



Developing a Searching Sheep Application Using Machine Learning

Chengyuan Dong^(✉) and Yihsin Ho

Takushoku University, Tatemati 815-1 Hachioji, Tokyo, Japan
22m212@st.takushoku-u.ac.jp

Abstract. Sheep hold significant economic value. While the loss of a few sheep may be insignificant on large-scale ranches, it poses an unacceptable burden on small and medium-sized ranches. Therefore, considerable manpower and time must be allocated for locating the lost sheep. In this paper, the authors focus on develop an application to help searching lost sheep. To correspond to the actual ranch environment, a drone is considered to use. The affordability of small drones, coupled with their mobility, offers an effective solution to address this challenge. The machine learning methods is used to identify sheep. Moreover, to build a lightweight, works quickly and correctly execute system, the system environment and several methods is discussed.

Keywords: Image Annotation Tools · YOLOv4 · Drone

1 Introduction

With the ongoing industrialization of society, the human's quality of life has been steadily improving. Animal husbandry stands as a vital component of the human social structure, with numerous enterprises relying on it for prosperity. Sheep serve as a representative species within animal husbandry. Their meat provides nourishment, their milk strengthens the immune system of human body, their fur is used for winter clothing, and their bones are utilized for pet food production. Sheep yield substantial profits for ranches and companies. As described above, for small-scale ranches or companies, the loss of a sheep imposes a significant cost. When searching for lost sheep, relying solely on manpower to locate stray sheep within ranches proves to be extremely time-consuming and energy intensive.

In this paper, the authors explore the utilization of drones for assisting in the search for sheep on a ranch. Specifically, the authors employ drones to efficiently capture images and videos within the ranch. For image processing, the authors use OpenCV as the library. As for machine learning, the authors adopt the YOLOv4 model. By leveraging machine learning techniques, the presented system utilizes YOLOv4 to swiftly detect and locate sheep within the data collected by the drone. The proposed method aims to effectively and expeditiously locate stray sheep.

2 Related Works

There are several research for searching animals. For example, there is a seasonally waterfowl called the white-fronted geese (*Anser albifrons*) that roost at Lake Miyajimamura, Hokkaido, Japan. Researchers, Kenta Ogawa etc. [1], use drones to observe the status of endangered animals, the white-fronted geese, and confirm the number of them. Then compare the measured number with the measured number of last years to determine whether they have been hunted. Using a drone flight could observe the entire lake, from an altitude of 100m above the water surface, with little disturbance to the roosting geese. The used of cascade classifier, which is a machine leaning technique, to automatically count the geese in the captured images. The counting accuracy ranger from -4.1% to +6.1% in four validation cases compared with manual counts on the drone images. They conclude that the combination of drone and machine leaning methods can get goose counts with an accuracy of $\pm 15\%$.

By using aerial portraits, Ostaki Tshuichihara etc. [2] presented a method that using GPS and accelerometers to manage cattle and grasslands, to achieve the purpose of reducing labor costs and feed costs. In order to identify harmful weeds, deep learning is performed on a large number of grassland portraits, and the playground in the pastoral area is segmented. Grassland images are manually labeled with 6 classes: grasses, broadleaf weeds, dry grass, bare land, forest, and other buildings. After dividing the 304*304-pixel image with the labels and reducing it to 256*256 pixels, the grassland is segmented using the region segmentation method using deep learning.

As described above, this research draw inspiration through above research to explore and expend use. This paper will descript the system environment include hardware and software of this research, and then, show the experiments and the results. Finally, describe the conclusion.

3 Environment

3.1 Hardware

To ensure the capture of high-quality images and videos, the authors opted for a drone manufactured by DJI [3]. This drone is equipped with a high-performance camera capable of capturing image data at a resolution of 5280*3956 pixels, ensuring the acquisition of clearer and more detailed visuals. Furthermore, it excels in capturing moving objects with remarkable clarity. With a prolonged standby time, it enables extended periods of target search in outdoor environments. Additionally, the drone offers a convenient one-key return home feature, significantly enhancing operational efficiency, particularly for users unfamiliar with drone control. The appearance of the drone is depicted in Fig. 1.

3.2 Software

In this study, the execution environment [4] is on Windows 10 and Ubuntu 18.04. In Windows10, the operation method is easy for computer beginners and can be started immediately. It has a high usage rate and is convenient for supporting and cooperating

with colleagues. Supports many drivers and takes full advantage of hardware performance. In Ubuntu18.04, there is very little system garbage, and the system doesn't get slower and slower as usage time increases. The system is safe, stable, and has few viruses. Powerful command line, basically all operations can be performed.

To support machine learning, the authors have installed CUDA developed by NVIDIA, and cuDNN developed by NVIDIA specifically for deep neural networks. However, since the graphics card driver is not installed, the maximum version of CUDA that can be installed is still relatively low, at version 9.0. Unfortunately, YOLOv4 cannot run on this version of CUDA. After conducting research [5], it was found that the most suitable CUDA version for YOLOv4 is version 11.0. To maximize the computing power of the GPU, the authors have downloaded GeForce Experience, which upgrades the supported CUDA version to version 12.0.

YOLOv4, in fact, is an algorithm that combines a large amount of previous research techniques and incorporates appropriate innovations to achieve a perfect balance between speed and accuracy. The processing is implemented as following: Compiling YOLOv4 using VS2019 in Windows 10 and testing its functionality by using the provided test weight file from YOLOv4 developers. Then, training can be initiated using annotated data for machine learning. Figure 2 is the comparison result by Alexey Bochkovskiy etc. [6], which illustrates the comparison between YOLOv4 and other object detectors. Furthermore, YOLOv4 typically utilizes rectangular bounding boxes for object detection, which include the coordinates of the top-left and bottom-right corners of the target object. Each bounding box also needs to be labeled with the corresponding category. Therefore, the dataset used for training YOLOv4 also consists of data with rectangular annotations.

In this research, the authors use LabelImg and labelme for image annotation. LabelImg [7] is a tool that allows users to directly label objects using four 90° rectangles. It can also convert the annotation results into the.TXT format required for YOLOv4 machine learning. On the other hand, labelme [8] is a tool that allows users to label objects as polygons. It can also label objects as rectangles, but its annotation result is different format that not suitable for YOLOv4 machine learning. Thus, its results need to be converted to the.TXT format by format conversion.

The authors installed both labelImg and labelme using Anaconda3. Anaconda is a Python distribution that includes over 180 scientific packages and their dependencies, such as Conda and Python. By default, it installs major Python programs and commonly used data-related packages for tasks like data importing, data cleaning, data processing, data computation, and data visualization. Conda, a package management command like pip, is provided to perform common operations like display, update, install, and uninstall packages. However, pip command can also be used since Anaconda by default includes it. Incidentally, OpenCV is the image processing method to be used in this research. Table 1 shows the detail information of research environment.

The software described above is mainly used on Windows 10, but the environment configuration in Ubuntu 18.04 is nearly identical to that of Windows 10, except for VS2019. In contrast with using VS2019 on Windows 10, CMake is utilized as an alternative on Ubuntu 18.04.



Fig. 1. The appearance of the drone

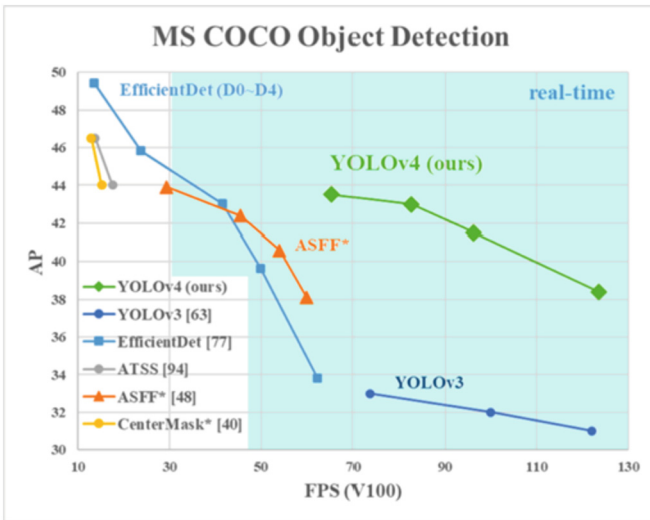


Fig. 2. The comparison of YOLOv4 and other object detectors [6].

Table 1. Research Environment

Hardware and Software	Version
Drone	Mavic 3
OpenCV	3.4.0
CUDA	11.0.2
cuDNN	11.0
Microsoft Visual Studio	2019
YOLO	4
Anaconda	3-5.3.0
GeForce Experience	528.02
CMake	3.10.2
Windows	10
Ubuntu	18.04
Graphics Card	GTX 1080

4 Experiment

In the experimental phase, the primary objective was to verify the successful processing of drone-captured data by YOLOv4. To accomplish this, the authors captured videos of cars and people using the drone and subsequently employed YOLOv4 for target recognition. Given the possibility of the recognition target being a living creature, with little disturbance to target, the authors maintained a flight height of approximately 10 to 15 m from the ground. This flight height was carefully chosen to ensure that the targets were not startled while preserving the quality of the captured images and videos. The results confirmed the successful recognition of the target.

After finishing verify the drone-captured data, the next phase is to accurately identify sheep in images. The authors conducted machine learning on sheep and created their own data weights. Since the authors were unable to capture actual images/videos themselves, they collected the required images/videos from the internet. Within the training data, the annotated targets consisted solely of sheep. For machine learning purposes, the authors created the necessary files such as `cfg.data`, `cfg.name`, and `yolov4.cfg` for YOLOv4 used. Then, at the image annotation step, the authors converted the XML files labeled with LabelImg to TXT files required by YOLOv4 [9–11]. Additionally, the authors converted the JSON files labeled with labelme to TXT files, regarding the `yolov4.cfg` file mentioned above.

To test different methods and conduct comparative experiments, the authors divided it into two types of YOLOv4-tiny and YOLOv4-normal. The difference between them is the number of Anchor box. All of them set Mask as 3, but in “tiny”, the number of Anchor box is 12, in “normal”, the number of Anchor box is 18. In this stage, the authors prepared 2 kinds of datasets, a total of 100 and 300 photographs, and within using the 2 different annotations, from LabelImg and labelme, there are 4 distinct datasets obtained. These 4 datasets were then used for machine learning with both YOLOv4-tiny and YOLOv4-normal, resulting in a total of 8 different outcomes. The same experiments were conducted on both Windows 10 and Ubuntu 18.04 systems. Ultimately, a total of 16 outcomes is obtained.

The results obtained from training with 100 photos using YOLOv4-tiny are referred to as “100-tiny”. Similarly, the results obtained from training with the same number of photos using YOLOv4-normal are referred to as “100-normal”. The naming convention follows in a similar manner for the other results.

4.1 The Result with 100-Tiny

In Fig. 3, it shows that the recognition rate of Result A is higher than the other results, followed by a relatively good performance that shows in Result D. The most peculiar result is Result C. Moreover, upon analyzing the accuracy of machine learning from Fig. 3, it finds that Result A shows the lowest accuracy, the accuracy of Result B, C, D becomes stable after approximately 1700 iterations of machine learning. However, Result C differs from the other before reaching 1700 iterations. Additionally, the machine learning duration was relatively short, lasting only about 1.5 h.

As the result, it can be said that the results may be attributed to the machine learning dataset, prepared consisted of only 100 photos, which is insufficient in quantity and the relatively simple learning method employed.

4.2 The Result with 100-Normal

In Fig. 4, it shows that Result A have the highest recognition rate, followed by Result C. Upon comparing the results of Results A and C, which both utilized the dataset annotated using the LabelImg tool, it is found that the machine learning results obtained using Windows, as in Result A, were able to recognize more targets compared to Result C. Furthermore, when comparing the results of Results A and C, both obtained using the LabelImg tool on the Windows system, the machine learning results were far superior to those obtained using labelme. Moving on to Fig. 4, only the accuracy of Result A declines, while the Result B, C, D stabilize in their machine learning accuracy after approximately 1700 iterations. However, the accuracy of Result B is unexpectedly high, which is quite abnormal. As a side note, running machine learning in normal mode takes at least 8 h.

4.3 The Result with 300-Tiny

Through Fig. 5, the recognition rate and the number of detections is highest in Result A, which corresponds to the machine learning results obtained using the LabelImg annotation tool on the Windows system. Furthermore, as shown in Fig. 5, it is found that Result A had the lowest machine learning accuracy, however, all four results stabilized after approximately 1700 iterations and remained within the range of 60% to 80% accuracy. Additionally, the machine learning duration was relatively short, lasting only about 1.5 h.

4.4 The Result with 300-Normal

Through Fig. 6, it shows a significant increase in both the labeling accuracy and the number of labeled objects across all four results. The results from Result A outperformed the others, as it was able to accurately label distant and occluded targets. Result B and C had similar labeling accuracy, but Result C had a slightly higher number of labeled objects. In Fig. 6, after approximately 1700 iterations of machine learning, the precision of all four results stabilized within the range of 70% to 90%. However, Result A exhibited significant instability in its precision. Incidentally, running machine learning in normal mode takes at least 8 h.

4.5 Discussion

Combining the results of 100-tiny and 100-normal, it can be observed that the machine learning method of normal mode outperformed that of tiny mode by a large margin. However, due to the limited dataset of only 100 photos, overfitting occurred, resulting in a recognition rate of 1.0 and machine learning accuracy exceeding 90%, which should not have happened.

Moreover, by comparing the results from 100-tiny and 300-tiny, the authors noticed that 300-tiny did not exhibit overfitting, and the average precision in 300-tiny was significantly higher than that in 100-tiny. Therefore, it is understood that by using a larger dataset for machine learning, the resulting performance tends to improve.

Furthermore, by comparing the results of 100-normal and 300-normal, it can be known that the accuracy and the number of recognized targets in the ADCD category of 300-normal align with normal expectations. However, in 100-normal, the category shows a significant number of target recognitions with an accuracy of 1.0. Additionally, the machine learning precision of category in Result A even exceeds 90%. It can be speculated that this is due to overfitting, caused by using a dataset labeled with only 100 photographs for machine learning in normal mode.

The authors compiled the average recognition rates and the number of recognized targets from these comparative experiments and created Table 2 and Table 3. It can be observed that the results of 100-normal were highly abnormal, confirming the previous notion of overfitting. In each comparative experiment, the best results were achieved when using the dataset labeled with labelImg on the Windows system for machine learning. Furthermore, in the results of 300-normal, the number of recognized targets exceeded 20 for each method. Therefore, the authors conclude that the choice of dataset creation tool, the quantity of photos in the dataset, the selection of tiny or normal machine learning methods, and the computer system, all above factors have an impact on the results of machine learning.

Table 2. The average recognition rates and the number of recognized targets using 100 photos.

	Windows 10.0			Ubuntu 18.04		
		Recognition Rate	Recognition Number of sheep		Recognition Rate	Recognition Number of sheep
100-tiny (Fig. 3)	LabelImg (Result A)	0.70	21	LabelImg (Result C)	0.65	6
	labelme (Result B)	0.70	4	Labelme (Result D)	0.7175	8
100-normal (Fig. 4)	LabelImg (Result A)	0.96	42	LabelImg (Result C)	0.78	24
	abelme (Result B)	0.592	15	labelme (Result D)	0.58	16

Table 3. The average recognition rates and the number of recognized targets using 300 photos.

	Windows 10.0			Ubuntu 18.04		
		Recognition Rate	Recognition Number of sheep		Recognition Rate	Recognition Number of sheep
300-tiny (Fig. 5)	LabelImg (Result A)	0.69	31	LabelImg (Result C)	0.61	17
	labelme (Result B)	0.57	16	Labelme (Result D)	0.54	12
300-normal (Fig. 6)	LabelImg (Result A)	0.74	34	LabelImg (Result C)	0.63	32
	labelme (Result B)	0.63	25	Labelme (Result D)	0.61	20



Fig. 3. The result of 100-tiny. A) the result about windows10 with LabelImg; B) the result about windows10 with labelme; C) the result about ubuntu with LabelImg; D) the result about ubuntu with labelme.



Fig. 4. The result of 100-normal. A) the result about windows10 with LabelImg; B) the result about windows10 with labelme; C) the result about ubuntu with LabelImg; D) the result about ubuntu with labelme.



Fig. 5. The result of 300-tiny. A) the result about windows10 with LabelImg; B) the result about windows10 with labelme;C) the result about ubuntu with LabelImg; D) the result about ubuntu with labelme.



Fig. 6. The result of 300-normal. A) the result about windows10 with LabelImg; B) the result about windows10 with labelme; C) the result about ubuntu with LabelImg; D) the result about ubuntu with labelme.

5 Conclusions

The goal of this study is to develop a sheep searching application using a drone and find the suitable machine learning method for recognition. To achieve this goal, one of the important parts is to construct an appropriate system environment and the appropriate method. This investigation is targeted to figure out the possible factors that affect the execution performance, by actual verification experiments. Through the comparative experiments, the authors have come to understand that in the Windows system, preparing a larger dataset of photographs, using the dataset labeled with the labelImg annotation tool, and employing the normal mode of machine learning method in YOLOv4, bring the best results in this study. This outcome is completely different from the initial presumption: It is anticipated using labelme, with its capability to handle irregular bounding box annotations, would provide more comprehensive results compared to datasets labeled with rectangular bounding boxes, moreover, presumed that the Ubuntu system would outperform Windows in automating the processing of large image datasets. Thus, this research is expected to provide a new direction for the future work in environment construction and machine learning methods.

Effectively finding stray sheep is an important issue in ranch management. This research offers a low-cost and potential solution. However, there are still many unresolved questions, such as how to identify targets from high-altitude and/or complex outdoor environments, improve recognition accuracy, accurately count the number of sheep, and differentiate sheep from other white animals, among others. To enhance recognition accuracy, the current idea is to prepare a larger dataset, such as over a thousand images. Another approach is to increase the maximum number of machine learning iterations, as the iterations in this study were set at 4000. Furthermore, there is room for modification in the choice of machine learning methods, particularly the normal approach, although this may result in longer learning times.

References

1. Ogawa, K., Ushiyama, K., Koneri, F.: Automated counting of waterfowl on water surface using UAV imagery. *J. Remote Sens. Soc. Jpn.* **42**, S1–S8 (2022)
2. Ostaki, T., et al.: Cows and Meadows Management Using Drone and GPS Sensors, SI2018, pp. 1110–1111. Osaka, Japan
3. DJI Japan corp.: DJI Mavic3 specs, Site: <https://www.dji.com/jp/mavic-3/specs>, Access date: 12 Dec. 2022
4. YOLOv4 configuration tutorial under Windows 10 (in Chinese), Site: <https://onl.la/D7z3q2R>, Access date: 12 Dec. 2022
5. Configuration of YOLOv3/YOLOv4(GPU)+Win10+VS2019 (starting from 0) (in Chinese), Site: <https://onl.la/gqN2Pf5>, Access date: 13 Jan 2023
6. Alexey, B., Wang, C.-Y., Liao, H.-Y.M.: YOLOv4: Optimal Speed and Accuracy of Object Detection, Site: https://www.researchgate.net/publication/340883401_YOLOv4_Optimal_Speed_and_Accuracy_of_Object_Detection, Access date: 15 May 2023
7. Label Studio community, LabelImg, Site: <https://github.com/heartexlabs/labelImg>, Access date: 15 May 2023
8. Labelme, Site: <https://github.com/wkentaro/labelme>, Access date: 15 May 2023

9. YOLOv4 model training process (in Chinese), Site: <https://onl.sc/AZ8D9XE>, Access date: 13 Jan 2023
10. Windows10 darknet yolov4 training process (in Chinese), Site: <https://onl.sc/H4EDqcp>, Access date: 13 Jan 2023
11. YOLOv4 training custom data set (in Chinese), Site: <https://onl.sc/CXVMmr4>, Access date: 13 Jan 2023