

# Development of a Deep-learning-based Pet Video Editor

Chun-Cheng Lin, Cheng-Yu Yeh,\* and Kuan-Chun Hsu

Department of Electrical Engineering, National Chin-Yi University of Technology,  
57, Sec. 2, Zhongshan Rd., Taiping Dist., Taichung 41170, Taiwan

(Received July 22, 2021; accepted November 4, 2021)

**Keywords:** pet video editing system, deep learning, convolutional neural network (CNN), object detection, you only look once (YOLO), pets' body movement recognition

Nowadays, a growing number of people have animals, particularly dogs and cats, as pets. A lot of pet owners spend much time taking care of their beloved pets, whose images are captured in daily life and at memorable moments. Edited video clips can be even widely shared with others via the Internet. However, it takes time to edit the captured pet videos. Accordingly, our team aimed to develop a pet video editor using an object detection and body movement recognition model. Pet videos can be captured and edited automatically as expected using AI techniques. For simplicity, the target was narrowed down to recognize the fundamental movements of dogs, namely, eating, tail raising, and yawning. As the first step, input videos were saved automatically once dogs' images were detected using a pretrained YOLOv4 object detection model. In this manner, video recordings are made easy and efficient. Subsequently, three types of dogs' body movements were recognized using a self-designed recognition model. Therefore, close-up images of dogs containing any of the three body movements can be instantly recognized, saved, and then shared with others. In this study, the presented body movement model was experimentally validated to give a recognition accuracy of up to 98.84%. We are currently working on increasing the number of movements that can be recognized by our system.

## 1. Introduction

Machine learning and deep learning remain two of the hottest research topics and have been well applied to a wide variety of disciplines. As a consequence, remarkable progress had been made in the development of deep-learning-based image recognition techniques.<sup>(1–8)</sup> Most of the models therein were actually built on the basis of convolutional neural networks (CNNs), among which representative studies are the AlexNet,<sup>(5)</sup> visual geometry group (VGG) Net,<sup>(6)</sup> Inception Net,<sup>(7)</sup> and ResNet.<sup>(8)</sup> Object detection<sup>(9–15)</sup> and image classification<sup>(16–20)</sup> stand as the most frequently used techniques in the field of image recognition.

In an object detection task, it is necessary to find and recognize specific objects in an image or a video. Nowadays, commonly adopted object detection models include the Single Shot MultiBox Detector (SSD),<sup>(10)</sup> EfficientDet,<sup>(11)</sup> and You Only Look Once (YOLO).<sup>(12–14)</sup> With the

---

\*Corresponding author: e-mail: [cy.yeh@ncut.edu.tw](mailto:cy.yeh@ncut.edu.tw)  
<https://doi.org/10.18494/SAM3566>

COCO dataset,<sup>(21)</sup> an object detection model can be trained to recognize up to 80 types of objects, including people, cars, cats, dogs, and so forth. Thus, such techniques can be widely applied to fields such as intelligent transportation, intelligent image analysis and retrieval, smart home, and smart security.

In the discipline of image classification, the development of facial recognition techniques has been a hot topic, and facial images are frequently recognized using deep-learning-based models such as FaceNet,<sup>(18)</sup> CosFace,<sup>(19)</sup> and ArcFace.<sup>(20)</sup> These models show high recognition accuracy and robustness, which gives the models advantages over traditional counterparts.

Currently, the population of persons who have pets, particularly dogs and cats, is growing. Many pet owners spend much time and money taking care of their beloved pets, and also capture their pets' images in daily life and at memorable moments, which can be even widely shared with friends via the Internet. However, it takes time to edit the captured pet videos.

In light of this, our team developed a pet video editor by which pet video capturing and editing have become easy and efficient. This was done using a YOLO model and another deep-learning-based model to recognize pets' body movements.<sup>(22)</sup> So far, nothing like this is available in the market, and the presented video editor is considered to be a user-friendly tool for pet owners once commercialized.

This paper is outlined as follows. Section 2 refers to the framework of the presented video editor, Sect. 3 details a model used to recognize dogs' body movements, Sect. 4 gives experimental results and discussion, and finally, Sect. 5 concludes this paper.

## 2. Pet Video Editor

In this paper, we present a deep-learning-based pet video editor by which users can record and edit the videos of their beloved pets efficiently. As illustrated in Fig. 1, the video editor has two features: (i) automatic editing of pet video clips and (ii) recognition of pets' body movements for close-up shots. The presented video editor uses a two-stage CNN-based approach to get the job done. In Stage 1, a pretrained YOLOv4 model<sup>(13)</sup> is employed to identify and then record pets' images in videos. In this manner, searching for and video editing of pet images can be made efficient. Subsequently, specific body movements can be recognized, and then close-up shots can be recorded in Stage 2.

Dog images were treated as subjects in the development of this work. As referenced previously, pet images were detected using the pretrained YOLOv4 model for the following reason: the YOLOv4 model has already been trained to recognize up to 80 types of objects, including 10 types of animals such as dogs and cats. Thus, the YOLOv4 model was directly applied to the presented pet video editor. Furthermore, the YOLOv4 model turns out to have twice as many frames per second (FPS) as EfficientDet at an average precision (AP) of 43.5%, as mentioned in Ref. 13.

As stated in Ref. 13, the vast majority of object detectors are composed of an input, a backbone, a neck, and a head. Object detectors are usually split into one- and two-stage detectors. In the neck, feature maps are extracted in different layers using techniques such as feature pyramid network (FPN),<sup>(23)</sup> path aggregation network (PAN),<sup>(24)</sup> and spatial pyramid

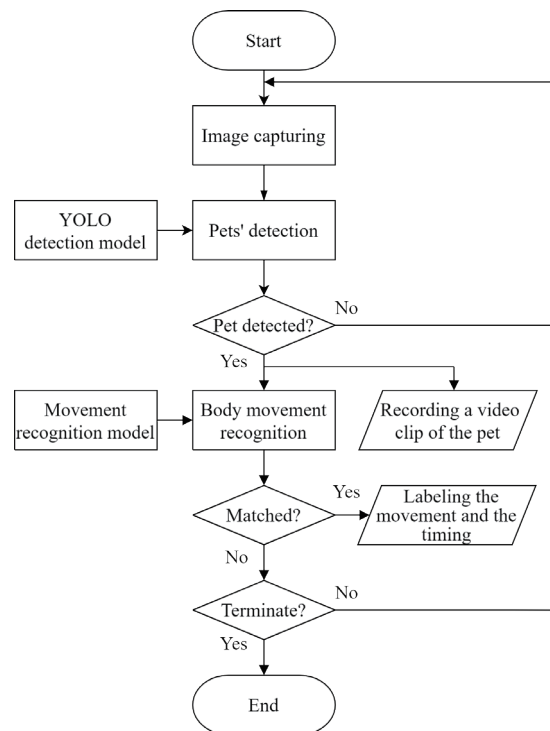


Fig. 1. Flowchart of presented pet video editor.

Table 1  
Development environment of the presented video editor.

Programming language	Python
Library	TensorFlow, Keras, OpenCV, numpy, threading, etc.
Detection model	YOLOv4
Recognition model	Proposed model
Hardware	PC (Windows 10 64-bit, Intel i7-7700 3.6 GHz CPU, 32 GB RAM), graphics card (GeForce GTX 1070Ti), web camera

pooling SPP.<sup>(25)</sup> Object detection happens in the head, and object types and boundary boxes thereof are provided. The YOLOv4 model employs CSPDarknet53<sup>(26)</sup> as the backbone. SPP and PAN are used for the neck, and a single-stage YOLOv3 detector is used for the head.

Table 1 gives the development environment of the video editor. As listed therein, the programming language is Python, and libraries such as TensorFlow, Keras, and OpenCV were used in the development of the video editor.

### 3. Pets' Body Movement Recognition Model

A model used to recognize pets' body movements is discussed here. As referenced previously, the target was narrowed down to recognize the fundamental movements of dogs, namely, eating, tail raising, and yawning, for simplicity.

Figure 2 gives the backbone of the model.<sup>(22)</sup> Color images of 128×128 pixels are applied to the model, and then feature maps are extracted through five layers of 'Conv's. Finally, the body

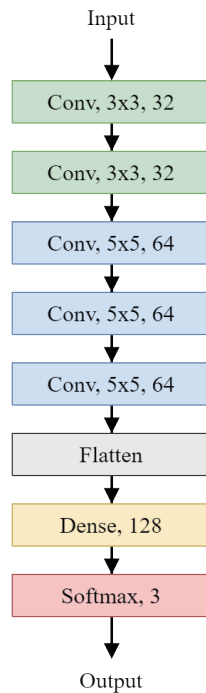


Fig. 2. (Color online) Backbone of proposed recognition model.

movements of dogs are recognized through the Dense and Softmax layers. In the first two convolutional layers, 32  $3 \times 3$  filters are used, whereas 64  $5 \times 5$  filters are used in the last three convolutional layers. In each convolutional layer, convolution, batch normalization, relu and max\_pooling ( $2 \times 2$ ) are performed sequentially. The body movement recognition model takes up only about 2.8 MB of memory. It is an efficient and simple solution to image classification.

Table 2 gives the number of collected images for dogs' body movement recognition. Eating, tail raising, and yawning are numbered as Classes 0, 1, and 2, respectively. Furthermore, Table 3 gives a sample image for each class of body movement. As listed in Table 2, up to 500 images were used to train the model in each class, i.e., a total of 1500 pieces of training data, whereas only 400 images in total were employed to test the model performance. The categorical cross-entropy loss function and the Adam optimizer were used to train the model with a batch size of 64 and an epoch of 200, and the model weights with the highest recognition accuracy were then recorded.


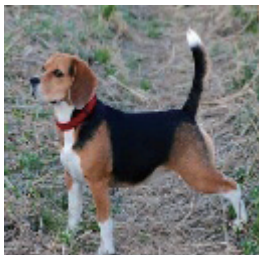
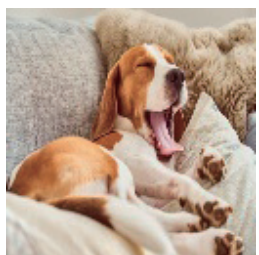
#### 4. Experimental Results

A two-stage model performance test was conducted in this work. In Stage 1, the recognition rate of dogs' body movements was measured, and the overall execution time was then measured in Stage 2. As tabulated in Table 2, as many as 1500 pieces of training data and 400 pieces of testing data were employed to train and test the model performance, and Fig. 3 gives confusion matrices as a way to demonstrate the training, testing, and overall (training + testing) results. In

Table 2  
Number of collected images for dogs' body movement recognition.

Class number	Type of movement	Total number of training data	Total number of testing data
0	Eating	500	132
1	Tail raising	500	140
2	Yawning	500	128

Table 3  
(Color online) Sample image for each type of movement.

Class number	0	1	2
Sample image			

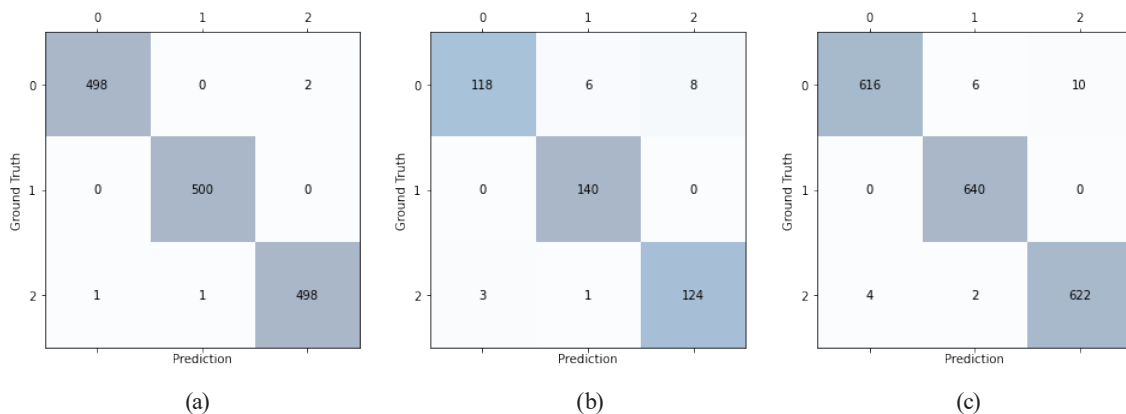


Fig. 3. (Color online) Confusion matrices for accuracy analyses. (a) Training. (b) Testing. (c) Overall.

a confusion matrix, diagonal and off-diagonal entries represent the numbers of times that body movements were recognized and misrecognized, respectively. Therefore, Fig. 3 gives recognition rates of 99.73% in training and 95.50% in testing, and finally, an overall recognition rate of 98.84%.

Table 4 gives the computational load required in each stage to process an image. The load test was conducted on 1000 images using a PC as specified in Table 1. As tabulated in Table 4, it took the YOLOv4 model 64.52 ms to detect an image in Stage 1, whereas it took the presented model as short as 0.38 ms to recognize a specific body movement in Stage 2. Then, the overall load is simply  $64.52 + 0.38 = 64.90$  ms, namely, 15.41 FPS, and it is obviously dominated by the load in Stage 1. This simple recognition model has been experimentally validated to give a high recognition rate.

Table 4

Execution time spent in each stage.

YOLOv4 detection model	64.52 ms per image
Proposed recognition model	0.38 ms per image
Overall load	64.90 ms per image

## 5. Conclusions

In this paper, we present a video editor for dogs by which video recording and editing become efficient for pet owners. Pet body movements were well recognized using a simple and efficient model, which takes up only approximately 2.8 MB of memory but gives a recognition accuracy of up to 98.84%. It must be stressed that nothing like this video editor is available yet in the market, and it is expected to be an easy-to-use and high-performance tool for pet owners when commercialized. As the next step, our team aims to upgrade the presented video editor, such that the upgrade version can recognize more types of body movements of dogs and even recognize those of other types of pet.

## Acknowledgments

This research was financially supported by the Ministry of Science and Technology under grant number MOST 110-2637-E-167-004 and the Ministry of Economic Affairs under grant number 108-EC-17-A-02-S5-008, Taiwan. The authors are deeply indebted as well to the coauthors of a cited paper, H. Y. Lai and H. H. Huang, for their contribution as the basis of this work.

## References

- 1 W. G. Hatcher and W. Yu: *IEEE Access* **6** (2018) 24411.
- 2 E. Min, X. Guo, Q. Liu, G. Zhang, J. Cui, and J. Long: *IEEE Access* **6** (2018) 39501.
- 3 O. I. Abiodun, A. Jantan, A. E. Omolara, K. V. Dada, A. M. Umar, O. U. Linus, H. Arshad, A. A. Kazaure, U. Gana, and M. U. Kiru: *IEEE Access* **7** (2019) 158820.
- 4 L. Jiao and J. Zhao: *IEEE Access* **7** (2019) 172231.
- 5 A. Krizhevsky, I. Sutskever, and G. E. Hinton: *Adv. Neural Inf. Process. Syst.* **25** (2012) 1097.
- 6 K. Simonyan and A. Zisserman: *CoRR* (2014). <http://arxiv.org/abs/1409.1556> (accessed April 2016).
- 7 C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich: *Proc. IEEE Conf. Computer Vision and Pattern Recognition (IEEE, 2015)* 1. <https://doi.org/10.1109/CVPR.2015.7298594>
- 8 K. He, X. Zhang, S. Ren, and J. Sun: *Proc. IEEE Conf. Computer Vision and Pattern Recognition (IEEE, 2016)* 770. <https://doi.org/10.1109/CVPR.2016.90>
- 9 S. Ren, K. He, R. Girshick, and J. Sun: *IEEE Trans. Pattern Anal. Mach. Intell.* **39** (2017) 1137.
- 10 W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C. Y. Fu, and A. C. Berg: *CoRR* (2015) <https://arxiv.org/abs/1512.02325> (accessed July 2018).
- 11 M. Tan, R. Pang, and Q. V. Le: *Proc. IEEE/CVF Conf. Computer Vision and Pattern Recognition (IEEE, 2020)* 10778. <https://doi.org/10.1109/CVPR42600.2020.01079>
- 12 J. Redmon and A. Farhadi: *CoRR* (2018). <https://arxiv.org/abs/1804.02767> (accessed July 2018).
- 13 A. Bochkovskiy, C. Y. Wang, and H. Y. Liao: *CoRR* (2020) <https://arxiv.org/abs/2004.10934> (accessed May 2020).

- 14 C. Y. Wang, A. Bochkovskiy, and H. Y. Liao: CoRR (2020) <https://arxiv.org/abs/2011.08036> (accessed December 2020).
- 15 K. He, G. Gkioxari, P. Dollár, and R. Girshick: Proc. IEEE Int. Conf. Computer Vision (IEEE, 2017) 2980. <https://doi.org/10.1109/ICCV.2017.322>
- 16 N. Audebert, B. L. Saux, and S. Lefevre: IEEE Geosci. Remote Sens. Mag. **7** (2019) 159.
- 17 P. Tang, X. Wang, B. Shi, X. Bai, W. Liu, and Z. Tu: IEEE Trans. Neural Netw. Learn. Syst. **30** (2019) 2244.
- 18 F. Schroff, D. Kalenichenko, and J. Philbin: Proc. IEEE Conf. Computer Vision and Pattern Recognition (IEEE, 2015) 815. <https://doi.org/10.1109/CVPR.2015.7298682>
- 19 H. Wang, Y. Wang, Z. Zhou, X. Ji, D. Gong, J. Zhou, Z. Li, and W. Liu: Proc. IEEE Conf. Computer Vision and Pattern Recognition (IEEE, 2018) 5265. <https://doi.org/10.1109/CVPR.2018.00552>
- 20 J. Deng, J. Guo, N. Xue, and S. Zafeiriou: Proc. IEEE Conf. Computer Vision and Pattern Recognition (IEEE, 2019) 4685. <https://doi.org/10.1109/CVPR.2019.00482>
- 21 COCO dataset: <http://cocodataset.org> (accessed July 2019).
- 22 C. Y. Yeh, H. Y. Lai, and H. H. Huang: IEEJ Trans. Electr. Electron. Eng. **16** (2021) 647.
- 23 T. Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie: Proc. IEEE Conf. Computer Vision and Pattern Recognition (IEEE, 2017) 2117. <https://doi.org/10.1109/CVPR.2017.106>
- 24 S. Liu, L. Qi, H. Qin, J. Shi, and J. Jia: Proc. IEEE Conf. Computer Vision and Pattern Recognition (IEEE, 2018) 8759. <https://doi.org/10.1109/CVPR.2018.00913>
- 25 K. He, X. Zhang, S. Ren, and J. Sun: IEEE Trans. Pattern Anal. Mach. Intell. **37** (2015) 1904.
- 26 C. Y. Wang, H. Y. Liao, Y. H. Wu, P. Y. Chen, J. W. Hsieh, and I. H. Yeh: Proc. IEEE/CVF Conf. Computer Vision and Pattern Recognition Workshops (IEEE, 2020) 1571. <https://doi.org/10.1109/CVPRW50498.2020.00203>