Cognitive Computation
https://doi.org/10.1007/s12559-024-10250-y

Deep Multi-task Learning for Animal Chest Circumference Estimation from Monocular Images

Hongtao Zhang[1] · **Dongbing Gu**[1]

Received: 20 September 2023 / Accepted: 6 January 2024

Abstract
The applications of deep learning algorithms with images to various scenarios have attracted significant research attention. However, application scenarios in animal breeding managements are still limited. In this paper we propose a new deep learning framework to estimate the chest circumference of domestic animals from images. This parameter is a key metric for breeding and monitoring the quality of animal in animal husbandry. We design a set of feature extraction methods based on a multi-task learning framework to address the challenging issues in the main estimation task. The multiple tasks in our proposed framework include object segmentation, keypoint estimation, and depth estimation of cow from monocular images. The domain-specific features extracted from these tasks improve upon our main estimation task. In addition, we also attempt to reduce unnecessary computations during the framework design to reduce the cost of subsequent practical implementation of the developed system. Our proposed framework is tested on our own collected dataset to evaluate its performance.

Keywords Convolutional neural network · Feature fusion · Keypoint detection · Depth estimation

Introduction

In animal breeding management systems, the administrator usually needs to regularly check the animal to obtain the physiological parameters of animal in time, so that the feeding strategy could be optimised accordingly. However, the collection of various physiological parameters adds a lot of complex daily workloads to the administrator. The purpose of this paper is to propose a new deep learning framework with images to simplify the process of animal monitoring and management.

There are many physiological parameters of animal. In this paper, we only focus on the use of a monocular camera as a low-cost device to be applied for this application to estimate the chest circumference of a dead cow lain on the ground. The workflow in our proposed framework is to capture a monocular RGB image of the cow from camera as input, then generate the intermediate results (depth, keypoints, and segmentation), finally output the regression result (chest circumference). The overall framework is illustrated in Fig. 1. The parameter (chest circumference) to be estimated is shown in Fig. 2. The intermediate results from our multi-task learning framework include the bounding box (the first image), the keypoints (the second image), the semantic segmentation mask (the third image), and the depth map (the fourth image) as shown in Fig. 3. This framework can be implemented by using a monocular camera and a GPU computer in real time, which greatly reduces the waste of human resources in the animal breeding management systems.

Traditional algorithms for object detection [1], object segmentation [2] and key point detection [3] are mostly based on 2D images. If we only analyse 2D images, it is difficult to retrieve the distance between two points in an image. Some 3D modeling methods [4] could consume a lot of computational power, making it difficult to implement the algorithm in real time. In order to balance the computational power and implementation demand, we choose to use the monocular depth estimation from RGB image to capture 3D information. We also estimate the keypoints of animal from segmented region. By fusing the depth and keypoint information, we are able to estimate the chest circumference of animal from the multi-task learning framework. Through our testing and evaluation results from the images we captured in real life,

✉ Hongtao Zhang
hz17842@essex.ac.uk

Dongbing Gu
dgu@essex.ac.uk

[1] School of Computer Science and Electronic Engineering, University of Essex, Colchester CO4 3SQ, UK

Published online: 12 February 2024

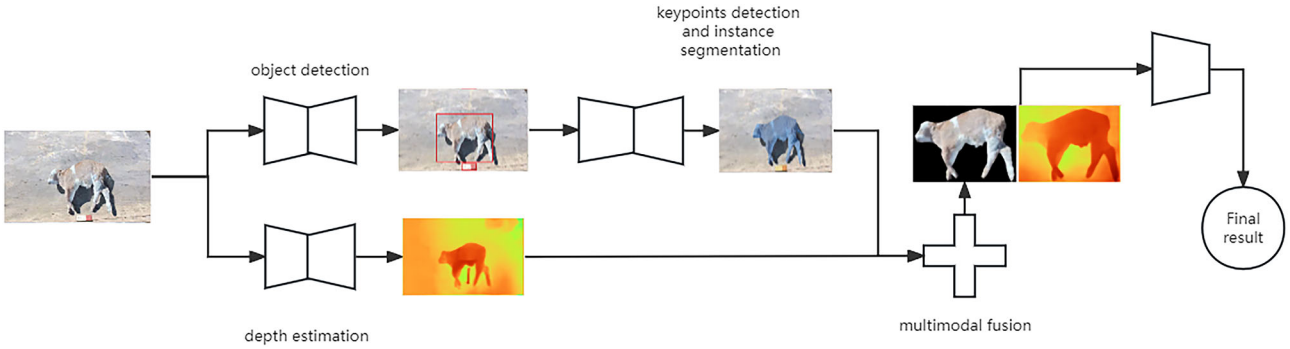
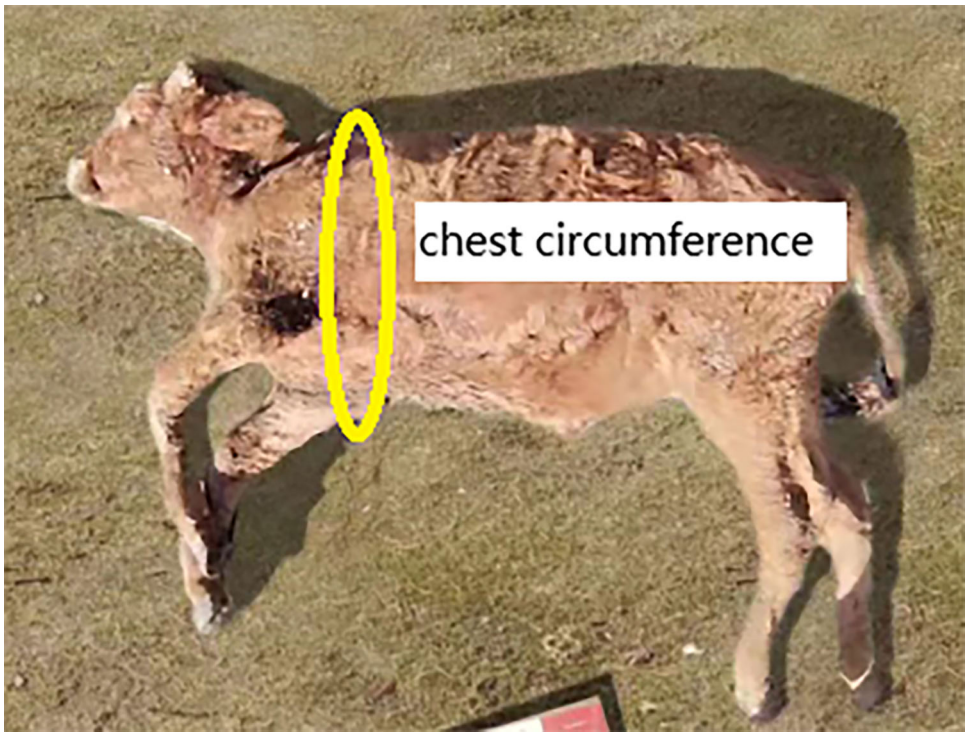


Fig. 1 Overview of our proposed framework, including object detection, object segmentation, keypoint detection, monocular depth estimation and regression network

Fig. 2 The chest circumference of cow



we found our proposed system is effective and performs well in various lightning conditions.

Our main contributions can be summarised below:

- A new deep learning framework for estimating the chest circumference of cow from monocular images is proposed. It is a multi-task learning framework which can extract the keypoints, mask, and depth information of a cow in an image and estimate the parameter of the cow.
- A new alignment method for depth estimation is developed to align the ground plane with the image plane. The alignment method is based on the ground plane estimation and the reference object estimation (the cigarette box in image, see Fig. 3).
- Our object segmentation and key point detection are based on the same network, which is built up on the top of the Deeplab-V3 [5]. They share the same features extracted from input image.

Fig. 3 The intermediate results from our multi-task learning framework, from left to right: bounding box, keypoints, semantic mask, and depth map generated from our framework

- The proposed network framework is tested and evaluated on real images collected in the field. The results demonstrate our framework is effective and efficient.

Related Works

There are some existing estimation methods for object length from images relevant to this work. They are based on object segmentation [2] and keypoint detection [6]. They estimate the length on 2D images, but do not use the depth information. In our work, we use a multi-task learning framework, which combines the information from multiple tasks, including depth estimation, object segmentation, and keypoint estimation, to implement the main parameter estimation task. In the following, we review some existing works on object detection, instance segmentation, key point detection and monocular depth estimation.

Object Detection Object detection is a common task in deep learning. The output of object detection is generally composed of three parts: category label, detection confidence, and object bounding box. They can be roughly divided into two directions: object detection based on anchor frame, and object detection independent of anchor frame. The former proposes anchor boxes or uses traditional computer vision technology to search for potential anchors, matches the proposed anchor with the possible ground truth box, and trains to correct the input proposal box to complete the prediction of the bounding box [7, 8]. The latter [6, 9] is mainly divided into two categories: dense prediction based and key point estimation based [10]. Our object detection network is based on the Yolo network architecture [7] with a reduced number of layers for network efficiency.

Instance Segmentation The instance segmentation [11] task aims to obtain the category information of specified objects in image. They work in end-to-end fashion, or perform the detection first and then segmentation [5]. Their outputs include mask, object label and confidence. They can be used to eliminate the influence of background on feature extraction areas [6]. In recent years, there are also some semi-supervised [12] and unsupervised [13] based results. The current mainstream object segmentation models can be roughly divided into three categories:

- End to end: it directly outputs the segmentation result. By using a regression network and the ground truth information, a segmentation network is directly trained.
- Bottom up: The idea is to perform the semantic segmentation at pixel level first, and then different instances are distinguished by clustering, metric learning or other means.
- Top down: The idea is to estimate the bounding box of the instance first, and then perform the semantic segmentation inside the box. The representative is the well-known Mask-RCNN [2].

Keypoint Detection In recent years, keypoint detection has been widely used in human pose estimation [3, 14] and face recognition tasks [15]. It outputs a sequence of 2D keypoints. They are also divided into two ideas: top down and bottom up. The former is to detect the object first, and then carry out the keypoint regression [16]. On the contrary, the latter is to regress multiple groups of keypoints first, and then group them together [3]. At present, most keypoint detection tasks are based on heatmap [6], and the improvement is observed by optimising the information loss in [14]. Our network uses the same feature extraction layers for both segmentation and keypoint selection, which can share the representation and help each other.

Monocular Depth Estimation At present, most monocular depth estimation methods are based on the Structure from Motion (SfM) [17] framework, which relates the depth estimation with camera pose estimation together. In many cases [18–20], monocular depth estimations are trained on the Kitti dataset [21] and CityScape dataset [22]. The scale problem is hard to be solved. The accuracy improvement in dynamic scenes has been the research focus for a period of time [23, 24]. Our depth is based on the pre-trained network Midas [18] and an alignment method, which is able to produce an aligned estimated depth map.

Multi-task Learning Recently, multi-task joint learning has been increasingly applied in more scenarios, such as optimising network structures and incorporating multi-modal information to optimise the results of object pose estimation in space [25], using multi-task joint learning for text correction in the field of text recognition [26], and applying multi-task joint learning in the field of genetics [27]. Inspired by these applications, our proposed framework is a multi-task joint learning paradigm.

Methods

In this section, we will describe our algorithm framework in detail. We use a monocular RGB image containing a dead cow lain on the ground as input. Our final goal is to estimate the chest circumference of the cow in the image. We employ a multi-task learning framework, which includes the use of results from multiple estimation tasks, such as depth map, keypoints, and object mask of the cow, to regress the final parameter. The framework is designed to facilitate the practical application, and focus on the design of light-weighted network models.

Figure 1 shows the overview of our proposed framework. The input is a monocular image. It uses an object detection network to detect the object and produce a bounding box from the input image. Then it uses a network to perform the keypoint detection and instance segmentation. At the same time, it uses a monocular depth network to estimate the depth map from the input image. The keypoints, object mask, and depth map are then used as input to the final regression network. The regression network outputs the chest circumference parameter.

Object Detection

Our object detection network is to estimate the bounding box of a cow from input image. It can effectively remove the background, and also filter out some images that do not meet the requirements. In the subsequent process, the cow to be detected needs to appear completely and clearly in the image. By filtering out poor images, the system only needs to detect the images that meet the requirements.

Our object detection network is based on the Yolo network architecture [7]. As shown in Fig. 4, we have modified the multi-scale output of Feature Pyramid Network (FPN) [28] by reducing the number of high-resolution layers in FPN (please see the deleted section). This effectively reduces the number of network parameters, ensures the effective output of large target results, and reduces the impact of incomplete small objects on detection results.

Keypoint Detection and Object Segmentation

The keypoints we select are distributed around the edge of the segmented object (see the second image in Fig. 3). As they are located around the outline of cow body, they provide important information for the parameter to be estimated. The keypoint detection result is to ensure that the cow in the image has an unified position and posture for next step processing.

The keypoint selection provides an opportunity to share the feature extraction parts of the network for the keypoint detection and object segmentation tasks as both tasks share similar features. This selection also makes the features of the two tasks interrelated, so that the two tasks can optimise each other in the feature extraction process. This could potentially improve the performance for both tasks, and also simplifies the network complexity.

Our object segmentation and keypoint detection network is shown in Fig. 5. It is built up on the top of the Deeplab-V3 [5]. We inherit the feature extraction part of Deeplab-V3 [5]. After the last convolution of feature extraction is completed in the network, the convolution result is divided into two branches. One branch produces the object segmentation result after performing the deconvolution operations in the decoder part. The other produces the keypoint detection result after performing a linear operation to stretch the feature extraction results into a one-dimensional tensor and passing a fully connected network.

Our object segmentation task has the following features:

- Single object: the object we need to segment is a single cow object and a reference object (cigarette box). The reference object is used for comparing the distances between keypoints with the length of the reference object (cigarette box) in the image. This is important for dealing with the scaling problem. See the regression subsection below for more details.
- The proportion of the object in the image is large: in general, the proportion of cow in the image should be more than half in the application scenario.

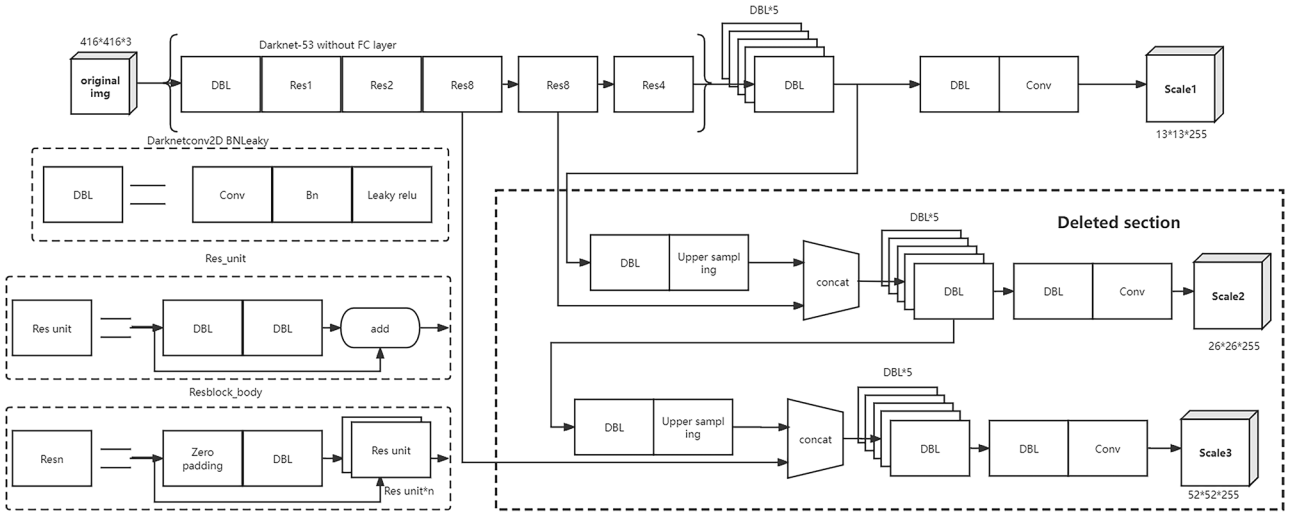


Fig. 4 The object detection network structure

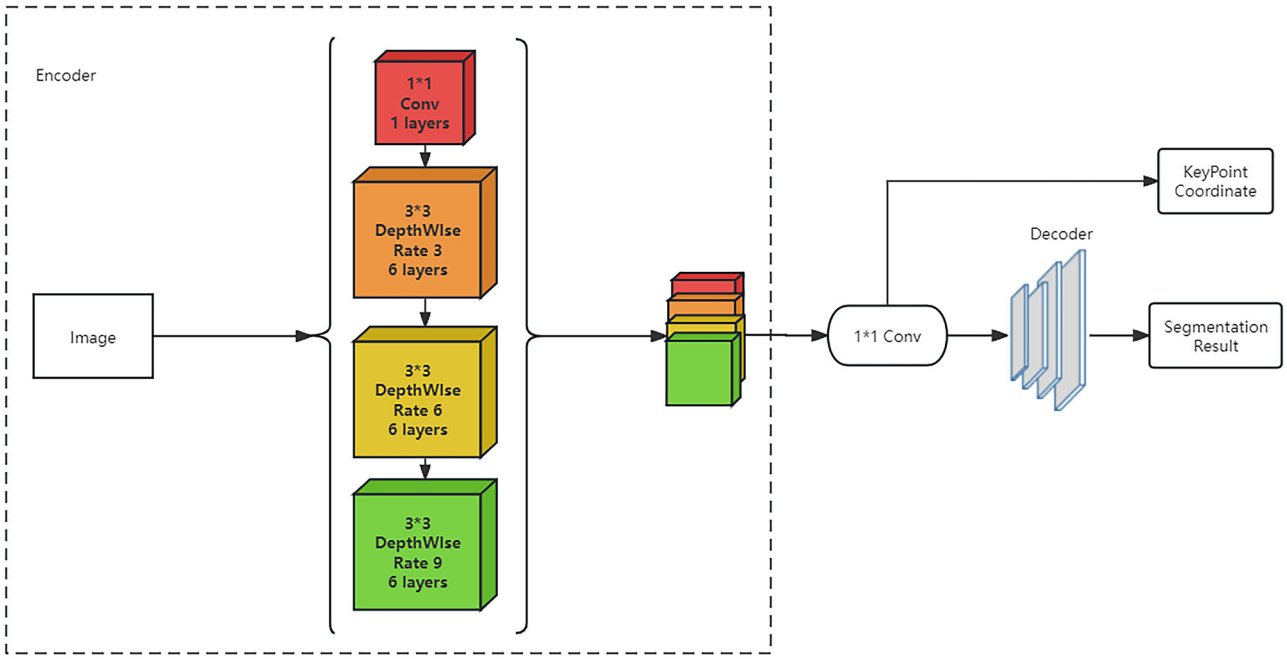


Fig. 5 The object segmentation and keypoint detection network structure. There are 19 layers in total in the feature extraction part

We choose an end-to-end method for the tasks mentioned above. This can also reduce the computational power required in the process.

Aligned Depth and Scale Estimation

Our depth estimation network uses the pre-trained network Midas [18] to estimate the depth for each pixel of input image. The depth from this network is the estimated distance $\hat{d}_i$ between a point i of the cow and the image plane. There are two problems to be solved before it can be used to estimate the chest circumference:

1. When the angle between the cigarette box plane and the image plane is small, the scales in the x and y directions are basically the same ($s_x = s_y$). But the scale in the z direction (s_z) is different with the x scale. We have to estimate them separately. When the angle between the cigarette box plane and the image plane is not small, we need to calculate each scale individually.
2. The ground plane is not parallel to the image plane. We have to estimate the ground plane and align the ground plane with the image plane. A new aligned depth estimate is required for the main parameter estimation.

These two problems are solved by using the reference object (cigarette box) in image. Given the known lengths of the cigarette box, we can retrieve the scales. By detecting the corner 3D coordinates of the cigarette box in image, we can retrieve the ground plane.

To solve problem 1, we use the results of object segmentation and depth estimation. We can obtain the estimated 3D coordinates of four corners of the cigarette box in image. When the scales in x and y directions are very different, we can assume the scales satisfy the linear relationship $s_x = js_y = ks_z$. We can obtain s_x from the estimated coordinates of four corners and the ground truth lengths of the cigarette box. Further, we can obtain k and j from the ratio of ground truth lengths of long and short sides of the cigarette box.

To solve problem 2, we assume that some areas in the image are the ground. We need to estimate the ground plane. The ground plane equation is $ax + by + cz + d = 0$. By detecting sufficient points in the image that are the ground and using the least squares method, we can obtain the equation parameters a, b, c, d. The normal vector of the ground $\vec{n_g}$ is (a, b, c).

From the estimated 3D coordinates of four corners of the cigarette box in image, we can obtain the direction vector l of any two points (x_1, y_1, z_1),(x_2, y_2, z_2). The angle between the straight line and the image plane is:

$$\cos\theta = \frac{\vec{n_i} \cdot \vec{l}}{|\vec{n_i}| \left|\vec{l}\right|} \tag{1}$$

where $\vec{n_i}$ is the normal vector of the image plane $(0, 0, 1)$, and the straight line is the segmentation result of the cigarette box in the ground plane.

Given the ground truth length L between two selected corners of the cigarette box, the depth difference between

the two points is $d_{diff} = L \sin\theta$. In this way we can obtain the depth differences between any two points of the cigarette box. Then we can compute a set of depths d_i for selected points in the cigarette box using d_{diff}. Their corresponding estimated depths from the network Midas are $\hat{d}_i$, $i = 1, ..., n$. Using a set of pairs $d_i, \hat{d}_i$, a least square method is used to compute the parameters D_{scale} and D_{shift}:

$$D_{scale} = \frac{\sum_{i=0}^{n} d_i \hat{d}_i - \frac{\sum_{i=0}^{n} d_i \sum_{i=0}^{n} \hat{d}_i}{n}}{\sum_{i=0}^{n} d_i^2 - \frac{(\sum_{i=0}^{n} d_i)^2}{n}} \tag{2}$$

$$D_{shift} = \frac{\sum_{i=0}^{n} \hat{d}_i - D_{scale} \sum_{i=0}^{n} d_i}{n} \tag{3}$$

Finally the aligned estimated depth $\bar{d}_i$ is:

$$\bar{d}_i = \hat{d}_i D_{scale} + D_{shift} \tag{4}$$

Regression Network

Our multi-task learning framework generates the object mask, depth map and keypoints, which are fed into the regression network to estimate the parameter: chest circumference. Before they are fed to the regression network, they have to be normalised to have a unified size. As we need to preserve the aspect ratio of the object, we use the padding method to normalise the size.

As the distances between each image and the camera are different, we have to use the reference object (cigarette box) with known lengths to find the scale for our regression task. We choose the ratio of diagonal lengths between the bounding box and the reference box as the scale to calculate the parameter.

The loss function for our regression task is designed by using a Mean Absolute Percentage Error (MAPE).

$$loss = \tan \frac{1}{n} \sum_{i=1}^{n} \left| \frac{\hat{y}_i - y_i}{y_i} \right| \tag{5}$$

where n is the number of images, y_i is the ground truth, and $\hat{y}_i$ is the estimated parameter for image i. The loss value is in the range of $(0 < \frac{1}{n} \sum_{i=1}^{n} \left| \frac{\hat{y}_i - y_i}{y_i} \right| < \frac{\pi}{2})$.

The regression network is shown in Fig. 6. We choose the A0 version of RepVgg [29], and use 3×3 convolution cores in the regression network. This design greatly improves the optimisation efficiency of CUDA for training speed. At the same time, the design of our residual blocks is similar to residual networks, which avoids the problems of gradient disappearance and gradient explosion in the training process. We add an attention network CBAM [30] in the input and output parts of the network, which enables the regression network to focus more on the effective region.

Experiment Results

In this section, we will present the details of our experiments and results. We use the GeForce GTX1080Ti graphic card to train and test various parts of the framework. All of them are implemented and tested on PyTorch.

Dataset

Our dataset is collected from real scenes where all the images contain a cow and a cigarette box lain on the soil and cement ground. The camera used is a short-focus camera with a focal length of 3 mm. More than 200 cows are used for image collection, each of them is recorded with a video clip. From all the video clips, we obtain 3000 training images and 300 test images. We manually label the training set and the test set with the ground truth segmentation mask and keypoint coordinates. And the bounding box is generated using the segmentation results we labeled, without the need for additional annotations. The deep estimation network is not trained in our framework and there is no need for labeling.

Evaluation Metrics

Our estimated parameter $\hat{y}_i$ is the chest circumference of cow in image i. The corresponding ground truth is y_i. The average error represents the average percentage of error between the estimated value $\hat{y}_i$ and the true value y_i of all test samples.

$$AE = \sum_{i=1}^{n} \frac{\hat{y}_i}{y_i} \tag{6}$$

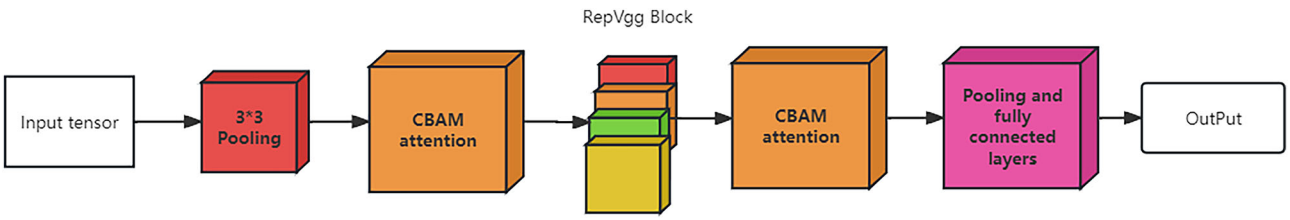


Fig. 6 The structure of our regression network

R2 score shows how well our regression model predicts the chest circumference of cow from observed images. It reflects the proportion of all variations of dependent variable y_i that can be explained through the regression model. It is computed as below

$$\bar{y} = \frac{1}{n}\sum_{i=1}^{n}\hat{y}_i \tag{7}$$

$$R^2 = \frac{SS_{reg}}{SS_{tot}} = \frac{\sum_i \left(\hat{y}_i - y_i\right)^2}{\sum_i \left(\hat{y}_i - \bar{y}\right)^2} \tag{8}$$

If the R2 score (R^2) is close to 0, the model cannot estimate the parameter at all. If it is 1, the model prediction is perfect.

Training

Our keypoint detection and object segmentation network is developed based on the Deeplab-V3 [5]. The input of the segmentation and keypoint network is a colour image with a side length of 300 and 3 channels. The output tensor shape of the target segmentation part is (1,300,300), and the tensor shape of the keypoint detection part is (4,2). The height and width of the input tensor of the regression network for the final output (chest circumference) are 224 pixels. The tensor shape of the final output of the network is 1, and the result of the network output will be multiplied by the scale obtained during the preprocessing of the segmented network to obtain the final output in centimetres. During the training process, we used AdamW [31] as the optimiser. In order to perform the effective data augmentation while preserving the effective information in the image, we only used three data augmentation methods: random rotation of no more than 45 degrees left and right, random flipping of image up, down, left and right, and adding a small amount of noise during the data augmentation process. We train this network by using our training dataset with labels.

Table 1 displays the testing results for the segmentation network. The object size in the table represents different object sizes in images. Due to the fact that the objects we detected are usually large, we only count the objects with a length and width greater than 32 pixels. When the detection object is between 32 and 96 pixels in length and width, the average precision and recall rate can reach 80%. When the detection object is over 96 pixels in length and width, the average precision can reach 87.8%, and the average recall rate can reach 90.6%.

At the same time, we separately compute the MIOU (Mean Intersection over Union) of the segmentation network for different objects. In order to verify the performance of the segmentation network in the boundary part, we include the testing on the Boundary IOU [32] when testing the segmentation network. Table 2 shows the testing results. It can be seen that all the values are above 94%, which indicates the network performs well in our dataset.

Table 1 Segmentation results for different object sizes

Indicator Name	IoU range	Target size	Value
Average Precision	0.50:0.95	all	0.878
Average Precision	0.50	all	1
Average Precision	0.75	all	1
Average Precision	0.50:0.95	32:96	0.800
Average Precision	0.50:0.95	96	0.878
Average Recall	0.50:0.95	all	0.906
Average Recall	0.50:0.95	32:96	0.800
Average Recall	0.50:0.95	96	0.906

In terms of keypoint detection, our average OKS (Object Keypoint Similarity) is 0.97.

Our monocular depth estimation network is developed based on the MidasV2 [18]. We do not need to train the network, only apply the alignment scale to the output of monocular depth estimation.

After training the above two networks individually, the regression network shown in Fig. 1 is trained on our dataset with the loss function in (5). The training process consists of 50 epochs. During the training process, we select AdamW [31] as the optimiser to achieve rapid convergence of model parameters.

Main Results

The main results are shown in Table 3. It shows the testing results for different network structures used in the regression network (see Fig. 6), including Res18, Res50, Res101, Ghost-ResNet-56 [33], FasterNet-T0 [34], RepVggB2 [29], RepVggB3 [29], and RepVggB2 +CBAM. The second and third columns show the network parameters and flops for different regression network structures.

The fourth column shows the percentage of the test samples whose estimation error is less than 10%. It can be seen that when RepVggB3 [29] is used, there are 47.72% of the test samples with an estimation error of less than 10%.

The fifth column of the table is the average error which represents the average percentage of error between the estimated value and the true value of all test samples. When using RepVggB2+CBAM as the regression backbone network, we obtain a minimum average error of 16.28%, which is better than the RepVggB2 without the attention module CBAM.

The last column is the R2 score, in which Res18 performs the best with a value of 0.9256.

Table 2 Segmentation results for different objects

Indicator Name	Target	Value
MIoU	Cow	0.984471
MIoU	Cigarette	0.978546
MIoU	Macro	0.981509
MIoU	Micro	0.985037
Boundary IoU	Cow(15 Pixels)	0.946831
Boundary IoU	Cigarette(5 Pixels)	0.953415
Boundary IoU	Macro	0.950123
Boundary IoU	Micro	0.949210

A number of samples are shown with their heatmaps (see Fig. 7). We can see that our algorithm has successfully noticed the effective regions in images, such as boundary regions. This demonstrates our network model is effective for the estimation task.

Ablation Experiments

To demonstrate the performance of our algorithm, we conduct a series of ablation experiments. We conduct ablation experiments with different regression network structures and different input channels.

Figure 8 shows the loss values during the training for the network structures: Res18, Res50, Res101, RepVggB2, and RepVggB3. During the 40 training epochs, we find the RepVggB2, RepVggB3, and RepVggB2+CBAM perform better than Res18, 50 and 101 in terms of convergent rate and loss value. This is mainly because the RepVgg block has more branches compared to the residual block in ResNet.

Figure 9 shows the loss values during the training for different inputs: 1 channel, 2 channels, and 4 channels. We find that the training performance for 4 channels performs better than other cases. When the input is 1 channel, we only use the depth map as the input. When the input is 2 channels, we use the depth map and mask as input. When the input is 4 channels, we use the depth map and the original RGB image as input.

We also test the impact of the depth estimation network on the estimation results. Figure 10 shows the estimated chest circumferences with and without the depth estimating network. The abscissa is the ground truth chest circumference. The vertical axis is the error between the estimated value and the true value. By comparing the results in these two figures, we find that the addition of our monocular depth estimation network can significantly improve the accuracy of the system.

Table 3 Evaluation results on own dataset

Structure	Parameters	Flops (GFlops)	Sample percentage (< 10%)	Average Error (AE)	R2 score
Res18	11689512	1.82	45.17%	18.65%	0.9256
Res50	25557032	4.12	43.54%	19.33%	0.9034
Res101	44549160	7.84	39.54%	21.50%	0.7956
Ghost-ResNet-56 [33]	4342924	0.63	32.32%	20.28%	0.7624
FasterNet-T0 [34]	3942924	0.34	30.43%	20.14%	0.7524
RepVggB2 [29]	89023016	20.47	43.35%	16.66%	0.8234
RepVggB3 [29]	123085928	29.18	47.72%	18.30%	0.8527
RepVggB2+CBAM	89842924	20.47	44.32%	16.28%	0.8934

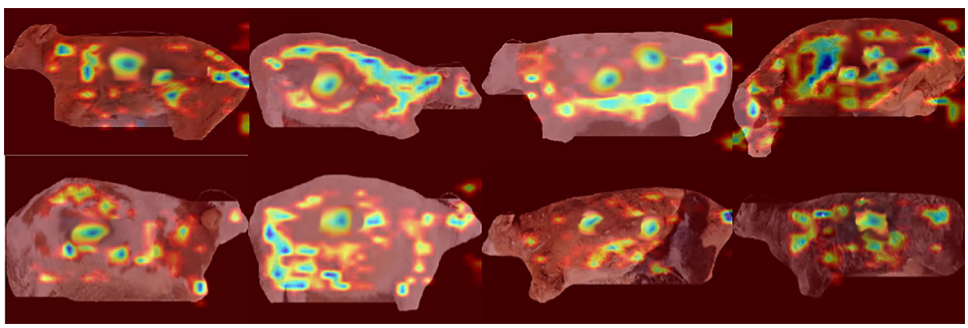


Fig. 7 Heatmaps for our testing results

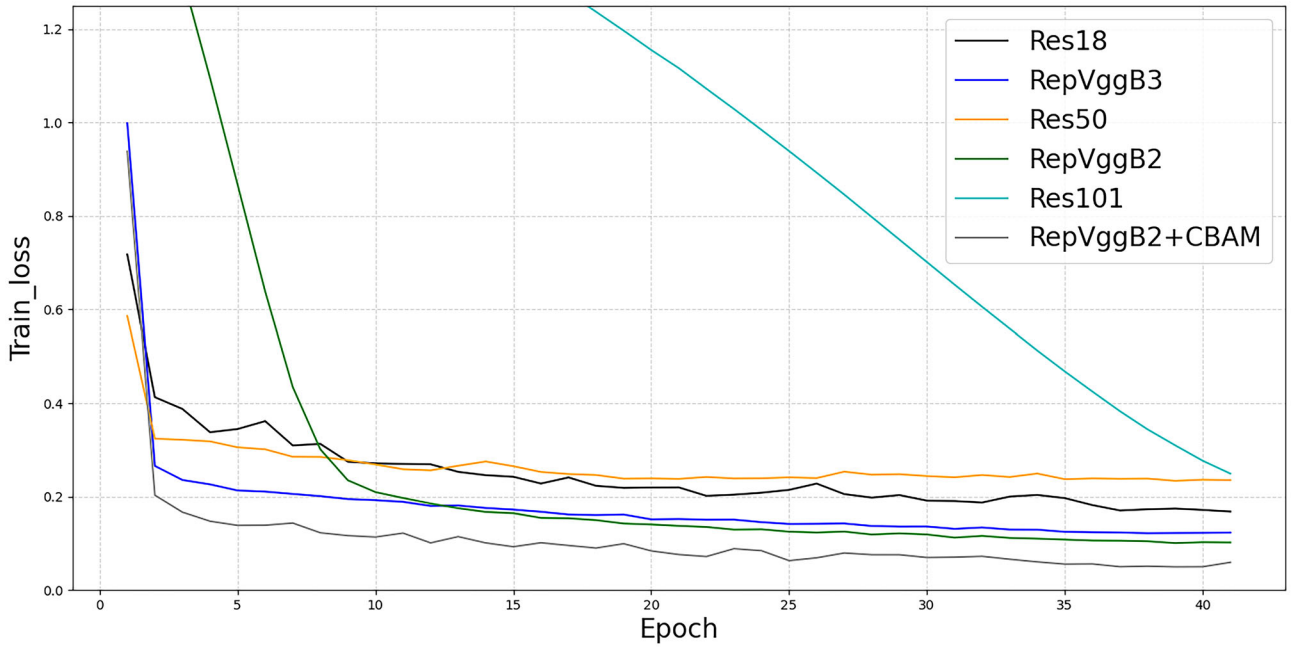


Fig. 8 The training performance of different network structures

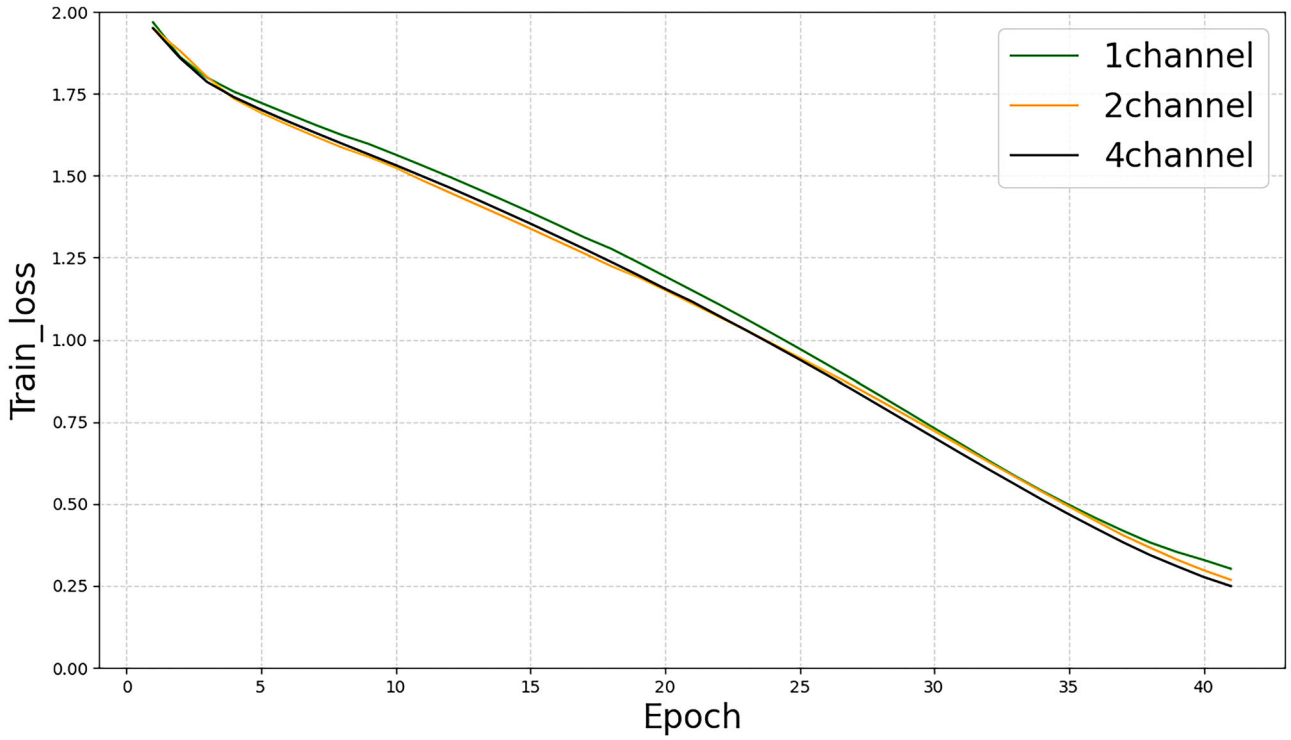


Fig. 9 The training performance of different input channels: 1 channel for depth map input, 2 channels for depth map and mask inputs, and 4 channels for depth map and the original RGB image inputs

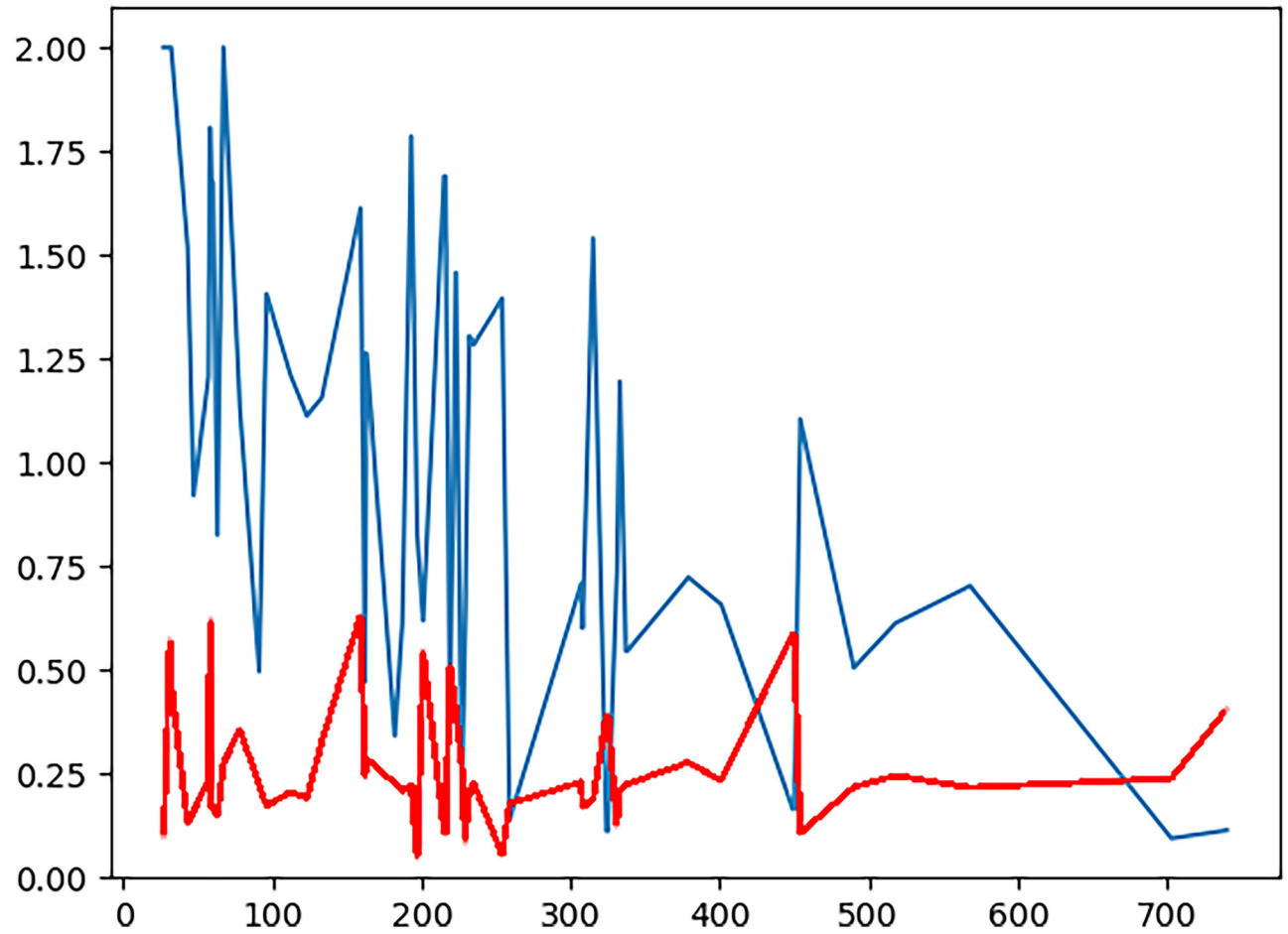


Fig. 10 The blue one is the estimation results without the depth network and the red one is the estimation results with the depth network. The abscissa is the ground truth chest circumference. The vertical axis is the error between the estimated value and the true value

Conclusion

In this work, we proposed a deep learning framework to estimate the chest circumference of cow from monocular images. This framework is based on the multi-task learning scheme, which incorporates the networks for monocular depth estimation, keypoint detection, object segmentation, and object detection. By fusing the results from multiple tasks, we can regress the chest circumference of cow from monocular images. We also made the contributions to the depth estimation network, and keypoint detection and object segmentation network in order for them to be used in our framework. We collected our own dataset for training and testing our models. The evaluation results show our framework is effective and provides a practical solution to the parameter estimation task.

In the future, we would like to extend our framework to include stereo camera images, which could provide more accurate information for the depth estimation. We would like to estimate other parameters, such as the body length or age of animals to enhance the rapid modeling and digitization of animal. The generalisation of the proposed network to other animals, such as pigs, is an essential work in the next step.

Data Availability Due to the nature of the research and commercial reasons, data support is not available.

Declarations

Ethics Approval This article does not contain any studies with human participants or live animals performed by any of the authors.

Competing Interests The authors declare no competing.

References

1. Liu W, Anguelov D, Erhan D, Szegedy C, Reed SE, Fu C-Y, Berg AC. SSD: Single shot multibox detector. In: European Conference on Computer Vision. 2015.
2. He K, Gkioxari G, Dollár P, Girshick RB. Mask R-CNN. In: 2017 IEEE International Conference on Computer Vision (ICCV). 2017. p. 2980–8.
3. Cao Z, Simon T, Wei S-E, Sheikh Y. Realtime multi-person 2D pose estimation using part affinity fields. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). 2016. p. 1302–10.
4. Deng R, Zeng G, Rui G, Zha H. Image-based building reconstruction with Manhattan-world assumption. In: Pattern Recognition. 2011.
5. Chen L-C, Papandreou G, Schroff F, Adam H. Rethinking atrous convolution for semantic image segmentation. arXiv:1706.05587 [Preprint]. 2017. Available from: http://arxiv.org/abs/1706.05587.
6. Zhou X, Wang D, Krähenbühl P. Objects as points. arXiv:1904.07850 [Preprint]. 2019. Available from: http://arxiv.org/abs/1904.07850.
7. Soviany P, Ionescu RT. Optimizing the trade-off between single-stage and two-stage deep object detectors using image difficulty prediction. In: 2018 20th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing (SYNASC). 2018. p. 209–14.
8. Ge Z, Liu S, Wang F, Li Z, Sun J. YOLOX: Exceeding YOLO series in 2021. arXiv:2107.08430 [Preprint]. 2021. Available from: http://arxiv.org/abs/2107.08430.
9. Tian Z, Shen C, Chen H, He T. FCOS: Fully convolutional one-stage object detection. In: 2019 IEEE/CVF International Conference on Computer Vision (ICCV). 2019. p. 9626–35.
10. Güler RA, Trigeorgis G, Antonakos E, Snape P, Zafeiriou S, Kokkinos I. DenseReg: Fully convolutional dense shape regression in-the-wild. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). 2016. p. 2614–23.
11. Wang X, Zhang R, Kong T, Li L, Shen C. SOLOv2: Dynamic, faster and stronger. arXiv:2003.10152 [Preprint]. 2020. Available from: http://arxiv.org/abs/2003.10152.
12. Qiao S, Shen W, Zhang Z, Wang B, Yuille AL. Deep co-training for semi-supervised image recognition. arXiv:1803.05984 [Preprint]. 2018. Available from: http://arxiv.org/abs/1803.05984.
13. Deng Y, Manjunath BS. Unsupervised segmentation of color-texture regions in images and video. IEEE Trans Patt Anal Mach Intell. 2001;23(8):800–10.
14. Sun K, Xiao B, Liu D, Wang J. Deep high-resolution representation learning for human pose estimation. In: 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). 2019. p. 5686–96.
15. Deng J, Guo J, Zhou Y, Yu J, Kotsia I, Zafeiriou S. RetinaFace: Single-stage dense face localisation in the wild. arXiv:1905.00641 [Preprint]. 2019. Available from: http://arxiv.org/abs/1905.00641.
16. Fang H, Xie S, Tai Y-W, Lu C. RMPE: Regional multi-person pose estimation. In: 2017 IEEE International Conference on Computer Vision (ICCV). 2016. p. 2353–62.
17. Godard C, Aodha OM, Brostow GJ. Digging into self-supervised monocular depth estimation. In: 2019 IEEE/CVF International Conference on Computer Vision (ICCV). 2018. p. 3827–37.
18. Bhatia S, Hooi B, Yoon M, Shin K, Faloutsos C. Midas: Microcluster-based detector of anomalies in edge streams. Proc AAAI Conf Artif Intell. 2020;34(4):3242–9.
19. Ranftl R, Lasinger K, Hafner D, Schindler K, Koltun V. Towards robust monocular depth estimation: Mixing datasets for zero-shot cross-dataset transfer. IEEE Trans Pattern Anal Mach Intell. 2019;44:1623–37.
20. Ranftl R, Bochkovskiy A, Koltun V. Vision transformers for dense prediction. In: 2021 IEEE/CVF International Conference on Computer Vision (ICCV). 2021. p. 12159–68.
21. Geiger A, Lenz P, Stiller C, Urtasun R. Vision meets robotics: The KITTI dataset. Int J Robot Res. 2013;32(11):1231–7.
22. Cordts M, Omran M, Ramos S, Scharwächter T, Enzweiler M, Benenson R, Franke U, Roth S, Schiele B. The cityscapes dataset. In: CVPR Workshop on The Future of Datasets in Vision. 2015.
23. Gordon A, Li H, Jonschkowski R, Angelova A. Depth from videos in the wild: Unsupervised monocular depth learning from unknown cameras. In: 2019 IEEE/CVF International Conference on Computer Vision (ICCV). p. 8976–85.
24. Yin Z, Shi J. GeoNet: Unsupervised learning of dense depth, optical flow and camera pose. In: 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2018. p. 1983–92.
25. Chen T, Gu D. CSA6D: Channel-spatial attention networks for 6D object pose estimation. Cognit Comput. 2021.
26. Wang F, Xie Z. An adversarial multi-task learning method for Chinese text correction with semantic detection. 2023.
27. Xu P, Cai J, Gao Y, Rong Z, Xin H. MIRACLE: Multi-task Learning based Interpretable Regulation of Autoimmune Diseases through Common Latent Epigenetics. 2023.
28. Lin T-Y, Dollár P, Girshick RB, He K, Hariharan B, Belongie SJ. Feature pyramid networks for object detection. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). 2016. p. 936–44.
29. Ding X, Zhang X, Ma N, Han J, Ding G, Sun J. RepVGG: Making VGG-style convnets great again. In: 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). 2021. p. 13728–37.
30. Woo S, Park J, Lee J-Y, Kweon I-S. CBAM: Convolutional block attention module. In: European Conference on Computer Vision. 2018.
31. Loshchilov I, Hutter F. Fixing weight decay regularization in adam. arXiv:1711.05101 [Preprint]. 2017. Availalble from: http://arxiv.org/abs/1711.05101.
32. Cheng B, Girshick R, Dollár P, Berg AC, Kirillov A. Boundary IoU: Improving object-centric image segmentation evaluation. 2021.
33. Han K, Wang Y, Tian Q, Guo J, Xu C, Xu C. GhostNet: More features from cheap operations. 2020.
34. Chen J, Kao S-H, He H, Zhuo W, Wen S, Lee C-H, Chan S-HG. Run, don't walk: Chasing higher FLOPS for faster neural networks. 2023.