JCSE Journal of Computing Science and Engineering

Regular Paper

Journal of Computing Science and Engineering,
Vol. 17, No. 3, September 2023, pp. 100-108

Exploration of Key Point Localization Neural Network Architectures for Y-Maze Behavior Test Automation

Gwanghee Lee, Sangjun Moon†, Dasom Choi†, Gayeon Kim†, and Kyoungson Jhang*
Department of Computer Engineering, Chungnam National University, Daejeon, Korea
manggu251@o.cnu.ac.kr, 202102544@o.cnu.ac.kr, choi252627@o.cnu.ac.kr, gyeon0907@o.cnu.ac.kr, sun@cnu.ac.kr

Abstract

The Y-maze behavioral test is a pivotal tool for assessing the memory and exploratory tendencies of mice in novel environments. A significant aspect of this test involves the continuous tracking and pinpointing of the mouse's location, a task that can be labor-intensive for human researchers. This study introduced an automated solution to this challenge through camera-based image processing. We argued that key point localization techniques are more effective than object detection methods, given that only a single mouse is involved in the test. Through an experimental comparison of eight distinct neural network architectures, we identified the most effective structures for localizing key points such as the mouse's nose, body center, and tail base. Our models were designed to predict not only the mouse key points but also the reference points of the Y-maze device, aiming to streamline the analysis process and minimize human intervention. The approach involves the generation of a heatmap using a deep learning neural network structure, followed by the extraction of the key points' central location from the heatmap using a soft argmax function. The findings of this study provide a practical guide for experimenters in the selection and application of neural network architectures for Y-maze behavioral testing.

Category: Human-Computer Interaction

Keywords: Deep learning; Computer vision; key point detection; Y-maze behavior test

I. INTRODUCTION

Behavioral experiments are commonly used to test the learning and memory functions in brain disease research and clinical trials such as Alzheimer and dementia. For example, researchers widely use the Y-maze behavior test to assess behavioral tasks in preclinical analyses of short-term spatial learning and memory [1]. The Y-maze is a simple maze with three arms shaped like Y, as shown in Fig. 1. Y-maze is a behavioral test that measures the willingness of normal rodents to explore new environments. Rodents usually tend to explore new environments rather than ones they have already explored [2]. Based on these rodents' behavior, the effect of the administered drug could be determined on the medically manipulated variables by observing the behavioral patterns of the experimental mice.

Experimental mice generate data continuously, independent of the observer's schedule, necessitating constant observation and recording of their behavior. Temporary interruptions can lead to inaccurate recording of sequential behavioral patterns. Moreover, changes in the observer's criteria due to factors such as environmental changes or fatigue can significantly undermine the reliability of the

Open Access http://dx.doi.org/10.5626/JCSE.2023.17.3.100 http://jcse.kiise.org

Received 17 July 2023; **Accepted** 09 September 2023
*Corresponding Author
†These authors contributed equally to this work.

 pISSN: 1976-4677 eISSN: 2093-8020

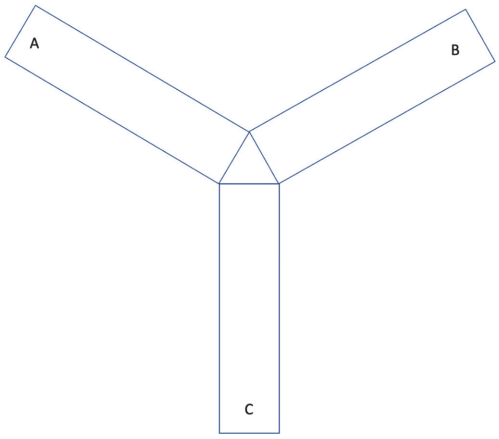

Fig. 1. A Y-maze comprises three arms A, B, and C.

Fig. 2. Prediction two cases such as device (green) and moues (red).

experiment. While non-stop observation could enhance data reliability, it is impractical due to the substantial increase in experimental costs.

The introduction of a deep learning model capable of processing Y-maze images from a simple camera offers a solution. This approach enables continuous experimentation and analysis, thereby enhancing the reliability of the experiment and reducing costs.

Commercial Y-maze automation systems, while expensive, are recognized for their superior performance. Previous research on Y-maze test automation has utilized high-computation depth cameras like Kinect or entrance detection sensors [3, 4]. These methods, akin to commercial products, also necessitate costs for environmental setup and software development. In this study, we provided performance test results for various deep learning models that utilize images from a basic camera to predict mouse location. Our findings aimed to assist experimenters in establishing cost-effective Y-maze experimental environments tailored to their specific needs.

While the object detection method based on deep learning models [5–10] could be employed to implement the Y-maze experiment, it may not be the most efficient approach. Given that only a single mouse is involved in the Y-maze, continuous positioning of the mouse is sufficient for result analysis. Therefore, a deep learning model that estimates three key locations—the tip of the mouse's nose, the center of its body, and the beginning of its tail—is more appropriate for locating the experimental mouse within the Y-maze. To further streamline the analysis of the Y-maze test, our models also predict four additional key points that provide the reference positions of the Y-maze device. These reference points, highlighted as green dots in Fig. 2, typically need to be manually drawn or set by the experimenters. These approaches, akin to key point localization methods used in applications such as pose estimation, face alignment, and facial landmark detection, can be effectively applied to estimate the location of the mouse or device within the Y-maze, thereby enhancing the efficiency and accuracy of the experiment.

Sections II and III introduce existing studies on position estimation techniques mainly used for pose estimation or face alignment. Section IV shows the experimental results of various architectures and hyperparameters. The last section describes a summary of the paper and future research. The key contributions of this paper are as follows:

1. The automation of the Y-maze behavior test was demonstrated using straightforward key point localization models, thereby circumventing the need for methods with high computational demands such as segmentation, object detection, or point cloud processing.
2. Deep learning models that utilize simple camera imagery was introduced, negating the need for depth cameras or additional sensors.
3. A comprehensive overview of the performance and computational requirements of various neural network models was provided facilitating their selection based on specific needs.
4. The performance outcomes achieved through the selection and combination of various neural network optimization options was illustrated on the ResNet-50 backbone.
5. The proposed models were designed to predict both device reference points and mouse key points, thereby simplifying the analysis process of the Y-maze behavior test.
6. In the spirit of open science, we make the source code of our models publicly available. This will enable experimenters to tailor their experimental environment according to the specific requirements of the Y-maze test and their individual research

needs. The source code can be accessed at https://github.com/gompanghee/KeypointDet_for_YMaze.

II. RELATED WORKS

Pose estimation and facial landmark detection are representative problems requiring feature point location prediction. Pose estimation is a problem of predicting the location of joint points in the human body and recognizing motions based on those locations. Facial landmark detection is a problem of predicting the locations of facial feature points, and its usage is for facial expression recognition, character identification, and facial pose estimation. Common to both problems is the key point location prediction. A heatmap in the form of gaussian distribution, as shown in Fig. 3, was created first. Then a method of estimating the key point position using gaussian approximation or soft argmax function [11] was applied to the heatmap. This process is called heatmap regression.

Pose estimation and facial landmark detection are key tasks that necessitate the prediction of feature point locations. Pose estimation involves predicting the positions of human body joints to discern movements, while facial landmark detection involves predicting facial feature points for applications such as facial expression recognition, character identification, and facial pose estimation. Both tasks share a common element: key point location prediction. This process, known as heatmap regression, begins with the creation of a heatmap in the form of a Gaussian distribution, as depicted in Fig. 3. Subsequently, a Gaussian approximation or soft argmax function [11] is applied to the heatmap to estimate the key point position.

Examples of deep learning models that use this process are UniPose [12], HRNet [13], and ADNet [14]. As shown in Fig. 4, UniPose and HRNet estimate the location of feature points using heatmap regression in pose estimation, and ADNet estimates the locations of feature points using heatmap regression in facial landmark detection.

The creation of a heatmap centered on each key point

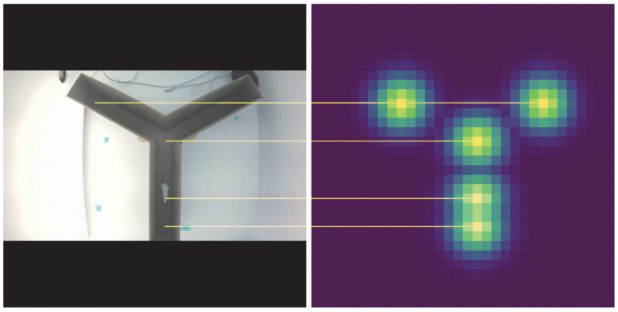

Fig. 3. (a) Source image and (b) Gaussian distribution heatmap.

Fig. 4. (a) Heatmap regression in pose estimation (http://human-pose.mpi-inf.mpg.de) and (b) facial landmark detection (https://thispersondoesnotexist.com).

necessitates a backbone network architecture capable of extracting relevant feature maps. In localization-related applications, including object detection, pose estimation, and facial landmark detection, backbone models such as VGGNet [15], ResNet [16, 17], Inception [18, 19], and EfficientNet [20, 21] are frequently employed. Researchers have explored methods for effective feature map extraction based on these backbone models.

OpenPose [22] employs VGGNet as its backbone, leveraging the deepened convolutional neural network (CNN) layer depth for effective feature extraction. EfficientPose [23] uses EfficientNet as its backbone, optimizing the combination of network width, depth, and input resolution to extract feature maps with minimal unnecessary computations. TFPose [24] utilizes a transformer-based backbone [25], which has a low dependency on training parameters, thus mitigating side effects such as gradient vanishing and hard backpropagation compared to CNN and multi-layer perceptron (MLP) [26]. This allows the transformer to extract features more easily by connecting layers that are relatively deep and wide.

Our experiment targets eight neural networks: VGG-19 [15], ResNet-50 [16, 17], Inception v3 [18, 19], MobileNet v2 [27], EfficientNet v2 [20, 21], Swin v2 (a transformer-based small neural network) [28, 29], U-Net [30], and Hourglass Net [31].

The method used to identify key point locations in pose estimation and facial landmark detection can be applied to localize mouse key points. Consequently, solving these two problems leads to similar backbones and processes. The subsequent section details the creation of a heatmap and the identification of key point locations within it.

III. HEATMAP REGRESSION

A. Gaussian Distribution Heatmap

A heatmap on a CNN can be used to represent probable key point locations in an image or feature map. Instead of using numerical coordinates, these locations can be

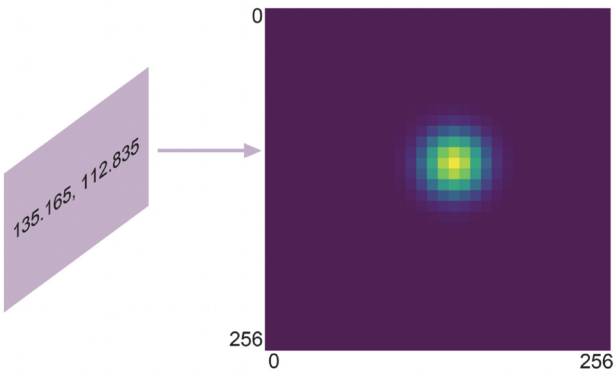

Fig. 5. Localization information represented by probability distribution.

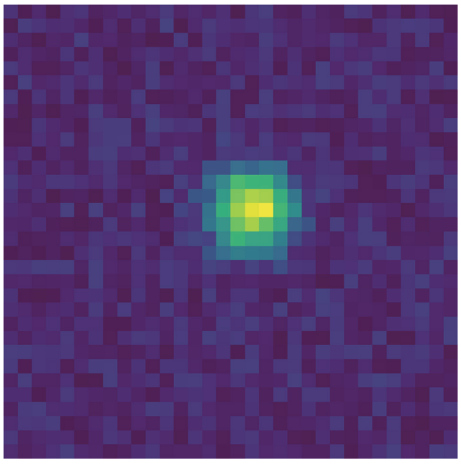

Fig. 6. Probability space represented by deep learning model.

depicted as bright spots on the heatmap. In essence, each pixel's brightness value in the image signifies the probability of it being a key point, creating a probability distribution as illustrated in Fig. 5.

As shown in Fig. 6, a deep learning model can be trained to output high values for locations with a high likelihood of being key points and low values for less probable locations. The resulting feature map closely resembles a heatmap.

Given that a probability distribution approximates a normal distribution as the population size increases, the Gaussian distribution, a standard normal distribution, serves as an excellent model for the heatmap.

B. Localization using Gaussian Approximation

To extract key point coordinates from the heatmaps, we can utilize the Gaussian distribution approximation. Key points can be easily found using least square approximation with the formula shown in Eq. (1). Since a heatmap is modelled with gaussian distribution, the method estimates the key point location with a Gaussian distribution. Approximated $\hat{\alpha}$ and $\hat{\beta}$ are the predicted coordinates ($\hat{x}$, $\hat{y}$). Z is a heatmap, X is a matrix with a set of integers from 0 to the width-1 of Z, and Y is a matrix with a set of integers from 0 to the height-1 of Z. σ is the standard deviation of the gaussian distribution.

$$\begin{pmatrix} \hat{\alpha} \\ \hat{\beta} \\ \frac{-\hat{\alpha}^2-\hat{\beta}^2}{2} \end{pmatrix} = \frac{1}{2}(X\ Y\ 1) + (2\sigma^2 \ln(2\pi\sigma^2 Z) - X^2 - Y^2)\,. \tag{1}$$

However, if an outlier not similar to the Gaussian distribution is included, the least square approximation makes a considerable error.

C. Localization using Soft Argmax

To estimate the key point coordinates corresponding to the position expressed in the heatmap, the coordinates of the pixels having the largest brightness value must be found. The argmax may be used for this purpose. Argmax function, expressed as Eq. (2), is a function of finding the location of the maximum value in a 1–d vector X, so it appropriately extracts the location of the highest probability value. Argmax function for a two-dimensional matrix X can be described as Eq. (3) for x-axis coordinates and Eq. (4) for y-axis coordinates.

$$argmax(X) = \{i | x_i \geq x, \forall x \in X\}, X \in R, \tag{2}$$

$$argmax(X)_i = \{i | x_{ij} \geq x, \forall x \in X\}, \tag{3}$$

$$argmax(X)_i = \{j | x_{ij} \geq x, \forall x \in X\}. \tag{4}$$

However, argmax returned only integer values and was not differentiable. Expressing the coordinates of the center of the heatmap only as integers has limitations in increasing the accuracy of coordinate prediction, especially since the height and width of the heatmap have already been scaled down several times. Therefore, we predicted granular locations using the soft argmax function, which can consider not only the maximum but also other values. Unlike the argmax function, the soft argmax function, shown in Eq. (5), is based on a continuous function. Not only does the maximum value affect the output, but also all the other values, including the minimum value, affect the output. Since the heatmap assumed a normal distribution, the soft argmax function was suitable for predicting the center coordinates in the heatmap. Soft argmax function for a two-dimensional matrix X can be represented as Eq. (6) for x-axis coordinates and Eq. (7) for y-axis coordinates.

$$soft - armax(X) = \sum_i \frac{e^{\beta x_i}}{\sum_j e^{\beta x_i}} i, \quad (5)$$

$$soft - armax(X)_i = \sum \frac{e^{\beta x_{ij}}}{\sum e^{\beta x}} i, \quad (6)$$

$$soft - armax(X)_i = \sum \frac{e^{\beta x_{ij}}}{\sum e^{\beta x}} j. \quad (7)$$

IV. EXPERIMENTS

A. Dataset

Images in the top view, such as in Fig. 7, are used as data sets for Y-maze test automation. Of the total images, 40,691 images were used as a train set, 5,155 images as a validation set, and 5,000 images as a test set.

Various data augmentations were applied to the images in the dataset for the regularization and generalization of the models. They were 50%–150% Random Contrast, 50%–150% Random Saturation, 70%–130% Random Brightness, 20%–100% Random JPEG Quality, -45° to 45° Random Rotation, and -5% to 5% Random Translation.

Truth data used for loss calculation was composed of a gaussian distribution. Considering that the network output was sigmoid-activated, the gaussian distribution for heatmap can be expressed as a function within the range [0,1], shown in Eq. (8). In Eq. (8), σ is 2.0, N is a matrix that listed integer sets from 0 to heatmap width as in Fig. 8(a), and M is a matrix that listed integer sets from 0 to heatmap height as shown in Fig. 8(b) width. In addition, x and y are key point coordinates.

$$f(x,y) = \exp\left(-\frac{(N-x)^2+(M-y)^2}{2\sigma^2}\right). \quad (8)$$

Fig. 7. An example image of the dataset.

$N =$

0	1	2	3	4	5	6	...
0	1	2	3	4	5	6	...
0	1	2	3	4	5	6	...
0	1	2	3	4	5	6	...
0	1	2	3	4	5	6	...
0	1	2	3	4	5	6	...
0	1	2	3	4	5	6	...
...	...	...	...	...	...	...	...

(a)

$M =$

0	0	0	0	0	0	0	...
1	1	1	1	1	1	1	...
2	2	2	2	2	2	2	...
3	3	3	3	3	3	3	...
4	4	4	4	4	4	4	...
5	5	5	5	5	5	5	...
6	6	6	6	6	6	6	...
...	...	...	...	...	...	...	...

(b)

Fig. 8. Heatmap index matrixes for (a) heatmap width and (b) heatmap height.

B. Loss and Metric

For heatmap regression, either binary cross entropy loss, as represented by Eq. (9), or Kullback-Leibler (KL)-divergence loss, as represented by Eq. (10), as a loss function, were employed. In this experimental context, clear contrast was crucial for accurate position representation. While predicting true positives is important, predicting true negatives was equally significant. Therefore, we chose binary cross-entropy loss, which considered both scenarios, over KL-divergence loss. In Eqs. (9) and (10), n is a batch size and m is a size of heatmap.

$$Loss = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{m}\sum_{j=1}^{m}(Y_{ij}\cdot\log\hat{Y}_{ij} - (1-Y_{ij})\cdot\log(1-\hat{Y}_{ij})), \quad (9)$$

$$Loss = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{m}\sum_{j=1}^{m}(Y_{ij}\cdot\log\frac{Y_{ij}}{\hat{Y}_{ij}}). \quad (10)$$

We may use as a metric a function that uses the coordinate error of each key point. However, when the scale of the image is different, as shown in Fig. 9, the

absolute error in the coordinates is meaningless because the zoomed-in image inevitably has a smaller absolute error.

Normalized mean error (NME) is a scale independent metric that avoids this problem by including a scale factor d representing scale like Eq. (11). As shown in Fig. 10(a), the metric of face landmark detection, i.e., NME$_{interocular}$, is defined by setting the pupil distance as a scale factor. Since the arm's length is clearly defined in the Y-maze test, NME$_{Arm}$ was defined with a scale factor d set to the arm's length, as shown in Fig. 10(b). In Eq. (11), k is a number of key points.

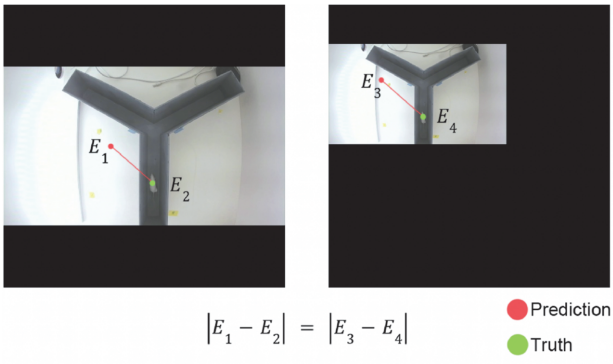

Fig. 9. Different scaled images.

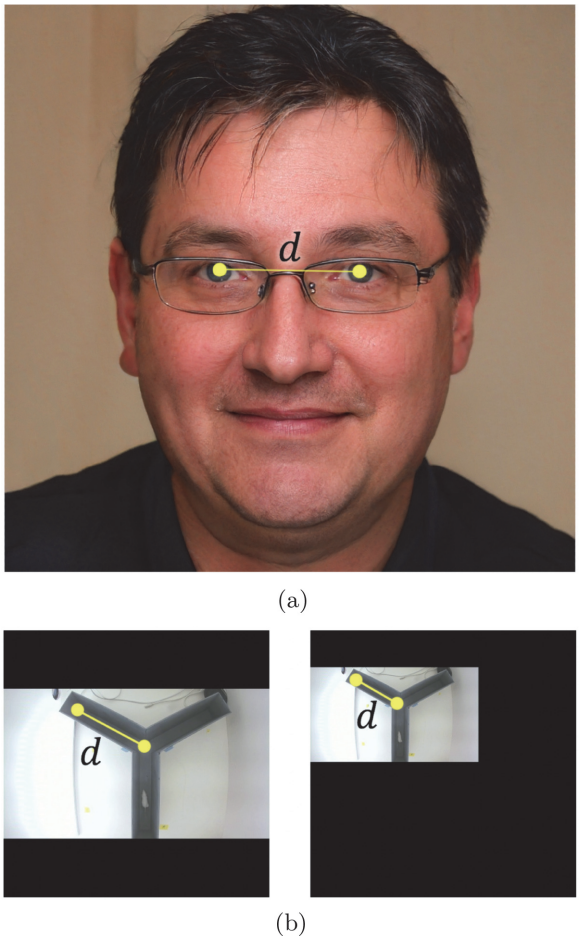

Fig. 10. Standard d between two tasks: (a) interocular *d* of face landmark detection (https://thispersondoesnotexist.com) and (b) length *d* of the arm of Y-maze.

$$\text{NME}\ (\%) = \frac{100}{k}\sum_{i=1}^{k}\frac{\|K_i - \hat{K}_i\|_2}{d}. \quad (11)$$

C. Results

Table 1 shows the results of experiments comparing various backbones. To maintain the same input resolution conditions, we set the input size to 224×224. However, in the case of Hourglass Net, the input size was set to 256×256 to prevent the problem of not recovering to the original size during up sampling. In addition, since each backbone has a different structure, the output size appeared differently.

The backbone neural network with the highest performance was EfficientNet v2, achieving 0.988% NME. The backbone, which showed the high performance with the least memory and computation, was MobileNet v2 [28], reaching 1.473% NME with only the 2.23 million parameter and 0.577 GFLOPs. Configuring EfficientNet v2 as the backbone was best when recognition performance is important. But it is better to use MobileNet v2 when memory and computational optimization were important.

Table 2 shows the results of experiments to find out the effects of various options for ResNet-50. The neural network options for ResNet-50 were different depths, widths, input resolution scaling, large kernel size application and Dropblock [32], which are denoted as Depth, Width, Resolution, Large Kernel and Dropblock, respectively, in Table 2. ResNet-50 was used as the reference model to observe the effect of only various options since it is used frequently as a basic model. Doubled input resolution performed best at 0.986% NME, but the computation volume increased four times to 28.922 GFLOPs. The experiment's highest performance with the smallest computations and number of parameters was maintained when a large kernel with depth wise separable convolution was applied, recording 2.037% NME with 13.413 million parameters and 4.919 GFLOPs. The reason may come from MobileNet [33, 27] which used the depth-wise separable convolution, where the number of parameters and computations decreased even though the kernel size increased. When the input resolution was 224, increasing the depth, as shown in the "Depth +102" option, showed a significant performance drop, but double input resolution, as shown in the "Resolution ×2 and Depth +102" option, made the drop very small [34]. Table 2 shows that the "Depth +102" option causes the performance to drop from 1.147 NME to 1.888 NME. However, the option "Resolution ×2" leads to the improved NME 0.986 and the option "Resolution ×2 and Depth +102" [34] to NME

Table 1. Results of the experiments comparing the performance of backbones

Backbone model	Input size	Output size (heatmap size)	Number of parameters ($\times 10^6$)	Computation (GFLOPs)	NME (%)
VGG-19	224×224	14×14	19.128	36.368	1.453
ResNet-50	224×224	7×7	22.618	7.230	1.147
Inception v3	224×224	5×5	20.916	5.308	1.098
MobileNet v2	224×224	7×7	2.230	0.577	1.473
EfficientNet v2, B0	224×224	7×7	5.722	1.362	0.988
Swin v2, tiny Window8	224×224	7×7	26.347	9.156	3.276
U-Net	224×224	14×14	9.028	22.529	1.074
HourglassNet	256×256	64×64	30.441	63.438	1.438

Table 2. Results of the experiments to analyze the effect of options

Option	Model	Input size	Number of parameters ($\times 10^6$)	Computation (GFLOPs)	NME (%)
Default	ResNet-50	224×224	22.618	7.230	1.147
Depth +102	ResNet-152, 102 more layers	224×224	55.790	21.099	1.888
Width ×2	Double channels	224×224	89.954	28.385	1.356
Large kernel (21×21)	21×21 kernels and separable convolution	224×224	13.413	4.919	2.037
Dropblock (20%, 5×5)	Added dropblock at several layers	224×224	22.618	7.305	3.540
Resolution ×2	Double resolution	224×224	22.618	28.922	0.986
Resolution ×2 and Depth +102	Double resolution and 102 more layers	448×448	55.790	84.395	1.066
Resolution ×2 and Width ×2	Double resolution and channels	448×448	89.954	113.542	1.040
Resolution ×2 and Large kernel	Double resolution and 21×21 kernels	448×448	22.593	28.316	1.053
EfficientNet and resolution ×2	Double resolution and with EfficientNet	448×448	5.722	5.444	0.681

1.066, the less improved performance. We could observe a similar phenomenon for the option "Width ×2" which incurs the performance drop from 1.147 NME to 1.356 NME. However, the option "Resolution ×2 and Width ×2" led to a less improved performance of 1.040 NME than the option "Resolution ×2" with 0.986 NME. A similar phenomenon happens with the option "Large Kernel (21×21)" [35]. Dropblock did not contribute to the performance improvement. In the ResNet-50 network option experiment, only the doubled input resolution option leads to better performance.

Combining the EfficientNet v2 B0, which showed the best result on the backbone experiment, and doubled input resolution, which showed the best result on the option experiment, achieved 0.681% NME. It is the best of all experiments. As a result, 0.681% NME indicated that, given d is 35 cm, the error for each key point is approximately 35 (cm) × 0.681 (%) × 1/100 = 0.23835 (cm). With this level of accuracy, the proposed model, which is EfficientNet and double resolution, is deemed sufficient to replace the observer.

V. CONCLUSION

We have experimented on various deep neural networks for key point localization to determine the location of mice, which is essential for Y-Maze behavior test automation, and confirmed that there are several networks with NME close to 1.0. In particular, it was observed that using EfficientNet as a backbone achieved the best NME. In addition, experiments were conducted to verify the effectiveness of width, depth, input resolution scaling, Dropblock, and large kernel, which can lead to performance improvement based on ResNet, which shows relatively good performance with little structural change. As a result, input resolution scaling is the most effective. Applying doubled input resolution to EfficientNet leads to the best results in both experiments, achieving 0.681 NME.

We will extend our approach to multiple Y-maze tests by choosing and cropping each area of the Y-maze. In addition to mouse localization, a Y-maze behavior analysis program must be developed for specific tests.

ACKNOWLEDGMENTS

This work was supported by research fund of Chungnam National University.

Conflict of Interest(COI)

The authors have declared that no competing interests exist.

REFERENCES

1. D. M. Bannerman, B. Niewoehner, L. Lyon, C. Romberg, W. B. Schmitt, A. Taylor, et al., "NMDA receptor subunit NR2A is required for rapidly acquired spatial working memory but not incremental spatial reference memory," *Journal of Neuroscience*, vol. 28, no. 14, pp. 3623-3630, 2008. https://doi.org/10.1523/JNEUROSCI.3639-07.2008
2. Stanford School of Medicine, "Y Maze Spontaneous Alternation Test," 2023 [Online]. Available: https://med.stanford.edu/sbfnl/services/bm/lm/y-maze.html.
3. Z. Wang, K. Murnane, and M. Ghovanloo, "An automated tracking system for Y-maze behavioral test using Kinect depth imaging," in *Proceedings of 2017 IEEE Biomedical Circuits and Systems Conference (BioCAS)*, Turin, Italy, 2017, pp. 1-4. https://doi.org/10.1109/BIOCAS.2017.8325222
4. F. J. Heredia-Lopez, F. J. Alvarez-Cervera, J. G. Colli-Alfaro, J. L. Bata-Garcia, G. Arankowsky-Sandoval, and J. L. Gongora-Alfaro, "An automated Y-maze based on a reduced instruction set computer (RISC) microcontroller for the assessment of continuous spontaneous alternation in rats," *Behavior Research Methods*, vol. 48, pp. 1631-1643, 2016. https://doi.org/10.3758/s13428-015-0674-0
5. J. Redmon and A. Farhadi, "Yolov3: An incremental improvement," 2018 [Online]. Available: https://arxiv.org/abs/1804.02767.
6. A. Bochkovskiy, C. Y. Wang, and H. Y. M. Liao, "Yolov4: optimal speed and accuracy of object detection," 2020 [Online]. Available: https://arxiv.org/abs/2004.10934.
7. C. Y. Wang, A. Bochkovskiy, and H. Y. M. Liao, "YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Vancouver, Canada, 2023, pp. 7464-7475.
8. S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: towards real-time object detection with region proposal networks," 2014 [Online]. Available: https://arxiv.org/abs/1506.01497v1.
9. W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C. Y. Fu, and A. C. Berg, "SSD: single shot multibox detector," 2015 [Online]. Available: https://arxiv.org/abs/1512.02325.
10. M. Tang, R. Pang, and Q. V. Le, "EfficientDet: scalable and efficient object detection," 2019 [Online]. Available: https://arxiv.org/abs/1911.09070.
11. A. Bulat, E. Sanchez, and G. Tzimiropoulos, "Subpixel heatmap regression for facial landmark localization," 2021 [Online]. Available: https://arxiv.org/abs/2111.02360v1.
12. B. Artacho and A. Savakis, "UniPose: unified human pose estimation in single images and videos," 2020 [Online]. Available: https://arxiv.org/abs/2001.08095.
13. K. Sun, B. Xiao, D. Liu, and J. Wang, "Deep high-resolution representation learning for human pose estimation," 2019 [Online]. Available: https://arxiv.org/abs/1902.09212.
14. Z. Liu, W. Lin, X. Li, Q. Rao, T. Jiang, M. Han, et al., "ADNet: attention-guided deformable convolutional network for high dynamic range imaging," 2021 [Online]. Available: https://arxiv.org/abs/2105.10697.
15. K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014 [Online]. Available: https://arxiv.org/abs/1409.1556.
16. K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," 2015 [Online]. Available: https://arxiv.org/abs/1512.03385.
17. K. He, X. Zhang, S. Ren, and J. Sun, "Identity mappings in deep residual networks," 2016 [Online]. Available: https://arxiv.org/abs/1603.05027.
18. C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, et al., "Going deeper with convolutions," 2014 [Online]. Available: https://arxiv.org/abs/1409.4842.
19. C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," 2015 [Online]. Available: https://arxiv.org/abs/1512.00567.
20. M. Tan and Q. V. Le, "EfficientNet: rethinking model scaling for convolutional neural networks," 2019 [Online]. Available: https://arxiv.org/abs/1905.11946v1.
21. M. Tan and Q. V. Le, "EfficientNetv2: smaller models and faster training," 2021 [Online]. Available: https://arxiv.org/abs/2104.00298.
22. Z. Cao, T. Simon, S. E. Wei, and Y. Sheikh, "Realtime multi-person 2D pose estimation using part affinity fields," 2018 [Online]. Available: https://arxiv.org/abs/1812.08008v1.
23. D. Groos, H. Ramampiaro, and E. A. Ihlen, "EfficientPose: scalable single-person pose estimation," *Applied Intelligence*, vol. 51, pp. 2518-2533, 2021. https://doi.org/10.1007/s10489-020-01918-7
24. W. Mao, Y. Ge, C. Shen, Z. Tian, X. Wang, and Z. Wang, "TFPose: direct human pose estimation with transformers," 2021 [Online]. Available: https://arxiv.org/abs/2103.15320.
25. A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, et al., "Attention is all you need," 2017 [Online]. Available: https://arxiv.org/abs/1706.03762v1.
26. A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, et al., "An image is worth 16x16 words: transformers for image recognition at scale," 2020 [Online]. Available: https://arxiv.org/abs/2010.11929v1.
27. M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L. C. Chen, "MobileNetv2: inverted residuals and linear bottlenecks," 2018 [Online]. Available: https://arxiv.org/abs/1801.04381v1.
28. Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo, "Swin transformer: hierarchical vision transformer using shifted windows," 2021 [Online]. Available: https://arxiv.org/abs/2103.14030v1.
29. Z. Liu, H. Hu, Y. Lin, Z. Yao, Z. Xie, Y. Wei, et al., "Swin transformer V2: scaling up capacity and resolution," 2021 [Online]. Available: https://arxiv.org/abs/2111.09883v1.
30. O. Ronneberger, P. Fischer, and T. Brox, "U-Net: convolutional networks for biomedical image segmentation," 2015 [Online]. Available: https://arxiv.org/abs/1505.04597.
31. A. Newell, K. Yang, and J. Deng, "Stacked hourglass networks

for human pose estimation," 2016 [Online]. Available: https://arxiv.org/abs/1603.06937v1.
32. G. Ghiasi, T. Y. Lin, and Q. V. Le, "DropBlock: a regularization method for convolutional networks," 2018 [Online]. Available: https://arxiv.org/abs/1810.12890v1.
33. A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "MobileNets: efficient convolutional neural networks for mobile vision applications," 2017 [Online]. Available: https://arxiv.org/abs/1704.04861.
34. I. Bello, W. Fedus, X. Du, E. D. Cubuk, A. Srinivas, T. Y. Lin, J. Shlens, and B. Zoph, "Revisiting ResNets: improved training and scaling strategies," 2021 [Online]. Available: https://arxiv.org/abs/2103.07579.
35. X. Ding, X. Zhang, J. Han, and G. Ding, "Scaling up your kernels to 31x31: revisiting large kernel design in CNNs," 2022 [Online]. Available: https://arxiv.org/abs/2203.06717v1.

Gwanghee Lee

He has been a Ph.D. student in Computer Science at Chungnam National University since 2021. His research interests include computer vision deep learning, face alignment, pose estimation, object detection, keypoint detection, semantic segmentation, instance segmentation, and face recognition.

Sangjun Moon

He is an undergraduate student majoring in artificial intelligence at Chungnam National University since 2021. His research interest is in computer vision and deep learning model architecture such as image classification, objection detection, and segmentation.

Dasom Choi

She is an undergraduate student in Department of Artificial Intelligence at Chungnam National University since 2021. She is interested in computer vision and deep learning such as image classification, segmentation, and objection detection.

Gayeon Kim

She is an undergraduat student in Artificial Intelligence at Chungnam National University since 2021. Her research interests are in computer vision and deep learning.

Kyoungson Jhang

He received B.S., M.S., and Ph.D. degrees in Department of Computer Engineering from Seoul National University in 1986, 1988, and 1995, respectively. Since September 2001, he has been working as a professor for the Department of Computer Science and Engineering at Chungnam National University, Daejeon, Korea. His research focuses on computer vision and deep learning.