Ekanayake, B. 2022. A deep learning-based building defects detection tool for sustainability monitoring. In: Sandanayake, Y.G., Gunatilake, S. and Waidyasekara, K.G.A.S. (eds). *Proceedings of the 10th World Construction Symposium*, 24-26 June 2022, Sri Lanka. [Online]. pp. 2-13. DOI: https://doi.org/10.31705/WCS.2022.1. Available from: https://ciobwcs.com/2022-papers/

A DEEP LEARNING-BASED BUILDING DEFECTS DETECTION TOOL FOR SUSTAINABILITY MONITORING

Biyanka Ekanayake[1]

ABSTRACT

To ensure sustainability of buildings, detection of building defects is crucial. Conventional practices of defects detection from building inspection data are mostly manual and error prone. With the advancements in computer vision, imaging technology and machine learning-based tools have been developed for real-time, accurate and efficient defects detection. Deep learning (DL), which is a branch of ML is more robust in automatically retrieving elements' semantics to detect building defects. Although DL algorithms are robust in object detection, the computational complexities and configurations of these models are high. Therefore, this study presents a process of developing a computationally inexpensive and less complicated DL model using transfer learning and Google Colab virtual machine to improve automation in building defects detection. Cracks is one of the major building defects that constraint the safety and durability of buildings thus hindering building sustainability. Building cracks images were sourced from the Internet to train the model, which was built upon You Only Look Once (YOLO) DL algorithm. To test the DL model, inspection images of five (05) buildings collected by the Facilities Management department of a University in Sydney city were used. The DL model developed using this process offers a monitoring tool to ensure the sustainability of buildings with its' ability of detecting cracks from building inspection data in real time accurately and efficiently. Although the current model is built to detect cracks, this process can be employed to automated detection of any building defect upon providing the training images of defects.

Keywords: *Computer Vision; Deep Learning; Defects; Google Colab; Sustainability.*

1. INTRODUCTION

Since the performance of buildings degrades over time, monitoring of defects is imperative to ensure sustainability of buildings (Mishra, et al., 2022). Crack formation, deformation, dampness, staining, settlement, and rebar corrosion are the most common building defects (Hauashdh, et al., 2021; Mishra, et al., 2022). In the operational stage of a building, the major share of the maintenance budget is allocated for preventing or repairing building defects (Ekanayake, et al., 2018; Hauashdh, et al., 2021).

Conventional approaches of detecting defects typically require facility managers or maintenance engineers conducting manual inspections of the building (Kong, et al., 2018). When the building is a mid to high-rise structure, collecting inspection data becomes cumbersome and can even pose safety hazards (Kung, et al., 2021). With the

[1] PhD Scholar, School of Built Environment, University of Technology Sydney, Australia, biyanka.ekanayake@uts.edu.au

advancements in computer vision (CV), unreliable access to inspection data and the time-consuming, labourious, erroneous manual methods of detecting defects have been improved (Lundkvist, et al., 2014; Şimşek, 2022). Computer vision involves acquiring, processing, and understanding image data with the use of high-definition cameras and machine learning (ML) algorithms (Szeliski, 2010) to replicate human vision for object detection.

The automated visual recognition ability of ML has been leveraged for defects detection in some pioneering studies such as Kwon, Park and Lim (2014); Lundkvist, Meiling and Sandberg (2014); Sankarasrinivasan, et al. (2015). However, traditional ML algorithms relying on manual feature extraction for object detection and classification are sensitive to input images, background noise and visual quality (Ying and Lee, 2019; Ekanayake, et al., 2021). Among the advances in CV, the use of deep learning (DL), which is a branch of ML is notable (Wang, et al., 2021). DL models automatically learn features by training data (Nanni , et al., 2017). Although the DL-based defects detection studies such as Chen and Jahanshahi (2017); Dizaji and Harris (2019); Kung, et al. (2021); Munawar et al. (2022) are emerging, the computational difficulties associated with developing and training DL models have not been fully addressed (Ekanayake, et al., 2021; Wang, et al., 2021).

Although DL performs far better than traditional ML algorithms, expensive hardware requirements, DL model training and configurations are constraining the application of DL models to their full potential (O'Mahony, et al., 2019; Wang, et al., 2021). Transfer learning is a solution to develop DL models using pre-trained networks without having to build the DL algorithm from the scratch (Pan and Yang, 2009; Liu, et al., 2021). With the proliferation of cloud computing, cloud providers offer inexpensive services for cloud enabled virtual machines (VM) such as Google Colab for training DL models (Pal and Hsieh, 2021). Therefore, this study presents a process of developing a computationally inexpensive and less complicated DL model using transfer learning and Google Colab virtual machine to improve automation in building defects detection. The DL models require a large amount of training images. Since this paper only intends to provide guidance on the process, it is not practical to train the DL model for all the defects. Among the defects, cracks are prominent (Mishra, et al., 2022). If the cracks are not properly detected at early stages, they can hinder the safety and durability of buildings thus threatening the building sustainability (Şimşek, 2022). Therefore, to demonstrate the process of the DL-based tool, building cracks images sourced from the Internet were used to train the DL model and inspection images of 5 buildings collected by the Facilities Management department of a University in Sydney city were used to test the model.

2. LITERATURE REVIEW

Understanding the impacts of defects on building sustainability, nature of cracks and the application of CV to enable automated visual recognition of defects are discussed in the subsequent subsections.

2.1 BUILDING DEFECTS AND THEIR IMPACTS ON SUSTAINABILITY

A building defect is characterised as a "failing or shortcoming in the function, performance, statutory or user requirements of a building and might manifest itself within the structure, fabric, services or other facilities of the affected building" (Watt, 2009). The growth of aged buildings has caused a surge in repair costs associated with building defects (Park, et al., 2019). While the old buildings with defects are typically unsustainable, in major cities, some newly built high-rise buildings have also proved to be unsustainable. Evidently, in early 2019, occupants were evacuated from two (02) newly built apartment complexes in Sydney due to building collapse risk caused by structural cracks (Crommelin, et al., 2021).

Explaining the relationship between the defects and building sustainability, Hauashdh, et al. (2021) pointed out that defects prevent the buildings from performing their functions, reducing the building life span, and impacting the building occupants' health and safety. Therefore, the social aspect of sustainability is immensely affected. On the other hand, the costs allocated for detecting, repairing, and preventing defects are escalating to create a negative impact on economic sustainability (Park, et al., 2019). The degradation of structural elements because of defects, generates waste and emissions making the buildings environmentally unsustainable (Lee, et al., 2018). Therefore, defects have a vast impact on the sustainability of buildings.

2.2 BUILDING CRACKS

Building cracks can be divided in to structural and non-structural cracks. The cracks in columns, beams, slabs, and footing are considered as structural cracks and may endanger the safety of a building over time (Mishra, et al., 2022). Structural cracks occur due to incorrect design and defective construction of structural elements. They can also be a result of overloading (Kunal and Killemsetty, 2014). Non-structural cracks mostly result from internally induced stresses in building material due to moisture variations and temperature variations. Cracks on walls and parapet walls are some examples. Even though non-structural cracks usually do not compromise safety of the building, they are not aesthetically appealing and can create impression of instability (Thagunna, 2014; Nama, et al., 2015). According to IS: 456 2000 Code of practice on plain and reinforced concrete, cracks are classified based on their width as follows (BIS, 2000).

- Thin cracks - less than 1mm in width
- Medium cracks - 1mm to 2mm in width
- Wide cracks - more than 2mm in width

Figure 1 illustrates the types of cracks.

a) b) c)

Figure 1: Types of cracks - a) thin crack, b) medium crack and c) wide crack

Source: Kunal and Killemsetty (2014)

There have been some studies related to defects analysis, which focussed on visual inspection of defects using the automated visual recognition ability of algorithmic approaches in CV.

2.3 AUTOMATED VISUAL RECOGNITION

Visual recognition by traditional ML approaches encompasses performing object detection in images by manual feature extraction, which is also referred to as handcrafted feature extraction (Ekanayake, et al., 2021). A feature extraction algorithm like Canny edge detector extracts edges that can be used to classify objects detected using edges. The inspection of a wall crack using Canny edge detector, which is further processed by hyperbolic tangent detector is displayed in Figure 2.

a) b) c)

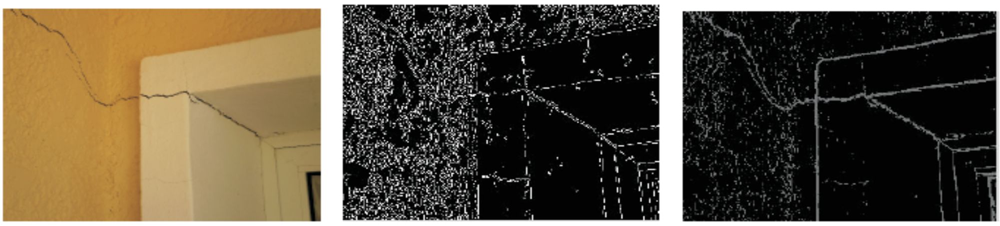

Figure 2: Detection of a crack - a) original image, b) Canny edge detector and c) Hyperbolic tangent detector

Source: Sankarasrinivasan, et al. (2015)

The main drawback of this traditional approach is that it is necessary for the programmer to decide which features are important in each image (O'Mahony, et al., 2019). For example, as in Figure 2, edge is the important feature to be extracted when detecting cracks. In addition to the information on shape, texture and edges of objects, a substantial level of manual pre-processing using algorithms is necessary to remove background noise such as unnecessary data in the image and to enhance visual quality. Instead of applying a single object detection algorithm, the handcrafted feature extraction necessitates conducting additional steps of pre-processing to make the region of interest in the image easily detectable (Ying and Lee, 2019). In recent years, CV-based DL, which is a branch of ML has been evolving rapidly and adopted in the facility inspection and defect analysis context (Marzouk and Zaher, 2020; Liu, et al., 2021). DL models, which are referred to as deep neural networks leverage input-to-target mapping by means of a deep sequence of layers to extract features from input data (LeCun, et al., 2015; Chollet, 2017). While the traditional ML algorithms use a shallow structure, DL network architecture learns the features of an image using a cascade of hidden layers and weights (Slaton, et al., 2020).

In supervised DL, a large amount of annotated data is trained so that the DL model can automatically learn features reducing the manual feature extraction. Objects are detected with rectangular bounding boxes (Wang, et al., 2018). The convolution neural networks (CNNs) are the widespread type of DL neural networks used for image processing (LeCun, et al., 2015; Chollet, 2017). The frameworks of DL-based CNN object recognition methods can mainly be categorised into two types. In one type, region proposals are generated initially, and each proposal is classified into different object categories. Region-based convolutional neural networks (R-CNN) belong to this category (Zhao, et al., 2019). The other type treats object detection as a regression problem, adopting a unified framework to achieve the outcomes. You Only Look Once (YOLO) (Redmon, et al., 2016) algorithm is the best example. Because of the unified framework approach, YOLO models are relatively fast, accurate and simple (Redmon, et al., 2016; Zhao, et al., 2019).

There have been few studies on CNN-based building defects detection. Lin, et al. (2017) built a CNN-based structural damage locations identification system. To facilitate crack detection in nuclear power plant components, Chen and Jahanshahi (2017) presented a CNN framework. Dizaji and Harris (2019) launched a CNN model for detecting surface cracks in concrete columns. In Kung, et al. (2021) and Munawar, et al. (2022), unmanned aerial vehicles (UAVs) were used to capture the defects in mid to high-rise buildings and CNN frameworks were developed for cracks detection.

3. RESEARCH METHODS

The aim of this study is to present a process to develop a computationally inexpensive and less complicated DL model using transfer learning and Google Colab VM for defects detection. This solution is built upon YOLO DL model and tested on cracks images. The subsequent sub sections provide details on preparing the training image database and mechanism behind transfer learning and Google Colab VM.

3.1 PREPARING THE IMAGE DATASET

Since this paper focusses on presenting a process, images were only required to train the DL model. When the number of diversified images is higher, DL model has sufficient features to learn and the accuracy increases (Wang, et al,. 2018). To create a diverse and large dataset of building cracks, publicly available 2000 online images were sourced. The label used was “crack” to annotate each image containing a building crack. To test the DL model developed using the process, inspection images of five (05) buildings collected by the Facilities Management department of a University in Sydney city were used.

3.2 TRANSFER LEARNING

There are two methods to build a DL model. It can either be developed from the scratch or a pretrained model which uses existing networks such as GoogleNet, AlexNet, ResNet (Simonyan and Zisserman, 2014) can be used. In this second approach, a mechanism called transfer learning is adopted to refine the pre-trained model to introduce previously unknown object classes and train the custom model (Torrey and Shavlik, 2010). A DL model can be pre-trained on large dataset of object classes such as Common Objects in Context (COCO) dataset. Using transfer learning, a new model with custom parameters is introduced to produce new weights corresponding to the new object classes. Transfer learning approach is not as much as prolonged and manually intervened as creating a model from the scratch (O’Mahony, et al., 2019). Programmers employ transfer learning to reuse many pre-trained DL models because of the competitive advantages in speed and accuracy (Nalini and Radhika, 2020).

3.3 VIRTUAL MACHINES: GOOGLE COLAB

High computational resources such as graphical processing units (GPUs), high performing memory, processors, and storage are essential for training DL models (O’Mahony, et al., 2019; Wang, et al., 2021). In addition to the hardware requirements, compute unified device architecture (CUDA), and CUDA-based deep neural networks (cuDNN) should be configured for GPU enabled DL model training (Jian, et al., 2013; Jorda, et al., 2019). With the advancements in modern computing systems, GPU-enabled gaming computers, embedded edge computing devices of single board have been used as training platforms for DL models (Pal and Hsieh, 2021). However, the hardware requirements are still expensive, and the configurations are complicated and time consuming. Generally, a functional computer with such hardware requirements costs approximately USD 2000.

With the proliferation of cloud computing and virtualisation, dedicated ML platforms and development environments are available to overcome the hardware and configuration issues (Carneiro, et al., 2018; Canesche, et al., 2021). Among them, Colaboratory (Colab) by Google, Azure Machine Learning by Microsoft and Watson Studio by IBM are prominent. Virtualisation using cloud computing creates a virtual version of the physical computer with a dedicated amount of processer, memory and GPU borrowed from a cloud providers' server. As a result of this, VMs remain independent of the local physical host computer (Rahman, et al., 2022). Colab is the most popular and cost-effective solution to be used as a VM (Pal and Hsieh, 2021). The working mechanism of Colab is illustrated in Figure 3.

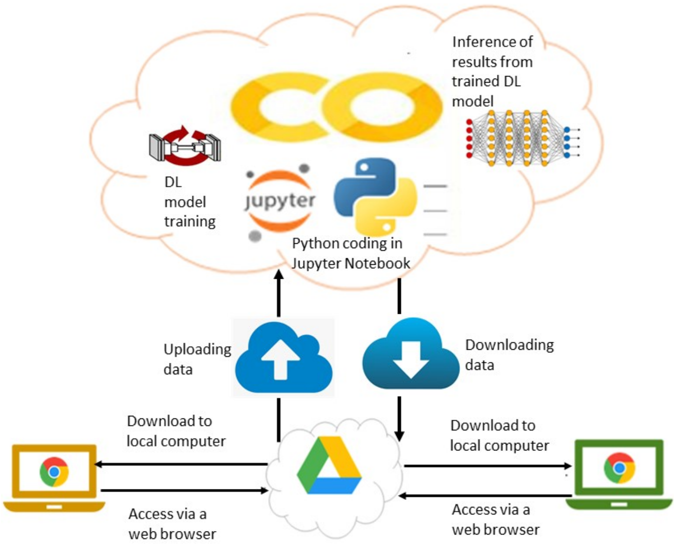

Figure 3: Working mechanism of Colab

Colab is a free-of-charge web-based Jupyter notebook accessed using a Google account to run Python codes. Colab notebooks are stored in Google Drive. Therefore, Google Drive acts as the storage unit accessed from any web browser (Ohkawara, et al., 2021). Colab enables setting up VM as runtime by connecting to GPUs hosted by Google or through Google cloud platform services (Google Research, 2022). Since zero configuration is required and most of the ML libraries are already installed, DL models can be trained in Colab with a few lines of code. Data can be uploaded from the local computer and downloaded to the local computers' hard drive through Google Drive (Pal and Hsieh, 2021).

3.4 THE YOLO VERSION 4 (YOLOV4) ALGORITHM

At the time this paper was written, YOLOv4 (Bochkovskiy, et al., 2020), which was introduced in 2020 has been the most stable and accurate version of YOLO with the optimal speed of detection. Since transfer learning was employed to build the DL model, understanding the mechanism of the YOLOv4 is important to identify the parameters to customise using transfer learning and training in Colab. In YOLOv4, the CNN backbone for object detection is Darknet53 (Bochkovskiy, et al., 2020). The original YOLOv4 model was trained on COCO dataset which comprises of day-to-day general objects of 80 different classes and weights were generated to detect and classify images containing those objects. In the current study, using transfer learning, new weights were trained in Colab for the object class of "crack".

4. PROCESS OF DEVELOPING THE DEEP LEARNING - BASED TOOL

An overview of the process is demonstrated in Figure 4. Initially, the annotated image dataset was prepared with the class label "crack". Next, YOLOv4 were customised using transfer learning followed by DL model training implemented in Colab. To test the cracks detection solution built upon a DL model, test images of building inspection data obtained from a University in Sydney were introduced.

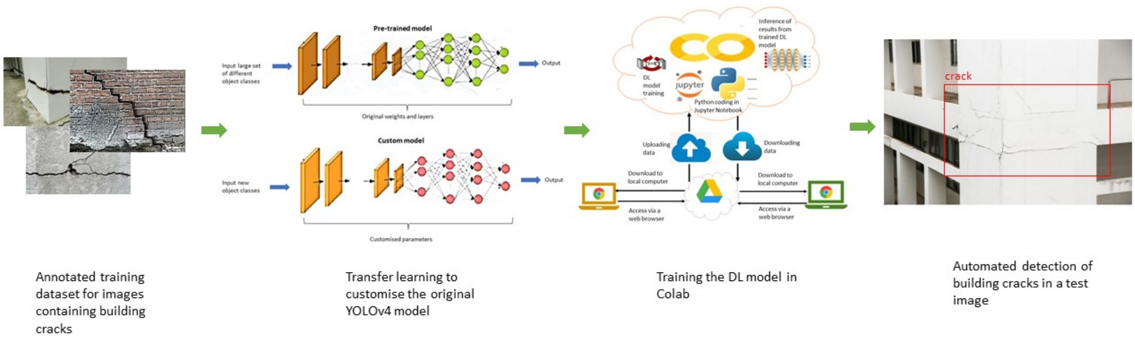

Figure 4: Steps of developing the DL-based solution to detect building cracks

Following are the detailed steps involved in the process of developing the DL-based solution.

- Step 1: Annotating the training images

To annotate the images of building cracks sourced from the Internet, an online annotation tool called *"Makes sense"* was used. The label used was "crack". As per the requirements of the YOLO model, a text file was generated with the details of the annotation for each of the image. Accordingly, 2000 images have corresponding 2000 text files in the training dataset.

- Step 2: Configuring Darknet architecture

To customise the detector backbone of the original YOLOv4 model, the yolov4 configuration file "*yolov4-custom.cfg*" was downloaded from the GitHub repository for YOLOv4, AlexeyAB (Bochkovskiy, et al., 2020). The batch size was changed to 64 images for one iteration and subdivisions were changed to 16, indicating the splitting of batch into 4 mini-batches, such that 64/4 = 16 images per mini-batch. The resolution size chosen for the current model is 416x416 such that the training images were resized to 416x416 pixel resolution. The maximum number of batches was set to 6000.

The parameters of the YOLO layers and the convolutional layers were modified to match the number of object classes. Before each of the YOLO layers, there are three (03) convolutional layers in Darknet architecture. The original Darknet used 255 filters depending on the number of classes of 80. The number of filters were calculated using the formula: (number of classes + 5) x 3. The number of classes were changed to 1 and number of filters were adjusted as 18.

- Step 3: Uploading files to Google Drive

Google Drive was connected to Colab and uploaded with data files needed for transfer learning-based customisation. The customised "*yolov4-custom.cfg"* file and a zip folder containing the images and their corresponding text files with annotation details were uploaded. The Python script containing the instructions to split the dataset into two (02) parts as 90% for training and 10% for validation was uploaded. The names file with the instructions on the class name "crack" and the data file with the instructions on the paths were also uploaded.

- Step 4: Linking the Google Drive and Colab notebook

A Colab notebook was created from the same Google account and was saved in the Google Drive. The runtime was set to GPU and high memory capacity. The *"mount drive"* command was executed to link the Colab notebook and Google Drive.

- Step 5: Cloning Darknet and enabling GPU

Darknet was cloned to Colab from the GitHub repository AlexeyAB followed by GPU enabled in Colab environment.

- Step 6: Building and customising the Darknet

Through transfer learning Darknet was customised using the instructions in the files that were uploaded in Step 3. To facilitate this, the files were copied to the current Darknet directory.

- Step 7: Training the customised DL model

YOLOv4 weights pre-trained on COCO dataset were downloaded from AlexeyAB GitHub. Upon executing the *"train custom detector"* command, as per the changes made in Step 6, weights of the custom YOLOv4 model were generated in every 1000 iteration, until 6000 iterations.

- Step 8: Testing the DL model

After the training was completed, the DL model was tested with building inspection images. Figures 5a and 5b depict how the cracks in the building inspection images were automatically detected by the DL model.

The robustness of the DL model was tested using mean average precision (mAP) and average loss. The metric, mAP is widely used to evaluate the detection accuracy of DL models. The 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 (O'Mahony, et al., 2019). The mAP is portrayed in red colour and average loss is displayed in blue in Figure 6. The best weight of the DL model has a mAP of 76% and the average loss of the model is 1.14.

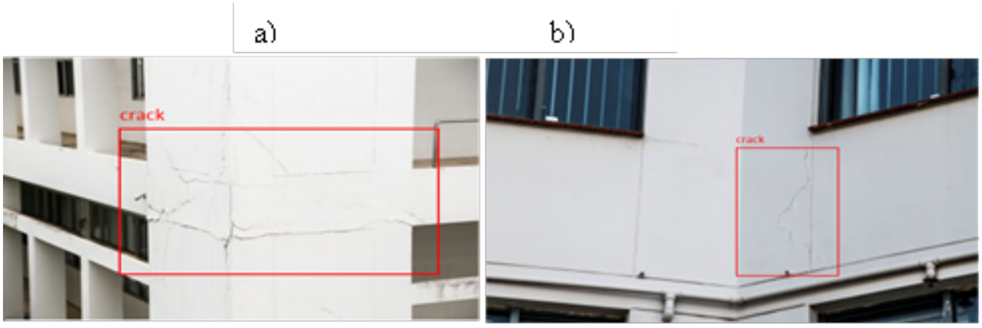

Figure 5: Test images of building inspection data

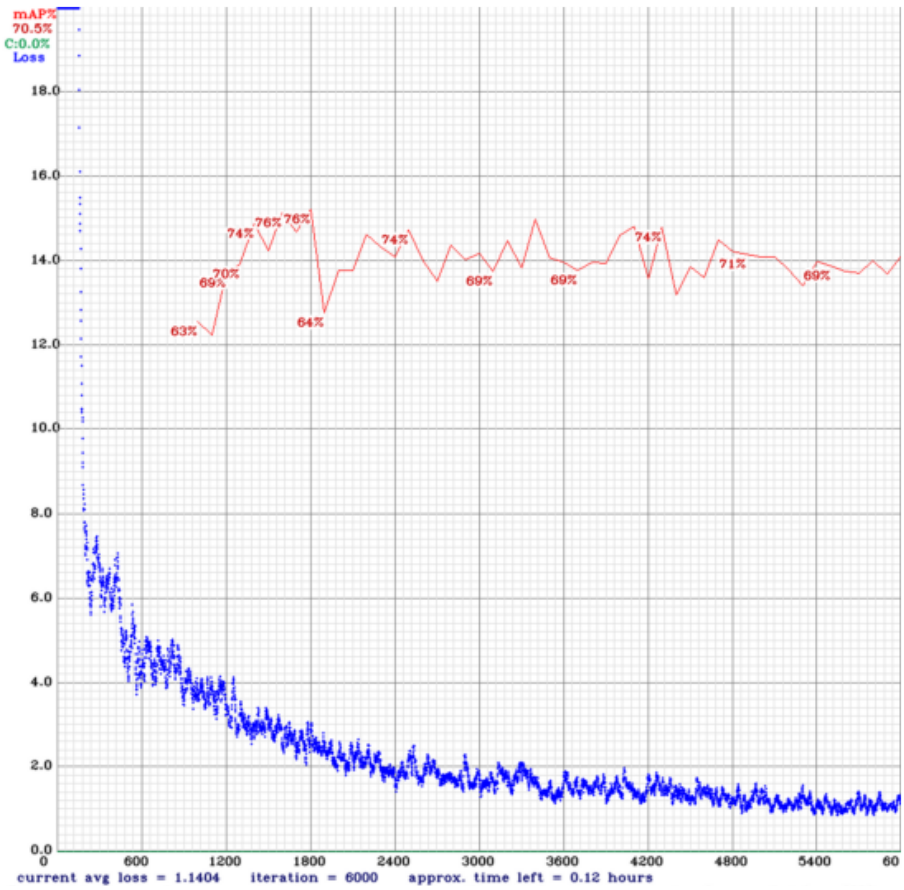

Figure 6: Performance evaluation of the deep learning model

It is noteworthy that these values are subjected to the visual quality of the images, which were sourced from the Internet. If the images are of high visual quality with less background noise, the mAP could be improved, and average loss could be reduced. On the other hand, this model was only trained and tested on images containing cracks to demonstrate the process. Since the process harnesses the DL model's ability in detecting objects, it can be extended to automated detection of any building defect upon providing the training images of the defects.

5. CONCLUSIONS AND RECOMENDATIONS

This paper presents a process of developing a DL-based tool for automated building defects detection. This solution exploits the DL model's ability in detecting objects in images using automated feature learning through training images. Compared to traditional ML algorithms, which use manual feature extraction, DL models proved to be more robust and reduce the manual intervention of pre-processing images. YOLOv4 was used to develop the DL model. Since it was not practical to train the DL model for all types of building defects, this model was only trained on building cracks images. The steps in the process provide guidance on using transfer learning to build DL models from pre-trained networks and training DL models in Google Colab VM platform. Both transfer learning and Colab-based training ensure reducing the computational complexities and expensive hardware requirements associated with DL models.

If defects are not properly detected at early stages, they can hinder the safety and durability of buildings. The DL model developed using this process offers an automated monitoring tool to ensure the sustainability of buildings. To improve automation, building inspection data can be captured using UAVs in mid to high-rise buildings. In terms of sustainable buildings, a step beyond defects detection would be to calculate the width and area of the defects to assess the damages using instance segmentation techniques. There is room for improving the performance of the current DL solution by introducing more training images and optimising the hyperparameters of the YOLOv4 algorithm in future studies.

6. REFERENCES

BIS, 2000. Indian Standard Code of Practice for General Structural use of Plain and Reinforced Concrete IS 456, Bureau of Indian Standards, New Delhi, India.

Bochkovskiy, A., Wang, C.Y. and Liao, H.Y.M., 2020. Yolov4: Optimal speed and accuracy of object detection', *arXiv preprint arXiv:2004*.10934.

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.

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 Collaboratory as a tool for accelerating deep learning applications. *IEEE Access*, 6, pp. 61677-61685.

Chen, F.C. and Jahanshahi, M.R., 2017. NB-CNN: Deep learning-based crack detection using convolutional neural network and Naïve Bayes data fusion. *IEEE Transactions on Industrial Electronics*, 65(5), pp. 4392-4400.

Chollet, F., 2017. *Deep learning with Python*. Manning, New York.

Crommelin, L., Thompson, S., Easthope, H., Loosemore, M., Yang, H., Buckle, C. and Randolph, B., 2021. Cracks in the compact city: Tackling defects in multi-unit strata housing, City Futures Research Centre, UNSW, Sydney.

Dizaji, M.S. and Harris, D.K., 2019. 3D InspectionNet: A deep 3D convolutional neural networks-based approach for 3D defect detection on concrete columns. Nondestructive Characterization and Monitoring of Advanced Materials, Aerospace, Civil Infrastructure, and Transportation XIII, vol. 10971, *International Society for Optics and Photonics*, p. 109710E.

Ekanayake, B., Fini, A.A.F. and Wong, J.K.W., 2021. Technical challenges for automated indoor construction progress monitoring. *44th AUBEA Conference 2021*, Deakin University, Melbourne, Australia.

Ekanayake, B., Sandanayake, Y.G. and Ramachandra, T., 2018. Challenges in hotel building refurbishment projects in Sri Lanka. *In Proceedings of the 7th World Construction Symposium*, Colombo, Sri Lanka.

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*, 127, p. 103705.

Google Research 2022, *Collaboratory,* [Online] Available from: https://research.google.com/colaboratory/faq.html [Accessed 19 January 2022].

Hauashdh, A., Jailani, J., Abdul Rahman, I. and Al-Fadhali, N., 2021. Factors affecting the number of building defects and the approaches to reduce their negative impacts in Malaysian public universities' buildings. *Journal of Facilities Management*, (ahead-of-print).

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*, 64(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*, 7, pp. 70461-70473.

Kong, Q., Allen, R.M., Kohler, M.D., Heaton, T.H. and Bunn, J., 2018. Structural health monitoring of buildings using smartphone sensors. *Seismological Research Letters*, 89(2)A, pp. 594-602.

Kunal, K. and Killemsetty, N., 2014. Study on control of cracks in a Structure through Visual Identification and Inspection. *IOSR Journal of Mechanical and Civil Engineering*, 11(5), pp. 64-72.

Kung, R.Y., Pan, N.H., Wang, C.C. and Lee, P.C., 2021. Application of Deep Learning and Unmanned Aerial Vehicle on Building Maintenance. *Advances in Civil Engineering*, 2021.

Kwon, O.S., Park, C.S. and Lim, C.R., 2014. A defect management system for reinforced concrete work utilizing BIM, image-matching and augmented reality. *Automation in Construction*, 46, pp. 74-81.

LeCun, Y., Bengio, Y. and Hinton, G., 2015. Deep learning. *Nature*, 521(7553), pp. 436-444.

Lee, S., Lee, S. and Kim, J., 2018. Evaluating the impact of defect risks in residential buildings at the occupancy phase. *Sustainability*, 10(12), p. 4466.

Lin, Y. Z., Nie, Z.h. and Ma, H. W., 2017. Structural damage detection with automatic feature-extraction through deep learning. *Computer-Aided Civil and Infrastructure Engineering*, 32 (12), pp. 1025-1046.

Liu, J., Zhang, Q., Li, X., Li, G., Liu, Z., Xie, Y., Li, K. and Liu, B., 2021. Transfer learning-based strategies for fault diagnosis in building energy systems. *Energy and Buildings*, 250, p. 111256.

Lundkvist, R., Meiling, J.H. and Sandberg, M., 2014. A proactive plan-do-check-act approach to defect management based on a Swedish construction project. *Construction Management and Economics*, 32(11), pp. 1051-1065.

Marzouk, M. and Zaher, M., 2020. Artificial intelligence exploitation in facility management using deep learning. *Construction Innovation*, 20(4), pp. 609-624.

Mishra, M., Lourenço, P.B. and Ramana, G.V., 2022. Structural health monitoring of civil engineering structures by using the internet of things: A review. *Journal of Building Engineering*, 48, p. 103954.

Munawar, H.S., Ullah, F., Heravi, A., Thaheem, M.J. and Maqsoom, A., 2022. Inspecting Buildings Using Drones and Computer Vision: A Machine Learning Approach to Detect Cracks and Damages. *Drones*, 6(1), p. 5.

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.

Nama, P., Jain, A., Srivastava, R. and Bhatia, Y., 2015. Study on causes of cracks and its preventive measures in concrete structures. *International Journal of Engineering Research and Applications*, 5(5), pp. 119-123.

Nanni, L., Ghidoni, S. and Brahnam, S., 2017. Handcrafted vs. non-handcrafted features for computer vision classification. *Pattern Recognition*, 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*, 4, p. 200022.

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

Pan, S.J. and Yang, Q., 2009. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, 22(10), pp. 1345-1359.

Park, M., Kwon, N., Lee, J., Lee, S. and Ahn, Y., 2019. Probabilistic maintenance cost analysis for aged multi-Family housing. *Sustainability*, 11(7), p. 1843.

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*, 126, pp. 196-210.

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.

Sankarasrinivasan, S., Balasubramanian, E., Karthik, K., Chandrasekar, U. and Gupta, R., 2015. Health monitoring of civil structures with integrated UAV and image processing system. *Procedia Computer Science*, 54, pp. 508-15.

Simonyan, K. and Zisserman, A., 2014. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409*.1556.

Şimşek, B., 2022. Investigation of self-healing ability of hydroxyapatite blended cement paste modified with graphene oxide and silver nanoparticles. *Construction and Building Materials,* 320, p. 126250.

Slaton, T., Hernandez, C. and Akhavian, R., 2020. Construction activity recognition with convolutional recurrent networks. *Automation in Construction*, 113, p. 103138.

Szeliski, R., 2010. *Computer vision: algorithms and applications,* Springer Science and Business Media.

Thagunna, G., 2014. Building cracks–causes and remedies. *3rd World Conference on Applied Sciences, Engineering and Technology*.

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, J., Ma, Y., Zhang, L., Gao, R.X. and Wu, D., 2018. Deep learning for smart manufacturing: Methods and applications. *Journal of Manufacturing Systems*, 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*, 35(1), p. 04020056.

Watt, D.S., 2009. *Building pathology: Principles and practice*, John Wiley and Sons.

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

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*, 30(11), pp. 3212-3232.