agriculture

MDPI

Article

ICNet: A Dual-Branch Instance Segmentation Network for High-Precision Pig Counting

Shanghao Liu [1,2], Chunjiang Zhao [1,2,*], Hongming Zhang [1], Qifeng Li [2], Shuqin Li [1], Yini Chen [2,3], Ronghua Gao [2], Rong Wang [1,2] and Xuwen Li [2,4]

[1] College of Information Engineering, Northwest A&F University, Xianyang 712100, China; lshrus@nwafu.edu.cn (S.L.); zhm@nwsuaf.edu.cn (H.Z.); lsq_cie@nwsuaf.edu.cn (S.L.); rongw@nwafu.edu.cn (R.W.)
[2] Research Center of Information Technology, Beijing Academy of Agriculture and Forestry Sciences, Beijing 100097, China; liqf@nercita.org.cn (Q.L.); cynmoon@ncepu.edu.cn (Y.C.); gaorh@nercita.org.cn (R.G.); 2209028114@stu.tjau.edu.cn (X.L.)
[3] School of Mathematics and Physics, North China Electric Power University, Beijing 102206, China
[4] School of Computer and Information Engineering, Tianjin Agricultural University, Tianjin 300384, China
* Correspondence: zhaocj@nercita.org.cn

Abstract: A clear understanding of the number of pigs plays a crucial role in breeding management. Computer vision technology possesses several advantages, as it is harmless and labour-saving compared to traditional counting methods. Nevertheless, the existing methods still face some challenges, such as : (1) the lack of a substantial high-precision pig-counting dataset; (2) creating a dataset for instance segmentation can be time-consuming and labor-intensive; (3) interactive occlusion and overlapping always lead to incorrect recognition of pigs; (4) existing methods for counting such as object detection have limited accuracy. To address the issues of dataset scarcity and labor-intensive manual labeling, we make a semi-auto instance labeling tool (SAI) to help us to produce a high-precision pig counting dataset named Count1200 including 1220 images and 25,762 instances. The speed at which we make labels far exceeds the speed of manual annotation. A concise and efficient instance segmentation model built upon several novel modules, referred to as the Instances Counting Network (ICNet), is proposed in this paper for pig counting. ICNet is a dual-branch model ingeniously formed of a combination of several layers, which is named the Parallel Deformable Convolutions Layer (PDCL), which is trained from scratch and primarily composed of a couple of parallel deformable convolution blocks (PDCBs). We effectively leverage the characteristic of modeling long-range sequences to build our basic block and compute layer. Along with the benefits of a large effective receptive field, PDCL achieves a better performance for multi-scale objects. In the trade-off between computational resources and performance, ICNet demonstrates excellent performance and surpasses other models in Count1200, AP of 71.4% and AP^{50} of 95.7% are obtained in our experiments. This work provides inspiration for the rapid creation of high-precision datasets and proposes an accurate approach to pig counting.

Keywords: pig counting; instance segmentation; deformable convolution; parallel modules; pig segmentation dataset

check for updates

Citation: Liu, S.; Zhao, C.; Zhang, H.; Li, Q.; Li, S.; Chen, Y.; Gao, R.; Wang, R.; Li, X. ICNet: A Dual-Branch Instance Segmentation Network for High-Precision Pig Counting. *Agriculture* **2024**, *14*, 141. https://doi.org/10.3390/agriculture14010141

Academic Editors: Luís Manuel Navas Gracia and Manuel Pérez-Ruiz

Received: 7 December 2023
Revised: 6 January 2024
Accepted: 11 January 2024
Published: 18 January 2024

Copyright: © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

Pig counting is an important part of livestock farming industry, as well as the vital process of recording and managing information such as pig growth, feeding, and reproduction costs in a pig farm. Accurate pig counting in pigsties can improve the efficiency of pig farm management. However, the traditional manual visual counting [1,2] and electronic ear tag counting [3] are inefficient, and difficult to reconcile at a later stage. With the arrival of the deep-learning era and the rapid development of computer graphics cards, the computing power of computers has been increasing, accelerating the development of computer vision

and deep-learning networks. Computer vision research is expanding, and it is widely used in object counting. Object counting algorithms provide the possibility of accurate counting, which is widely used in the field of agronomy [4–8] and accelerates the development of animal husbandry informationization.

Pig counting is a type of object counting which implements several mainstream methods like object recognition, object classification and object detection. For example, object detection counting can quickly achieve accurate counting for sparser scenes. The R-CNN proposed by Girshick et al [9], introduced a two-stage detection method for the first time. Furthermore, Wang [10] proposed a detection method that recognizes ear tags to check the specific situation of pigs in a production environment. Object detection methods use deep convolutional networks to obtain excellent target detection accuracy, but the accuracy of target recognition decreases when faced with scenes with occlusion between pigs and the sticky edges of pigs, which means it is not conducive for us to deploy it in real-world scene applications. For denser scenes, the current mainstream method is the density map estimation [11–13] based on convolutional neural networks (CNN). Feng et al. [11] proposed an architecture composed of density map network and a K-means clustering algorithm, which can quickly estimate the number of high-density and high-overlap crowds in the scene, but this method has a lower counting accuracy, which does not meet our requirements for accurate counting. In addition, some researchers focus on keypoints detection to monitor the status of pigs. Chen et al. [14] created a model to detect keypoints of pigs' bodies and identify individual pigs, which can associate pigs across video frames. Nonetheless, since only three points are used to mark a pig in frames, the method is prone to fault detection and identifying the background as a pig.

Instance segmentation is a technique used to perform pixel-level classification with high accuracy. There are a number of segmentation networks that have performed well so far. Kaiming He et al. proposed Mask R-CNN [15] by incorporating a Feature Pyramid Network (FPN) [16] for feature fusion and hierarchical detection, employing a more accurate bilinear interpolation approach RoIAlign instead of the traditional RoIPooling, which can generate richer feature mapping maps. Since then, Mask-RCNN has become the primary method of instance segmentation. Then, SOLOv1 [17] introduced the concept of "instance categories" to look at the instance segmentation task from a completely new perspective. The "Instance category" assigns a category to each pixel in an instance based on the location and size of the instance, effectively transforming instance mask segmentation into a categorizable problem to be solved. SOLOv1 is a simpler and more flexible instance segmentation framework that outperforms Mask R-CNN in terms of accuracy. However, SOLOv1 currently faces the problems of inadequate representation and learning of masks, insufficiently high resolution of masks obtained from prediction and is too time-consuming NMS for post-processing. Therefore, SOLOv2 [18] adopted the idea of dynamic convolution, which enabled the network to dynamically segment the instances based on the position information, and proposed Matrix NMS, which greatly improved the speed of post-processing. Maskformer [19] as well as Mask2former [20] were proposed to refine the task of instance segmentation even more, using a mask classification model that predicts class-specific masks for each instance in an image. Since Maskformer suffers from the problem of high memory occupation, leading to training difficulties, Cheng et al. further proposed Mask2former, which is keyed to the Masked-attention Mask Transformer structure, is based on the self-attention mechanism [21], and uses the mask technique to generate the segmentation masks. Compared to the traditional encoder–decoder-based neural network model, Mask2former achieves better results in instance segmentation tasks. Nonetheless, existing methods still face challenges such as interactive occlusion, overlapping and inaccurate counting, and the lack of a substantial high-precision pigcounting dataset, which is attributed to the inefficiency in human production of labels. In summary, to complete the task of precise counting, we choose instance segmentation to detect and count pigs instead of objection detection, density estimation, etc., owing to the advan-

tages of recognizing the outline of an object independently and clearly separating it from the background.

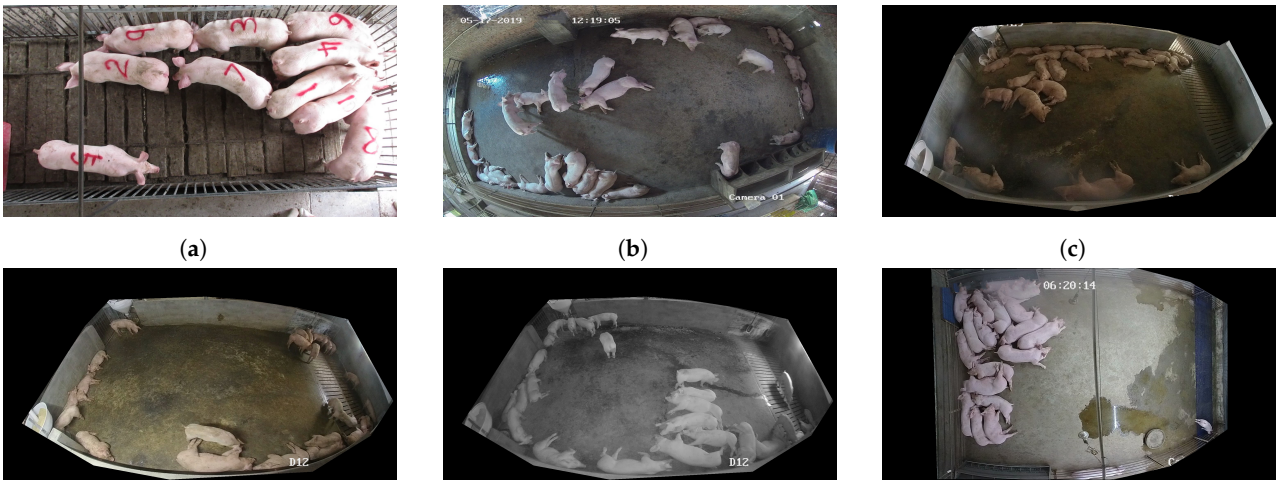
In this paper, on the one hand, in order to acquire high-precision datasets, we design a semi-automatic labeling software based on Segment Anything Model (SAM) [22], which are faster and more accurate than the manual labeling methods in labeling segmentation labels. On the other hand, for accurate pig counting in realscenario, we adopt the framework of Mask-RCNN, combining the advantages of DCNv3 [23] and dual-branch structure to design a pig counting backbone named ICNet, which achieves a better performance in high-precision dataset and less computational resource. In general, the contribution of our paper are summarized in the following three aspects:

- We create a semi-auto instance labeling tool called SAI, which is based on the SAM, to produce faster and more accurate segmentation labels, and a high-precision pig counting dataset called Count1200.
- We reconstruct a more robust backbone called ICNet, with a Pipe Layer and four Parallel Deformable Convolutional Layers (PDCLs), which effectively exploits the long time-series modeling property and is able to expand the sensory field while saving computational resources.
- We design a module named Parallel Deformable Convolutional Block (PDCB) that consists of a double-branch structure and skip-connection, which is used to form the PDCL. PDCB is based on DCNv3, which is able to achieve more features with fewer parameters, and is more suitable for our detection work on multi-scale datasets.

2. Materials and Methods

2.1. Count1200 Dataset

Count1200 is an instance segmentation dataset primarily designed for the problem of pig counting. The dataset comprises 1220 images, encompassing situations from various scenarios, including different times of day and varying lighting conditions. We collect images from the Internet and take pictures at the livestock farm, representing 75% and 25% of the total number of images, respectively, consisting of situations of small-scale objects (Figure 1c,d), occlusion between pigs and other obstructions such as beams and feeding troughs (Figure 1a,f), overlapping or clustering among pigs (Figure 1e,f), and low-light challenges posed by nighttime perspectives (Figure 1e); this enables our dataset to possess diversity and robustness. Considering the need for multi-scale counting, the data we collected include 9–28 pigs per image.



(**a**) (**b**) (**c**)

(**d**) (**e**) (**f**)

Figure 1. Samples of Count1200 capturing in different situations. (**a**) Large-scale objects. (**b**) Normal condition. (**c**) Small-scale objects. (**d**) Small-scale objects. (**e**) Nighttime perspectives. (**f**) Overlapping and clustering.

It is a tedious and difficult task for researchers to make the ground truth of instance segmentation, which has persistently hindered the development of instance segmentation [24]. In traditional manual annotation methods, researchers often use the method of clicking around the pigs multiple times to draw polygons for mask labels. When there are too many objects in the image, this type of annotation process is very time-consuming and labor-intensive. With its strong ability to identify masks of natural objects, SAM can segment objects easily; that means this model can rapidly generate numerous points to outline the shape of anything, even in complicated environments. The details of SAI are shown in Figure 2.

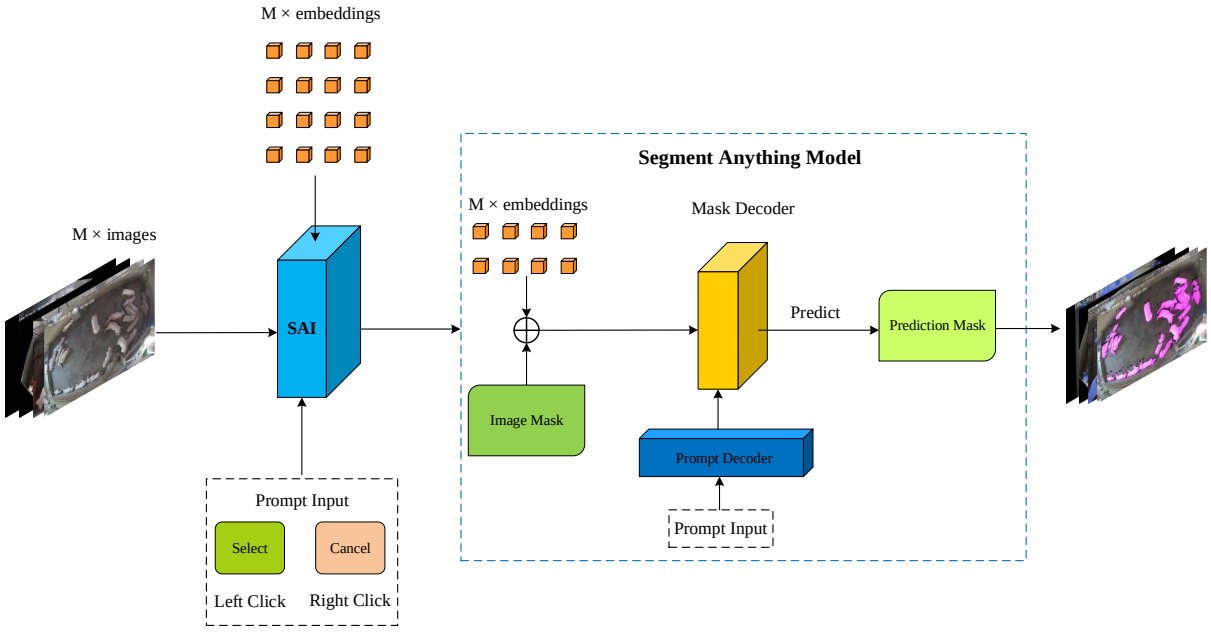
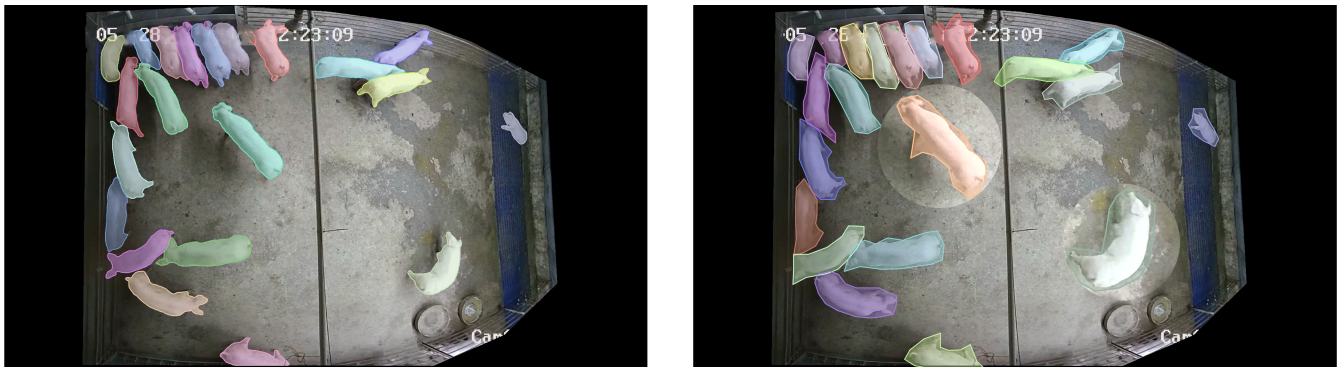


Figure 2. The process and details of SAI.

Firstly, M-embedding files are generated by the encoder of ViT and correspond one-to-one with the images, which contain feature information and positional encoding. Prompt input is a series of mouse operation instructions used to select regions of concern. With the Graphical User Interface (GUI) of SAI, we can use left click to select pigs and right click to fine-tune the areas of interest in real time. When it comes to the details of the SAM model, the mask decoder plays a pivotal role and the peripheral operations are sent to the mask decoder after passing through the prompt decoder. In addition, the combination of embeddings and image masks serves as input for the mask decoder, too. The mask decoder utilizes the above resources to make mask predictions and provide real-time feedback in the image. The specific visualization effect is illustrated in the right size of Figure 2.

We successfully leverage the features above to develop a semi-auto instance labeling tool named SAI with a convenient GUI. With just left and right clicks, we can generate a mask in 5 s, which is seven times faster than manual annotation. The comparison of the two methods is illustrated in Figure 3. As opposed to the manual annotation with coarse-grained contour, the mask produced by SAI seamlessly fits around the objects, which helps models to separate pigs and background quickly and allows the feature of instances to be extracted efficiently without interference from irrelevant information, as shown in Figure 3a,b.

Along with the great performance of SAI, we create a COCO-type dataset [25] that consists of 25,762 instances and divide them into training and testing sets in an 8:2 ratio, containing 975 and 245 images, respectively. This dataset presents significant challenges for both instance segmentation and counting tasks. We aim to provide a foundational idea for subsequent research.



(**a**) (**b**)

Figure 3. Comparison of semi-auto and manual annotation methods. (**a**) Semi-auto annotation (96 points, 5s per instance). (**b**) Manual annotation (17 points, 35s per instance).

2.2. Methods

This research concentrates on a pig-counting task based on instance segmentation, with the aim of achieving clear separation among objects and between objects and the background that facilitates accurate counting. With its great performance in CNN models, the whole structure of our model follows the design of Mask-RCNN [15], as shown in Figure 4. The results of prediction consist of coordinates of bounding boxes, masks, and categories of instances.

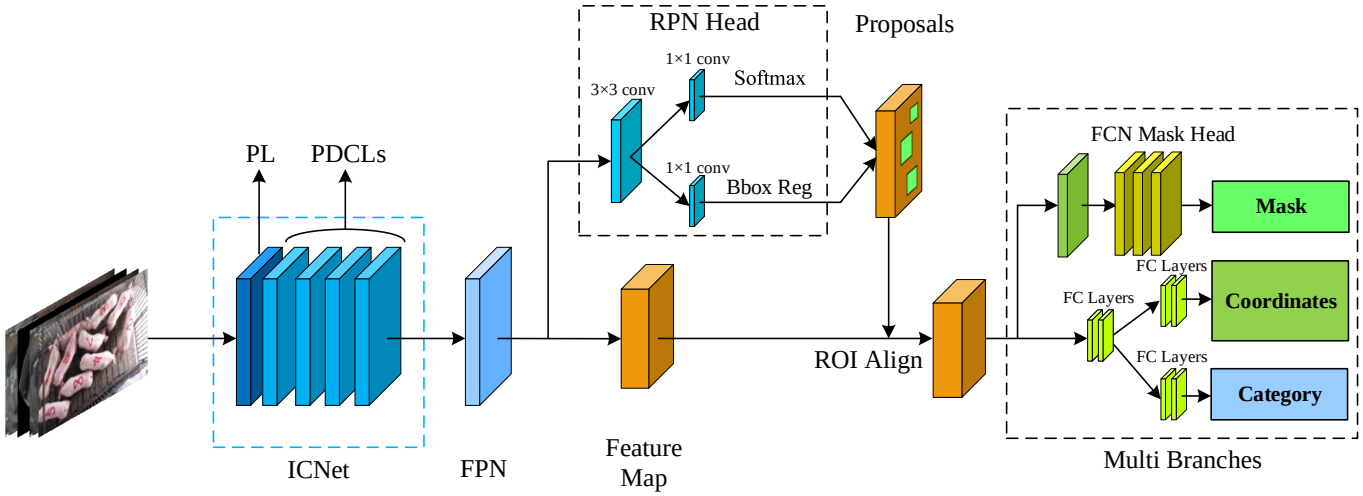


Figure 4. Architecture of the model.

2.2.1. Instances Counting Network (ICNet)

Deviating from the traditional pipeline of Mask-RCNN using ResNet [26] as a feature-extracting module, we suggest ICNet as a more robust and appropriate backbone for instance segmentation tasks (the area outlined by the blue dashed box in Figure 3). The details of ICNet are illustrated in Figure 5.

By combining it with a series of layers, we construct a network with the ability to achieve long-range dependence, adaptive spatial aggregation and efficient utilization of memory and computation. That means, compared to normal convolution, we have a larger receptive field and can better preserve information such as long-distance spatial information and dense pixel interaction information extracted from images at the same kernel size. One pipe layer and four PDCLs serve as the main building blocks of ICNet, with a downsampling layer positioned after each PDCL. The feature map is sent to FPN and RPN [27] for additional information compression and proposal generation after semantic and spatial information have been extracted from the input. After implementing ROI

Align, we use a Fully Convolutional Networks Mask Head to generate a mask of regions proposals, and we follow the design of Mask R-CNN to use the fully convolutional layers to predict the coordinates and categories of instances.

2.2.2. Pipe Layer

The pipe layer is a combination of two cycles comprising 3×3 convolution, Batch Normalization [28] and *ReLU* [29], whose function is to preprocess the data and adjust the resolution of input, improving the resolution of basic semantic information. A single combination cycle is used to reduce the input resolution by two times; the configuration of the convolutions is $stride = 2$ and $padding = 1$. The specific pipeline of two cycles can be described using Formulas (1) and (2), where $X_{input} \in \mathbb{R}^{H \times W \times 3}$ and $X_{output} \in \mathbb{R}^{H \times W \times 64}$.

$$X_{temp}^{H \times W \times 32} = ReLU(BN(Conv(X_{input}))) \tag{1}$$

$$X_{output} = BN(Conv(X_{temp})) \tag{2}$$

2.2.3. Downsampling Layer

Instead of using a single normalization layer in other networks, we combine Batch Normalization (BN) and Layer Normalization (LN) [30] for the downsampling layer, which preserves the benefits of modeling long-range sequences and addresses internal covariate shift. The function of the downsampling layer is to reduce scales of input by half and double the number of channels, which is similar to the function of the pipe layer, as indicated by the blue box in the lower part of Figure 5.

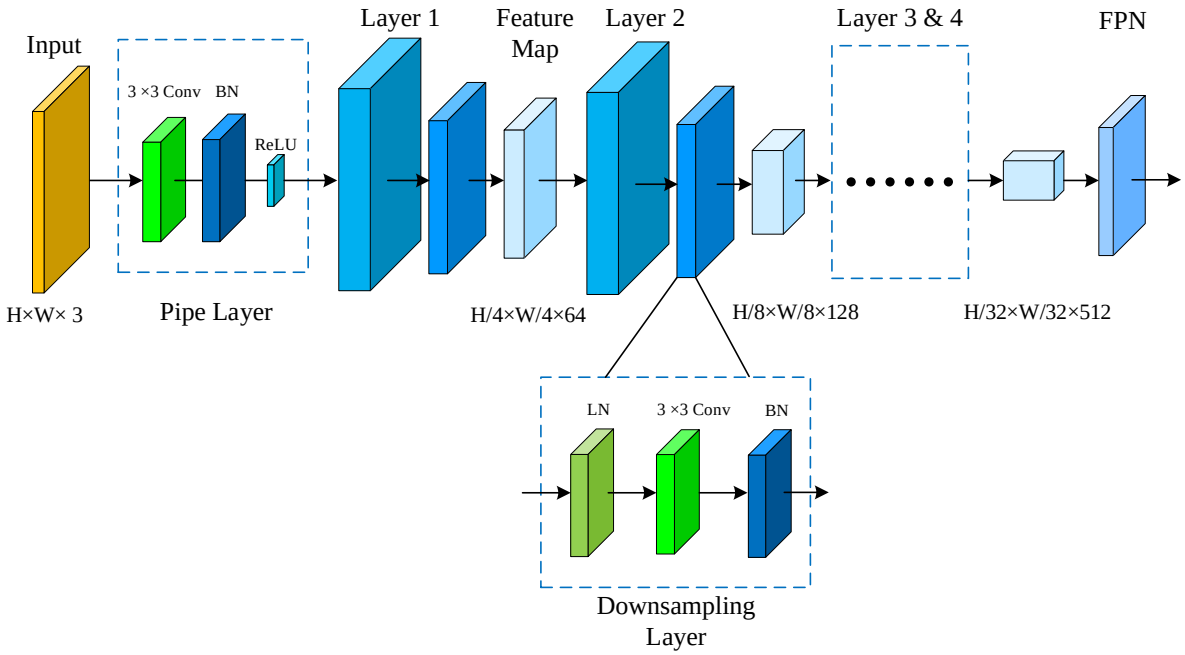


Figure 5. The overall structure of the ICNet, where BN and LN stand for Batch Normalization and Layer Normalization, respectively.

The downsampling layer ensures that the model incorporates multi-scale information, making it more conducive to downstream tasks when data comprise objects of various sizes. The details of the downsampling are shown in Formula (3), where $Permute()$ is a dimension transformation in the order of (0, 3, 1, 2).

$$X'^{H \times W \times C/2} = BN(Permute(LN(X^{H \times W \times C}))) \tag{3}$$

2.2.4. Parallel Deformable Convolutions Layer (PDCL)

The PDCL is a sequence composed of computing basic blocks named parallel deformable convolution blocks (PDCBs). The implementation details of this computing layer are illustrated in Figure 6.

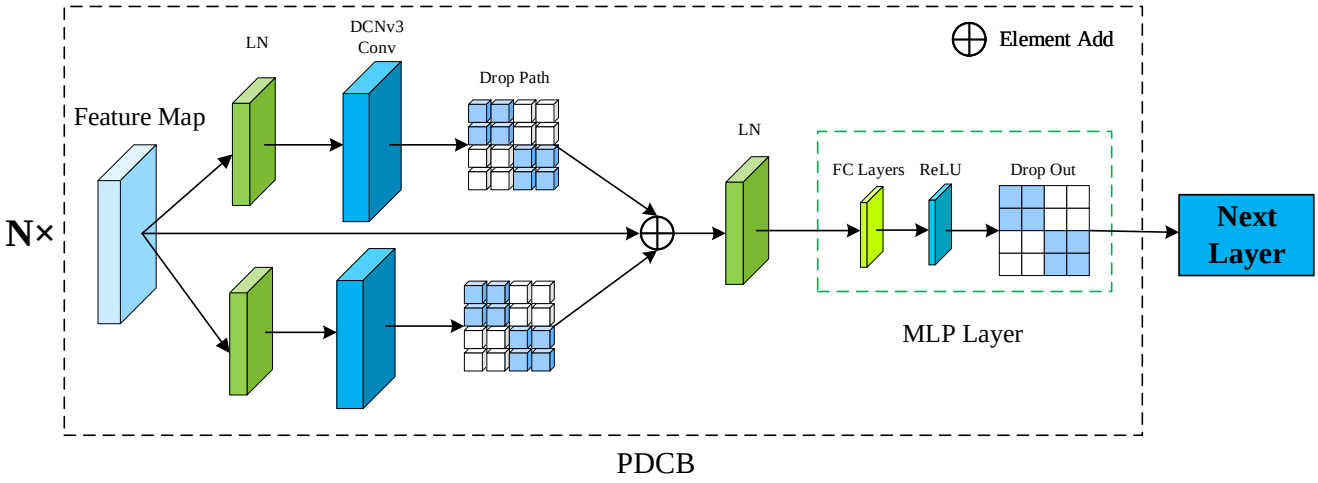


Figure 6. Illustrating the specific computational components of PDCL.

An aggregation of N basic blocks is represented by PDCL, where N determines the size of each layer and serves as a hyper-parameter for the number of PDCB. For instance, in this work, we stack four layers to build ICNet, where the array for N is $N_i \in [2, 2, 8, 2]$ indicating that there are two blocks in $Layer_1$, eight blocks in $Layer_3$, and so on.

In order to achieve a balance between performance and resources consumption, a dual-branch structure is developed by us, whose core operation is DCNv3 that occurs after LN and before a drop path layer [31]. In our experiment, layer normalization exhibits strong adaptability with DCNv3 and we make good use of this feature to build our basic block. Compared to the basic block in InternImage [23], our PDCB uses less parameters to achieve a better performance owing to the creative unities of double branches features and skip-connection [26]. The dual-branch blocks can be depicted as Formula (4):

$$X_i = Drop\_Path(\gamma \times DCNv3(LN(X))), \quad (i = 1, 2) \tag{4}$$

Integer i represents the ith branch in the current block, while γ is a hyper-parameter used to scale the results and we set $\gamma = 1$ in this research.

$Drop\_path()$ is a layer of network regularization method, which is proposed for branch networks. Compared to the common method used to prevent overfitting—drop out [32], which applies to specific feature maps—$Drop\_path()$ involves randomly dropping network branches and is particularly suitable for multi-branch structured networks.

In summation, the principal procedure of PDCL can be described as Formula (5):

$$Layer_j = F_N(MLP(LN(X + \sum_{i=1}^{2} X_i))), \quad (j = 1, 2, 3, 4) \tag{5}$$

where j is the jth layer in ICNet, and F_N means that we recursively call N PDCBs to process the input.

2.2.5. Deformable Convolutions v3 (DCNv3)

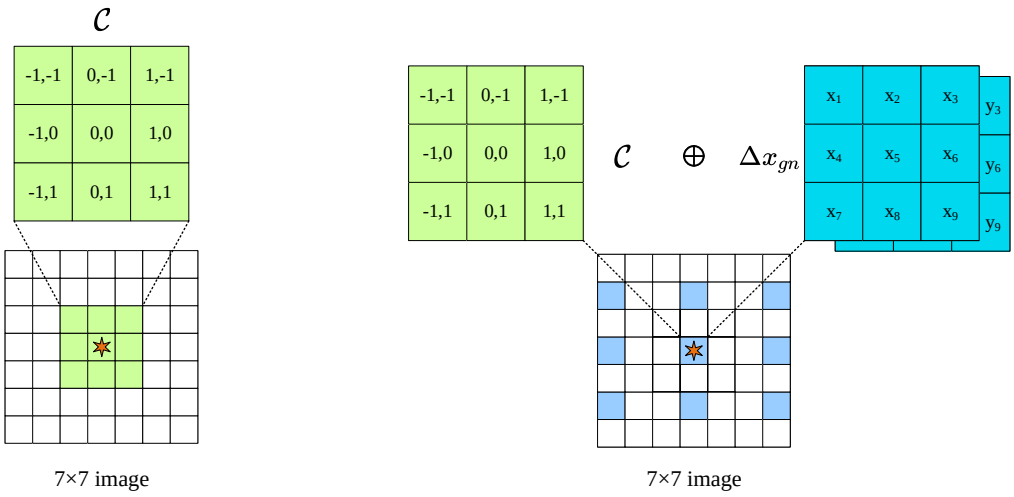
Deformable Convolutions was created in 2017 by Jifeng Dai [33] and further improved by Xizhou Zhu in 2019 [34]. It has been frequently used in instance segmentation and object detection-related tasks, since it has an irregular convolution shape, which allows a more flexible range of receptive fields when scanning images. Owing to the advantages

listed above, deformable convolution is adapted to this work with the aim of achieving multi-scale detection of targets.

Regular shapes are always equipped in common convolution such as square (1×1, 3×3 kernel), rectangle (1×7 kernel), and so on, which can traverse images patch by patch to a generate feature map. This can be summarized by the following formula, taking 3×3 kernel as an example:

$$M_{out}(x_0) = \sum_{x_n \in \mathcal{C}} w(x_n) \cdot M_{in}(x_0 + x_n) \tag{6}$$

where M_{in} and M_{out} are the input feature map and output feature map, respectively, and x_0 is the current pixel point handled by the kernel. $\mathcal{C}$ represents a grid region of coordinates that determines the dilation and receptive field size of kernel, the visualization effect is as shown in Figure 7a and x_n traverses the position in $\mathcal{C}$. $w(x_n)$ is a corresponding weight coefficient of enumeration in $\mathcal{C}$.



(**a**) Classical convolution. (**b**) Deformable convolution.

Figure 7. Examples displayed under 3×3 kernel within 7×7 image. The pentagram of image represents the center of convolution. The green and blue squares in these two 7×7 images represent the receptive field range of current convolution.

For the DCNv3, an offset matrix and ideas of groups dividing are incorporated into the convolution operation.

The offset matrix is a learnable convolutional layer that indicates the expansion scale and direction of the current convolution kernel. This particularity means common convolution had a deformable receptive field and the visible range of convolution is irregular. The values in the offset are highly flexible, granting characteristics of long-range dependencies and spatial aggregation which is similar to that of Transformer.

The concept of a multi-group [35] is initially applied to image classification tasks and significantly excels in MHSA. Partitioning operators into multiple groups is equivalent to dividing them into multiple subspaces, facilitating models to focus on different pieces of information from various zones. The calculation process of deformable convolution v3 is displayed as

$$M_{out}(x_0) = \sum_{g=1}^{G} \sum_{n=1}^{N} w_g m_{gn} \cdot M_{in}(x_0 + x_n + \Delta x_{gn}) \tag{7}$$

G is the number of groups we set [2, 4, 8, 16] for each PDCL, meaning that the dimensions of the feature map in each group are $C_g = C_i / G_i$, where C_i represents the total channels in $Layer_i$. Positive integer N is the number of sampling pixels that equal $|\mathcal{C}|$ in Formula (6). The offset matrix is denoted by $\Delta x_{gn} \in \mathbb{R}^{H \times W \times 2S}$, where S is the kernel size

of convolution and 2S rules the offset in the $\mathcal{X}$ and $\mathcal{Y}$ directions. $w_g \in \mathbb{R}^{C_i \times C_g}$ on behalf of the projection weights of the current group, which is similar to $w(x_n)$ in Formula (6). As for the m_{gn}, it represents a scale factor of x_n in this group.

Throughout the stages of growth, pigs are housed in different pigsties, since pigs vary greatly in their body shape at different periods. Additionally, differences in the capture distance during imaging can result in variations in individual sizes within the same pen [36]. Moreover, the shooting edges of the camera can also cause image distortion. The truths mentioned above can easily cause inaccuracies in detection models with datasets collected from real farming scenarios. With the help of DCNv3, our model can adapt to the changing feature distributions, which can greatly alleviate the impact of the above issues.

2.3. Experimental Environment Setup

All experiments are conducted under the Pytorch & CUDA framework in a version of 1.12.1 and 11.3, respectively. We use a NVIDIA (Santa Clara, CA, USA) RTX A6000 to train these models from scratch with a batch size of 32 and 200 epochs.

2.3.1. Data Preprocessing

We set the size of input images to [512, 512], a common setting in many segmentation models, which creates a balance between segment accuracy and memory consumption. Apart from this, data augmentation methods are employed to enhance the generalization capability, including random flipping and random cropping, with a flipping ratio of 50% and crop size of [512,512].

2.3.2. Optimization Strategy

AdamW [37] is used in our research with the strategy of layer decay [38]. We initialize the $lr = 1 \times 10^{-4}$ with the weight decay = 5×10^{-2}. In addition, we perform a linear warm-up [39] in [180, 190] epochs with $\gamma = 0.1$. The remaining parameter configurations remain consistent with Mask R-CNN [15] and InternImage [23].

2.3.3. Evaluation Metric

COCO evaluation metrics are used in this research to estimate our results. True Positive (TP), False Positive (FP), True Negative (TN), False Negative (FN), and Intersection over Union (IoU) are applied to compute the precision of segmentation. The IoU describes the relationship between the predicted mask and the ground truth in terms of their overlapping areas, illustrated in Formula (8).

$$IoU = \frac{Area(O)}{Area(U)} \tag{8}$$

where $Area(O)$ represents the overlapping area between label and prediction and $Area(U)$ is the sum of these two areas. Meanwhile, the precision and recall at a specific threshold are represented by the following formula, where t is the threshold of IoU.

$$P = \frac{TP(t)}{TP(t) + FP(t)} \tag{9}$$

$$R = \frac{TP(t)}{TP(t) + FN(t)} \tag{10}$$

AP is calculated using Formula (11) and mAP is the average of AP values across all categories.

$$AP = \int_1^0 P(r)\, dr \tag{11}$$

We will display the AP and AR at different IoU thresholds [0.5:0.05:0.95] of results in the next section.

Beyond that, we select the Mean Absolute Error (MAE) and Root Mean Square Error (RMSE) to assess the precision of counting [40]. These two metrics are calculated by the formula as follows:

$$MAE = \frac{1}{N}\sum_{n=1}^{N}|y_n - \hat{y}_n| \tag{12}$$

$$RMSE = \sqrt{\frac{1}{N}\sum_{n=1}^{N}(y_n - \hat{y}_n)^2} \tag{13}$$

where N is the total number of images and n is nth image in this set. The number of actual instances in the nth image is denoted by y_n, while the number of predicted masks is indicated by $\hat{y}_n$. MAE and $RMSE$ are inversely correlated with the model's accuracy and robustness.

3. Results

3.1. Results on the Test Set

We train our model with an input size of 512 × 512, and we conduct a comparative experiment between our model and mainstream instance segmentation models in terms of # params, # FLOPs, AP, and AR. The test results are summarized as follows in Table 1.

ICNet arrives at AP of 71.4% and AP^{50} of 95.7% with parameters of 33M, the highest score and the fewest parameters of all the models. In addition, we also outperform other networks in the rest of the metrics. Compared to SOLOv1 with similar parameter values, we achieve a 24.1% increase in AP and a 13.1% increase in AP^{50}. In contrast, SOLOv2 obtains AP of 60.6 % and AP^{50} of 90.9%, which are close to our results, but it uses more parameters. Compared to the model based on Vision Transformer(ViT) [41] and Mask2Former, Mask2Former still cannot achieve matching results with different backbones, and due to the use of ViT's architecture, its number of parameters is about twice that of ICNet. In comparison with InternImage, which also builds basic blocks based on DCNv3, ICNet surpasses InternImage in all metrics using only 33M parameters while InternImage has parameters of 49M. As for the AP and AP^{50}, the results obtained using ICNet are 3.8% and 2.2% higher than those obtained using InternImage.

ICNet and InternImage have better segmentation accuracy compared to other models in various metrics, which verifies the superiority of DCNv3 in instance segmentation and pig counting. Furthermore, the novel design of the dual branch structure allows us to more effectively extract the feature information contained in the images, while the addition of a drop path prevents the model from overfitting. In addition, the use of skip connections helps us to mitigate feature loss after multiple layers of convolutional processing, which is crucial for information transmission. In addition, the combination of downsampling layers and the use of BN and LN between layers grants us abundant features compared to other models. Our ablation experiment also indirectly proves this point.

Table 1. Comparison of the Count1200 test set. The volume of parameters and FLOPs are obtained across the entire model workflow. We show the key evaluation metrics from the COCO dataset.

Methods	Scale	Param.	# FLOPs	AP	AP^{50}	AP^{75}	AP^{M}	AP^{L}	AR
SOLOv1	512^2	36M	**106G**	47.3	82.6	51.7	9.7	53.4	56.5
SOLOv2	512^2	65M	133G	60.6	90.9	67.4	33.2	64.5	67.4
Mask R-CNN	512^2	62M	134G	47.8	77.1	54.2	13.3	52.7	54.9
Mask2Former (R)	512^2	63M	117G	53.5	85.1	59.9	15.5	58.9	60.0
Mask2Former (S)	512^2	69M	125G	54.1	84.6	61.0	15.0	59.7	60.6
InternImage	512^2	49M	117G	67.6	93.5	81.4	37.7	71.6	70.7
ICNet	512^2	**33M**	**106G**	**71.4**	**95.7**	**86.4**	**47.5**	**74.5**	**74.3**

Mask2Former (R) and Mask2Former (S) means that use the ResNet and Swin Transformer as backbones, respectively. The bold numbers in the label represent the optimal value.

3.2. Ablation Study

We use one branch without drop path and a single LN as a normalization layer to build up the simple block, which is the foundation of the entire ablation study. We attempt to add BN, dual branch structure, and drop path for testing. The test results are shown in the Table 2 below.

The experiment demonstrates that the model's performance cannot be enhanced by a single LN layer. Combining BN and LN can better induce and compress data, achieving 70.7% and 74.3% on AP and AR, respectively. In addition, when we introduce the dual branch structure into the model, the model improves by 0.3% on AP compared to Experiment No. 2, but achieve the same 74.3% on AR as Experiment No. 5, which is the highest value in the entire experiment. This fully demonstrates the effectiveness of the dual branch structure. Drop path is introduced in experiment No. 4, but our AR decreases by 0.1% and our AP only improves by 0.1%, suggesting that the effect of drop path is not significant for a single branch. We conduct a No. 5 experiment by combining drop path with double branches, which shows significant improvements in both AP and AR compared to the initial simple block. We also choose this structure as the foundation for our PDCB.

Table 2. The ablation experiment of ICNet. AP and AR are displayed to represent the improvement with different module combinations.

No.	Simple Block	BN	Dual	Drop Path	AP	AR
1	√				66.5	69.7
2	√	√			70.7	73.8
3	√	√	√		71.0	**74.3**
4	√	√		√	70.8	73.7
5	√	√	√	√	**71.4**	**74.3**

The √ symbol represents that we use this component in the current experiment. The bold numbers in the label represent the optimal value

3.3. Evaluation of Counting

In the evaluation of counting results, the test results of each model in MAE and RMSE are shown in the Table 3.

It is evident from the table that ICNet performs best in terms of both MAE and RMSE (the smaller the MAE value, the higher the model evaluation accuracy, and the smaller the RMSE value, the stronger the model robustness). Combining it with Table 1 shows that models with better segmentation performance also have higher accuracy when it comes to counting. Therefore, the instance segmentation method for pig counting is very feasible and effective, and the model we build is accurate and robust enough.

Table 3. Comparison of counting result evaluations.

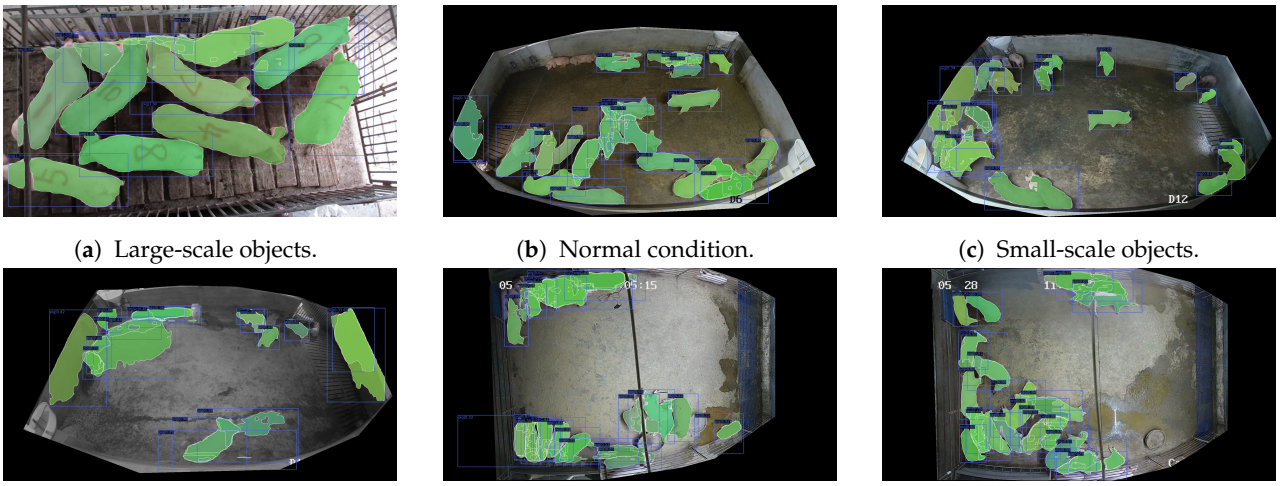
Metrics	MAE	RMSE
SOLOv1	3.71	4.69
SOLOv2	1.20	1.83
Mask R-CNN	2.86	3.55
Mask2Former (R)	1.73	2.37
Mask2Former (S)	2.39	3.13
InternImage	0.93	1.38
ICNet	**0.68**	**1.07**

The bold numbers in the label represent the optimal value.

3.4. Visualization Results

We test ICNet, InternImage and Mask R-CNN on the test set of Count1200 and the visualization results are displayed below.

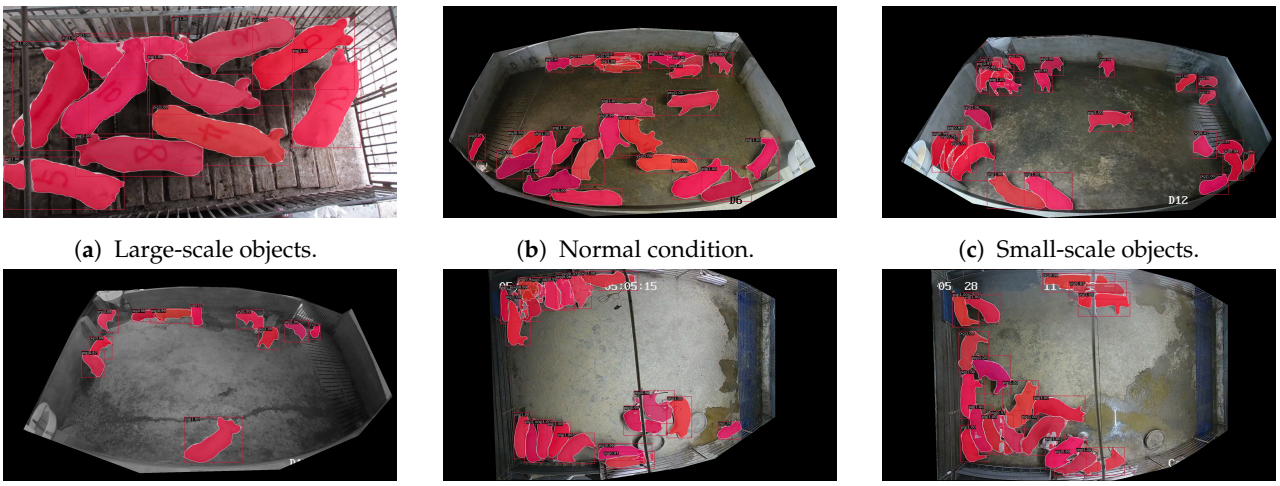
We select scenes with multi scales and different lighting intensities. The visualization results of Mask R-CNN are shown in Figure 8, showcasing issues such as repeatedly segmenting the same pig, missing pigs and identifying the background as pigs, which is a undesirable situation for detection and counting purposes. Owing to its close accuracy compared with ours, as shown in Figure 9, InternImage exhibits similar detection performance on large-scale targets to ours, such as in the clear segmentation results in Figure 9a. In addition, when facing scenes with a large number of objects, InternImage may fail to count a significant number of pigs, as illustrated in Figure 9b,d. Compared to the rough detection results of Mask R-CNN and the coarse-grained segmentation results of InternImage, the masks we produce almost seamlessly fit the pigs and can still be effectively recognized even when the pigs overlap. We can even accurately segment pigs when they are obstructed by objects such as the iron pipes in Figure 10a,e,f. Besides, there are few cases of missed or incorrect detections, which are consistent with the results in the Table 3.



(**a**) Large-scale objects. (**b**) Normal condition. (**c**) Small-scale objects.

(**d**) Nighttime perspectives. (**e**) Overlapping. (**f**) Occlusion.

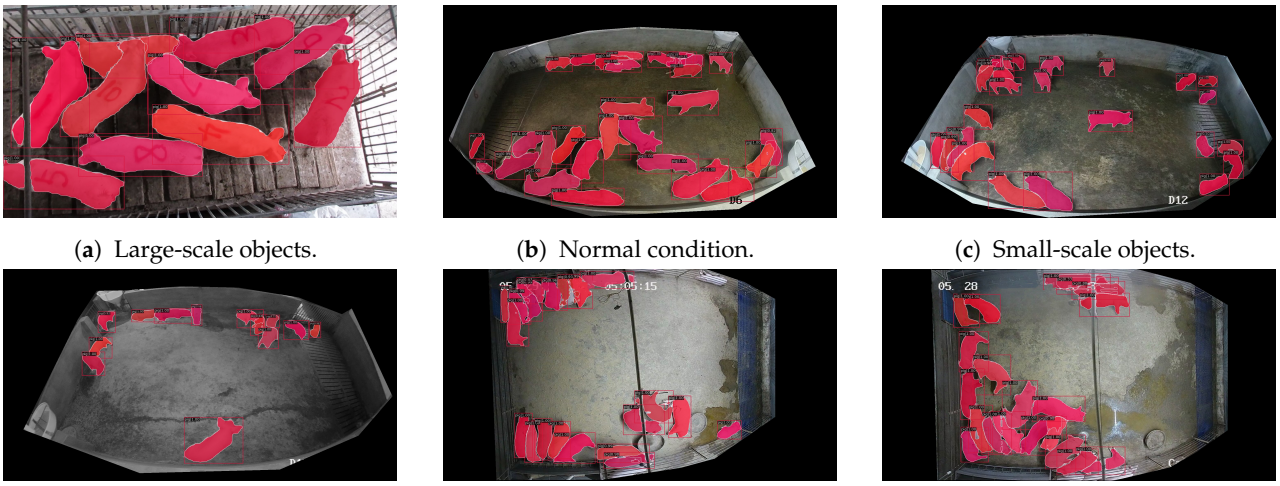
Figure 8. Visualization results of test set including different situations based on Mask R-CNN. The numbers on pigs of subfigure (**a**) represent the identifiers of pigs.



(**a**) Large-scale objects. (**b**) Normal condition. (**c**) Small-scale objects.

(**d**) Nighttime perspectives. (**e**) Overlapping. (**f**) Occlusion.

Figure 9. Visualization results of test set including different situations based on InternImage.



(**a**) Large-scale objects. (**b**) Normal condition. (**c**) Small-scale objects.

(**d**) Nighttime perspectives. (**e**) Overlapping. (**f**) Occlusion.

Figure 10. Visualization results of test set including different situations based on ICNet.

4. Discussion

To accomplish the goals of instance segmentation and pig counting, a dual-branch network and a well-prepared dataset are presented in this work. Count1200 is a dataset which is highly effective in counting and identification tasks owing to the advantages of instance segmentation. ICNet is a backbone specifically designed for segmentation and counting, consisting of a sequence of pipe layers and PDCLs which can enable the model to obtain multi-scale features. A dual branch structure of PDCB is used to design the PDCL using deformable convolution v3. This model benefits from the characteristics of long-range dependencies and spatial aggregation provided by DCN in conjunction with the idea of multi-group design, as shown in the experimental results table (Tables 1 and 3), we achieve the best performance in both instance segmentation and counting tasks. Moreover, the visual results, which showcase the seamless masks and highly accurate counting, further highlight the performance of our dataset and model.

Although the test results of our model are good, far better than those of the rest of the models, it still encounters a few issues. The masks we predict are correct and accurate in number, but there are white areas in the middle, failing to completely cover the entire pig. In addition, our method still has a very small number of false positives and missed detections. The problems above will be solved in future model enhancements.

5. Conclusions

In this paper, with the help of SAI, a well-curated dataset named Count1200 is created. Beyond that, leveraging the variable receptive field range of DCNv3 and the robust extraction capabilities of the dual-branch structure, we develop a network with a small parameter count and outstanding performance named ICNet. We make good use of the strengths of our dataset and model to achieve better performance in tasks of instance segmentation and pig counting. Through a series of repeatable comparative experiments, we validate that our model exhibits better segmentation and counting capabilities compared to other models. We intend to further enhance the model by considering both lightweight and high-precision aspects.

Author Contributions: Conceptualization, S.L. (Shanghao Liu); methodology, S.L. (Shanghao Liu); software, Y.C., R.W.; validation, X.L.; formal analysis, S.L. (Shanghao Liu); investigation, R.G.; resources, H.Z., S.L. (Shuqin Li); data curation, Q.L.; writing—original draft preparation, S.L. (Shanghao Liu), Y.C.; writing—review and editing, S.L. (Shanghao Liu), Y.C., R.G.; visualization, S.L. (Shanghao Liu); supervision, C.Z. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Special Program for Cultivating Outstanding Scientists of Beijing Academy of Agriculture and Forestry Sciences (JKZX202214), Shaanxi Key Industry

Innovation Chain Project (2023-ZDLNY-69), and Yangling Livestock Industry Innovation Center Double-chain Fusion Project (2022GD-TSLD-46).

Institutional Review Board Statement: Not applicable.

Data Availability Statement: The original contributions presented in the study are included in the article, further inquiries can be directed to the corresponding author/s.

Acknowledgments: We would like to express our gratitude to iFLYTEK for organizing the competition (https://challenge.xfyun.cn/topic/info?type=pig-check (accessed on 7 December 2023)). It was a great help in collecting online images for us.

Conflicts of Interest: The authors declare no conflicts of interest.

Abbreviations

The following abbreviations are used in this manuscript:

SAI	Semi-auto Instance Labeling Tool
SAM	Segment Anything Model
ICNet	Instances Counting Network
PDCL	Parallel Deformable Convolutions Layer
PDCB	Parallel Deformable Convolution Block
MHSA	Multi-head Self-attention
GUI	Graphical User Interface
ViT	Vision TransFormer

References

1. Neethirajan, S. Recent advances in wearable sensors for animal health management. *Sens. Bio-Sens. Res.* **2017**, *12*, 15–29. [CrossRef]
2. Zhang, T.; Liang, Y.; He, Z. Applying image recognition and counting to reserved live pigs statistics. *Comput. Appl. Softw.* **2016**, *33*, 173–178.
3. Schleppe, J.; Lachapelle, G.; Booker, C.; Pittman, T. Challenges in the design of a GNSS ear tag for feedlot cattle. *Comput. Electron. Agric.* **2010**, *70*, 84–95. [CrossRef]
4. Chen, S.; Wang, Q.; Chen, D.R. Effect of pleat shape on reverse pulsed-jet cleaning of filter cartridges. *Powder Technol.* **2017**, *305*, 1–11. [CrossRef]
5. Rahnemoonfar, M.; Sheppard, C. Deep count: Fruit counting based on deep simulated learning. *Sensors* **2017**, *17*, 905. [CrossRef]
6. Shen, Y.; Zhou, H.; Li, J.; Jian, F.; Jayas, D.S. Detection of stored-grain insects using deep learning. *Comput. Electron. Agric.* **2018**, *145*, 319–325. [CrossRef]
7. Mohanty, S.P.; Hughes, D.P.; Salathé, M. Using deep learning for image-based plant disease detection. *Front. Plant Sci.* **2016**, *7*, 1419. [CrossRef]
8. Sa, I.; Ge, Z.; Dayoub, F.; Upcroft, B.; Perez, T.; McCool, C. Deepfruits: A fruit detection system using deep neural networks. *Sensors* **2016**, *16*, 1222. [CrossRef]
9. Girshick, R.; Donahue, J.; Darrell, T.; Malik, J. Rich feature hierarchies for accurate object detection and semantic segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Columbus, OH, USA, 23–28 June 2014; pp. 580–587.
10. Wang, F.; Fu, X.; Duan, W.; Wang, B.; Li, H. Visual Detection of Lost Ear Tags in Breeding Pigs in a Production Environment Using the Enhanced Cascade Mask R-CNN. *Agriculture* **2023**, *13*, 2011. [CrossRef]
11. Feng, W.; Wang, K.; Zhou, S. An efficient neural network for pig counting and localization by density map estimation. *IEEE Access* **2023**, *11*, 81079–81091. [CrossRef]
12. Jiang, K.; Xie, T.; Yan, R.; Wen, X.; Li, D.; Jiang, H.; Jiang, N.; Feng, L.; Duan, X.; Wang, J. An attention mechanism-improved YOLOv7 object detection algorithm for hemp duck count estimation. *Agriculture* **2022**, *12*, 1659. [CrossRef]
13. Li, Y.; Zhang, X.; Chen, D. Csrnet: Dilated convolutional neural networks for understanding the highly congested scenes. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–22 June 2018; pp. 1091–1100.
14. Chen, G.; Shen, S.; Wen, L.; Luo, S.; Bo, L. Efficient pig counting in crowds with keypoints tracking and spatial-aware temporal response filtering. In Proceedings of the 2020 IEEE International Conference on Robotics and Automation (ICRA), Paris, France, 31 May–31 August 2020; IEEE: Piscataway, NJ, USA, 2020; pp. 10052–10058.
15. He, K.; Gkioxari, G.; Dollár, P.; Girshick, R. Mask r-cnn. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 2961–2969.

16. Lin, T.Y.; Dollár, P.; Girshick, R.; He, K.; Hariharan, B.; Belongie, S. Feature pyramid networks for object detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 2117–2125.
17. Wang, X.; Kong, T.; Shen, C.; Jiang, Y.; Li, L. Solo: Segmenting objects by locations. In Proceedings of the Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, 23–28 August 2020; Part XVIII 16; Springer: Berlin/Heidelberg, Germany, 2020; pp. 649–665.
18. Wang, X.; Zhang, R.; Kong, T.; Li, L.; Shen, C. Solov2: Dynamic and fast instance segmentation. *Adv. Neural Inf. Process. Syst.* **2020**, *33*, 17721–17732.
19. Cheng, B.; Schwing, A.; Kirillov, A. Per-pixel classification is not all you need for semantic segmentation. *Adv. Neural Inf. Process. Syst.* **2021**, *34*, 17864–17875.
20. Cheng, B.; Misra, I.; Schwing, A.G.; Kirillov, A.; Girdhar, R. Masked-attention mask transformer for universal image segmentation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, New Orleans, LA, USA, 19–20 June 2022; pp. 1290–1299.
21. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, Ł.; Polosukhin, I. Attention is all you need. *Adv. Neural Inf. Process. Syst.* **2017**, *30*, 1–11.
22. Kirillov, A.; Mintun, E.; Ravi, N.; Mao, H.; Rolland, C.; Gustafson, L.; Xiao, T.; Whitehead, S.; Berg, A.C.; Lo, W.Y.; et al. Segment anything. *arXiv* **2023**, arXiv:2304.02643.
23. Wang, W.; Dai, J.; Chen, Z.; Huang, Z.; Li, Z.; Zhu, X.; Hu, X.; Lu, T.; Lu, L.; Li, H.; et al. Internimage: Exploring large-scale vision foundation models with deformable convolutions. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Vancouver, BC, Canada, 18–22 June 2023; pp. 14408–14419.
24. Cheng, H.K.; Oh, S.W.; Price, B.; Schwing, A.; Lee, J.Y. Tracking anything with decoupled video segmentation. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Paris, France, 2–6 October 2023; pp. 1316–1326.
25. Lin, T.Y.; Maire, M.; Belongie, S.; Hays, J.; Perona, P.; Ramanan, D.; Dollár, P.; Zitnick, C.L. Microsoft coco: Common objects in context. In Proceedings of the Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, 6–12 September 2014; Part V 13; Springer: Berlin/Heidelberg, Germany, 2014; pp. 740–755.
26. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778.
27. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *Adv. Neural Inf. Process. Syst.* **2015**, *28*, 1–14. [CrossRef] [PubMed]
28. Ioffe, S.; Szegedy, C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In Proceedings of the International Conference on Machine Learning, PMLR, Lille, France, 7–9 July 2015; pp. 448–456.
29. Glorot, X.; Bordes, A.; Bengio, Y. Deep sparse rectifier neural networks. In Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics, JMLR Workshop and Conference Proceedings, Fort Lauderdale, FL, USA, 11–13 April 2011; pp. 315–323.
30. Ba, J.L.; Kiros, J.R.; Hinton, G.E. Layer normalization. *arXiv* **2016**, arXiv:1607.06450.
31. Larsson, G.; Maire, M.; Shakhnarovich, G. Fractalnet: Ultra-deep neural networks without residuals. *arXiv* **2016**, arXiv:1605.07648.
32. Srivastava, N.; Hinton, G.; Krizhevsky, A.; Sutskever, I.; Salakhutdinov, R. Dropout: A simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.* **2014**, *15*, 1929–1958.
33. Dai, J.; Qi, H.; Xiong, Y.; Li, Y.; Zhang, G.; Hu, H.; Wei, Y. Deformable convolutional networks. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 764–773.
34. Zhu, X.; Hu, H.; Lin, S.; Dai, J. Deformable convnets v2: More deformable, better results. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 9308–9316.
35. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. Imagenet classification with deep convolutional neural networks. *Adv. Neural Inf. Process. Syst.* **2012**, *25*, 1–9. [CrossRef]
36. Tangirala, B.; Bhandari, I.; Laszlo, D.; Gupta, D.K.; Thomas, R.M.; Arya, D. Livestock Monitoring with Transformer. *arXiv* **2021**, arXiv:2111.00801.
37. Loshchilov, I.; Hutter, F. Decoupled weight decay regularization. *arXiv* **2017**, arXiv:1711.05101.
38. Bao, H.; Dong, L.; Piao, S.; Wei, F. Beit: Bert pre-training of image transformers. *arXiv* **2021**, arXiv:2106.08254.
39. Goyal, P.; Dollár, P.; Girshick, R.; Noordhuis, P.; Wesolowski, L.; Kyrola, A.; Tulloch, A.; Jia, Y.; He, K. Accurate, large minibatch sgd: Training imagenet in 1 hour. *arXiv* **2017**, arXiv:1706.02677.
40. Tian, M.; Guo, H.; Chen, H.; Wang, Q.; Long, C.; Ma, Y. Automated pig counting using deep learning. *Comput. Electron. Agric.* **2019**, *163*, 104840. [CrossRef]
41. Dosovitskiy, A.; Beyer, L.; Kolesnikov, A.; Weissenborn, D.; Zhai, X.; Unterthiner, T.; Dehghani, M.; Minderer, M.; Heigold, G.; Gelly, S.; et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv* **2020**, arXiv:2010.11929.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.