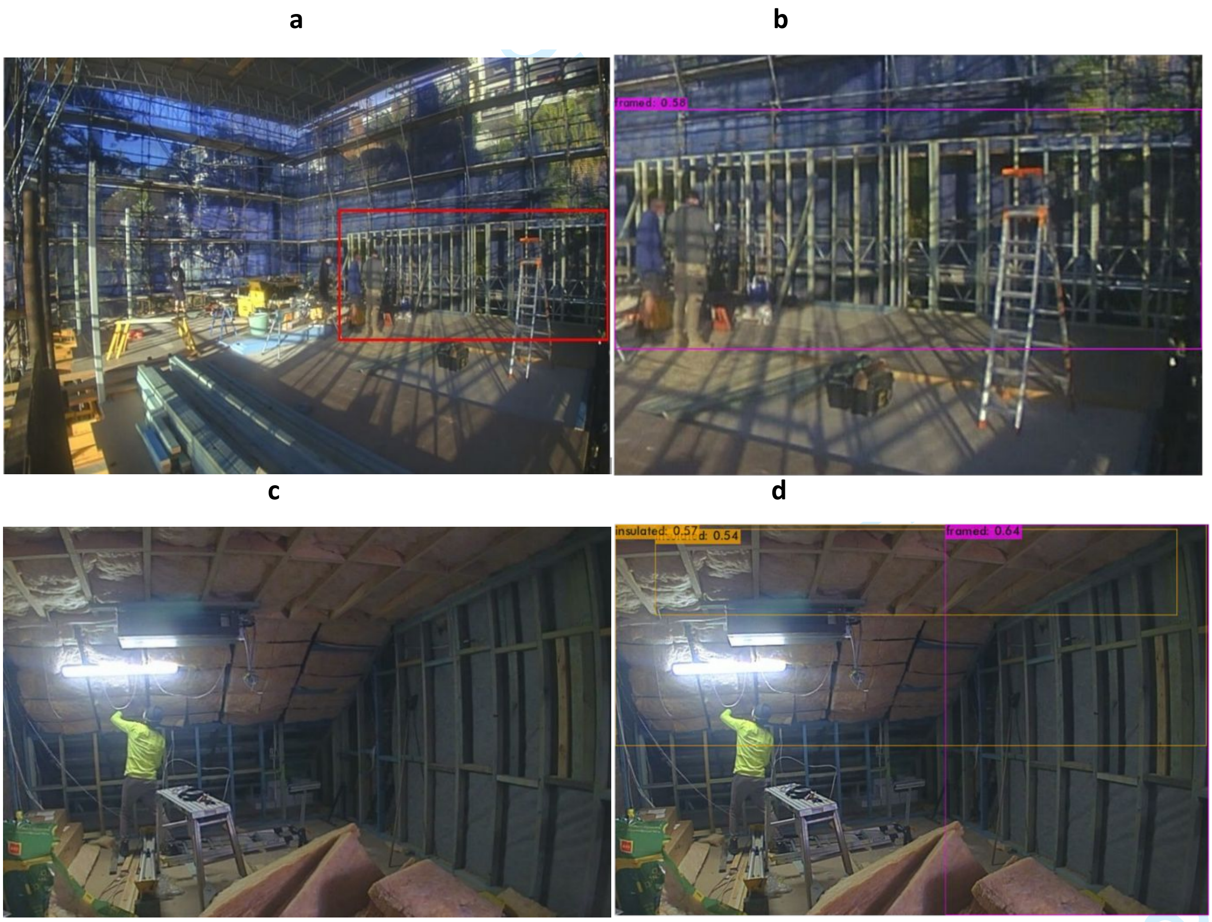

Figure 7: Indoor elements state recognition by overcoming the technical challenges

Figure 7c depicts further technical challenges in the indoor sites. The recognition of the insulation state was significantly affected by the presence of an artificial lighting fixture in the middle of the ROI. Previous studies by Ekanayake et al. (2021b) and Hamledari and McCabe (2016) have highlighted that artificial lights cause non-uniform illumination constraining robust feature extraction. Indoor objects related challenges such as construction material including batt insulation blankets and insulation tools caused clutter in the indoor scenes and the ROI was blocked by construction workers carrying out the insulation. Even though this indoor scene was heavily cluttered and poor visual quality was evident, the DL model could accurately recognise the framing and insulation states as evidenced in Figure 7d. The confidence scores of existence for insulation are 54% and 57% whilst framing has a score of 64%.

Additionally, when the ROIs captured from the fixed cameras were of irregular shapes and orientations, the confidence scores of detections tended to be low. Nonetheless, without using any pre-processing algorithms to make improvements to the images, the indoor elements as-built states in the ROIs were recognised by using the YOLOv4-based DL approach used in this study.

Comparison with the previous studies which employed DL models

The pioneering studies of Ying and Lee (2019) and Shamsollahi et al. (2021) only provide evidence on the recognition of the indoor elements using Mask R-CNN. The automation level is heavily manually intervened during the data collection in Ying and Lee (2019). Since synthetic images are used as the training images, Shamsollahi et al. (2021) fails to reflect the impacts of challenging indoor construction environment. While Wei et al. (2022) calculates indoor work completion percentage using Mask R-CNN-based segmentation and maps the relationship between 2D images and 3D building information models, the high manual intervention in the data collection and algorithmic pre-processing steps is noteworthy. The use of a small training dataset and testing and training images being collected from the same location are the other limitations.

Compared to these studies, the automation level of the current study is high due to the use of fixed time-lapse cameras for data collection and zero manual intervention from the pre-processing algorithms to enhance visual quality of indoor images. The current study also reflects the impacts of challenging indoor construction environment on automated visual recognition of indoor elements. These previous studies have not used a DL model other than

Mask R-CNN and have not addressed the means to overcome the DL model training related complications by using a VM platform such as Colab.

Challenges encountered in training the DL model in Colab

Despite offering a cost effective and pre-configured environment to train DL models, the free-of-charge Colab platform poses certain limitations. Even with a steady internet connection, idle timeouts of more than 90 minutes can cause disruptions. When Colab gets disconnected during the training, backbone executables of DL models will not work. Considering these limitations, Colab Pro was employed for the current study. When trained in Colab Pro, the VM was connected to a stable runtime with faster GPU, RAM, and processor. To avoid being disconnected from Colab Pro, an auto click code was executed.

Despite the current limitations, the advancements in cloud computing are highly promising to train DL models in the cloud enabled VMs. One such advancement is that Colab users can setup a Google cloud platform (GCP) account and connect to a GCP marketplace VM as the runtime (Rahman *et al*., 2022). These VMs offer complete flexibility and provide a consistent environment removing all the Colab enforced runtime limitations. At a reasonable subscription fee, the deployment of DL models for very large image datasets through cloud enabled platforms has become relatively fast and easy to set up with less configurations compared to the edge computing counterparts.

Conclusions and future directions

Traditional methods of as-built state recognition practices lack accuracy, are inefficient and costly. Compared to exterior sites, the as-built object recognition in indoor site images is hindered by the technical challenges related to indoor objects, lighting conditions and camera positioning. Since traditional ML algorithms employ manual feature extraction and are sensitive to image quality, recognising indoor construction elements with poor visual quality is challenging. By harnessing YOLOv4 algorithms' ability in real-time efficient and accurate object detection and classification through training images, this study presents a DL-based approach to facilitate the as-built state recognition of indoor construction works.

Using transfer learning, trained weights were generated for the customised YOLOv4 model for the selected indoor as-built scenario of framing, insulation, and drywalls installation. This DL model proves high accuracy with a best trained weight reporting a mAP of 92% and an average loss of 0.83. Different from the recent DL-based indoor construction progress monitoring studies, this study contributes to the body of knowledge and the industry practitioners from the following two aspects. (1) The current study offers an efficient, accurate and readily shareable workflow of training DL models in a VM platform based on Google Colab. (2) Upon providing training images, the accurate detection and classification ability of the customised YOLOv4 model can be extended to recognise the as-built states of other indoor scenes such as tiling, ceiling sheets installation, interior glazing.

There are some limitations to this study despite its contributions. The images collected from complex environments such as indoor construction sites pose challenges for the detection and classification ability of DL models. Therefore, in future studies, there is room for improving the performance of the current DL model by introducing more training images and fine-tuning the hyperparameters such as learning rate, loss function of the YOLOv4 algorithm.

References

Ahmadian Fard Fini, A., Maghrebi, M., Forsythe, P.J. and Waller, T.S. (2022), 'Using existing site surveillance cameras to automatically measure the installation speed in prefabricated timber construction', *Engineering, Construction and Architectural Management*, vol. 29, no. 2, pp. 573-600.

Akbari, A., Awais, M., Bashar, M. & Kittler, J. (2021), 'How Does Loss Function Affect Generalization Performance of Deep Learning? Application to Human Age Estimation', paper presented to the 38th International Conference on Machine Learning, Proceedings of Machine Learning Research, https://proceedings.mlr.press/v139/akbari21a.html.

Bochkovskiy, A., Wang, C.Y. and Liao, H.Y.M. (2020), 'Yolov4: Optimal speed and accuracy of object detection', *arXiv preprint arXiv:2004.10934*.

Bosché, F. (2012), 'Plane-based registration of construction laser scans with 3D/4D building models', *Advanced Engineering Informatics*, vol. 26, no. 1, pp. 90-102.

Canesche, M., Bragança, L., Neto, O.P.V., Nacif, J.A. and Ferreira, R. (2021), 'Google Colab CAD4U: Hands-on Cloud Laboratories for Digital Design', *2021 IEEE International Symposium on Circuits and Systems (ISCAS)*, IEEE, pp. 1-5.

Canny, J. (1986), 'A computational approach to edge detection', *IEEE Transactions on pattern analysis and machine intelligence*, no. 6, pp. 679-698.

Carneiro, T., Da Nóbrega, R.V.M., Nepomuceno, T., Bian, G.B., De Albuquerque, V.H.C. and Reboucas Filho, P.P. (2018), 'Performance analysis of google colaboratory as a tool for accelerating deep learning applications', *IEEE Access*, vol. 6, pp. 61677-61685.

Chollet, F.(2017), *Deep learning with Python*, Simon and Schuster, New York, NY.

Deng, H., Hong, H., Luo, D., Deng, Y. and Su, C. (2020), 'Automatic Indoor Construction Process Monitoring for Tiles Based on BIM and Computer Vision', *Journal of Construction Engineering and Management*, vol. 146, no. 1.

Ekanayake, B., Fini, A. and Wong, J.K.W (2021), 'Technical challenges for automated indoor construction progress monitoring', paper presented to the 44th AUBEA Conference 2021, Deakin University, Melbourne, Australia.
Ekanayake, B., Wong, J.K.W., Fini, A.A.F. and Smith, P. (2021), 'Computer vision-based interior construction progress monitoring: A literature review and future research directions', *Automation in Construction*, vol. 127, p. 103705.
Golparvar-Fard, M., Peña-Mora, F. and Savarese, S. (2015), 'Automated Progress Monitoring Using Unordered Daily Construction Photographs and IFC-Based Building Information Models', *Journal of Computing in Civil Engineering*, vol. 29, no. 1.
Google Research (2022), *Colaboratory*, available at https://research.google.com/colaboratory/faq.html(accessed 19 January 2022).
Guo, X., Li, Y. and Ling, H. (2016), 'LIME: Low-light image enhancement via illumination map estimation', *IEEE Transactions on Image Processing*, vol. 26, no. 2, pp. 982-993.
Guven, G. and Ergen, E. (2021), 'Tracking major resources for automated progress monitoring of construction activities: masonry work case', *Construction Innovation*, vol. 21, no. 4, pp. 648-667.
Hamledari, H. and McCabe, B. (2016), 'Automated visual recognition of indoor project-related objects: Challenges and solutions', *Construction Research Congress 2016*, pp. 2573-2582.
Hamledari, H., McCabe, B. and Davari, S. (2017), 'Automated computer vision-based detection of components of under-construction indoor partitions', *Automation in Construction*, vol. 74, pp. 78-94.
Jabbar, H. and Khan, R.Z. (2015), 'Methods to avoid over-fitting and under-fitting in supervised machine learning (comparative study)', *Computer Science, Communication and Instrumentation Devices*, pp. 163-172.
Jian, L., Wang, C., Liu, Y., Liang, S., Yi, W. and Shi, Y. (2013), 'Parallel data mining techniques on graphics processing unit with compute unified device architecture (CUDA)', *The Journal of Supercomputing*, vol. 64, no. 3, pp. 942-967.
Jorda, M., Valero-Lara, P. and Pena, A.J. (2019), 'Performance evaluation of cudnn convolution algorithms on nvidia volta gpus', *IEEE Access*, vol. 7, pp. 70461-70473.
Kardovskyi, Y. and Moon, S. (2021), 'Artificial intelligence quality inspection of steel bars installation by integrating mask R-CNN and stereo vision', *Automation in Construction*, vol. 130, p. 103850.
Kartika, I. and Mohamed, S.S. (2011), 'Frame differencing with post-processing techniques for moving object detection in outdoor environment', *2011 IEEE 7th International Colloquium on Signal Processing and its Applications*, IEEE, pp. 172-176.
Kopsida, M., Brilakis, I. and Vela, P.A. (2015), 'A review of automated construction progress monitoring and inspection methods', *Proc. of the 32nd CIB W78 Conference 2015*, pp. 421-431.
Kropp, C., Koch, C. and König, M. (2014), 'Drywall state detection in image data for automatic indoor progress monitoring', *Computing in Civil and Building Engineering*, pp. 347-354.
Kropp, C., Koch, C. and König, M. (2018), 'Interior construction state recognition with 4D BIM registered image sequences', *Automation in Construction*, vol. 86, pp. 11-32.
Kropp, C., Koch, C., König, M. and Brilakis, I. (2012), 'A framework for automated delay prediction of finishing works using video data and BIM-based construction simulation', *Proceedings of the 14th International Conference on Computing in Civil and Building Engineering*.

Kropp, C., König, M. and Koch, C. (2013), 'Object recognition in BIM registered videos for indoor progress monitoring', *EG-ICE International Workshop on Intelligent Computing in Engineering*.

LeCun, Y., Bengio, Y. and Hinton, G. (2015), 'Deep learning', *Nature*, vol. 521, no. 7553, pp. 436-444.

Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.Y. and Berg, A.C. (2016), 'SSD: Single shot multibox detector', *European conference on computer vision*, Springer, pp. 21-37.

Li, Y., Lu, Y. and Chen, J. (2021), 'A deep learning approach for real-time rebar counting on the construction site based on YOLOv3 detector', *Automation in Construction, 124*, 103602.

Martinez, P., Ahmad, R. and Al-Hussein, M. (2019), 'A vision-based system for pre-inspection of steel frame manufacturing', *Automation in Construction*, vol. 97, pp. 151-163.

Nalini, M. and Radhika, K. (2020), 'Comparative analysis of deep network models through transfer learning', *2020 Fourth International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud)(I-SMAC)*, IEEE, pp. 1007-1012.

Nanni, L., Ghidoni, S. and Brahnam, S. (2017), 'Handcrafted vs. non-handcrafted features for computer vision classification', *Pattern Recognition*, vol. 71, pp. 158-172.

O'Mahony, N., Campbell, S., Carvalho, A., Harapanahalli, S., Hernandez, G.V., Krpalkova, L., Riordan, D. and Walsh, J. (2019), 'Deep learning vs. traditional computer vision', *Science and Information Conference*, Springer, pp. 128-144.

Ohkawara, M., Saito, H. and Fujishiro, I. (2021), 'Experiencing GPU path tracing in online courses', *Graphics and Visual Computing*, vol. 4, p. 200022.

Pal, A. and Hsieh, S.H. (2021), 'Deep-learning-based visual data analytics for smart construction management', *Automation in Construction*, vol. 131, p. 103892.

Pan, S.J. and Yang, Q. (2009), 'A survey on transfer learning', *IEEE Transactions on knowledge and data engineering*, vol. 22, no. 10, pp. 1345-1359.

Rahman, F.H., Newaz, S.H.S., Au, T.W., Suhaili, W.S., Mahmud, M.A.P. and Lee, G.M. (2022), 'EnTruVe: ENergy and TRUst-aware Virtual Machine allocation in VEhicle fog computing for catering applications in 5G', *Future Generation Computer Systems*, vol. 126, pp. 196-210.

Razavi, S., Young, D., Nasir, H., Haas, C., Caldas, C.H., Goodrum, P. and Murray, P. (2008), 'Field trial of automated material tracking in construction', *Annual Conference of the Canadian Society for Civil Engineering 2008-Partnership for Innovation*, pp. 1503-11.

Redmon, J. (2013), 'Darknet: Open source neural networks in C'.

Redmon, J., Divvala, S., Girshick, R. and Farhadi, A. (2016), 'You only look once: Unified, real-time object detection', *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 779-788.

Redmon, J. and Farhadi, A. (2018), 'Yolov3: An incremental improvement', *arXiv preprint arXiv:1804.02767*.

Rice, L., Wong, E. and Kolter, Z. (2020), 'Overfitting in adversarially robust deep learning', *International Conference on Machine Learning*, PMLR, pp. 8093-8104.

Shamsollahi, D., Moselhi, O. and Khorasani, K. (2021). 'A Timely Object Recognition Method for Construction Using the Mask R-CNN Architecture'. *ISARC. Proceedings of the International Symposium on Automation and Robotics in Construction*,

Shorten, C. and Khoshgoftaar, T.M. (2019), 'A survey on image data augmentation for deep learning', *Journal of Big Data*, vol. 6, no. 1, pp. 1-48.

Simonyan, K. and Zisserman, A. (2014), 'Very deep convolutional networks for large-scale image recognition', *arXiv preprint arXiv:1409.1556*.

Slaton, T., Hernandez, C. and Akhavian, R. (2020), 'Construction activity recognition with convolutional recurrent networks', *Automation in Construction*, vol. 113, p. 103138.

Torrey, L. and Shavlik, J. (2010), 'Transfer learning', *Handbook of research on machine learning applications and trends: algorithms, methods, and techniques*, IGI global, pp. 242-264.

Wang, C.Y., Liao, H.Y.M., Wu, Y.H., Chen, P.Y., Hsieh, J.W. and Yeh, I.H. (2020), 'CSPNet: A new backbone that can enhance learning capability of CNN', *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*, pp. 390-400.

Wang, J., Ma, Y., Zhang, L., Gao, R.X. and Wu, D. (2018), 'Deep learning for smart manufacturing: Methods and applications', *Journal of Manufacturing Systems*, vol. 48, pp. 144-156.

Wang, Z., Zhang, Q., Yang, B., Wu, T., Lei, K., Zhang, B. and Fang, T. (2021), 'Vision-Based Framework for Automatic Progress Monitoring of Precast Walls by Using Surveillance Videos during the Construction Phase', *Journal of Computing in Civil Engineering*, vol. 35, no. 1, p. 04020056.

Wei, W., Lu, Y., Zhong, T., Li, P. and Liu, B. (2022), 'Integrated vision-based automated progress monitoring of indoor construction using mask region-based convolutional neural networks and BIM', *Automation in Construction*, vol. 140, p. 104327.

Yang, J., Park, M.W., Vela, P.A. and Golparvar-Fard, M. (2015), 'Construction performance monitoring via still images, time-lapse photos, and video streams: Now, tomorrow, and the future', *Advanced Engineering Informatics*, vol. 29, no. 2, pp. 211-224.

Ying, H. and Lee, S. (2019), 'A Mask R-CNN based Approach to Automatically Construct As-is IFC BIM Objects from Digital Images', paper presented to the Proceedings of the 36th International Symposium on Automation and Robotics in Construction (ISARC).

Zhao, Z.Q., Zheng, P., Xu, S.T. and Wu, X. (2019), 'Object Detection with Deep Learning: A Review', *IEEE transactions on neural networks and learning systems*, vol. 30, no. 11, pp. 3212-3232.

Reviewers Comments to Author	**Authors Response to Reviewers Comments**
We sincerely appreciate all the valuable suggestions and the constructive feedback given by the reviewers. • Appropriate changes suggested by the reviewers have now been presented in red-coloured text in the revised manuscript. • The title has been revised as **"A deep learning-based approach to facilitate the as-built state recognition of indoor construction works".** After addressing the reviewers' comments, we determined that the revised title better communicates main objective of this manuscript. • To meet the word limit of the journal without hampering the quality of the revisions made, Table 1 has now been converted to text.	
Reviewer 1	
1- The abstract needs to be summarised the main points and avoid unnecessary parts to understand better and readability. The abstract is very general, with unnecessary statements. Please revise the abstract.	The "purpose" paragraph has been entirely revised. Lines 12-13 and17-20 in page 1 have now been revised to address this comment.
2- Please underscore the scientific value added in the abstract. Add some of the most critical quantitative results to the Abstract.	Lines 17-20 in page 1 have now been revised to address this comment. The key findings of the study are now highlighted with the quantitative values.
3- The objective of the study is not clear and needs to be specified in the introduction section.	Lines 6-8 and14-18 in page 3 have been revised to address this comment.
4- The results should be explained in more detail to a better understanding by readers and compared the findings with the existing literature. The given information is not sufficient.	A new section has been added under discussion in page 19 (lines 17-32). This section includes a comparison of the current study with the recent deep learning based indoor construction progress monitoring studies.
5- The conclusions need to be revised and improved. Please make sure the conclusion section underscores the scientific value added to the paper and the applicability of the findings/results. The conclusions are very general, with unnecessary statements.	The first paragraph of the conclusion provides an overview to the background and the purpose of this study in page 20 (lines 21 -26). The second paragraph of the conclusion has now been revised to highlight the key results and the contributions of this study (pages 20-21).
6- Please refer to more recent and relevant papers.	Lines 3-12 in page 6 have now been revised to refer to the most recent (2021, 2022) deep learning applications.

Reviewer 2	
1- page 4 (line 46 to 53). The literature review: overview of the literature review is not necessary. it makes it more like book or thesis. it is therefore suggested to be removed	This paragraph, "as recapitulated from the literature review….." has now been removed (lines 18-22 in page 7).
2- page 5, line 37, the subsection on neural network appeared suddenly without any prior mention or link from the preceding section to make a meaningful reading to the reader. As such, flow of thoughts should be applied in all the other sections of the literature review sections	In the paragraph before Figure 1, (line 24-27 of page 4), the convolution neural networks (CNNs) have been introduced as the widespread type of deep neural networks. Furthermore, Figure 1 has been used to demonstrate the differences of object recognition between a deep neural network (CNN) and a traditional ML algorithm (Canny edge detector). In page 5, lines 4-6, have now been reworded to communicate the explanation on CNN clearly.
3- page 6, Figure 1, line 28. when an image is from the Authors, there is no need to put the "source as Authors" (this applies to all other figures). However, are the Authors sure that the figures are theirs? remember this is literature review section.	"Source: The Authors" has now been removed. Although Figure 1 appears in the literature review section, it was created by the authors to clearly visualise the mechanism behind traditional machine learning algorithms and deep neural networks. Apart from the images collected from the indoor construction sites, all the other figures have been created by the authors.
4- page 9, are line 10, the word "below" should be removed. once a figure number is mentioned, there is no need to mention its location	This has now been corrected in page 7, line 20.

List of Figures

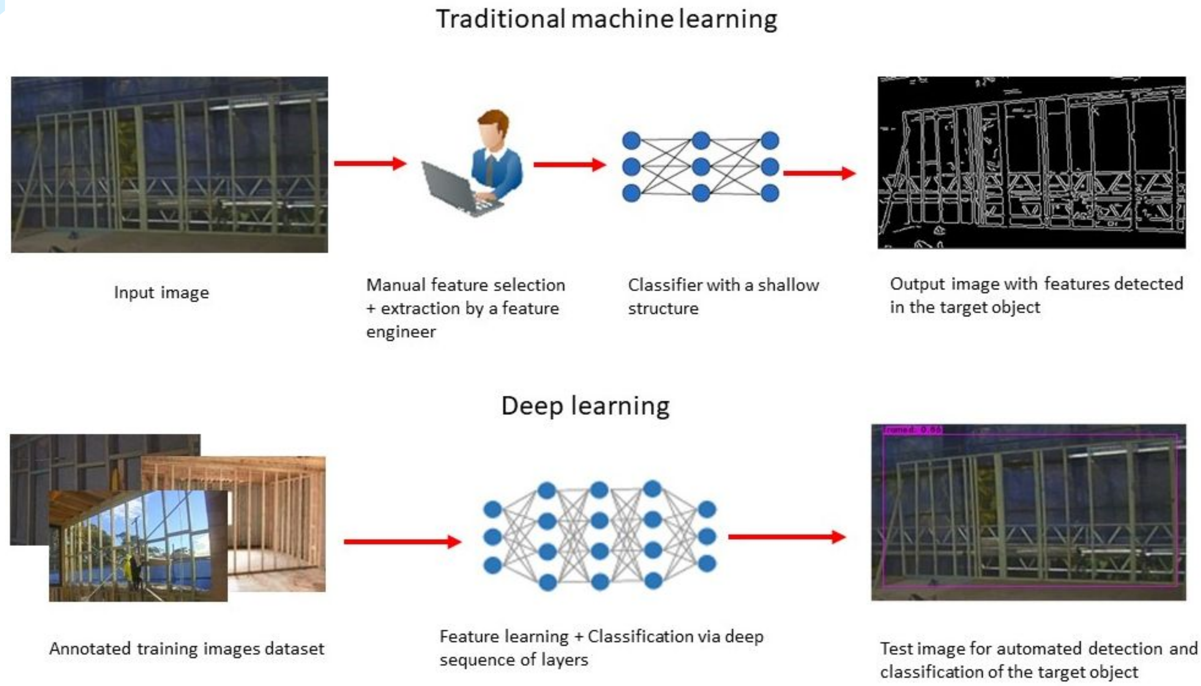

Figure 1: Mechanism behind (a) traditional machine learning (Canny edge detector); (b) deep learning (CNN)

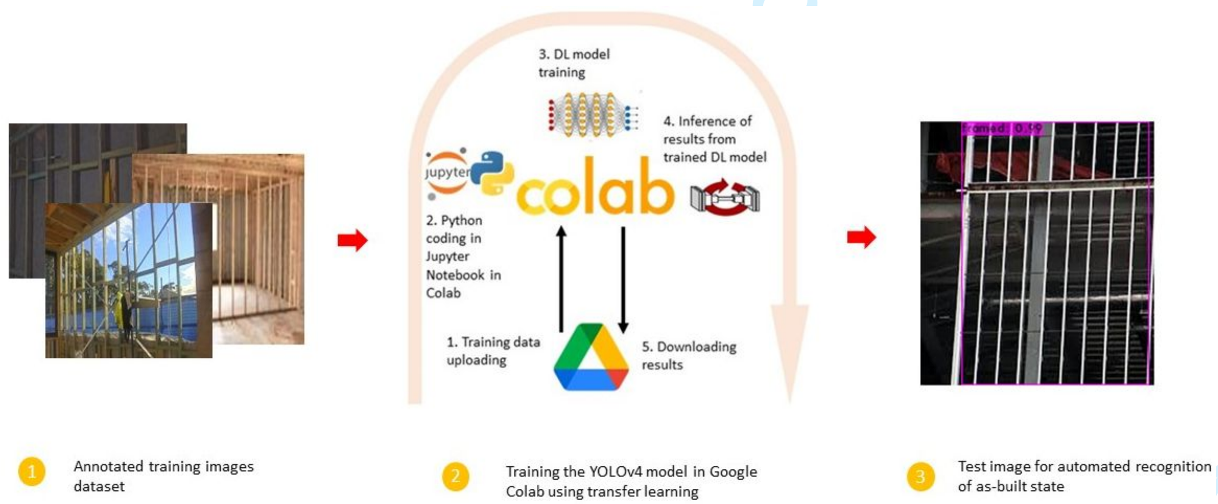

Figure 2: Process of developing the DL-based approach to recognise indoor as-built elements

a) b)

Figure 3: a) Image captured from Site 1; b) image sourced from the Internet

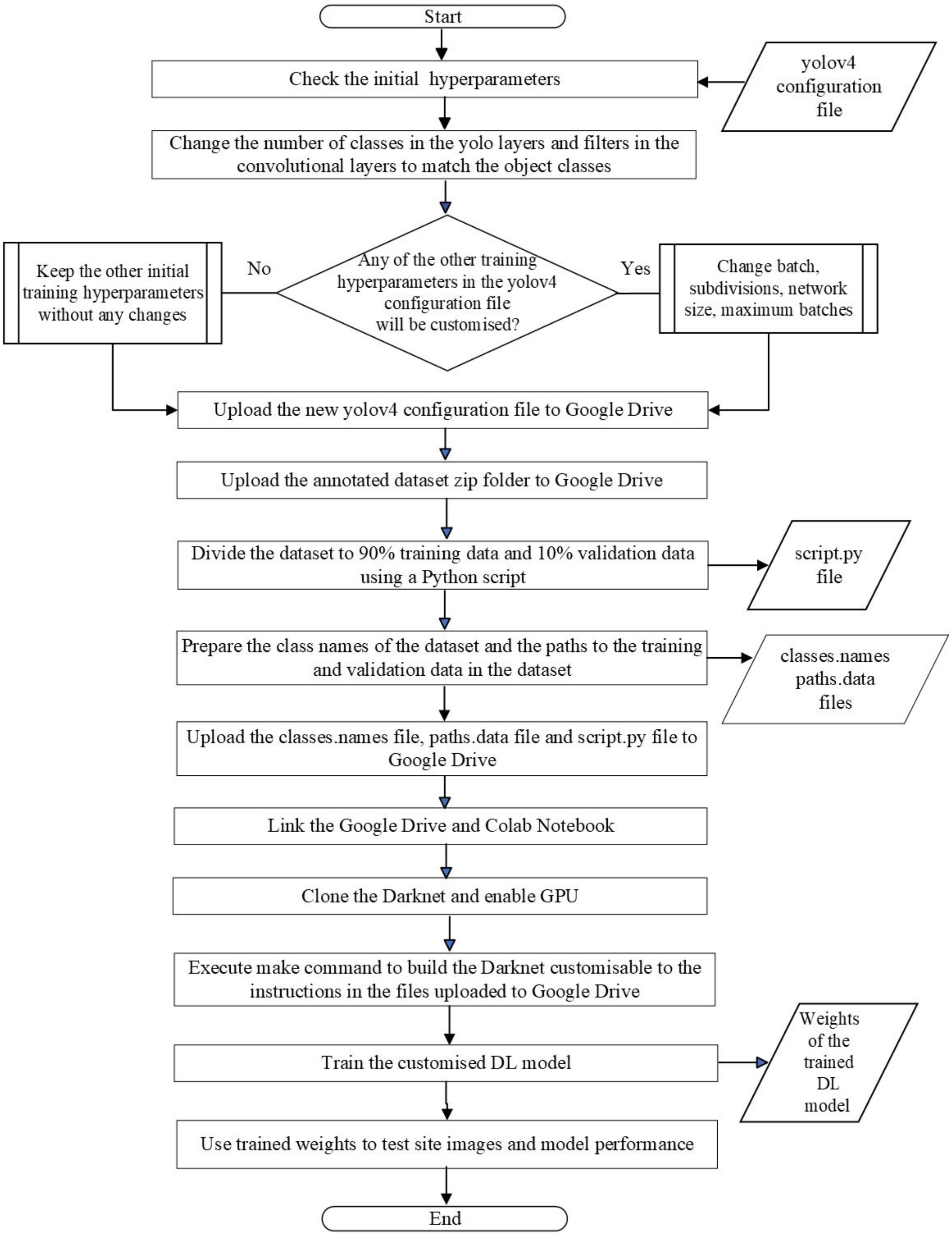

Figure 4: Technical algorithm for DL-based object recognition approach using YOLOv4

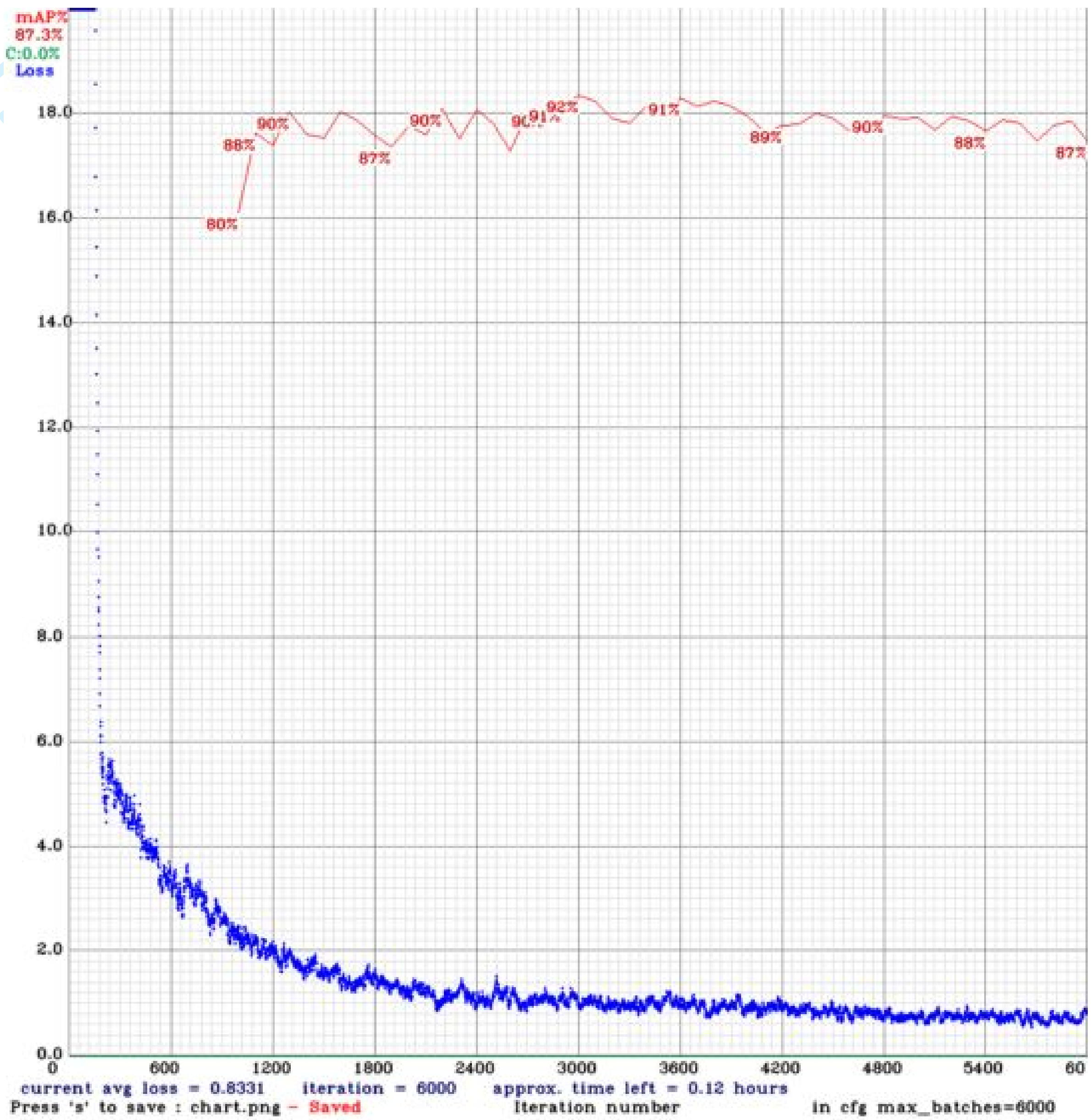

Figure 5: Performance chart of the proposed DL model

a b

Figure 6: Test image for a) insulation and drywall installation; b) framing

a b

c d

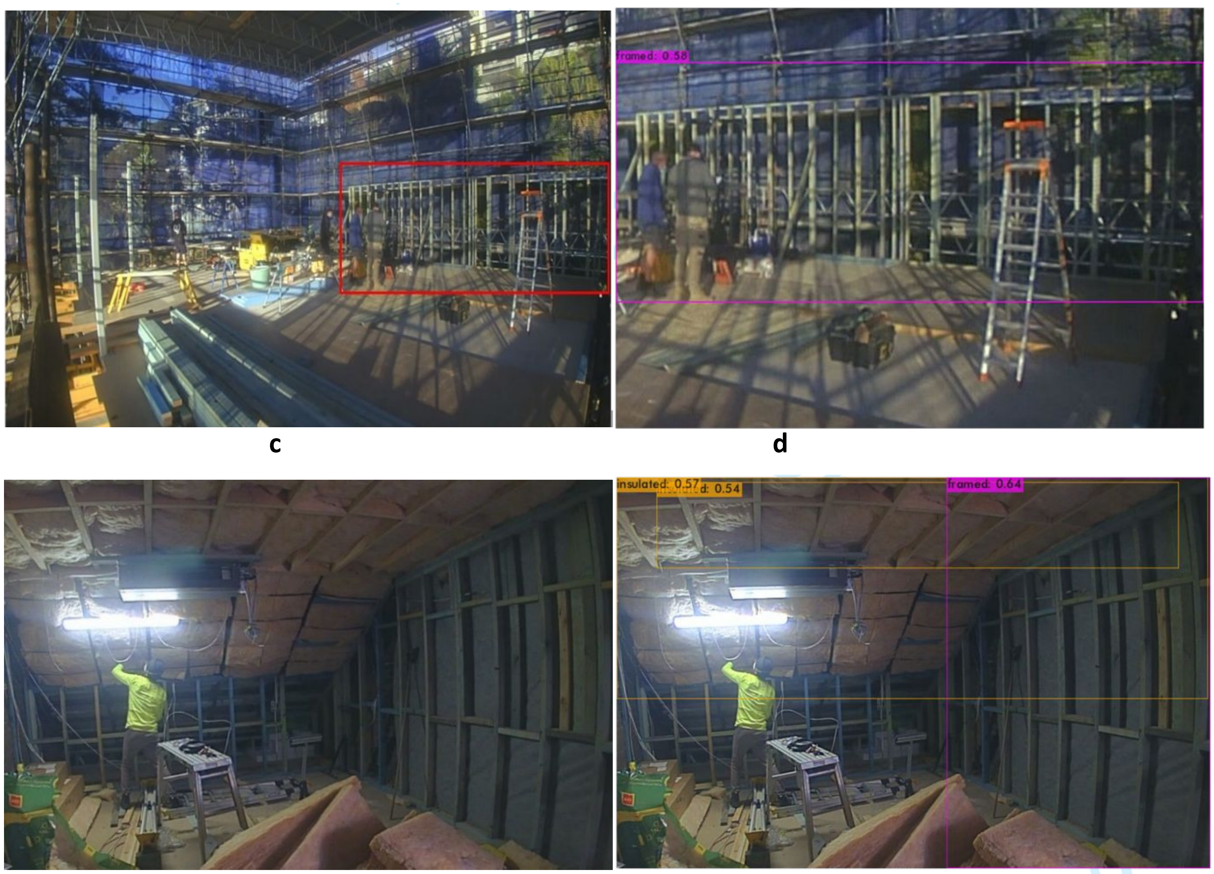

Figure 7: Indoor elements state recognition by overcoming the technical challenges