# A Study on Tracking Moving Objects: Pig counting with YOLOv5 and StrongSORT

Seunggwan Lee[1], Wonhaeng Lee[2], Junghoon Park[3*]

[*]*Corresponding author: Junghoon Park, Professor, College of Computing and Informatics, Applied Artificial Intelligence*

*Ajou University, Republic of Korea*
*206, World cup-ro, Yeongtong-gu, Suwon-si, Gyeonggi-do*
{gwan7801, bc005007, stevejobs}@ajou.ac.kr

**Abstract.** Counting the number of pigs is a significant issue in pig farms for efficient farm management. However, in pig farms in Korea due to the small land size, pigs are crowded together in a confined space. It makes it hard to count the number of pig stocks on the farm and it gets much worse when the farm scale grows larger. To solve this problem, we suggest an automatic method for counting the number of pigs, using an AI model with CCTV that is already equipped in farms. By doing this, we can count the accurate number of pig stock which can systemize internal and external environment management and be helpful to sales and distribution. We used YOLOv5 for object detection and StrongSORT for tracking. We optimized the model by tuning epoch and batch-size and got a 0.97832 mAP score. Using the detection model with the tracking model we could assign an ID to each pig in the video, which leads to a more accurate result. By this paper, we could be close to building a smart pig farm system at a low cost.

**Keywords:** computer vision, pig counting, object detection, object tracking YOLOv5, StrongSORT, smart pig farm.

## 1 Introduction

The annual per capita pork consumption in Korea (2020) was 26 kg, similar to the sum of beef (13 kg) and chicken (14.7 kg). From statistics since 1995, it is easy to see that pork consumption has been overwhelmingly high compared with other meat consumption [1]. However, the number of pig farms dropped below 6,000 for the first time in 2021, due to a lack of manpower and high feed costs [2]. If this situation continues, when consumption exceeds supply pork prices would skyrocket.

In this paper, we suggest deep learning based real-time pig farm monitoring module using images from CCTV on the farm. The previous paper used only YOLO in object detection and only DeepSORT in object tracking [3]. For higher accuracy and faster computation, we compare YOLO and R-CNN in object detection and StrongSORT in object tracking. This would perform accurate pig counting and assign an ID to each pig without an additional physical device to facilitate management. Therefore,

it is possible to establish a Smart Pig Farm system that can accurately count the number of pigs and manage them systematically even with a little manpower. Through this Smart Pig Farm system, it is possible to lower the entry barrier of Korean pig farms, which are being reduced. Moreover, this AI model is not limited to the pig farming business and domestic market, it could be applied to other industries that need an accurate count of the object or global market.

## 2 Background

The final goal of this research is to count the accurate number of pigs for the Smart Pig Farm system. In this paper, object detection and object tracking techniques are used. We compared Fast R-CNN and YOLOv5, and for object tracking, the Strong-SORT algorithm is used.

### 2.1 Object Detection Module

**1. Faster R-CNN**

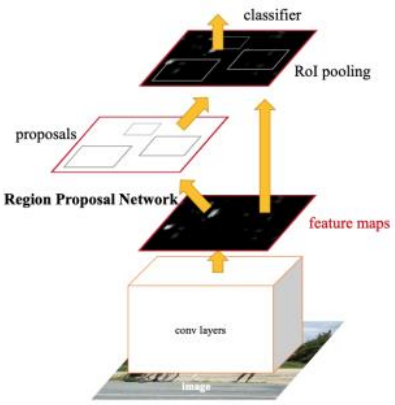


**Figure 1.** Process of Faster R-CNN. Convolution layers extract the Feature Map and pass it to the RPN to compute the RoI. The calculated RoI value conducts Pooling, then conducts classification for object detection [4].

Faster R-CNN inherits Fast R-CNN structure. Faster R-CNN calculates RoI through RPN that has features using Anchor boxes while Fast R-CNN uses selective search. RPN shows higher accuracy by calculating about 800 RoI, while Selective Search calculates 2,000 RoIs. Moreover, Faster R-CNN operates on GPU. This raises accuracy and speed by calculating little RoI using GPU.

## 2. YOLOv5

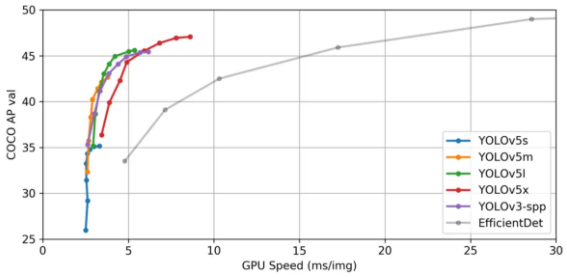


**Figure 2.** YOLOv5 type, GPU Speed, COCO AP Validation [5].

Previous research used YOLOv4 in object detection [3], but this paper uses an updated version, YOLOv5. YOLOv5 has a different Backbone and Head compared to other YOLO models [6]. The Backbone part extracts the Feature Map from the image, which is similar to YOLOv4, but works on lower capacity and the computing speed is faster. Backbone uses BottleneckCSP(CSPNet-based) and the Head part finds the object's position based on the Feature Map. After setting the Anchor Box (Default Box), uses it to generate the final Bounding Box. While YOLOv3 has a high FPS but mAP was relatively low. YOLOv5 however, performs well in both FPS and mAP.

## 3. Comparing Faster R-CNN with YOLOv5

We compared it with Faster R-CNN, which has high performance among R-CNN. By comparing the result of YOLOv5 and Faster R-CNN, Faster R-CNN AP was 0.7373, and YOLOv5 AP was 0.9837. As a result, YOLOv5 showed higher accuracy in our dataset.

Moreover, YOLOv5 inference speed was faster than Faster R-CNN. Also, unlike Faster R-CNN, YOLOv5 detects small or far away objects and has very few overlapping boxes. So, we chose YOLOv5 for Object detection [7].

### 2.2 Object Tracking Module

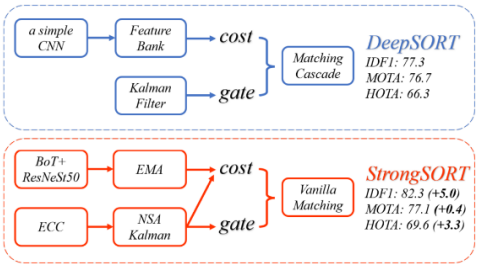


**Figure 3.** Framework and performance comparison between DeepSORT and StrongSORT. Performance is evaluated on the MOT17 validation set based on detections predicted by YOLOX [8].

StrongSORT improvement from DeepSORT lies mainly in the two branches, as shown in the bottom half of Figure 3. For the appearance branch, a stronger appearance feature extractor, BoT [9], is applied to replace the original simple CNN. By taking ResNeSt50 as the backbone [10], it can extract much more discriminative features.

For the motion branch, StrongSORT adopted ECC for camera motion compensation. Furthermore, StrongSORT replaced the vanilla Kalman filter with the NSA Kalman algorithm. Since it is vulnerable w.r.t. low-quality detections and ignores the information on the scales of detection noise [11].

Furthermore, to solve the assignment problem, instead of employing only the appearance feature distance during matching, StrongSORT adopted both appearance and motion information. The matching cascade algorithm limits the performance as the tracker becomes more powerful and it would limit the matching accuracy. StrongSORT replaced the matching cascade with the vanilla global linear assignment.

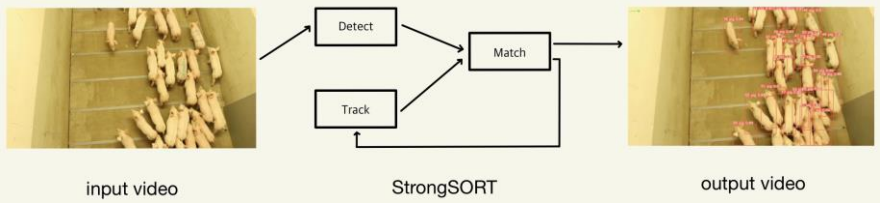


**Figure 4.** StrongSORT algorithm process

The main principle of StrongSORT is to predict the trajectory of the current frame from the trajectory after the Kalman filtering algorithm and make a judgment on whether to confirm it or not. After that, this detects the current frame, then correlates the data between the detection result and the predicted trajectory. Then update it after matching is completed.

After that, this continues to predict the current frame, observes the next frame, updates it, and so on cycle. If the track does not match, it will be deleted after exceeding the maximum age, and if the detection does not match, a new track is created, and the prediction is continued by Kalman filtering and repeated.

## 3 Experimental Results

### 3.1 Experimental Environment and Dataset

In this paper, the pig detecting and tracking model was trained on the KISTI National Supercomputing Center supercomputer system. For further information, the system model is Lenovo nx360-m4 with Intel Xeon Ivy Bridge (E5-2670) / 2.50GHz (10-core) / 2 socket (CPU), two V100(GPU), and 128GB DDR3(RAM). OS installed in the model is CentOS 7.9 (Linux, 64-bit).

**Figure 5**. Sample images from three different pig farms [12].

The dataset is a manually annotated open dataset provided by KISTI. Intflow Inc. built this dataset by using CCTV in three pig farms. This dataset consisting of image file(jpg) and annotation file(json) has 2,700 images and one json file that has labels and bbox informations of the pigs in the images. As shown in Figure 5, captured image files were directly collected from the pig farm with CCTV which was already installed. Since it is CCTV, captured images are a top-down view, showing only pigs.

As a result, for the detection module, a total of 2,700 images that contain 79,326 pigs were used. 1,890 images (70%) were used for training data, 540 images (20%) were used for validation data, and 270 images (10%) were used for test data. For our final model, we used all data for training which is 2,700 images.

### 3.2 Results with Object Detection Module

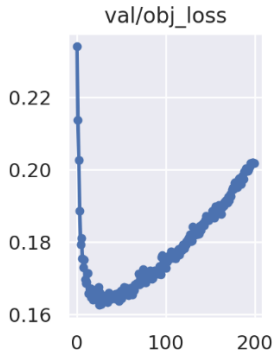


**Figure 6.** val/obj loss of Epoch 200, batch-size 16 model.

**Table 1.** Results for different epoch and batch-size.

| epoch | batch-size | metrics/mAP 0.5 | metrics/mAP 0.5:0.95 | val /box loss | val/obj loss |
|---|---|---|---|---|---|
| 30 | 16 | 0.96392 | 0.69898 | 0.029694 | 0.16369 |
| 30 | 8 | 0.97210 | 0.68231 | 0.026231 | 0.16034 |
| 30 | 4 | 0.97092 | 0.72676 | 0.027831 | 0.16216 |
| 30 | 2 | 0.97132 | 0.71402 | 0.026895 | 0.97779 |
| 50 | 16 | 0.97533 | 0.70647 | 0.028947 | 0.16665 |
| 50 | 8 | 0.97832 | 0.73375 | 0.026913 | 0.16552 |
| 50 | 4 | 0.97269 | 0.73252 | 0.027197 | 0.16694 |
| 50 | 2 | 0.97319 | 0.71294 | 0.027839 | 0.16832 |
| 100 | 16 | 0.97256 | 0.72234 | 0.027868 | 0.18307 |
| 100 | 8 | 0.97123 | 0.72429 | 0.027371 | 0.19229 |
| 100 | 4 | 0. 97239 | 0.72301 | 0.027316 | 0.19319 |
| 100 | 2 | 0. 97301 | 0.73125 | 0.026982 | 0.18507 |
| 200 | 16 | 0.96843 | 0.73142 | 0.027487 | 0.20177 |
| 200 | 8 | 0.97125 | 0.73372 | 0.026646 | 0.19872 |
| 200 | 4 | 0.97012 | 0.73192 | 0.027192 | 0.19983 |
| 200 | 2 | 0.97492 | 0.72915 | 0.027842 | 0.20091 |

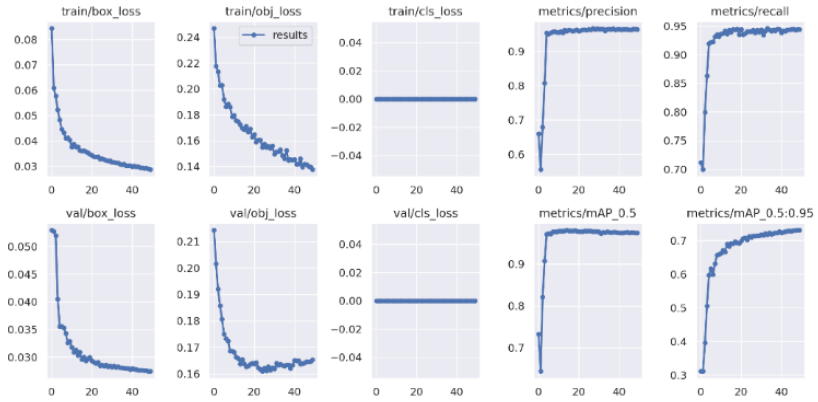


**Figure 7.** Evaluating performance of 50 epoch, 8 batch-size model YOLOv5

Figure 6. is obj_loss of the model tuned with 200 epoch and 16 batch-size which shows the overfitting problem. From the Figure 6. we can see that loss value rebounds at epoch 30~50. So, we conducted experiments with epoch sizes of 30, 50, 100, 200 and batch sizes of 2, 4, 8, 16 to find the best model. By the result shows at Table 1., for our final model we tuned parameter as batch-size 8 at epoch 50 which had the highest metrics/mAP 0.5:0.95.

Unlike Figure 6., which proceeded with 200 epochs, Figure 7. shows decrease of box_loss and obj_loss values as train and validation epochs progress. In addition, precision, recall, and mAP increase as progress.

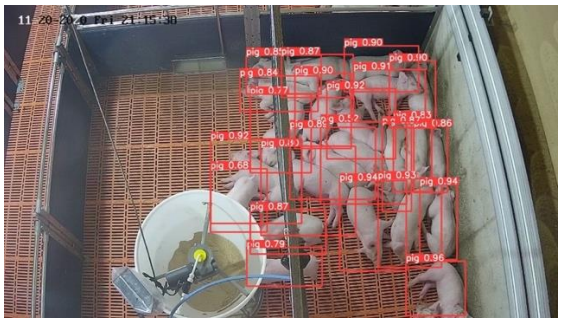

**Figure 8.** The results of pig detection with YOLOv5

We tested our final detection model which is YOLOv5 tuned by 200 epoch and 8 batch-size with our test dataset. And we compared the number of pigs that our final model detected with labeling data from json file (annotation file). We calculated RMSE and the result was 0.215.

### 3.3 Results with Object Tacking Module

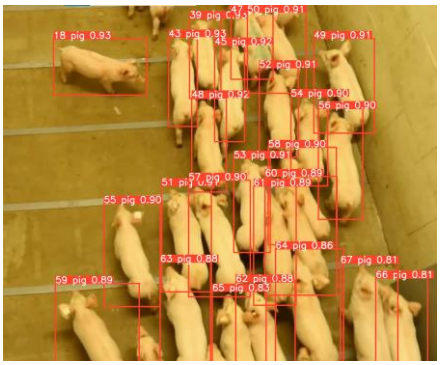

**Figure 9.** The results of pig tracking with StrongSORT

We use a 7-second-long video to check if our final models work well. The pig with ID 18, goes out of the frame and come back with same ID number. This shows that our final model performed better than using DeepSORT. The result of the full video is uploaded in the YouTube link below.

(https://www.youtube.com/watch?v=BOOLM5Bl4OM)

## 4 Conclusions

From the result of this paper, we compared YOLOv5 and Faster R-CNN in object detection and chose a better model which was YOLOv5. In addition, by adjusting the epoch and batch-size of YOLO in a total 16 combinations, we find the optimal model with high accuracy. In object tracking, we got more sophisticated results by using StrongSORT instead of DeepSORT [11].

Each farm can use this AI model by putting real-time CCTV video to calculate the accurate number of pigs right away for farm management and low-cost pig individual management without an additional physical device. It is possible to create a systematic Smart Pig Farm.

Moreover, the system is not limited to pigs, it could be also used for other various livestock such as cows, sheep, horses, and chickens. Furthermore, it makes it easier to manage each livestock individually and figure out the number of objects. So instead of rearing them in cages, they can be raised free at pasture.

For future research, we will create a tail that can track the movement path of the object by connecting the bboxes center points of the object. Also, not only we could identify the past path but also predict the future path. Moreover, we could study an algorithm that tracks objects moving fast.

## References


1. KMTA Consumption Status Available online: http://www.kmta.or.kr/kr/data/stats_spend.php (accessed on 25 January 2021).
2. KOSTAT Livestock trend survey dataset. Available online: http://kostat.go.kr/portal/korea/kor_nw/1/8/1/index.board?bmode=read&bSeq=&aSeq=416443&pageNo=1&rowNum=10&navCount=10&currPg=&searchInfo=&sTarget=title&sTxt= (accessed on 20 January 2022).
3. Jonggwan Kim.; Yooil Suh.; Junhee Lee. EmbeddedPigCount: Pig Counting with Video Object Detection and Tracking on an Embedded Board. Sensors. 2022.
4. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. IEEE Trans. Pattern Anal. Mach. Intell. 2017.
5. Junfan WANG1.; Yi CHEN1.; Mingyu GAO. Improved YOLOv5 network for real-time multi-scale traffic sign detection. arXiv 2021.
6. Redmon, J.; Divvala, S.; Girshick, R.; Farhadi, A. You Only Look Once: Unified, Real-Time Object Detection. arXiv 2016.
7. Yang, Q.; Xiao, D.; Lin, S. Feeding Behavior Recognition for Group-Housed Pigs with the Faster R-CNN. Comput. Electron. Agric. 2018.
8. Ge, Z., Liu, S., Wang, F., Li, Z., Sun, J.: Yolox: Exceeding yolo series in 2021. arXiv preprint arXiv:2107.08430 (2021)
9. Luo, H., Jiang, W., Gu, Y., Liu, F., Liao, X., Lai, S., Gu, J.: A strong baseline and batch normalization neck for deep person re-identification. IEEE Transactions on Multimedia 22(10), 2597–2609 (2019)
10. Zhang, H., Wu, C., Zhang, Z., Zhu, Y., Lin, H., Zhang, Z., Sun, Y., He, T., Mueller, J., Manmatha, R., et al.: Resnest: Split-attention networks. arXiv preprint arXiv:2004.08955 (2020)
11. Yunhao Du.; Yang Song.; Bo Yang.; Yanyun Zhao. StrongSORT: Make DeepSORT Great Again. arXiv 2022.
12. Intflow Pig Farming image dataset: https://aida.kisti.re.kr/competition/main/problem/PROB_000000000000171/detail.do