DEEP-LEARNING-BASED SCRATCHING BEHAVIOR QUANTIFICATION SYSTEM FOR LABORATORY MICE

[1]Chi-Jui Chang, [2]Oscar Tai-Yuan Chen, [3]Chih-Hung Lee, [4]Vincent S. Tseng*

[1]Institute of Computer Science and Engineering, National Yang Ming Chiao Tung University
E-mail: jerryyyyy708.cs12@nycu.edu.tw
[2]Institute of Computer Science and Engineering, National Yang Ming Chiao Tung University
E-mail: oscarchen.cs10@nycu.edu.tw
[3]Department of Dermatology, Kaohsiung Chang Gung Memorial Hospital and Chang Gung University College of Medicine
E-mail: zieben@cgmh.org.tw
[4]Department of Computer Science, National Yang Ming Chiao Tung University
E-mail: tseng@cs.nycu.edu.tw
[*]Vincent S. Tseng is the corresponding author.

ABSTRACT

Accurately detecting mouse scratching behavior is essential for biomedical research, particularly in pruritus studies. Manual recording or observation is time-consuming for the researchers. To address this, we developed a deep-learning-based automated laboratory mice scratching behavior quantification system on web-based platform for efficient analysis.

Considering the performance and efficiency, we applied YOLOv4-CSP as the detector to identify mouse scratching behavior in our system. The web platform provides an end-to-end solution, with the automated detection pipeline, the researchers can upload videos, and the system will process them through automated detection, visualize results, and generate reports. The system is designed for scalability and accessibility, the researchers can access the system remotely and process large datasets in batch.

Our work presents a practical and user-friendly solution for automated mouse scratching detection and quantification, which ensures more consistent and reproducible results in related research.

Keywords: Dermatology; Mouse Scratching Analysis; Object Detection; Deep Learning; Web-based System

1. INTRODUCTION

Mouse scratching behavior is essential in various biomedical research, particularly in studies related to pruritus or evaluating drugs [1][2]. The researchers can obtain valuable insights into the effect of medicine or skin condition observation by analyzing the scratching behavior of mice. Traditionally, monitoring and recording the mouse scratching behavior relies on manual observation, which means that the researchers must watch the whole length of mouse scratching videos or observe mice behavior directly and record each instance of scratching events by hand. However, such an approach is inefficient. The researchers might have to monitor the mice for hours and hours, which is significantly time-consuming and might lead to human error. Fatigue and environmental distractions can result in inconsistencies in the record, which can affect the reliability and reproducibility of the recorded results.

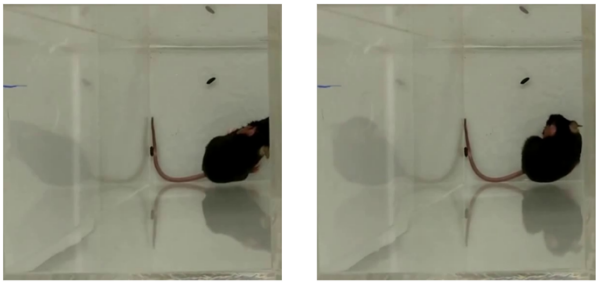

Fig. 1: Example frames of non-scratching (left) and scratching (right) mouse.

To address these challenges, we developed a deep-learning-based automated laboratory mice scratching behavior quantification system, applying object detection techniques to catch the scratching behavior of mice. With our system, the researchers are no longer required to analyze the videos manually. Instead, simply upload the recorded video of mice into the system. Next, the system will process the video automatically and then generate quantitative reports with the details of detected scratching behavior, eliminating the need for manual intervention. The automated process significantly reduces human labor, which makes the whole

observation more efficient and consistent. As a result, this system makes large-scale observation data collection possible, which allows the researchers to conduct more extensive studies that were previously impractical due to the limitations of manual analysis.

Object detection is a popular task in deep learning which has been widely applied in various real-world applications in the medical field. For example, disease diagnosis and behavioral analysis [3][4][5]. In the context of mouse scratching detection, object detection enables the model to determine whether the mouse being observed is performing a scratching behavior or not by analyzing the motion characteristics [6][7]. The model can be trained to recognize different movement patterns associated with scratching, enabling it to replace human observation with automated prediction effectively. The detection model follows the following process: the model first identifies the mouse's location and predicts the bounding box, then classifies whether the mouse is scratching or not. The bounding box can help enhance visual interpretability for real-world applications, allowing the researchers to verify whether the model correctly caught the mouse's location, and the binary label of scratching/non-scratching is the main target to predict.

For the object detection model, we selected YOLOv4-CSP [8], a model used in various object detection tasks, as the architecture. Although several versions of the YOLO-based model have been introduced in recent years, YOLOv4 remains one of the most stable architectures and was well validated, making it a reliable choice for our web system in real-world applications [9][10]. For automated analysis, stability and efficiency are the key factors. Therefore, YOLOv4-CSP is particularly suitable for our study, since it has well accuracy with balanced computational cost. YOLOv4-CSP serves as the detection module and performs frame-wise analysis of the video to determine the mouse scratching behavior in our system.

A critical challenge in training deep learning models of mouse scratching detection is that the binary label distribution of the dataset does not often remain consistent. In most mouse scratching research, the number of non-scratching frames is much larger than the number of scratching frames [6][7]. In some cases, the ratio can even reach 1:250, which makes it challenging to train and evaluate the model performance accurately. The model might tend to prioritize the majority class prediction, which might affect the performance in detecting rare but more important events–in this case, the frames of mouse scratching. The usefulness of the system will be significantly reduced if the model cannot effectively predict the scratching behavior of the mouse.

To mitigate this issue, we construct a mouse scratching dataset, ensuring the distribution of scratching and non-scratching frames to be more consistent. We collected a dataset where the videos with too few scratching frames will be carefully filtered. The total number of annotated videos is 555. All of the videos in the dataset are carefully selected to ensure the quality of data samples. The final dataset has a scratching-to-non-scratching frame ratio of approximately 1:2. By using this dataset for training, the model can better accurately detect the scratching event and prevent being overly influenced by the majority class. Since our primary objective in the task is to detect the scratching events as accurately as possible, we must deal with this problem to ensure the reliability and robustness of the model.

The developed web system can effectively replace the process of manual observation and recording, which is time-consuming. It provides an efficient and scalable solution for mouse scratching analysis with an automated deep learning-based approach. With YOLOv4-CSP integrated as the detection module, ensuring the dataset is balanced and enhancing the interpretability with visualization and quantitative report generation, the system not only improves the research efficiency but also provides a more consistent and reproducible behavior study in biomedical research. It is designed to provide a more convenient user experience for the researchers. The processing pipeline of the web system includes video preprocessing, cage segmentation, scratching detection, and post-processing for accurate event counting. Moreover, the user interface provides several functionalities, such as tracking processing status, scratching event visualization, and report generation.

The functionalities will be explained in detail in the Method section. With our developed system, researchers can ensure consistent and reproducible results when observing mouse scratching behavior. Furthermore, with the post-processing module, the system not only detects scratching events by frame but also provides the result of the actual scratching count. As a result, the researchers can obtain a quantitative and reliable measurement of scratching behavior using our web system. Our main contributions are as follows:

1. We built a balanced dataset for mouse scratching detection with 555 videos collected carefully, which mitigate the label distribution issue compared to previous studies. The balanced dataset ensures the model to better detecting the scratching behavior and reduce the occurrence of missed detections.
2. We developed a deep-learning-based automated laboratory mice scratching behavior quantification system on a web-based platform that greatly reduces the human effort and time required to observe mouse behavior directly. The system provides a user-friendly and scalable web-based solution and supports large-scale predictions.

The remaining of this paper is organized as follows. Section 2 reviews object detection in medical research and mouse scratching detection strategies. Section 3 presents our developed system, including the overall workflow, cage segmentation, detection model, and

post-processing. Section 4 provides detailed information on the datasets and metrics, experimental results, and the demonstration of our developed web system. Finally, Section 5 concludes the paper and discusses future directions.

2. RELATED WORKS

Automated mouse scratching or related behavior analysis has been a growing field of biomedical studies, particularly in animal behavior tracking and classification applications. Traditional methods for mouse scratching behavior analysis heavily rely on manual observation and recording, which is time-consuming and unstable due to possible human error. To address these challenges, increasing research focuses on applying computer vision and deep learning techniques to behavioral studies to provide efficient and scalable analysis.

2.1. Object Detection in Biomedical Research

Object detection is one of the most critical tasks in computer vision, and it has been widely used in medical and behavioral research. Deep-learning-based object detection models, such as Faster R-CNN, SSD, and YOLO, have shown robust performance in various biomedical applications [8][11][12]. For example, disease diagnosis, cell detection, and animal behavior tracking [13][14][15][16]. Among these, YOLO-based models are the most widely selected for real-world application due to their high detection speed and stable model performance. As a result, YOLOv4-CSP was selected for our system as it provides a good balance between detection accuracy and computational cost. Its robustness makes it suitable for real-world applications in our system.

2.2. Mouse Scratching Detection Approaches

Several recent studies have explored automated methods for mouse scratching detection. Most approaches rely on post-estimation or object detection techniques. Post-estimation models such as DeepLabCut and OpenPose have been widely applied to track mouse movements and infer scratching behavior based on keypoint detection [17][18]. However, such methods require complicated, high-quality datasets containing precise skeletal annotation, which significantly increases the costs in large-scale studies.

On the other hand, object detection approaches offer a more practical and scalable solution. With object detection models, we can directly classify scratching behavior based on the video frames instead of precisely tracked key points. Previous works have employed CNN-based models for several animal behavior classifications. For example, grooming detection and pain recognition demonstrate the effectiveness of using CNN-based object detection for behavioral analysis [19][20]. However, many of these models are trained on highly imbalanced datasets, especially in the research of mouse scratching detection, where the number of non-scratching frames is significantly more than scratching frames, which might limit the model performance [6][7].

In our work, we address the issue by constructing a balanced dataset and using it to train our YOLOv4-CSP

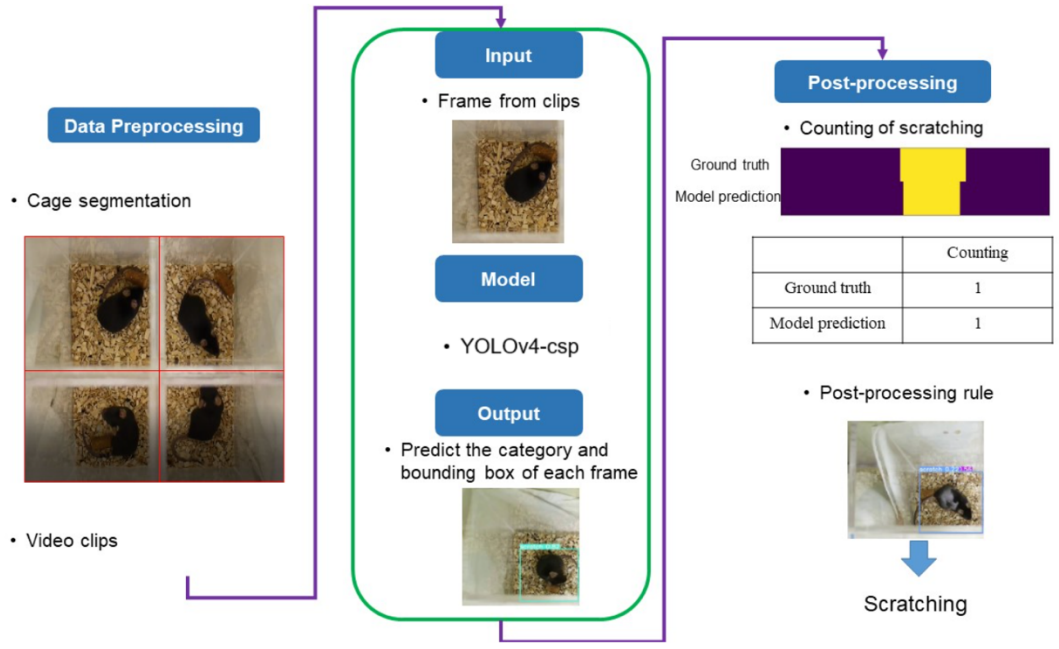

Fig. 2: The overall workflow of our developed laboratory mice scratching behavior quantification web system.

object detection model. This mitigates the model bias and improves the detection performance. This approach ensures that the main target – scratching events – is correctly recognized, which is crucial for reliable biomedical studies.

2.3. Web-Based Platforms for Behavioral Analysis

There are increasing applications in biomedical research that integrate deep learning models into web-based systems due to their scalability and accessibility. Previous research has developed several web-based applications, such as cloud-based medical image analysis platforms and video-based movement tracking [21][22]. This system allows researchers to process large datasets remotely and reduce the need for high human efforts.

For animal behavioral studies, web-based solutions have been implemented in zebrafish movement tracking and rodent activity monitoring [23][24]. However, limited research has focused on integrating deep learning-based object detection techniques into web applications for mouse scratching analysis. Our work bridges this gap by developing an automated web-based scratching detection system that enables the researchers to obtain scratching behavior observation reports by simply uploading the video into the system. They can also gain information from the visualization result provided by the video, which significantly reduces manual workload and makes large-scale studies more feasible.

3. METHODS

In this section, we will introduce the web system we developed for real-world application. We deployed the detection model to the web system and integrated it with the post-processing module. The system is a fully automated analysis pipeline that allows the users to upload mouse-scratching videos, which will then be processed through cage segmentation, scratching detection, visualization, and report generation.

3.1. Overall Workflow

The overall web system workflow is shown in Figure 2. There are several subpages with different functionalities. The homepage displays the user instructions and the process information, including pending and processed files. The system automatically processes pending videos every 12:00 daily. The upload page is where the user can submit the videos for mouse scratching analysis, and the videos will be converted into a compatible format and prepared for cage segmentation. The analysis progress page allows the users to track the status of the video processing. The information provided includes cage segmentation results, analysis stages, scratching records, and visualizations of the detected scratching behavior. Finally, the analysis results page lists all reports of the mouse detection that are ready for the user to download. The report includes key information, such as the starting and ending time of the scratching event and the total scratching counts.

3.2. Cage Segmentation

The input video of the system is required to contain four mice separated in different cages. The web system provides an interface for the users to perform cage segmentation. The default coordinates of the system are set to (50, 100) for general use, and the users can manually adjust the coordinates to match their input videos. After the segmentation, the video will be split into four separate clips with a single mouse in each clip. These individual videos will then be forwarded to the detection model for further analysis.

3.3. Detection Model

With strong detection performance and tolerable inference speed, we selected YOLOv4-CSP as the detection model. Specifically, YOLOv4-CSP utilizes CSPDarknet53 as its backbone, effectively reducing computational costs. Moreover, Spatial Pyramid Pooling (SPP) and Path Aggregation Networks (PAN) are applied to enhance the ability of the model to extract features. To train the detection model, we preprocessed the input video using the following steps. First, we convert the video into frames for the model to take as input images. Next, the input frame will be resized to 640 × 640. We then apply several augmentations to the input video frame, including rotation, translation, scaling, shearing, and perspective transformation. Additionally, we enhance the brightness, contrast, and saturation of video frames. Random cutouts, flipping, and mosaic augmentation are also applied to prevent overfitting. After training, we deploy the trained YOLOv4-CSP model on the web system as our primary object detection model for mouse scratching analysis. At this stage, each of the four segmented videos is processed frame by frame, with the model detecting both the bounding box of the mouse and whether a scratching is occurring.

3.4. Post-Processing

After obtaining the frame-wise predictions, the system translates the prediction of the video frames into the actual count of scratching. Each time successive counts occur, more than a manually selected threshold will be counted as a scratch. The actual scratching count and the starting and ending point of each scratch occurrence will be recorded in the report to provide users with a comprehensive analysis. This web system is designed to automate the analysis process, which reduces the time for the researchers to observe and record the mouse scratching events manually.

The visualization and process status provided in each subpage also makes it more convenient and flexible for user experience. The system can provide precise detection results and detailed scratching frequency and duration reports. This significantly reduces the required time for mouse scratching analysis.

4. EXPERIMENTS

4.1. Experimental Settings

4.1.1. Dataset

We collected 555 videos filmed above the mice cage as the dataset. The total number of frames in our collected videos is 59,340. The videos are selected carefully to ensure the ratio of scratching to non-scratching frames remains consistent. Specifically, the ratio of scratch and non-scratch frames is 1:2. We split the dataset into training, validation, and testing sets with a ratio of 3:1:1 in a manner of 5-folds label-stratified validation.

4.1.2. Metrics

In this work, we focus on the correctness of predicting scratching in the video instead of the bounding box region. As a result, we use accuracy, f1-score, precision, and recall to determine the performance of our detection model. Each frame will be assigned a label either scratch or non-scratch. We check if the prediction of the detection model of each frame is the same as the label.

Table 1: The performance of scratching detection (%).

	Accuracy	*Recall*	*Precision*	*F1-Score*
YOLOv4 -CSP	86.77± 1.66	83.46± 2.05	89.02± 2.20	85.04± 1.96

4.2. Experiment Results

4.2.1. The Performance of Scratching Detection

In this section, we analyze the performance of the YOLOv4-CSP model used in our Web System. Table 1 shows the experimental results of scratching detection. YOLOv4-CSP demonstrated a strong prediction performance. The model achieved an accuracy of 86.77%, with 83.46% recall. Based on the recall score, we can observe that even if there is a 2:1 ratio between non-scratching and scratching labels, the model can effectively identify most scratching events in the dataset. This result shows that YOLOv4-CSP is a suitable selection for this task. Due to the strong performance and relatively low computational cost, we integrated the trained YOLOv4-CSP into the web system as the detection module.

4.2.2. System Demonstration

In this section, we present the actual workflow of the web system when a video is uploaded to demonstrate its functions and the results generated that are available for researchers. The system is designed to provide an end-to-end scratching detection process from data input to the final report generation, reducing the need for manual observation and annotation to enhance research efficiency. The following paragraph will outline each step of the process in detail.

The workflow begins with the web system homepage. An illustration of the homepage is shown in Figure 3. The interface consists of three main sections: (1) The user instructions block, which provides guidance on how to operate the system; (2) The waiting video block, which lists videos currently in the queue for cage segmentation, allowing users to track the progress of their uploaded data; and (3) The un-processed video block, which displays previously uploaded videos, where the listed videos are waiting for the system to process the videos. This structure ensures users can efficiently manage their video datasets and monitor the analysis pipeline.

The user can upload the video from the upload interface. After selecting and submitting the video files required to perform prediction. After the video is uploaded to the system, the website will proceed to the cage segmentation page. The segmentation process

Fig. 3: The homepage of our developed web system.

Fig. 4: The user interface of cage segmentation.

搔抓記錄

No.	Start time(mm:ss:msms)	End time(mm:ss:msms)
1	02 : 07 : 07	02 : 07 : 10
2	02 : 08 : 09	02 : 08 : 21
3	02 : 09 : 06	02 : 13 : 07
4	02 : 13 : 17	02 : 14 : 03
5	02 : 14 : 13	02 : 15 : 20
6	02 : 30 : 15	02 : 30 : 19

Fig. 5: The user interface of scratching log.

	A	B	C	D	E	F	G
1	Total count	Start	End	Time duration	Start	End	Time duration
2	4	01:43.6	01:43.8	0.17	10:45.7	10:46.1	0.33
3							
4							
5							
6							
7							
8							
9							

Fig. 7: The content of the CSV report.

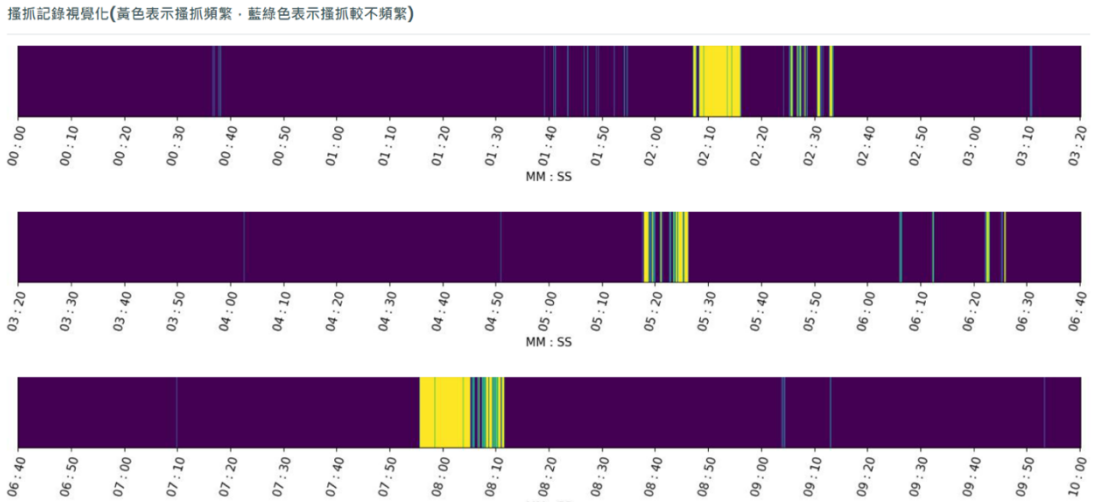

Fig. 6: The frame sequence visualization of detected scratching behaviors in a video.

allows the user to separate multiple mice in the video into different videos, ensuring each mouse is analyzed individually.

As shown in Figure 4, the segmentation interface allows users to input specific coordinates, and the original video will be divided into four separate clips with the given coordinates, with each clip containing a single mouse. The system will then independently apply mouse scratching detection process to each video. The segmentation results are displayed on the screen to allow researchers to verify if the separation results correctly separated the mice before proceeding to detection. This approach ensures the bounding boxes and analysis results provided later will be correctly aligned with individual mice. Once the user confirms the segmentation, the system proceeds to the detection process. During the detection phase, the system applies the YOLOv4-CSP model to detect and classify the mouse scratching behavior by frame within each cage (i.e., each segmented video). The detection results will be displayed directly on the web system, which allows the researcher to assess the prediction result of the scratching behavior. Each video will be associated with a scratching log, shown in Figure 5, which provides precise timestamps of when a scratching event occurs. This log makes it easy for researchers to determine when scratching behavior was detected, eliminating the need to manually watch the whole video. Additionally, as shown in Figure 6, the system generates visualized analysis results, where the more frequent scratching

Fig. 8: The list of downloadable report files.

occurs will be highlighted more, allowing the researchers to examine the scratching trends over time. The visualization charts provide an overview of scratching behavior throughout the whole timeline, making it easier to identify when periods of frequent scratching activity occurred. For example, in our demonstration dataset, the system detected increased scratching activity at 2:10, 5:17, and 7:55, highlighting key intervals that may be relevant for biomedical analysis. These visual representations enhance data interpretability and facilitate more effective decision-making in scratching behavior studies.

Finally, the platform generates a CSV report for each cage after completing the process. The report summarizes the prediction results for the mouse

behavior. These reports can be downloaded and used for further analysis, enabling researchers to integrate the behavior observation into broader experimental studies. An example of the content of the report is shown in Figure 7. It provides a structured summary of all detected events, including the total number of scratches and their corresponding timestamps. The report can be downloaded in the final analysis result page shown on Figure 8. All downloadable reports will be listed on the table of this page.

This section demonstrates the complete workflow using our automated laboratory mice scratching behavior quantification system, from video input and segmentation to detection, visualization, and report generation. With the automated scratching detection process, the manual workload required for behavioral analysis is significantly reduced, and the system ensures consistent and reproducible results. This approach provides researchers with a scalable and efficient tool for studying mouse scratching behavior in biomedical research.

5. CONCLUSION

In this study, we developed a deep-learning-based automated scratching behavior quantification web system for laboratory mice to address the challenges of inefficiencies and inconsistencies in manual observation and recording. With YOLOv4-CSP integrated as the detection model, our system ensures the analysis process is scalable, consistent, and reproducible, eliminating the need for extensive human supervision.

One of contributions in our work is to collect and build a balanced mouse-scratching video dataset. Unlike traditional datasets where non-scratching frames are usually much more than scratching frames, we carefully constructed a dataset with a 1:2 ratio, allowing the model to learn from the features of both behaviors effectively.

Additionally, our user-friendly web system provides a comprehensive pipeline to automatically process the video and generate the report and visualization results. This can serve as an efficient and accessible tool for large-scale research. The benefit of our developed system can not only minimize human effort and time cost but also enhance the reproducibility of results.

Our experimental results demonstrate that the YOLOv4-CSP model can effectively detect mouse scratching behaviors, achieving an accuracy of 86.77%. The system provides a practical and scalable solution for biomedical research, which is beneficial for future researchers who require large-scale analysis. For future work, we will explore using different model architectures to further enhance the detection result and integrate advanced post-processing techniques for a more accurate event count prediction.

ACKNOWLEDGEMENT

This research was supported by National Yang Ming Chiao Tung University, Taiwan, R.O.C and Kaohsiung Chang Gung Memorial Hospital, Taiwan, R.O.C. under grant no. CGMH-NYCU-113-CORPG8P0441.

REFERENCES

[1] S. G. Shimada and R. H. LaMotte, “Behavioral differentiation between itch and pain in mouse,” *Pain*, vol. 139, no. 3, pp. 681–687, Oct. 2008.

[2] K. Matsuda, Y. Kitano, M. Sawahata, T. Kume, and D. Uta, “Mirogabalin inhibits scratching behavior of spontaneous model mouse of atopic dermatitis,” *Frontiers in Pharmacology*, vol. 15, 2024.

[3] R. Qu, Y. Yang, and Y. Wang, “COVID-19 Detection Using CT Image Based on YOLOv5 Network,” in *2021 3rd International Academic Exchange Conference on Science and Technology Innovation, IAECST 2021*, Institute of Electrical and Electronics Engineers Inc., 2021, pp. 622–625.

[4] S. Schneider, G. W. Taylor, and S. Kremer, “Deep learning object detection methods for ecological camera trap data,” in *Proceedings - 2018 15th Conference on Computer and Robot Vision, CRV 2018*, Institute of Electrical and Electronics Engineers Inc., Dec. 2018, pp. 321–328.

[5] S. Jahn, G. Schmidt, L. Bachmann, H. Louton, T. Homeier-Bachmann, and A. K. Schütz, “Individual behavior tracking of heifers by using object detection algorithm YOLOv4,” *Frontiers in Animal Science*, vol. 5, 2024.

[6] K. Kobayashi, S. Matsushita, N. Shimizu, S. Masuko, M. Yamamoto, and T. Murata, “Automated detection of mouse scratching behaviour using convolutional recurrent neural network,” *Scientific Reports*, vol. 11, no. 1, Dec. 2021.

[7] H. Yu, J. Xiong, A. Y. Ye, S. L. Cranfill, T. Cannonier, M. Gautam, M. Zhang, R. Bilal, J. E. Park, Y. Xue, V. Polam, Z. Vujovic, D. Dai, W. Ong, J. Ip, A. Hsieh, N. Mimouni, A. Lozada, M. Sosale, A. Ahn, M. Ma, L. Ding, J. Arsuaga, W. Luo, “Scratch-AID, a deep learning-based system for automatic detection of mouse scratching behavior with high accuracy,” *Elife*, vol. 11, p. 84042, 2022.

[8] Wang, Chien-Yao, Alexey Bochkovskiy, and Hong-Yuan Mark Liao. "Scaled-yolov4: Scaling cross stage partial network." *Proceedings of the IEEE/cvf conference on computer vision and pattern recognition*. 2021.

[9] X. Hu, Y. Liu, Z. Zhao, J. Liu, X. Yang, C. Sun, S. Chen, B. Li, and C. Zhou, “Real-time detection of uneaten feed pellets in underwater images for aquaculture using an improved YOLO-V4 network,” *Computers and Electronics in Agriculture*, vol. 185, Jun. 2021.

[10] D. Wu, S. Lv, M. Jiang, and H. Song, “Using channel pruning-based YOLO v4 deep learning algorithm for the real-time and accurate detection of apple flowers in natural

environments," *Computers and Electronics in Agriculture*, vol. 178, Nov. 2020.

[11] R. Girshick, "Fast R-CNN," in *2015 IEEE International Conference on Computer Vision (ICCV)*, IEEE, Dec. 2015, pp. 1440–1448.

[12] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "SSD: Single Shot MultiBox Detector," in *Computer Vision – ECCV 2016*, B. Leibe, J. Matas, N. Sebe, and M. Welling, Eds., Cham: Springer International Publishing, 2016, pp. 21–37.

[13] E. A. Aldakheel, M. Zakariah, and A. H. Alabdalall, "Detection and identification of plant leaf diseases using YOLOv4," *Frontiers in Plant Science*, vol. 15, 2024.

[14] M. Kang, C. M. Ting, F. F. Ting, and R. C. W. Phan, "ASF-YOLO: A novel YOLO model with attentional scale sequence fusion for cell instance segmentation," *Image and Vision Computing*, vol. 147, Jul. 2024.

[15] M. Kang, C.-M. Ting, F. F. Ting, and R. C.-W. Phan, "CST-Yolo: A Novel Method For Blood Cell Detection Based On Improved Yolov7 And CNN-Swin Transformer," in *2024 IEEE International Conference on Image Processing (ICIP)*, IEEE, Oct. 2024, pp. 3024–3029.

[16] M. Choiński, M. Rogowski, P. Tynecki, D. P. J. Kuijper, M. Churski, and J. W. Bubnicki, "A First Step Towards Automated Species Recognition from Camera Trap Images of Mammals Using AI in a European Temperate Forest," in *Computer Information Systems and Industrial Management*, K. Saeed and J. Dvorský, Eds., Cham: Springer International Publishing, 2021, pp. 299–310.

[17] A. Mathis, P. Mamidanna, K. M. Cury, T. Abe, V. N. Murthy, M. W. Mathis, and M. Bethge, "DeepLabCut: markerless pose estimation of user-defined body parts with deep learning," *Nature Neuroscience*, vol. 21, no. 9, pp. 1281–1289, Sep. 2018.

[18] Z. Cao, G. Hidalgo, T. Simon, S. E. Wei, and Y. Sheikh, "OpenPose: Realtime Multi-Person 2D Pose Estimation Using Part Affinity Fields," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 43, no. 1, pp. 172–186, Jan. 2021.

[19] N. Sakamoto, K. Kobayashi, T. Yamamoto, S. Masuko, M. Yamamoto, and T. Murata, "Automated Grooming Detection of Mouse by Three-Dimensional Convolutional Neural Network," *Frontiers in Behavioral Neuroscience*, vol. 16, Feb. 2022.

[20] M. Feighelstein, I. Shimshoni, L. R. Finka, S. P. L. Luna, D. S. Mills, and A. Zamansky, "Automated recognition of pain in cats," *Scientific Reports*, vol. 12, no. 1, Dec. 2022.

[21] J. Egger, D. Wild, M. Weber, C. A. Ramirez Bedoya, F. Karner, A. Prutsch, M. Schmied, C. Dionysio, D. Krobath, Y. Jin, C. Gsaxner, J. Li, and A. Pepe, "Studierfenster: an Open Science Cloud-Based Medical Imaging Analysis Platform," *Journal of Digital Imaging*, vol. 35, no. 2, pp. 340–355, Apr. 2022.

[22] R. Liu, T. Lu, S. Yuan, H. Zhou, and M. Gowda, "SmartDampener: An Open Source Platform for Sport Analytics in Tennis," *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, vol. 8, no. 3, Sep. 2024.

[23] G. Teicher, R. M. Riffe, W. Barnaby, G. Martin, B. E. Clayton, J. G. Trapani, and G. B. Downes, "Marigold: a machine learning-based web app for zebrafish pose tracking," *BMC bioinformatics*, vol. 26, no. 1, p. 30, Dec. 2025.

[24] E. Zentrich, S. R. Talbot, A. Bleich, and C. Häger, "Automated Home-Cage Monitoring During Acute Experimental Colitis in Mice," *Frontiers in Neuroscience*, vol. 15, Oct. 2021.