Impact of Traditional Augmentation Methods on Window State Detection

Seunghyeon Wang[a], *Ivan Korolija*[a], *and Dimitrios Rovas*[a]

[a] Institute for Environmental Design and Engineering, University College London, London, UK,
E-mail: seung-hyun.wang@ucl.ac.uk, i.korolija@ucl.ac.uk, d.rovas@ucl.ac.uk.

Abstract. Window state information and changes can help understand ventilation patterns or be used as input in energy models. State identification can be achieved by capturing time-lapse images and processing these through a deep learning model. Deep learning methods have shown reliable performance in object detection tasks such as window and door detection, but have not been applied for window states detection. One of the challenges in setting up such models is to collect a large number of images of window states. In this case, image augmentation can be a critical pre-processing step to enhance the training dataset artificially. Image augmentation has been beneficial in similar contexts and applications. This paper investigates image augmentation methods, adjusting for brightness, scale, and weather. Windows images were used as the starting dataset to demonstrate the proposed methods, and augmented images were artificially generated from the original images. Using the expanded dataset, the Faster R-CNN (faster region-based convolutional neural network) trained a model to detect the binary window states. The augmented dataset model showed better performance than when the original dataset was used. The findings are a testament to the utility of image augmentation methods in the training model of window states detection using deep learning methods.

Keywords. Window states detection; Image augmentation methods; Deep learning
DOI: https://doi.org/10.34641/clima.2022.375

1. Introduction

In residential and non-residential buildings, indoor environments that meet high thermal and visual comfort standards are conducive to occupants' health and productivity. One way of controlling indoor environment parameters are manually openable windows. Operational patterns of each window, also called window states, can be very complex since they can be affected by many behavioural aspects driven by environmental or non-environmental factors (e.g. indoor conditions and personal characteristics) [1]. For example, when occupants feel uncomfortable with the indoor environment, windows may be partially opened, even during the cold season. On the other hand, other occupants may choose to fully close windows if they are sensitive to outside noise.

Knowing the window states and how they vary over time can be helpful in many applications. For example, window states are used as an influential input of energy, ventilation, and lighting simulations. There are two widely used methods for window state detection. One way is to monitor the windows' binary (open or closed) state through the magnetic contacts attached at the interface between the window frame and the wall. However, installing magnetic sensors' in large numbers requires high capital and installation cost, which diminishes the benefits from the available information [2]. Another approach to window state detection is to classify the window states after taking a picture of each building façade from places where window states can be recognisable accurately [3]. As photography can capture many windows simultaneously through a single camera, it is relatively inexpensive. However, time-lapse photography is limited when window states are recognisable using naked eyes, and such recordings can raise privacy concerns. For this reason, this method may be more appropriate in non-residential buildings such as offices and schools during occupied hours (generally 9 am to 5 pm) [4]. In addition, such manual classification of window states from images is time consuming and labour-intensive. In particular, this problem may become more serious when the sampling interval is short, and there are many windows on each façade.

Computer vision techniques can be introduced as an automatic method for analysing the images. While deep learning methods for computer vision have been proved in high generalisation ability to recognise different components such as window and door from building façade images [4], they have not been applied for window states detection yet. As

one of the data-driven methods, deep learning may have billons of parameters to be trained for high generalisation ability and requires a sufficiently large number of annotated datasets [5]. Curation of such large data sets for training the deep learning model can be time consuming and tedious. As an alternative, artificially expanding labelled training datasets from original data, known as image augmentation, could be introduced to address this data scarcity problem [6]. Image augmentation methods change the pixel values to produce artificial images based on original ones. However, computational time and generalisation ability may be increased and decreased, respectively if unrelated augmented data is trained. For this reason, appropriate augmentation methods based on a possible scenario in a real site needs to be applied. However, in previous studies, the data augmentation methods for window states detection have not been investigated yet.

Two image augmentation methods are often used in practice: traditional and elastic. Traditional methods involve various techniques such as rotation, flipping and brightness adjustment and take advantage of simple operations. While they are easy to apply and have low computational requirements [7], they often result in limited variation of the augmented images with regular change of pixel. On the other hand, elastic methods such as Generative Adversarial Network (GAN) generate artificial images using deep learning approaches. They make more substantial inconstant and irregular changes to the original image, but require significant computational resources and the augmented images that can be generated are limited by the available real data [8]. Although there is no doubt that both approaches may be useful for the augmentation of training data, traditional augmentation is considered in this research as a research scope.

In this paper, we propose image augmentation methods for improving the accuracy of the window states detection model based on deep learning. The developed model can detect the hinged and sliding window and its states with binary level (open, closed). The development of model with this binary level is still meaningful since existing automated research for sliding window states detection [9] manually performs localization of window and an automated method for hinged window states with reliable accuracy are not developed yet. Furthermore, given existing studies have still used a magnetic sensor and manual photographic method for 2-discretization level [10], binary window states detection can be utilized practically. To demonstrate the proposed methods, different datasets, which are original and augmented datasets, are created, respectively. Finally, the models with or without proposed augmentation methods are implemented and compared.

2. Image augmentation methods

As influential augmentation methods for window states in this research, the following traditional augmentation techniques: brightness, scale, and weather augmentation are selected. This chapter describes the reasons for selecting these methods with its predicted effectiveness.

2.1 Brightness augmentation

Since it is challenging to distinguish window states for dark lighting conditions, images with enough brightness should be collected. The brightness of each image can be ambient and different for reasons such as time of day and weather when picture of building facades from outdoor environment is taken. In general, images taken on a sunny day are likely to have a relatively higher intensity than those taken on a rainy day. Thus, various images with different lighting conditions are collected in real site. To effectively collect the dataset on this variation of the irregular lighting distribution, brightness augmentation can be applied. This one can artificially change original images into either brighter or darker by increasing or decreasing pixel intensity in original images [11], which is shown in equation 1:

$$I'(x) = I(x) + J(x); \quad (1)$$

where $I'(x)$ and $I(x)$ mean the augmented image that brightness is changed, and original image, respectively; $J(x)$ denotes a matrix having the brightness adjustment factor, which is a same matrix size as $I(x)$. By adding or subtracting $J(x)$ to current intensity of each pixel in original image $I(x)$, the augmented images with different lighting condition are generated. The acceptable intent of changed pixel by the brightness adjustment should be enough for recognition of window states. For example, suppose images of a sunny day with high illumination are collected mostly in whole dataset. In that case, dark day with low illumination is needed to be augmented by decreasing the pixel intensity with appropriate adjustment. However, if the pixel intensity of such a sunny day is highly increased, the visual information for recognising window states may be insufficient.

2.2 Scale augmentation

Various buildings have windows of different sizes. In addition, distance between building façade and photographic vantage point could be different due to visual obstacles such as cars and trees, even though similar buildings are recorded. By this reason, scale variation of building façade images is caused. To address this issue, scale augmentation, which is an affine transformation, can be utilised. The application process of scale augmentation is to first take an image and change its size along the coordinate axis relative to the original one. An

operation of this augmentation can be defined as followed in equation 2:

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} S_x & 0 \\ 0 & S_y \end{pmatrix} * \begin{pmatrix} x \\ y \end{pmatrix} \quad (2)$$

where x' and y' denote the resized coordinates of x and y in a new image, respectively and, S_x and S_y are scaling factors for determining how much coordinates of original image are scaled by entering scaling factors into the matrix. And x and y denote the coordinates of the original image, respectively. Scaled images are newly generated by multiplying x and y by the scaling factors. Images are enlarged or reduced when scaling factors are more or less than 1, respectively. Moreover, the values of directions should be carefully adjusted. Too zoomed in or out images that unlikely to happen in real world observations should be not generated.

2.3 Weather augmentation

The window states detection model can be used for the investigation of different weather effects on window states. It may be possible for different weather such as sunny, cloudy, and rainy days except for the unrecognisable cases where visual information for identification of window states disappears (e.g., heavy fog and torrential rain). Therefore, weather-influenced scenarios should be considered to develop a model robust to variable weather conditions. However, collecting weather data can take a long time. Specifically, rainy days are not frequent in some countries, making assembling these images more difficult. Weather augmentation can help address this problem. The following simple operation, which is called an image blending technique, for making weather-related images can be formally written in equation 3:

$$g(x) = (1 - \alpha) * f_1(x) + \alpha * f_2(x) \quad (3)$$

With $g(x)$ an artificially weather-influenced image, $f_1(x)$ the original clear image, and $f_2(x)$ an image including weather information. Here a is a linear weighting factor. To obtain the image $g(x)$, transferring each source pixel's intensity value related to weather in the image $f_2(x)$ into a pixel position in the clear image $f_1(x)$. For weather image $f_2(x)$, according to the included information in pixels, different weather is represented. For example, a rainy image can be made with the following steps: Gaussian noise following gaussian distribution is created over a binary image, representing black (0) or white colour (1). And the motion blur filter that travels in a single direction horizontally is applied to the created binary image. This filter $h(x, y)$ can be in the form of equation (4) [12]:

$$h(x,y) = \begin{cases} \frac{1}{L}, if\sqrt{x^2 + y^2} \leq L/2, y/x = \tan\theta \\ 0, otherwise \end{cases} \quad (4)$$

Two parameters govern the motion blur filter: length L and angle θ. Length affects the perceived distance that the pixels are moved from their original positions so that a more significant length value will result in more blurring. Angle describes the actual angle of the movement. A setting of $\alpha = 90^o$ will produce a vertical blur, and a setting of $\alpha = 0^o$ will produce a horizontal blur. Like other techniques, ranges of parameters should be properly adjusted for being more realistic rainy image. In addition, too strong weather effects over images make window states unrecognisable with naked eyes.

3. Methodology

This research aims to demonstrate the effectiveness of traditional augmentation techniques (brightness, scale, and weather augmentation) on window states detection. Fig. 1 shows an overview of the experimental process for the demonstration. The augmented datasets are generated by independently applying each augmentation method to original training data. After that, the original and augmented datasets are prepared to train a Faster R-CNN model for detecting the window states on building façade images. Finally, the models based on Faster R-CNN with or without proposed data augmentation methods are compared. In this chapter, the experimental process is described in detail.

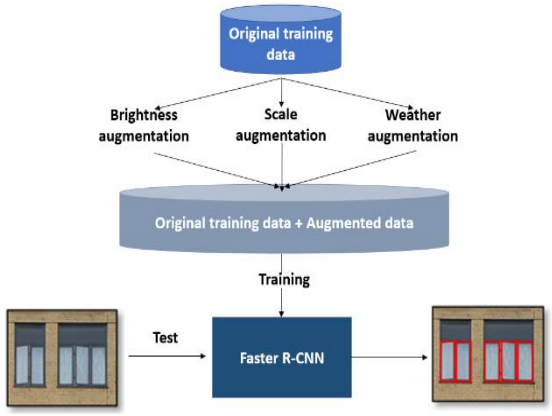

Fig. 1 – Experimental process using different augmentation techniques

3.1 Environmental settings

The simulations were performed on Windows 10 with an Intel Core i7-7700HQ CPU @ 2.80 GHz×8, an NVIDIA GeForce GTX 1080ti GPU and 32G RAM.

3.2 Dataset preparation

In this experiment, only non-residential buildings in the city of London, UK were captured as a case study. Various architectural designs including traditional (e.g. Victorian and Renaissance style)

and modern style (e.g. Sustainable and Postmodern architecture) were included in the dataset. Moreover, there exist different types of hinged and sliding windows including fixed window at the same time (e.g. awning, hopper window and, horizontal, and vertical sliding window). Concerning layout of window, while the most of windows are arranged as a symmetric structure, an unsymmetrical one was also contained. The detailed typology of building façade observed in this research can be downloaded here: (https://doi.org/10.5522/04/14993589)

The augmentation techniques may be ineffective for sample data collection if the collected dataset includes many variations enough for reliable accuracy. However, it is difficult to a priori define the required number of images. In this research, the used device for RGB camera was a smartphone with a polarising filter to reduce the veiling reflections. In total, 750 raw images of building façade with a resolution of 853×1440 pixels were collected, providing sufficient visual information for recognising the binary discretisation level. All images were taken during office hours (from 9am to 5pm) in the summer (from 21 April to 3 May). In addition, since photographs with different angles have severe angle variations, which can have an detrimental effect on the model accuracy [13]. For this reason, building facade images with the frontal view were collected as much as possible. In the case that tilted images were collected due to absence of photographic vantage points, they were rectified manually into frontal images, which is shown in Fig. 22. The manual rectification of building façade image is also time consuming and labour-intensive process.

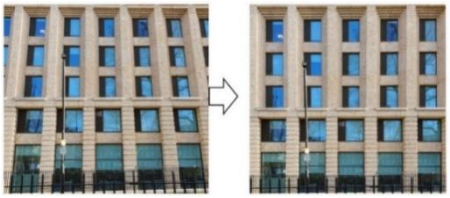

Fig. 2 - An example of rectified images (left: tilted images, right: rectified images)

After the rectification process, the collected whole images were randomly divided into the training (60%) and test (40%) set with 450 and 300 images respectively. Then, defined parameters, brightness, scale, and weather augmentation were applied to only training data. The used hyperparameters were selected after a trial-and-error process ensuring whether images were realistic or not. Created dataset with ranges of parameters is shown in Tab. 11. Firstly, for brightness parameters, pixel intensity between -30 and 30 randomly was added to the pixel intensity of original images. In scale augmentation, original images were randomly scaled up or down to a value of 80% to 120% in x and y-coordinates of their original size. This was performed independently per axis since there can exist a variety of window sizes according to buildings. In addition, the scaled-down images have void space. Since deep learning-based methods require fixed length vectors as input data, the augmented images should be the same size. Therefore, zero-value pixels were padded on the void space in the reduced image when scale is less than 1. In the case of weather augmentation, when images are taken, original images included only rainy images so that among weather, rainy effect was considered. To achieve this, two parameters length L and angle θ, which range from 3 to 7 and 45 to 60 randomly, respectively were selected. Based on 450 original training images, respective augmented image dataset (450 images) from each augmentation method using the described parameters were generated. Examples of the original and each augmented image is shown in Fig. 33. For demonstrating the effectiveness of augmented datasets, five different datasets were prepared: one original training dataset, four augmented datasets (original training dataset with brightness, scale, weather augmentation, and a sum of all augmentation techniques.

Tab. 1 - Created dataset with ranges of parameters

Dataset	Ranges of parameters	Number of images
Original	-	450
Brightness	[-30, 30]	450
Scale	[0.8, 1.2]	450
Rainy	L = [3, 7], θ = [45, 60]	450
Total	-	1,800

The subsequent process is to annotate the window states with binary discretisation level. As a result, localisation of each window as bounding box and binary state of corresponding window as each bounding box's class, which is "background", "opened", or "closed" were annotated. LabelImg [14] was used for this annotation process of ground-truth labels, which generates XML files containing information on the class of each object and the corresponding bounding boxes. In addition, when openable windows are together with fixed window, fixed window needs to be regarded as background since it is not openable, which is always closed states. However, especially distinguishing fixed and hinged window is difficult in certain case. For this reason, fixed windows were annotated as closed window. In addition, to effectively label a lot of augmented images, the following method was applied. In the case of the two methods (brightness and weather), we used the same label information as that of original images because the bounding box position of objects in augmented images does not change. On the other hand, scale transformation is a method to augment images by changing aspect ratio of original images at 2D axis. Thus, the label information representing the bounding box position

of windows were adjusted in accordance with the scale change of corresponding images.

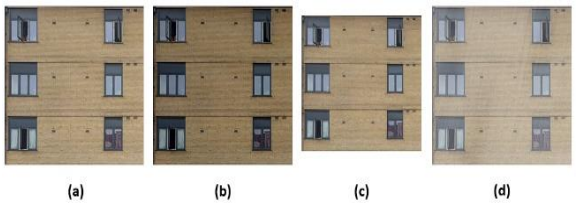

Fig. 3 - Examples of images: (a) original image, (b) brightness augmentation, (c) scale augmentation, (d) rainy augmentation

3.3 Faster R-CNN

To perform the binary window-state detection, window data with localisation and state information is required. To achieve this, object detection, which creates a bounding box for localisation and classification, can be applied. Faster R-CNN [15] has been widely used as an object detection method based on deep learning due to its high generalisation ability in computer vision tasks [16]. In particular, Faster R-CNN has shown reliable accuracy in window detection tasks [17], similar to window states detection. Therefore, Faster R-CNN was selected as a method for the object detection in this research. We used Detectron2 [18], which is a Facebook AI Research's next-generation software system for the implementation of Faster R-CNN.

3.3.1 Architecture

The Faster R-CNN model mainly comprises three components: (1) feature extractor, (2) RPN (Region Proposal Network), and (3) classifier and regressor. The feature extractor extracts features related to a task of window states detection from building façade images using CNN (Convolutional Neural Network). The function of RPN is to generate possible region proposals containing objects with a wide range of aspect ratios and scales using anchors. The classifier and regressor components compute the class probability of identified objects, and each such thing is localised.

The components of Faster R-CNN can have various architectures for specific tasks by modifying the original Faster R-CNN [15]. In deep learning, deeper architectures often have an advantage over shallow ones since the stacking of multiple layers can improve the representational capacity of model [19]. In this research, ResNet-101 was used as an architecture of feature extractor. One main reason is that deeper neural networks in deep learning have a limitation: with the network depth increasing, the accuracy becomes saturated and then degrades rapidly. This issue is called gradient vanishing. During training the model through backpropagation, the gradient of earlier layers is calculated by multiplying the gradients of later layers. When the gradient of later layers is less than one, the gradients in earlier layers close to almost zero. To overcome this problem, ResNet-101 preserves the information of prior layers by adding the output of the last layer to the next, thus allowing the network to remember the previous layer's information [20]. Faster R-CNN models with ResNet-101 developed for many tasks have shown a faster convergence early and a good trade-off relationship between the network speed and accuracy in many research [21]. For classifier, as an objective of this research is to detect binary classifications (open/closed) of window states, the number of output neurons in the last layer were modified into three neurons, including background class. The rest of the architectures including RPN and regressor followed the same original paper [15].

Meanwhile, generally training deep learning architecture, including ResNet-101, requires a lot of data and takes a long time to train a model from scratch. As an alternative, a pre-trained model can be utilised by using a model developed in advance. In other words, once trained using another dataset, the model can use the learned features to perform many other tasks. Moreover, its effectiveness can be more increased when different but more related data is pre-trained to target model tasks [22]. ImageNet [23], which was designed for the academic purpose of computer vision research, could be a pre-trained model. ImageNet is a large database of over 14 million images with many categories, including building façade images. Therefore, in this research, the feature extractor based on ResNet-101 was initialised using pre-training with ImageNet.

3.3.2 Hyperparameter settings

The training process aims to minimise the overall loss in Faster R-CNN. There are many tuneable hyperparameters (e.g. batch size, momentum, and epochs) for training the model. Since hyperparameters have a great impact on the model's performance, their combinations are carefully optimised through experiments. However, the main aim of this research is not to achieve reliable accuracy through careful optimisation but to demonstrate the impact of traditional augmentation techniques on window states detection. In this research, setting the hyper parameters were achieved by modifying the Detectron2 configuration. Expressly, the possible value for the batch size is limited by the available GPU memory. It was set to 2, considering the capabilities of the hardware system used. In the case of epochs, it can significantly affect our research question. Although the training time is shortened with fewer epochs, the original and augmented dataset may be less trained. Thus, 20,000 epochs were considered enough value for demonstrating the proposed method. All other configurations were kept as the default settings from Detectron2. Interested readers can refer to here [18] for detailed configurations.

Tab. 2 – Model performance with different training dataset

Rank	Training Dataset	Final training loss	AP: opened window	AP: closed window	mAP
5	Original images	0.21	0.875	0.905	0.89
3	Original + Brightness	0.19	0.91	0.94	0.925
4	Original + Scale	0.20	0.896	0.93	0.913
2	Original + Rainy	0.17	0.924	0.94	0.932
1	Original + All techniques	0.18	0.94	0.96	0.95

4. Results

To evaluate the performance of proposed augmentation techniques, the overall loss of training set, AP50 (Average Precision), and mAP (mean Average Precision), of test set were used as metrics. These metrics have been used generally in academia to measure object detection performance. Detailed descriptions of each metric can be found in the paper [24]. Tab 2. shows the experimental results. All models were trained enough until reaching the described training loss. A model using only original training data showed reliable accuracy with 0.89 in mAP to some extents. In addition, all augmentation methods improved the detection accuracy of Faster R-CNN. In independent applications of each augmentation method, while weather augmentation relatively more increased the model performance, brightness and scale augmentation relatively less increased one. The most effective method was using original images with brightness, scale, and weather augmentation simultaneously. Compared to only using the original dataset, the model performance in mAP of such method increased from 0.89 to 0.95. Fig. 4 shows the examples of visually detected results in the same scenes of testset using Faster R-CNN with original images (left) and images with all techniques (right). In Fig. 4 (the first row), both models detected all window states correctly with 100% accuracy. As shown from Fig. 4 (the second middle), only a model using all augmentation techniques had 100% accuracy, but the original model regarded opened window states as closed ones. Although a model with all augmentation techniques showed better accuracy than the original one, wrong results were still observed. In Fig. 4 (the third row), while both models incorrectly detected door parts as a closed window, a model using the augmented dataset was less frequently wrong. This result may be caused by one possible reason that images including such similar objects to windows are not included enough in the training dataset.

While augmentation techniques showed better accuracy than the original one, the extent of effectiveness for real applications of proposed

Fig. 4 - Comparison of the visual results using the models trained with different datasets: original images (left), images with all techniques (right)

methods may differ from the collected dataset on window states detection. In the case where a data distribution of train and test dataset in variations is bigger, the certain augmentation may be more effective. For example, weather augmentation may be not useful if the weather condition of application sites is constant (e.g., only summer or winter). Besides, a Faster R-CNN model with all augmentation techniques could achieve a good detection result in most complex scenarios included in an experimental dataset (e.g. intensive lighting conditions, different scales). However, a low generalisation ability was observed in one scenario where similar objects such as doors exist. For practical applications, the generated model may be used practically to investigate window states except for one scenario of similar objects. In addition, although incorrectly detected closed window will always be closed states and its implication on indoor environment might be negligible, the improvement of model in this scenario is required in the future research.

The proposed methods can be reproduced with the following guideline. Deep learning-based models typically are initialised with a random sampling approach before training the weights in trainable layers. For this reason, even with the exact same dataset, different and uncontrolled weight initialisation may yield different models with each performance. However, even if the exact same initialisation for the weights is applied, a lack of reproducibility may still be observed with many reasons such as software versions, implementation variations, and hardware differences.

5. Conclusion

This study described the impact of traditional augmentation techniques (brightness scale and rainy augmentation) on window states detection using deep learning. For demonstrating the proposed methods, window states detection with binary discretisation level was performed using Faster R-CNN. Building façade images (750 images) acquired by the RGB camera were spilt into two sets: a training set (450 images) and a test set (300 images). By applying each augmentation method to only train datasets, five datasets were prepared: the original dataset, and four augmented datasets with brightness, scale, rainy technique, and a combination of all methods. Each augmentation technique showed better accuracy than only using original images. The most effective augmentation method was simultaneously utilising original images with brightness, scale, and weather augmentation. It could improve the accuracy of detection in mAP by 95%, which was higher than a model accuracy of 89% when only the original dataset was used. The result shows that proposed augmentation techniques can be used for simply and quickly creating a large size dataset without much effort of collecting the data in the real sites.

Although the effectiveness of augmentation methods was demonstrated to window states detection, such ways may also be practical for other similar applications such as building age classification and window detection as potential applications. Furthermore, the generated model can be used as a pre-trained model for subsequent future research of window states detection. And, when the annotation of images of window states is required, the generated model can make pre-annotated boxes.

In this research, 95% accuracy was finally achieved through the used dataset and proposed method with defined hyperparameters. More datasets with careful optimisation should be performed to reach more reliable accuracy. As a limitation of this study, elastic augmentation methods such as techniques based on GAN are not considered. It may generate realistically visual information such as texture or colour for window states detection. Through such augmentation methods, further research is needed to demonstrate that window states detection models with more discretised levels (e.g., three or percentage states level).

6. References

[1] S. Wei, R. Buswell, D. Loveday, Factors affecting "end-of-day" window position in a non-air-conditioned office building, Energy Build.2013;62: 87-96. https://doi.org/10.1016/j.enbuild.2013.02.060.

[2] S. Gilani, W. O'Brien, Review of current methods, opportunities, and challenges for in-situ monitoring to support occupant modelling in office spaces, J. Build. Perform. Simul.2017;10:444–470. https://doi.org/10.1080/19401493.2016.1255258.

[3] A. Wagner, W. O'Brien, B. Dong, Exploring occupant behavior in buildings: Methods and challenges, 2017. https://doi.org/10.1007/978-3-319-61464-9.

[4] H. Hu, L. Wang, M. Zhang, Y. Ding, Q. Zhu, Fast and Regularized Reconstruction of Building Fac Ades from Street-View Images Using Binary Integer Programming, in: ISPRS Ann. Photogramm. Remote Sens. Spat. Inf. Sci., 2020;365–371. https://doi.org/ 10.5194/ isprs-annals-V-2-2020-365-2020.

[5] C. Shorten, T.M. Khoshgoftaar, A survey on Image Data Augmentation for Deep Learning, J. Big Data. 2019;6. https://doi.org/10.1186/s40537-019-0197-0.

[6] E.D. Cubuk, B. Zoph, D. Mane, V. Vasudevan, Q. V. Le, Autoaugment: Learning augmentation strategies from data, in: Proc. IEEE Comput. Soc. Conf. Comput.

Vis. Pattern Recognit., 2019;113-123, https://doi.org/10.1109/CVPR.2019.00020.

[7] S.C. Wong, A. Gatt, V. Stamatescu, M.D. McDonnell, Understanding Data Augmentation for Classification: When to Warp?, in: 2016 Int. Conf. Digit. Image Comput. Tech. Appl. DICTA 2016, 2016;1-6 https://doi.org/ 10.1109/DICTA.2016.7797091.

[8] A. Mikołajczyk, M. Grochowski, Data augmentation for improving deep learning in image classification problem, in: 2018 Int. Interdiscip. PhD Work. IIPhDW 2018, 2018;117-122. https://doi.org/10.1109/IIPHDW.2018.8388338.

[9] H. Zheng, F. Li, H. Cai, K. Zhang, Non-intrusive measurement method for the window opening behavior, Energy Build. 2019;197:171–176. https://doi.org/10.1016/j.enbuild.2019.05.052.

[10] H.B. Gunay, W. O'Brien, I. Beausoleil-Morrison, A critical review of observation studies, modeling, and simulation of adaptive occupant behaviors in offices, Build. Environ. 2013;70:31–47. https://doi.org/10.1016/j.buildenv.2013.07.020.

[11] Q. Zheng, M. Yang, X. Tian, N. Jiang, D. Wang, A full stage data augmentation method in deep convolutional neural network for natural image classification, Discret. Dyn. Nat. Soc. 2020. https://doi.org/10.1155/2020/4706576.

[12] S. Jana, Y. Tian, K. Pei, B. Ray, DeepTest: Automated testing of deep-neural-network-driven autonomous cars, in: Proc. - Int. Conf. Softw. Eng., 2018;303-314. https://doi.org/ 10.1145/3180155.3180220.

[13] M. Neuhausen, A. Martin, M. Obel, P. Mark, M. König, A cascaded classifier approach to window detection in facade images, in: ISARC 2017 - Proc. 34th Int. Symp. Autom. Robot. Constr., 2017;690-697. https://doi.org/10.22260/isarc2017/0096.

[14] Tzutalin, LabelImg, Git code (2015). https://github.com/tzutalin/labelImg

[15] S. Ren, K. He, R. Girshick, J. Sun, Faster R-CNN: Towards real-time object detection with region proposal networks, in: Adv. Neural Inf. Process. Syst., 2015;1:91-99

[16] J.C.P. Cheng, M. Wang, Automated detection of sewer pipe defects in closed-circuit television images using deep learning techniques, Autom. Constr. 2018;95:155-171 https://doi.org/ 10.1016/j.autcon. 2018.08.006.

[17] S. Hensel, S. Goebbels, M. Kada, FACADE RECONSTRUCTION for TEXTURED LOD2 CITYGML MODELS BASED on DEEP LEARNING and MIXED INTEGER LINEAR PROGRAMMING, in: ISPRS Ann. Photogramm. Remote Sens. Spat. Inf. Sci., 2019;37–44. https://doi.org/10.5194/isprs-annals-IV-2-W5-37-2019.

[18] Y. Wu, A. Kirillov, F. Massa, W.-Y. Lo, R. Girshick, Detectron2, (2019). https://Github.Com/Facebookresearch/Detectron2.

[19] D. Rolnick, M. Tegmark, The power of deeper networks for expressing natural functions, in: 6th Int. Conf. Learn. Represent. ICLR 2018 - Conf. Track Proc., 2018.

[20] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit., 2016. https://doi.org/10.1109/CVPR.2016.90.

[21] S. Srivastava, A.V. Divekar, C. Anilkumar, I. Naik, V. Kulkarni, V. Pattabiraman, Comparative analysis of deep learning image detection algorithms, J. Big Data. Volume 8, 66, (2021). https://doi.org/10.1186/s40537-021-00434-w.

[22] L. Torrey, J. Shavlik, Transfer learning, in: Handb. Res. Mach. Learn. Appl. Trends Algorithms, Methods, Tech., 2009;242-264 https://doi.org/10.4018/978-1-60566-766-9.ch011.

[23] A. Krizhevsky, I. Sutskever, G.E. Hinton, ImageNet classification with deep convolutional neural networks, Commun. ACM. 2017. https://doi.org/10.1145/3065386.

[24] P. Martinez, B. Barkokebas, F. Hamzeh, M. Al-Hussein, R. Ahmad, A vision-based approach for automatic progress tracking of floor paneling in offsite construction facilities, Autom. Constr. 2021;125.https://doi.org/10.1016/j.autcon.2021.103620

Data Statement

The datasets generated and analysed in the current study are available in the repository (https://doi.org/10.5522/04/14993589).