Turkish Online Journal of Qualitative Inquiry (TOJQI)
Volume 12, Issue 6, July, 2021: 10209 - 10219

Research Article

Pedestrian Recognition Technology Using YOLO

Min-Hui Heo[1], Jin Lee[1], Dong-Jin Shin[2], Yong-Soo Lee[3], Jeong-Joon Kim[4*]
[1]Department of Computer Mobile Convergence, Gyeonggi University of Science and Technology, South Korea
[2]Department of Computer Engineering, Anyang University, South Korea
[3]Professor, Department of Software and Communications Engineering, Yeoju Institute of Technology, South Korea
[4*]Assistant Professor, Department of ICT Convergence Engineering, Anyang University, South Korea
hmh4902@naver.com[1], ing013@naver.com[1], djshin@ayum.anyang.ac.kr[2], diclee@yit.ac.kr[3], jjkim@anyang.ac.kr(Corresponding Author)[4*]

Abstract:

Recently, the performance of pedestrian recognition techniques has been improved rapidly with the introduction of deep learning, and the scope of their use has also been widely used in various fields. Currently, most pedestrian recognition technology focused on is detecting a standing person. These general pedestrian recognition techniques are inappropriate for situations where detailed pedestrian information is needed, such as complex traffic conditions, child protection zones, and disasters that make it difficult to identify children. Well-known algorithms for pedestrian recognition techniques include Faster R-CNN, YOLO, and SDD. In this paper, we use the Yolo algorithm to differentiate between adults and children among pedestrians. Furthermore, we confirm the change in detection results with the number of learning images by gradually increasing the number of learning images, and propose an improvement method to improve the detection accuracy of pedestrians.

Keywords: *Yolo, pedestrian recognition, Image Detect*

I. Introduction

As deep learning has been combined with AI, one of the main technologies of the Fourth Industrial Revolution, pedestrian recognition technologies, including computer vision, are being used in various fields. Pedestrian recognition technology is being used in safety protection systems to identify pedestrians in real time on devices capable of filming images such as CCTV and to inform them when a risk is detected. However, although it is still determined whether the object is a pedestrian, detailed research on detecting objects such as adults, children, the elderly, and the disabled is still insufficient. Therefore, in this paper, we attempt to distinguish pedestrians using Yolo, which has recently been in the spotlight among various image recognition algorithms. The used version is v3, and the photos required for learning were collected by implementing a crawling program, and the tools used for labeling were Yolo-produced Yolo_Mark. Paper is composed as follows: Chapter 2 introduces a study related to the algorithms used in this study. Chapter 3 describes the labeling method and main code for object recognition in this work. Finally, Chapter 4 discusses the conclusions of this study and the direction of improvement in future research.

II. Related Technologies and Research

2.1. Labeing Operatins

There is a HOG detector as a basic method of pedestrian recognition technology. The HOG detector divides the area of the object into a constant cell format, obtaining a histogram of the orientation of pixels with a constant magnitude of

slope or higher for each cell, and then extracting the distribution features to identify pedestrians.[1] Traditional HOG detectors have a slow drawback, and there is a Cascade HOG method that improves this. Cascade HOG is a method of applying cascade technology to a HOG detector, which computes the HOG from a variety of blocks and selects the appropriate blocks through boosting learning.[2]

2.2. YOLO

yolo (You Only Look Once) is an algorithm for detecting multiple objects and predicting images using a representative single network.[3] Algorithms can predict the type and location of objects, with single convolutional probabilities of bounding boxes and class probabilities within images trained around the center of the grid using grids of the same size. Based on the prediction box, we calculate the reliability from the learned image to identify the categories by selecting objects with high object reliability.

2.3. Yolo labeling

Data labeling refers to the process of categorizing human-made photos, documents, etc. into categories using data processing tools and deriving location coordinate data information values from objects for images. YOLO labeling uses YOLO mark, a subprogram of YOLO, as a process of collecting learning data and labeling them to generate individual models based on YOLO. yolo mark is a labeling tool for learning directly yolo's image data information[4] The labeling tool labels objects in the image to be learned by drawing them with bounding boxes by constructing the center coordinates x, y, and bounding box width and height w, h.

2.4. Related Studies

Based on video analysis, there is a study that uses biometric functions to classify children/adults.[5] This study uses the Haar cascade algorithm, which aims to classify and detect children and adults by measuring biometrics considering the relative

proportional length of the head and body. The results of the study show a slightly higher accuracy for adults, with 100% and 64.5% for children and adults, respectively. Furthermore, there is a study that constructs and learns a novel artificial neural network model based on Deep Learning YOLO's artificial neural network model, which belongs to a public data portal.[6] This study aims to implement and study real-time pedestrian detection and alarm systems that can be linked to intelligent transportation systems. The results of this study implement My_Tiny_Model3 artificial neural network model with higher detection accuracy and optimized real-time processing than conventional Yolo models.

In addition, there is a study based on YOLO v3 that implements real-time license plate character recognition and model recognition systems using virtual data.[7] As a result of this study, license plate characters and vehicle recognition systems are divided into YOLO v3 tiny and YOLO v3 to reveal their respective accuracy.

III. YOLO Labelling Operations

Labeling based on the YOLO algorithm used in this paper is a necessary process to create a weight file for object detection. The tool that provides labeling based on the YOLO algorithm is the YOLO mark. To build a labelling environment using GPUs in Yolomark, we installed CUDA 8.0 and opencv 3.2.0 versions available on Windows 10 and GTX1060, and Visual Studio 2015 compatible with Yolov3. If you open yolo_mark.vcxproj, you can see where CUDA is used. It modifies the corresponding code and connects it with CUDA to establish an environment where GPU is used. To build an environment where opencv is available in yolo_mark.sln, we edit the user variable Path for Rey to create a path to access opencv. When a path is set and a directory is added, the environment for building the program is configured.

3.1. Labeing Operatins

Before proceeding with the labeling task, we modify the yolo-cfg file to determine the number of objects to be learned and change the value of the filter according to the number of objects. The filter value is 5 * (number of object classifications + 5). Choose pictures of objects under 40 years of age and infants to learn put them inside the /Release/data/img folder. Open the obj.data file that exists inside the x64/Release/data folder and save the classes as few as the number of object classes to learn. It also enters and stores the classification names Kid and Adult of the objects to be learned in the obj.names file. The object id of the object is given in order from zero in the object name stored in the obj.names file. This paper is designated as Kid No. 0 Adult No. 1

After successful construction of yolo_mark.sln, yolo_mark.exe is created in the x64/Release folder, and yolo_mark.cmd can run to proceed with labeling as shown in Figure 1.

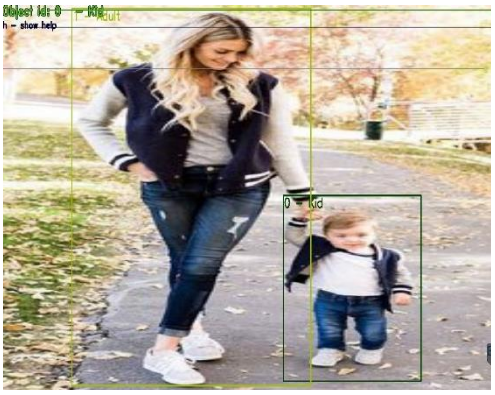


Fig 1. adult & kid is Labeled appearance

The labeling opera selects the desired learning object among the objects in the picture and draws a bounding box, so only the objects that exist in the bounding box are used for learning. Select the object id you want to learning is draw a bounding box, obj.names the location coordinates of the bounding box (four vertex) with the object name you specified in the names file are created in the img folder as the object photo name.txt file, At the same time, a path to the photo file located in the data/imge file is created in the train.txt file. Afterwards, will proceed with the learn by dividing the object image into grids around the position coordinates written in the .txt inside the img folder.

In this paper, the criteria for images set to clearly indicate the distinction between adults and children are kid/adult Whole body, adults are adults in their 40s under, child was chosen as an infant.

The reason chose a kid as an infant is because in the process of defining the characteristics of adults and children, with infants or older were not clearly recognized after learning and two objects were confused, becoming they were recognized kid and adult at the same time so chose to be an infant.

3.2. Ckeck labeling

Related files after labeling are the img directory, obj.data, obj.data obj.names, train.txt, yolo-obj.cfg file exists.

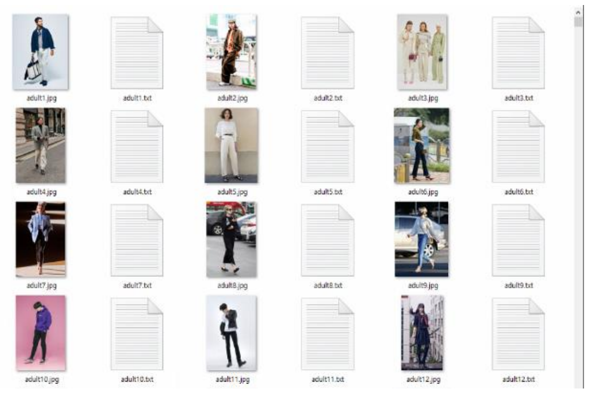
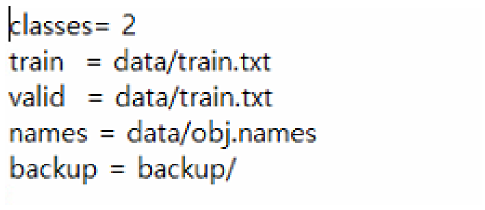


Fig 2. The img directory used for yolo training

As shown in Figure 2, the img directory contains photo needed for labeling, and when the object id of the labeled object and the object name with the coordinates of the corresponding bounding box are generated when, indicating that the bounding box was drawn normally.

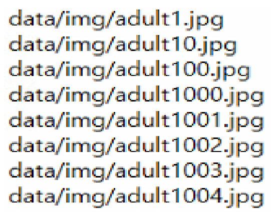


```
classes= 2
train  = data/train.txt
valid  = data/train.txt
names = data/obj.names
backup = backup/
```

Fig 3. obj.data file for configuration of Yolo_mark

As shown in Figure 3, the obj.data file contains the number of classes to classify, train.txt that contains the location of the data to be learned, Path to the names file where the object's name is stored, location of the backup copy file that will be saved when the weights file is created.

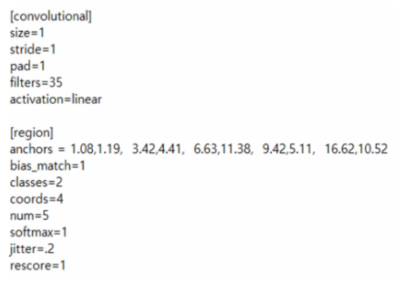
obj.names file is a file that stores the name of the object to be learned. object id is generated from zero in the order in which it is written in the file, and when it is labeling, it draws a bounding box by distinguishing objects through that object id. object id is generated from zero in the order in which it is written in the file, and when it is labeling, it draws a bounding box by distinguishing objects through that object id.



```
data/img/adult1.jpg
data/img/adult10.jpg
data/img/adult100.jpg
data/img/adult1000.jpg
data/img/adult1001.jpg
data/img/adult1002.jpg
data/img/adult1003.jpg
data/img/adult1004.jpg
```

Fig 4. train.txt to save the labeled image path file.

while assigning a bounding box to the object through a labeling making as shown in Figure 4, train.txt file to the determined saves the path of the object picture Automatically. Stored paths are used to load images as they progress through learning.



```
[convolutional]
size=1
stride=1
pad=1
filters=35
activation=linear

[region]
anchors = 1.08,1.19,  3.42,4.41,  6.63,11.38,  9.42,5.11,  16.62,10.52
bias_match=1
classes=2
coords=4
num=5
softmax=1
jitter=.2
rescore=1
```

Fig 5. yolo-obj.cfg for setting of yolo

The yolo-obj.cfg file is a file that defines blocks unit in Yolo's layout, as shown in Figure 5, this is a setup file for labeling. Before starting labeling, filters and classes must be modified to label the desired number of objects. The classes number, which specifies the number of objects, were modified by setting to kids and adult, two, and the filter value described in this section 1 was calculated according to the number of objects and entered below [convolutional], filters as 35.

3.3. Course of study (maincode)

The Learning is proceeded using the Darknet Yolov3 algorithm.

Yolomark's labelling information files img directory, train.txt, obj.names, obj.data, yolo-obj-cfg files is move to Darknet Yolo file and proceed with the learning. Depending on the environment, the batch of yolo-obj-cfg with the amount of RAM needs to be modified to a lower level.

Figure 6 shows darknet.c, the main code of the algorithm used in the learning process.

1	#include "darknet.h"
	#include <time.h>
	#include <stdlib.h>
	#include <stdio.h>
	#if defined(_MSC_VER) && defined(_DEBUG)
	#include <crtdbg.h>
	#endif
	
	#include "parser.h"
	#include "utils.h"
	#include "dark_cuda.h"
	#include "blas.h"
	#include "connected_layer.h"

2	extern void run_detector(int argc, char **argv);

3	} else if (0 == strcmp(argv[1], "detector")){
	run_detector(argc, argv);
	} else if (0 == strcmp(argv[1], "detect")){

```
float thresh = find_float_arg(argc, argv, "-thresh", .24);
        int ext_output = find_arg(argc, argv, "-
ext_output");
char *filename = (argc > 4) ? argv[4]: 0;
test_detector("cfg/coco.data", argv[2], argv[3], filename,
thresh, 0.5, 0, ext_output, 0, NULL, 0, 0);
}
```

Fig 6. Main code required for learning

Figure 6 shows the main code used to detect objects in darknet.cs, which exists in the src folder inside the darknet. 1) This is where the library required is loaded when using darknet. Bring in darknet.h and CUDA. 2) When turning learning around, The first part to start is to load an externally existing detector file. 3) When using detector, the part where conditions can be modified can be adjusted to –thersh of Yolov3. In this paper, more than 0% of objects are found to be found set to display.

At the same time as we proceed with the learning, a graph indicating the progress of the learning is raised and the average loss of object classification for the current learning progress is known. When the learning is completed, a weights file is stored in the backup folder in 1000 units. Weighing files allow for Darknet Yolov3 algorithm-based object detection.

3.4. Learning command language

After completing the labeling process, the learning command language for generating Weights files are as follows.

at "darknet.exe detector train data/obj.data yolo-obj.cfg darknet53.conv.74" darknet.exe means a prebuilt darknet executable included in Yolo3, To a

prebuilt Darknet executable included in Yolo3, and among the parameters used, among the parameters used, train calls a function that means proceeding with the learning., and specifies the path to the previously described obj.data and yolo-obj.cfg files.

The following parameter, darknet53.conv.74, is a darknet pretrain model provided by the manufacturer.

IV. Result

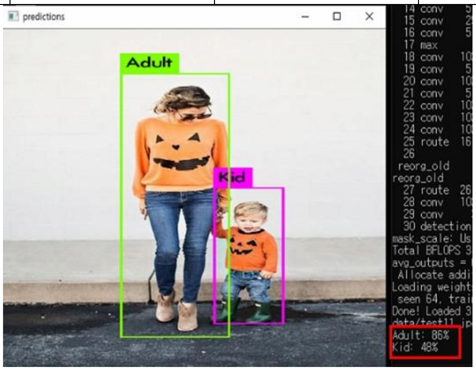
4.1. Object Detection Command

The following command is used to detect objects using a weight file: darknet.exe detector test data/obj.data yolo-obj.cfg yolo-obj_45000.weights data/sample_image.jpg Similar to the previously described learning command, 'test' of the parameters used in the detection command calls a function that detects the object. Put the learned '.weights' file. Finally, put the path of the sample photo.

4.2. Check detection of objects.

The sample data used to detect the object were selected from a photo of an adult and a child together when visually verified by a human being. Object detection uses weight files learned by increasing adult and child images to 1500, 2000 and 2500 respectively. The weight file was studied 45000 times. The conditions for detection are, as mentioned in the main code earlier, when the accuracy of the detected adult and kid is greater than 0%, a bounding box is generated. Figure 7 detects objects using a weight file that learns 1500 sheets from sample photos.

	Sample1.jpg	**Sample2.jpg**	**Sample3.jpg**	**Sample4.jpg**	**Sample5.jpg**	**Sample6.jpg**
Sample Pictures Used						
Study Results for 1500 sheets	Adult : 86 Kid: 48	Adult : x Kid: 81	Adult 1 : x Adult 2 : x Kid 1: 86 Kid 2: 86	Adult : 59 Kid: 86	Adult : x(kid: 34) Kid: 34	Adult : x Kid: 88
Study Results for 2000 sheets	Adult : 82 Kid: 60	Adult : 63 Kid: 76	Adult 1 : x Adult 2 : x Kid 1: 90 Kid 2: 84	Adult : 84(kid: 28) Kid: 88	Adult : x(kid: 60) Kid: 63	Adult : 28 Kid: 83
Study Results for 2500 sheets	Adult : 90 Kid: 70	Adult : 83 Kid: 64(Adult: 62)	Adult 1 : x Adult 2 : 40 Kid 1: 85 Kid 2: 84	Adult : 82 Kid: 78	Adult : 48(kid: 38) Kid: 56	Adult : 42 Kid: 87



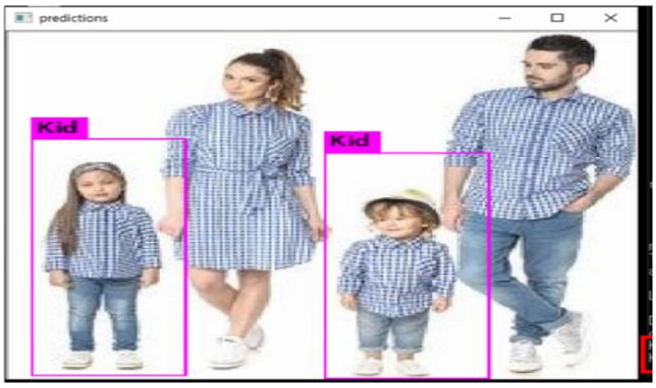
```
 14 conv    5
 15 conv    2
 16 conv    5
 17 max
 18 conv   10
 19 conv    5
 20 conv   10
 21 conv    5
 22 conv   10
 23 conv   10
 24 conv   10
 25 route  16
 26
 reorg_old
reorg_old
 27 route  26
 28 conv   10
 29 conv
 30 detection
mask_scale: Us
Total BFLOPS 3
avg_outputs =
 Allocate addi
Loading weight
 seen 64, trai
Done! Loaded 3
data/test11.jp
Adult: 86%
Kid: 48%
```

Fig 7. 1500sheets object detection using a weight file

The first of the sample data using a weight file learned from 1500 sheets shows object accuracy of 86% for adults and 48% for children. Subsequently, as shown in Sample 1.jpg of Figure 9, the results of detecting the first sample data with weight files learned from 2000 and 2500 sheets show that the bounding box is detected closer to the object and the accuracy is also improved.

Table 1 shows the accuracy when objects are detected using weights learned from 1500, 2000 and 2500 sheets for each sample plot and sample plot used for object detection.

table 1. table of object detection results based on number of lessons



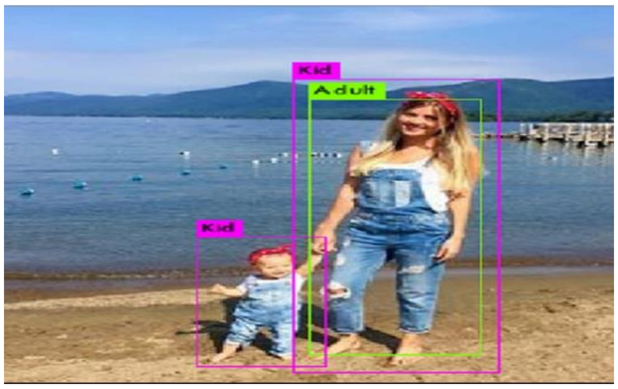
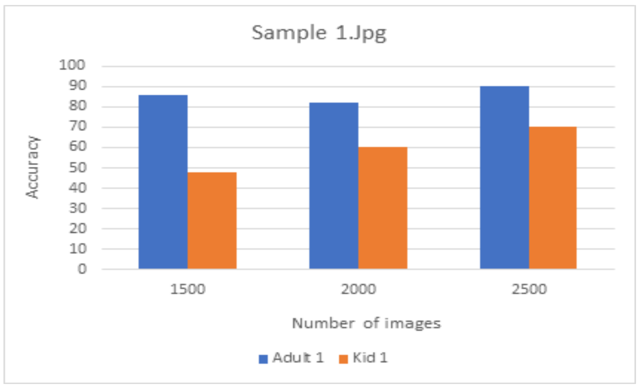
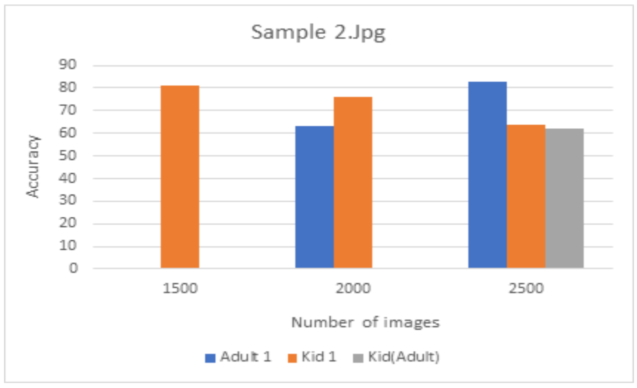


Fig 8. Sample3.jpg is 1500sheets(Left) and Sample5.jpg is 2500sheets(Right) object detection results





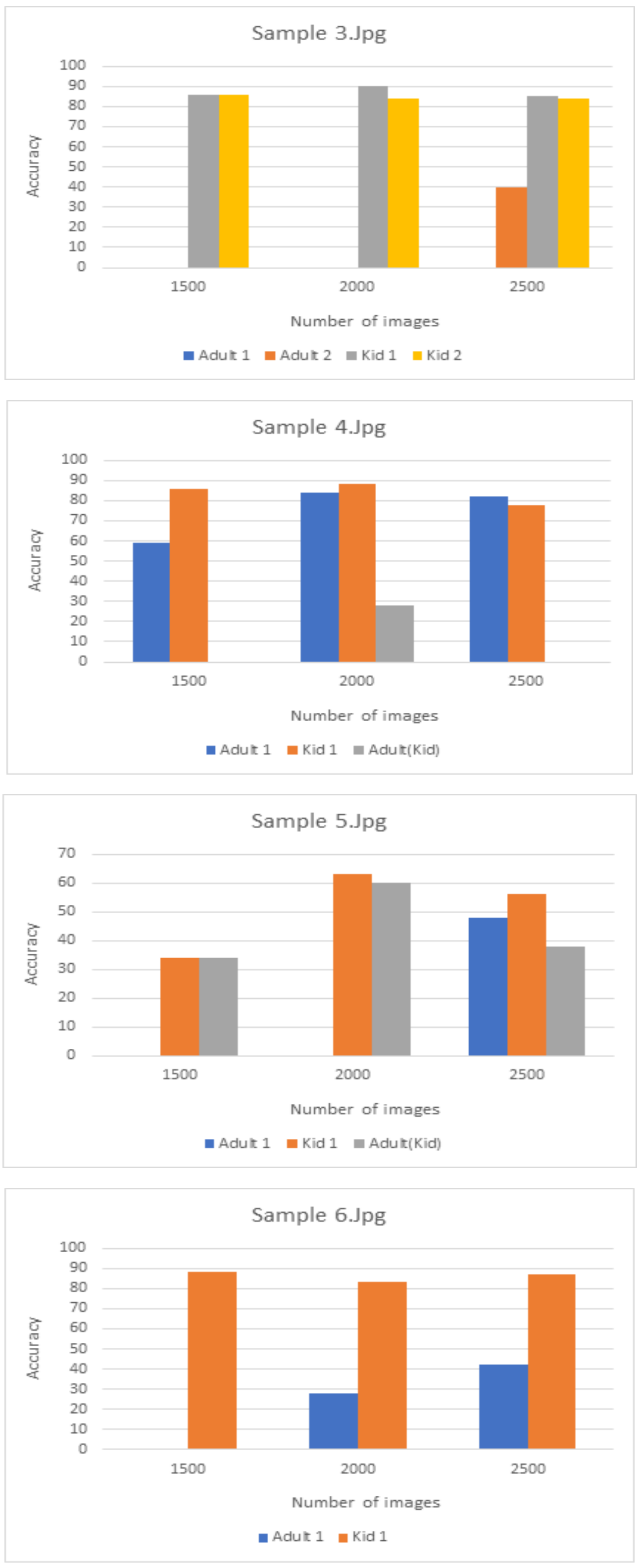


Fig 9. graph of object detection results by number of images

The left image in Figure 8 shows the result of object detection, which is learned from 1500 chapters of Sample 3.jpg. This was set to be detected when the accuracy was more than 0%, but Given the results of no adult detection as shown in Figure 8, it can be determined that the accuracy has not exceeded 0%. The right image in Figure 8 shows that adult objects seen in the detection results after learning 2,500 sheets of Sample 5.jpg overlap with children. The overlap of detection results is a phenomenon in which the accuracy of children is measured together when object detection is performed and accuracy is determined. This represents the result of failing to distinguish objects clearly.

Table 1 shows a diagram that is studied using 1500, 2000 and 2500 learning data and evaluated accurately through 6 sample photos. Furthermore, to check the accuracy according to the number of images, the tables are visualized and charted as shown in Figure 9.

Table 1 shows that two objects were detected from one object. In Figure 9, we represent overlapping objects using '()' after object names that should have been detected in the table. Object detection of adults and children who learned 2000 sheets in Sample 4.jpg showed 84% accuracy as Adult, but the results of detection together with Kid overlapping with 28%.

Figure 9 shows the results as a whole, and the higher the number of lessons, the clearer the object detection becomes and the higher the accuracy of object detection in adults. On the other hand, the detection accuracy of children does not increase significantly and results can be seen to maintain a high probability. This shows that increasing the number of lessons clarifies detection results and improves accuracy.

V. Conclusion

YOLO algorithms can detect objects faster and with higher accuracy than other object detection algorithms. However, when learning objects based on YOLO algorithm, it is not easy to set the criteria for objects, so if you proceed without clear criteria, Similar learning content between learning objects can clear recognition becomes difficult and accuracy can be reduced. I was able to experience differences between CPU environments and GPU environments in the course of the process of this paper. In a CPU environment, learning 50 object images of adults and children, respectively, was forced to end at the 10th day of 400 learning. However, in a GPU environment using CUDA, each of the 2,500 object images of adults and children was taught 45,000 times is takes about 36 hours, there is a difference in learning speed between CPU and GPU environments.

As a result of the detection by increasing the number of photo, Compared to the results learned with 1500 sheets, the results detected with 2000 sheets increase in accuracy and part that clearly distinguishes objects becomes larger. The results of the detection with weights learned from 2,500 sheets indicate that the accuracy of the two objects increases to a similar level and distinguishes the objects. As part of increasing the accuracy of objects recognition by viewing these results, increasing the number of learning photos was also a consideration. However, as noted in the results and performance evaluation, increasing the number of data to be learned can also result in lower performance, and we can see that the detection of objects by adults/children is expressed in duplicate. Therefore, we intend to analyze the source code of the corresponding Yolo algorithm and study the direction in which it can be improved.

REFERENCES

[1] Jae-Kyu Park, “A Study on Design and Experimental Verification of Deep Learning Based CCTV Pedestrian Detection and Tracking System: Focused on CIC7P Model”, Doctoral dissertation, Soongsil University, 2017.

[2] Joo-Young Lee, “Implementation of Pedestrian Recognition Based on HOG using ROI for Real Time Processing”, Institute of Korean Electrical and Electronics Engineers, vol. 18, no. 4, pp. 581-585,2014.
DOI : http://dx.doi.org/10.7236/JIIBC.2016.16.2.

[3] Yong-Hwan Lee, Young-Seop Kim, "Comparison of CNN and YOLO for Object Detection," Journal of the semiconductor & display technology, vol. 19, no. 1, pp. 85-92, 2020.

[4] Song-Won Lim, "Generation of Domestic Vehicle identification Classifier System using R-CNN Algorithm and Composition of the Vehicle Dataset", Masters dissertation, Seoul National University, 2020.

[5] Omer F. Ince, J. S. Park, J. Song, and B. W. Yoon, "Child and Adult Classification Using Ratio of Head and Body Heights in Images", ICIC International, vol. 8, no. 5, pp.819-825 2017
DOI:http://www.ijcce.org/papers/304-E045.pdf

[6] Jeong-Hwan Kim, Yong-Hyeon Shin "A Study on Deep Learning-based Pedestrian Detection and Alarm System," The Journal of The Korea Institute of Intelligent Transportation Systems, vol. 18, no. 4, pp. 58-70, 2019.
DOI : https://doi.org/10.12815/kits.2019.18.4.58

[7] Seung-Ju Lee, Goo-Man Park, "Proposal for License Plate Recognition Using Synthetic Data and Vehicle Type Recognition System" Journal of Broadcast Engineering, vol. 25, no. 5, pp. 776-788,2020.
DOI :https://doi.org/10.5909/JBE.2020.25.5.776