J. Korean Soc. Ind. Appl. Math. Vol.26, No.4, 311–323, 2023 https://doi.org/10.12941/jksiam.2023.26.311

ANALYSIS OF THE FLOOR PLAN DATASET WITH YOLO V5

MYUNGHYUN JUNG[1,†], MINJUNG GIM[1], AND SEUNGHWAN YANG[2]

[1]NATIONAL INSTITUTE FOR MATHEMATICAL SCIENCES, SOUTH KOREA
Email address: [†]mhjung07@nims.re.kr

[2]DEFINE INC., SOUTH KOREA

ABSTRACT. This paper introduces the industrial problem, the solution, and the results of the research conducted with Define Inc. The client company wanted to improve the performance of an object detection model on the floor plan dataset. To solve the problem, we analyzed the operational principles, advantages, and disadvantages of the existing object detection model, identified the characteristics of the floor plan dataset, and proposed to use of YOLO v5 as an appropriate object detection model for training the dataset. We compared the performance of the existing model and the proposed model using mAP@60, and verified the object detection results with real test data, and found that the performance increase of mAP@60 was 0.08 higher with a 25% shorter inference time. We also found that the training time of the proposed YOLO v5 was 71% shorter than the existing model because it has a simpler structure. In this paper, we have shown that the object detection model for the floor plan dataset can achieve better performance while reducing the training time. We expect that it will be useful for solving other industrial problems related to object detection in the future. We also believe that this result can be extended to study object recognition in 3D floor plan dataset.

1. INTRODUCTION

Image classification is the discipline of recognizing and classifying objects in images. Since the unveiling of AlexNet at the ILSVRC (ImageNet Large Scale Visual Recognition Challenge) competition in 2012, which showed a significant improvement in performance over the previous SOTA (State-of-the-art) model, the field of image classification has made significant progress. The main terms used in this field are:

- Classification: classifying an object in an image as to what it is.
- Localization: For a single object in an image, the location of the object is recognized through a rectangular box, which is called a bounding box.
- Object detection: When multiple objects exist in an image, classification and localization are performed for each object.

Received November 29 2023; Revised December 18 2023; Accepted in revised form December 20 2023; Published online December 25 2023.

2000 *Mathematics Subject Classification.* 68T20.

Key words and phrases. Floor plan dataset, Object detection, Faster R-CNN, YOLO v5.

[†] Corresponding author.

- Segmentation: Recognizing and classifying objects at the pixel level, meaning that not detecting objects with the bounding box having the empty space in object detection but finding and classifying only the area where the object is present.

This paper describes the problem-solving conducted with Define Inc., a client company, and deals with how to upgrade the performance of the object detection model with the floor plan dataset.

Define Inc. is a company in the application software development and supply industry. Its business includes system application software development and supply, software consultancy and development supply, computer equipment consultancy and development supply, and research and development (natural science research).

The client company has been researching using deep learning techniques to create an object detection model that recognizes and detects objects in the floor plan dataset. The model used by the client company is the Faster R-CNN. It has a mAP@60 of 0.89, which is not bad. However, the problem is that it takes a long time to train due to the large size of the images, the large amount of data and the complexity of the model. In addition, there is a lack of confidence that the current method is optimal, due to a lack of understanding of deep learning models. To address these issues, the client company commissioned the verification of the existing model and performance improvement through mathematical data analysis.

2. Related Research

Research in object detection has a long history. In 2001, the Viola-Jones detector [15, 16] by P. Viola and M. Jones was the first to achieve real-time detection of human faces without any constraints. In 2008, the Deformable Part-based Model (DPM) [4] proposed by P. Felzenszwalb won the VOC-07, -08 and -09 detection competitions and is an example of traditional object detection.

Since the emergence of AlexNet [6] in 2012, research on object detection using deep learning has been active, along with the rapid development of deep learning [5]. Figure 1 is the graph showing the increase in the number of published papers in the field of object detection [5].

The operation process of the object detection model using deep learning is mainly divided into region proposal and classification. Region proposal refers to the process of proposing the location of objects in the form of bounding box, and classification refers to the process of identifying the object detected as the proposed bounding box. A model that performs these two processes sequentially is called a two-stage detector, and a model that performs them simultaneously is called a one-stage detector.

Because the two-stage detector performs the above two operations sequentially, it has good accuracy but the disadvantage of slow speed. A model developed to improve this is the one-stage detector, which has the advantage of high speed and can be used mainly for real-time detection. However, compared to the two-stage detector model, it has the disadvantage of poor performance in detecting dense objects and small objects. Table 1 shows typical examples of

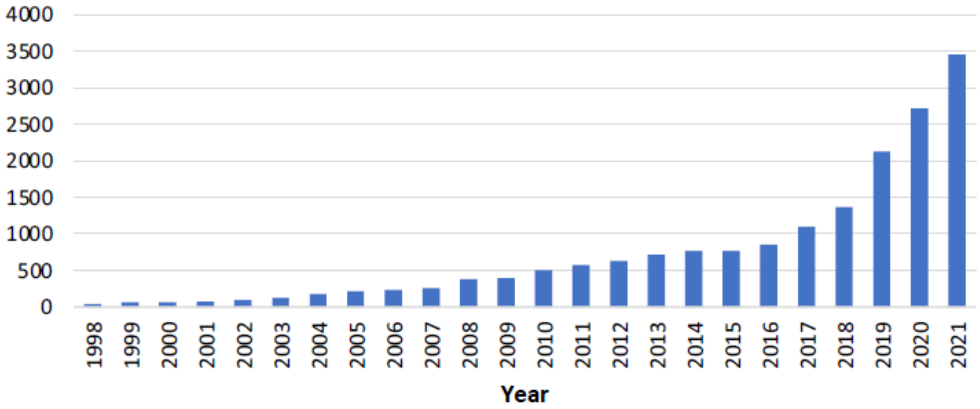


FIGURE 1. Number of publications in object detection [5].

Two-stage detector	One-stage detector
R-CNN [7]	YOLO [12]
SPPNet [8]	Single Shot MultiBox Detector (SSD) [13]
Fast R-CNN [9]	RetinaNet [14]
Faster R-CNN [10]	CornetNet [15]
Feature Pyramid Networks [11]	CenterNet [16]
	DETR [17]

TABLE 1. Examples of two-stage and one-stage detectors [5].

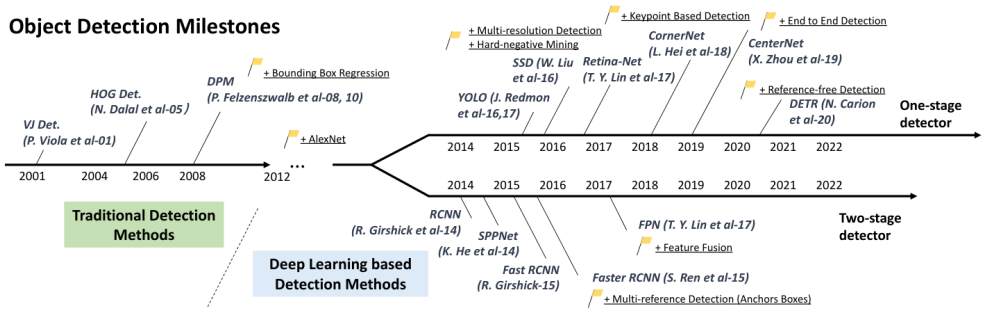


FIGURE 2. Object detection milestones [5].

a two-stage detector and a one-stage detector. Figure 2 is the graph showing the genealogy of the papers that are considered milestones in the field of object detection [5].

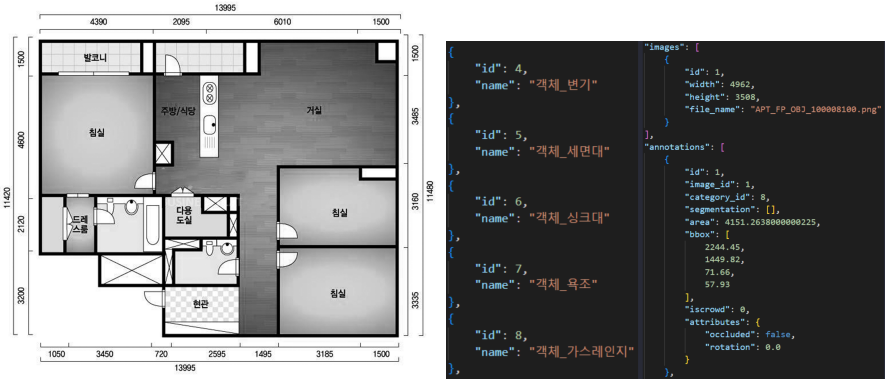


FIGURE 3. An example of the floor plan image and annotation file.

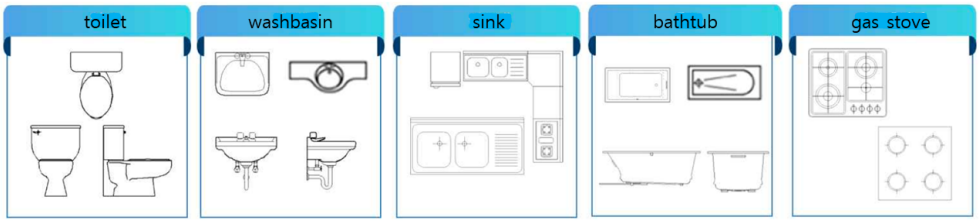


FIGURE 4. Sample images of the objects.

3. PROPOSED METHOD

In this chapter, we introduce the data provided by the client company and the data preprocessing process to solve the problem. It explains the operational principle, advantages and disadvantages of the Faster R-CNN model used by the client company for object detection, and introduces YOLO v5, a deep learning model proposed by us.

3.1. **Data description.** Common types of drawings include layout, perspective, plan, elevation, section, bird's eye view, etc. Of these, the drawing we will use is a floor plan. To build a model that detects objects in the floor plan dataset, we select and train a model with the floor plan images and annotation files containing object information. The object information includes the object name, the category_id of the object, and the location of the object (x, y coordinates of the top left corner and the horizontal and vertical lengths of the bounding box). Figure 3 is an example of an actual floor plan image and part of an annotation file.

The objects we want to detect in the floor plan are a toilet, a washbasin, a sink, a bathtub, and a gas stove. Figure 4 shows sample images of the objects.

Figure 5 represents the visualization of information about objects in a floor plan.

Given a floor plan and an annotation file, you can draw information about objects on the floor plan, but you also want to create a model that can locate, recognize, and classify the object on

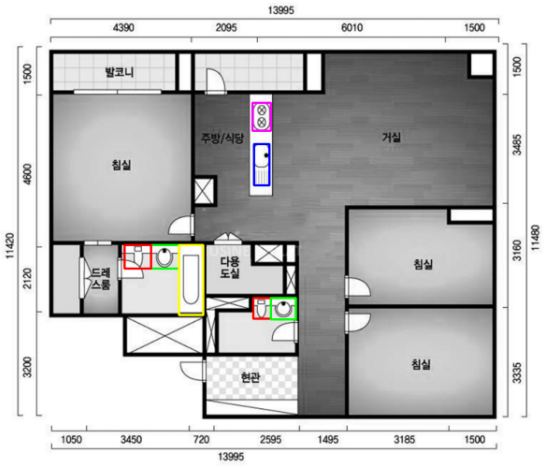


FIGURE 5. Visualization of objects in the floor plan.

its own, even if you only have the floor plan. This is object detection. Here's more about the data:

- File extensions: floor plan images are given as PNG files and annotation files as JSON files.
- File name: house type_FP_OBJ_file index. The housing types are APT (apartment), DEH (detached house), and ROW (row house). FP means 'floor plan' and OBJ means 'object detection'.
  ex) APT_FP_OBJ_000212509.png, DEH_FP_OBJ_989138376.json, ROW_FP_OBJ_970361746.png
- Number of data: 10,127 pairs of floor plans and annotation files.
- Image size: the sizes of 10,047 images are between 4,961 and 4,964 in width and 3,508 and 3,510 in height. The remaining images are of varying sizes and small numbers, so we excluded them from the dataset.
- From the 10,047 data, we excluded the data with missing bounding box name and location information, and finally used 10,041 data. From the final data, we randomly extracted 6,025 training data, 2,008 validation data, and 2,008 test data.
- The total number of each item is as follows: 41,122 toilets, 41,029 washbasins, 20,864 sinks, 19,660 bathtubs and 20,504 gas stoves.

3.2. **Data pre-processing.** The sizes of 10,041 images we'll use range from 4,961 to 4,964 in width and 3,508 to 3,510 in height, but we need them to be the same size to train the model. So we resize all the images to (4961, 3508) using the resize function in the OpenCV library.

We also reduced the image size to 0.3x to solve the problem that the large image size makes the training too long. The value of 0.3x is an experimental value obtained through various tests, taking into account the trade-off between learning speed and accuracy, as the smaller the

Reduction ratio	Original	70%	50%	30%	20%
Training time (seconds/epoch)	5625.5012	3489.8909	2471.8303	1910.0603	1867.5478
mAP@60	0.9097	0.9014	0.8972	0.8904	0.7223

TABLE 2. Change in training time and mAP@60 of Faster R-CNN as a function of image reduction ratio.

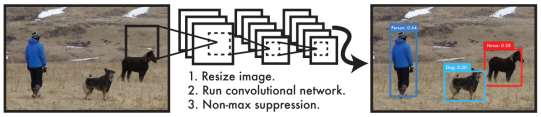


FIGURE 6. YOLO detection system [12].

image size, the faster the training speed, but the lower the accuracy. Table 2 shows that the mAP@60 for Faster R-CNN decreases slowly at the reduction ratio of from 100% to 30%, but does rapidly at the ratio of 20%.

3.3. **Faster R-CNN.** Let's introduce Faster R-CNN, the model that the client used before.

The previous model, Fast R-CNN, uses a selective search algorithm for region proposal, which is the biggest cause of slowdown because it operates on the CPU and causes a bottleneck in the neural network. To solve this problem, Fast R-CNN adopts Region Proposal Network (RPN) at the region proposal stage. It is called Faster R-CNN.

$$\text{Faster R-CNN} = \text{Fast R-CNN} + \text{RPN}.$$

Although the Faster R-CNN model is an improved version over the previous model, improving speed and accuracy, it has the disadvantage of having a complex model structure. So, it takes a lot of time to train and infer.

To address these shortcomings, we searched for a one-stage detector with a simpler structure to improve speed, and came up with YOLO v5, a model that is fast, accurate, and easy to use.

3.4. **YOLO v5.** YOLO, which stands for "You Only Look Once", is a model that restructures object detection as a regression problem so that the image can be viewed once and computed once to process the location, classification, and class probability of the bounding box representing the object. Figure 6 briefly shows the inference principle of YOLO v1 [12]: (1) resize the image to a fixed size, (2) run the convolutional network once, and (3) select only the detection results that exceed the confidence-based threshold.

YOLO v1 uses a neural network with 24 convolutional layers and 2 fully connected layers to compute the entire image only once, resulting in very fast inference speeds. This makes it a popular model for real-time object detection. Table 3 shows a performance comparison of the

Real-time detectors	Train	mAP	FPS
100Hz DPM	2007	16.0	100
30Hz DPM	2007	26.1	30
Fast YOLO	2007+2012	52.7	155
YOLO	2007+2012	63.4	45
Less than real-time			
Fastest DPM	2007	30.4	15
R-CNN Minus R	2007	53.5	6
Fast R-CNN	2007+2012	70.0	0.5
Faster R-CNN VGG-16	2007+2012	73.2	7
Faster R-CNN ZF	2007+2012	62.1	18
YOLO VGG-16	2007+2012	66.4	21

TABLE 3. Comparison of Real-time Systems [12].

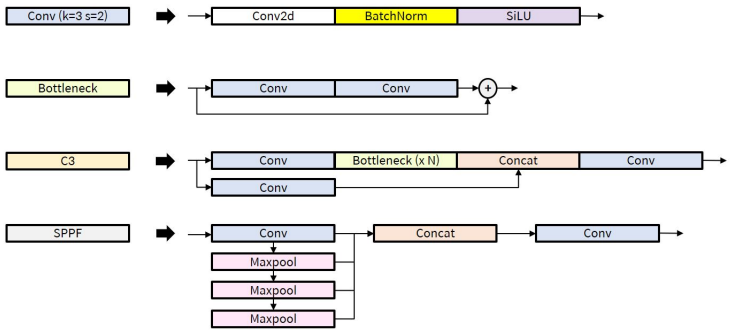


FIGURE 7. Modules of YOLO v5 [19].

models when YOLO model was released [12]. You can see that YOLO has a high mAP and fast FPS.

Our proposed YOLO v5 is the fifth model in the YOLO series. Figure 7 shows the modules of YOLO v5 model structure. Figure 8 shows the model structure utilizing the modules in Figure 7 [19].

3.5. **Description of mAP.** The metric we used to compare the performance of the existing and alternative models is mean Average Precision (mAP). This is the metric mainly used to determine the performance of object detection models. To understand it, we need to know the concepts of Precision, Recall, and Intersection of Union (IoU).

Precision is the percentage of true bounding boxes in positive ones made by the object detection model. Recall is the proportion of positive predictions made by the model in true

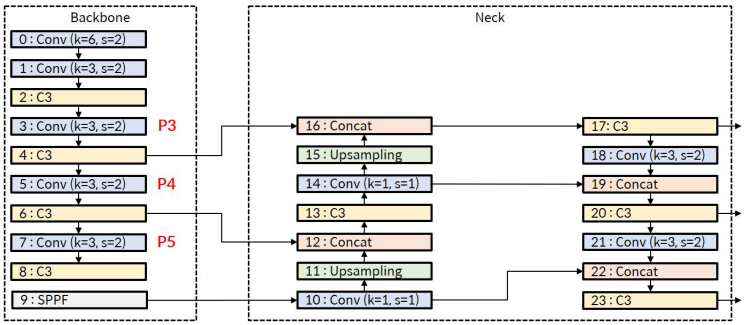


FIGURE 8. Model structure of YOLO v5 [19].

bounding boxes. The formulas for calculating these are as follows:

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \text{ and Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

where TP, TN, and FP means [18]:

- True positive (TP): A correct detection of a ground-truth bounding box;
- False positive (FP): An incorrect detection of a nonexistent object or a misplaced detection of an existing object;
- False negative (FN): An undetected ground-truth bounding box.

IoU is a value that indicates how much the predicted bounding box overlaps with the actual bounding box and is obtained by the following formula:

$$\text{IoU} = \frac{\text{area of overlap}}{\text{area over union}}.$$

To determine whether a predicted bounding box is true or false, we first determine an IoU threshold. If the IoU value exceeds the threshold, the bounding box is determined to be true, otherwise it is determined to be false. Therefore, the TP, FP and FN values will change depending on the IoU threshold. In this paper we use 0.6 as the IoU threshold.

After determining whether a bounding box is true or false, the bounding boxes are sorted in descending order of confidence as shown at table in Figure 9. The graph showing precision versus recall based on the table is called the PR curve (blue solid line). If you interpolate the PR curve (red dotted line) and calculate the area between the graph and the x-axis, it becomes Average Precision (AP). The AP value is calculated for each class and the average of AP values of all classes is defined as mean Average Precision (mAP).

4. RESULTS AND CONCLUSION

4.1. **Results.** Here are the specifications of the computer we used to train the data into the model:

- Intel(R) Xeon(R) CPU E5-2620 v4 @ 2.10GHz

detection	confidence	TP	FP	acc TP	acc FP	precision	recall
R	95%	1	0	1	0	1	0.0666
Y	95%	0	1	1	1	0.5	0.0666
J	91%	1	0	2	1	0.6666	0.1333
A	88%	0	1	2	2	0.5	0.1333
U	84%	0	1	2	3	0.4	0.1333
C	80%	0	1	2	4	0.3333	0.1333
M	78%	0	1	2	5	0.2857	0.1333
F	74%	0	1	2	6	0.25	0.1333
D	71%	0	1	2	7	0.2222	0.1333
B	70%	1	0	3	7	0.3	0.2
H	67%	0	1	3	8	0.2727	0.2
P	62%	1	0	4	8	0.3333	0.2666
E	54%	1	0	5	8	0.3846	0.3333
X	48%	1	0	6	8	0.4285	0.4
N	45%	0	1	6	9	0.4	0.4
T	45%	0	1	6	10	0.375	0.4
K	44%	0	1	6	11	0.3529	0.4
Q	44%	0	1	6	12	0.3333	0.4
V	43%	0	1	6	13	0.3157	0.4
I	38%	0	1	6	14	0.3	0.4
L	35%	0	1	6	15	0.2857	0.4
S	23%	0	1	6	16	0.2727	0.4
G	18%	1	0	7	16	0.3043	0.4666
O	14%	0	1	7	17	0.2916	0.4666

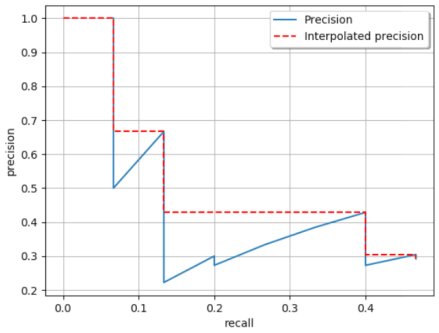


FIGURE 9. Table for mAP calculation and PR-curve from the table [18].

	conf_thre	batch-size	lr	momentum
Faster R-CNN	0.048	2	0.005	0.9
YOLO v5	0.001	4	0.01	0.937

TABLE 4. Parameters used.

	Training time (seconds/epoch)	Training time (seconds/image)	Inference time (seconds/image)	FPS	mAP@60
Faster R-CNN	1910.06	0.32	5.66e-2	17.68	0.89
YOLO v5	546.61	0.09	1.20e-2	83.33	0.98

TABLE 5. Comparison of the performance of Faster R-CNN and YOLO v5.

- NVIDIA TITAN V

For each model, the following parameters commonly used:

- Common: IoU threshold=0.6, optimizer=SGD, weight_decay=0.0005.

Table 4 shows the parameters applied differently for each model.

Table 5 compares the training time (seconds/epoch), inference time (seconds), FPS, and mAP@60 for the Faster R-CNN and YOLO v5 models. YOLO v5 had 71% less training time, 25% less inference time, 20x faster FPS, and 0.08 higher mAP@60 than Faster R-CNN.

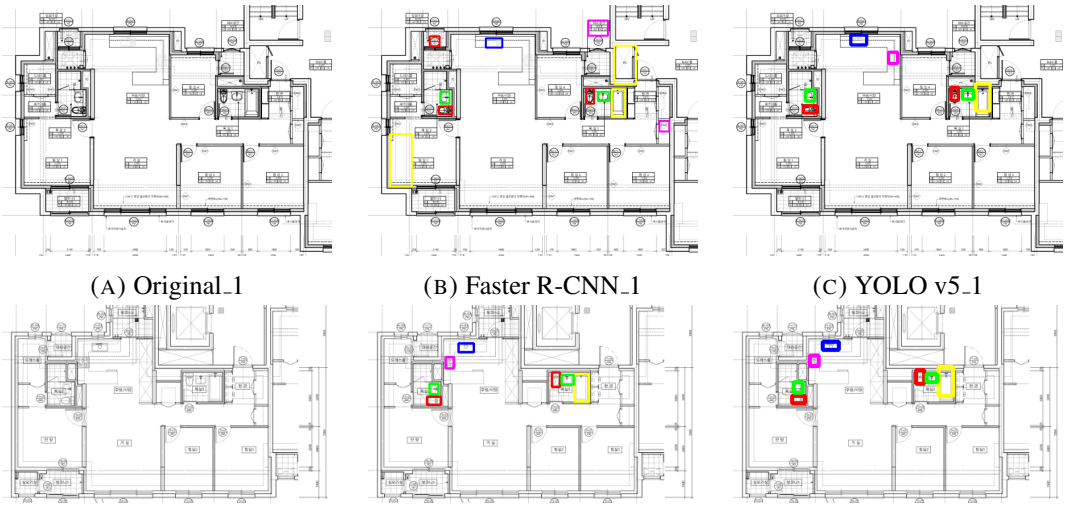
Table 6 compares APs of each class for Faster R-CNN and YOLO v5 models.

Figure 10 shows the original floor plan images and those predictions made by Faster R-CNN and YOLO v5, respectively. The first row represents that YOLO v5 predicted exactly the same as the original data, while Faster R-CNN incorrectly predicted that there were objects in the space without objects.

	Toilet	washbasin	Sink	Bathtub	gas stove
Faster R-CNN	0.91	0.84	0.90	0.93	0.89
YOLO v5	0.98	0.98	0.97	0.99	0.99

TABLE 6. APs for each class.

This doesn't mean that Faster R-CNN always under-predicts. As Faster R-CNN also has a high mAP@60, there are cases where both models predict perfectly, as shown in the second row of Figure 10.



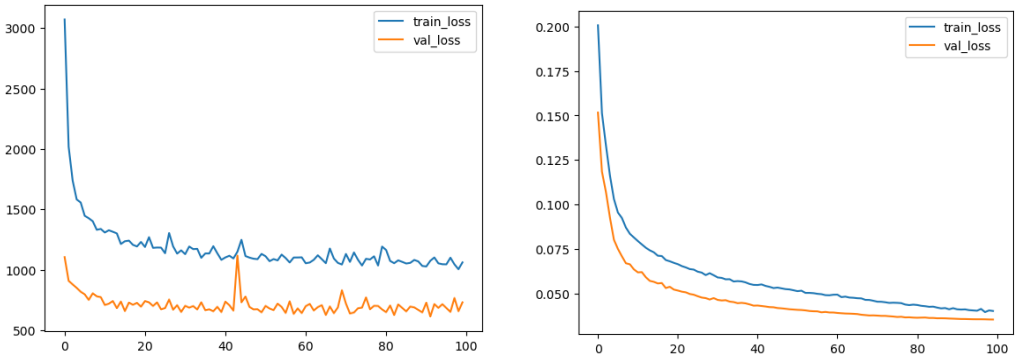
(A) Original_1 (B) Faster R-CNN_1 (C) YOLO v5_1

(D) Original_2 (E) Faster R-CNN_2 (F) YOLO v5_2

FIGURE 10. Comparison of Faster R-CNN and YOLO v5 for test data.

Figure 11 shows the variation of train_loss and val_loss when Faster R-CNN and YOLO v5 are trained up to 100 epoch. Faster R-CNN had the lowest train_loss at 92 epoch and YOLO v5 had the lowest train_loss at 100 epoch. The val_loss of Faster R-CNN reached nearly the bottom at epoch 17 faster than that of YOLO v5. The difference of val_loss of Faster R-CNN at epochs 17 and 92 is not significant. The val_loss of YOLO v5 slowly decreased until it reached around 100 epoch. The reason two graphs have different scales in Figure 11 is that two models have different loss functions for each step of classification, objectness, and location. Table 7 shows loss functions for each step.

Figure 12 shows the change of mAP@60 according to the variation of confidence threshold for Faster R-CNN and YOLO v5. As you can see, both graphs decrease from the beginning. This leads us to choose the lowest conf_thre to obtain the highest mAP@60 values for each model.

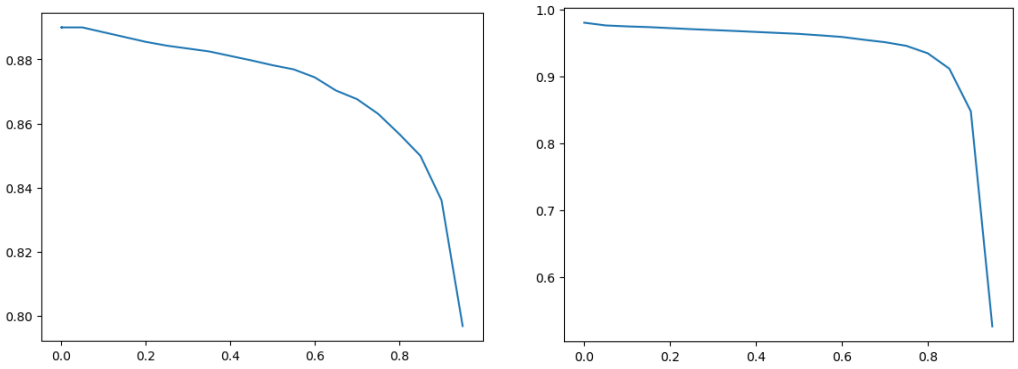


(A) Faster R-CNN (B) YOLO v5

FIGURE 11. Comparison of train loss and val loss of Faster R-CNN and YOLO v5.

	Classification	Objectness	Location
Faster R-CNN	cross entropy	binary cross entropy	smooth L1 loss
YOLO v5	binary cross entropy with logits	binary cross entropy with logits	Complete-IoU

TABLE 7. Loss functions for Faster R-CNN and YOLO v5 [10, 19].



(A) Faster R-CNN (B) YOLO v5

FIGURE 12. Comparison of mAP@60 of Faster R-CNN and YOLO v5.

4.2. **Conclusion.** Through data analysis, data pre-processing and feature analysis of various object detection models, we obtained the results of the problem-solving to improve the performance of object detection models for the floor plan dataset. To solve this problem, we examined the characteristics of the floor plan dataset, the operational principle and characteristics of the

model used by the client company, and identified the related prior research. Faster R-CNN, a model previously used by the client company, had a good level of accuracy but had the disadvantage of taking so long time to train due to the complexity of the model. Therefore, we proposed to use YOLO having a simple structure, which is a representative model of the one-stage detector method, and YOLO v5, which is a stabilised and easy-to-use version. In addition, we found that the large size of the floor plan images made the training time too long. To reduce the training time, we reduced the size of images through data pre-processing. Also, we utilized various parameters to get the high performance of the model.

From this problem-solving, the client company improved the performance of the object detection model on the floor plan dataset. In addition, we were able to understand the operational principle and performance of various object detection models and experienced in applying object detection models to real-world floor plan datasets. As the field of object detection is rapidly developing and related research is constantly being published, we expect to be able to further improve the performance of object detection models by applying new research results to the floor plan dataset in the future. In addition, we plan to further investigate the segmentation of objects in the floor plan dataset by studying models related to instance segmentation, such as UNet and DeepLabV3+ models.

The client plans to utilize the results of this industry problem-solving to

- Reducing inference errors in 2D-based architectural floor plan inference web services.
- Developing a model for optical character recognition in architectural floor plan datasets.
- Recognizing various architectural information, including objects, from 2D drawings and utilize them to generate 3D drawings.

ACKNOWLEDGMENTS

This work was supported by National Institute for Mathematical Sciences(NIMS) grant funded by the Korean government(MSIT) (No.NIMS-B23810000).

REFERENCES

[1] Li, Fei-Fei, et al., *Spatial Localization and Detection.*, CS231n, Stanford, 2016. Retrieved from `http://cs231n.stanford.edu/slides/2016/winter1516_lecture8.pdf`.

[2] Viola, Paul, and Michael Jones, *Rapid object detection using a boosted cascade of simple features*, IEEE, Proceedings of the CVPR 2001, HI, USA 2001.

[3] Viola, Paul, and Michael J. Jones, *Robust real-time face detection*, International journal of computer, **57** (2004), 137-154.

[4] Felzenszwalb, Pedro, David McAllester, and Deva Ramanan, *A discriminatively trained, multiscale, deformable part model*, IEEE, Proceedings of the CVPR 2008, AK, USA 2008.

[5] Zou, Zhengxia, et al, *Object detection in 20 years: A survey*, Proceedings of the IEEE, **111** (2023), 257-276.

[6] Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E. Hinton, *Imagenet classification with deep convolutional neural networks*, Advances in neural information processing systems 25, NIPS, Proceedings of NIPS 2012, NV, USA 2012.

[7] Girshick, Ross, et a, *Rich feature hierarchies for accurate object detection and semantic segmentation*, IEEE, Proceedings of the CVPR 2014, OH, USA 2014.

[8] He, Kaiming, et al, *Spatial pyramid pooling in deep convolutional networks for visual recognition*, IEEE transactions on pattern analysis and machine intelligence, **37** (2015), 1904-1916.

[9] Girshick, Ross, *Fast r-cnn*, IEEE, Proceedings of the IEEE international conference on computer vision, Santiago, Chile 2015.

[10] Ren, Shaoqing, et al, *Faster r-cnn: Towards real-time object detection with region proposal networks*, Advances in neural information processing systems 28, NIPS, Proceedings of NIPS 2015, Montreal, Canada 2015.

[11] Lin, Tsung-Yi, et al, *Feature pyramid networks for object detection*, IEEE, Proceedings of the CVPR 2017, HI, USA 2017.

[12] Redmon, Joseph, et al, *You only look once: Unified, real-time object detection*, IEEE, Proceedings of the CVPR 2016, NV, USA 2016.

[13] Liu, Wei, et al, *Ssd: Single shot multibox detector*, Computer Vision–ECCV 2016, Springer, Proceedings of the14th European Conference, Amsterdam, The Netherlands, 2016.

[14] Lin, Tsung-Yi, et al, *Focal loss for dense object detection*, IEEE transactions on pattern analysis and machine intelligence, **42** (2018), 318 - 327.

[15] Law, Hei, and Jia Deng, *Cornernet: Detecting objects as paired keypoints*, Proceedings of the European conference on computer vision (ECCV), Munich, Germany 2018.

[16] Zhao, Zhong-Qiu, et al, *Object detection with deep learning: A review*, IEEE transactions on neural networks and learning systems, **30** (2019), 3212-3232.

[17] Carion, Nicolas, et al, *End-to-end object detection with transformers*, Cham: Springer International Publishing, Proceedings of the European conference on computer vision (ECCV), online, 2020.

[18] Padilla, Rafael, Sergio L. Netto, and Eduardo AB Da Silva, *A survey on performance metrics for object-detection algorithms*, IEEE, Proceedings of 2020 international conference on systems, signals and image processing (IWSSIP), Niteroi, Brazil, 2020.

[19] Jocher, Glenn, et al. Ultralytics/yolov5: V5.0 - Yolov5-p6 1280 Models, AWS, Supervise.ly and Youtube Integrations. v5.0, Zenodo, 2021, doi:10.5281/zenodo.4679653.