Computers and Electronics in Agriculture 163 (2019) 104840

Contents lists available at ScienceDirect

Computers and Electronics in Agriculture

journal homepage: www.elsevier.com/locate/compag

Original papers

Automated pig counting using deep learning

Mengxiao Tian[a], Hao Guo[a,b,*], Hong Chen[a,*], Qing Wang[a], Chengjiang Long[c], Yuhao Ma[a]

[a] College of Information and Electrical Engineering, China Agricultural University, Beijing 100083, China
[b] College of Land Science and Technology, China Agricultural University, Beijing 100083, China
[c] Kitware Inc, Clifton Park, NY 12065, USA

ARTICLE INFO

Keywords:
Deep learning
Pigs counting
Automatic

ABSTRACT

Pig counting is one of the most critical topics in farming management and asset estimation. Due to its complexity, traditional agriculture method relies on manual counting, which is obviously inefficient and a waste of manpower. The challenging aspects like partial occlusion, overlapping and different perspectives even limit the usage of traditional computer vision techniques. In recent years, deep learning has become more and more popular for computer vision applications, because of its superior performance comparing to traditional methods. In this paper, we propose a deep learning solution to address the pig counting problem. We present a modified Counting Convolutional Neural Network (Counting CNN) model according to the structure of ResNeXt, and tune a series of experimental parameters. Our CNN model learns the mapping from the image feature to the density map, and obtains the total number of pigs in the entire image by integrating the density map. In order to validate the efficacy of our proposed method, we conduct experiments on a real-world dataset collected from actual piggery farming with 15 pigs in an image averagely. We achieve 1.67 Mean Absolute Error (MAE) per image and outperforms the competing algorithms, which strongly demonstrates that our proposed method can accurately estimate the number of pigs even if they are partially occluded in different perspectives. The detection speed, 42 ms per image, meets the requirements of agricultural application. We share our code and the first pig dataset we collected for pig counting at https://github.com/xixiareone/counting-pigs for livestock husbandry and science research community.

1. Introduction

Pig counting is a very important work in current large-scale agricultural production management and asset management in the piggery. Accurate pig counting can improve management in pigs feeding, piggery construction and etc, which can help farmers with cost reduction and unnecessary losses, and further make the farms more competitive.

However, it is challenging to count pigs accurately, due to pigs overlapping, variations in group density, camera perspective, and illumination changes, as illustrated in Fig. 1. Manual counting misses some pigs or adds extra pigs easily; it is time-consuming and expensive endeavor, and false-reporting and underreporting (Zhang et al., 2016a). These issues are common in large-scale breeding enterprises. Present computer vision techniques (Kashiha et al., 2013; Thanapongtharm et al., 2016) cannot solve the problems above effectively. They work only in a stable environment and after a complicated processing procedure.

Deep learning has been proven to be the most promising solution for objects counting in different environments. It is widely deployed in almost all fields of agriculture, such as object recognition (Zheng et al., 2018; Sladojevic et al., 2016; Picon et al., 2018), object classification (Amara et al., 2017; Park et al., 2018; Dyrmann et al., 2016), object detection (Mohanty et al., 2016; Sa et al., 2016; Shen et al., 2018). In the identification and counting of crops about fruits and leaves (Chen et al., 2017; Rahnemoonfar and Sheppard, 2017; Uzal et al., 2018). According to a recent survey of deep learning in agriculture (Kamilaris and Prenafeta-Bold, 2018), no research in quick and accurate counting livestock is mentioned.

In this paper, we propose a modified version of Counting Convolutional Nerual Network (Counting CNN) (Onoro-Rubio and Lopez-Sastre, 2016) in a fashion of end-to-end as a homogeneous, multi-branch architecture for pig counting. As shown in Fig. 2, we combine both Counting CNN and ResNeXt (Xie et al., 2016) in our deep learning architecture. Therefore, our proposed CNN model does not need to depend on foreground segmentation results since our model only takes appearance information in consideration.

Since there exist on public datasets available for this task, we collect an image dataset for pig counting from multiple websites and also

* Corresponding authors at: College of Information and Electrical Engineering, China Agricultural University, Beijing 100083, China (H. Guo, H. Chen).
E-mail addresses: guohaolys@cau.edu.cn (H. Guo), chenhong@cau.edu.cn (H. Chen).

https://doi.org/10.1016/j.compag.2019.05.049
Received 7 September 2018; Received in revised form 14 May 2019; Accepted 25 May 2019

Fig. 1. Pig images are from internet and real life. The challenging aspects for pig counting include (a) occlusion between targets and other obstructions, (b) overlapping among pigs, (c) illumination changes, (d) multiple perspectives.

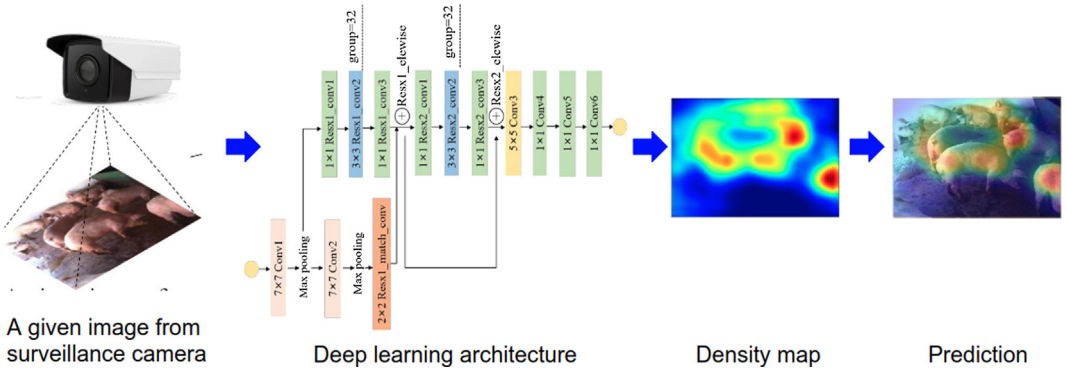


Fig. 2. The proposed framework for pig counting in real life. Given an image taken from surveillance camera, we feed it into our deep learning architecture to produce a density map, from which we are able to estimate the number of pigs appearing in the input image. As illustrated, different from the existing Counting Convolutional Neural Networks, our proposed deep learning architecture incorporates multiple kernel sizes in ResNeXt architecture, as well as skip connection.

include the real images captured from a real farm. The collected images contain different levels of crowding and overlapping, different scenes and perspectives, illumination changes and resolution. All the images are preprocessed and augmented. We crop multiple patches, with size 72×72 to make it fit model. When training is done, the model with best performance is picked and image patches are mapped to the corresponding density map. The result comes from the integration of the density map value. Comparing to traditional methods, our solution provides more robust performance especially when illumination changes and occlusion.

To the best of our knowledge, our dataset is the first one supporting pig counting with manual annotation provided for each image. Overall the contributions of our paper can be summarized in three folds:

- We collect a new herd dataset containing nearly 30,000 pigs to meet different scenarios and different perspectives for pig counting, which is available at https://github.com/xixiareone/counting-pigs.
- A modified Counting CNN network is proposed for automated pig counting, which significantly improves the accuracy and standard error of the counting. When compared to the original Counting CNN model (Onoro-Rubio and Lopez-Sastre, 2016), our model further reduces the mean error, from 3.23 to 2.78 in test data (from internet) and from 7.15 to 1.67 in test data (from real life).
- The detection speed of our model is 42 ms. This fulfils the requirements of agricultural application.

The rest of this paper is organized as follows: Section 2 describes the background and related work. Our proposed CNN model for pig counting is explained in Section 3. Section 4 provides the experimental results and discussion in detail. Conclusion with future research directions are shown in Section 5.

2. Related work

Some researches in agriculture (Hodgson et al., 2016; Sirmacek et al., 2012; Gemert et al., 2015) proposed drone or other aerial photography to count animal populations. Counting based on an aerial view is accurate, however aerial photography is only suitable in large farms that cover large areas. Since pigs are often concentrated in a small region that the width of the piggery is usually less than 15 meters, it is not practical for drones. Using knowledge of region growing (Liang et al., 2017) with a morphological algorithm (Zhang et al., 2016a), the objects counting by these traditional methods can count the number of pigs in a single pig house. However, the disadvantage of this method is that the image recognition heavily depends on the experience of researchers. The reason is that the objects in the image are morphologically diverse, and the images might suffer from illumination changes. These uncontrollable factors increase the inaccuracy when processing images with relatively dense targets.

Fish counting (Zhang et al., 2013) uses an adaptive thresholding segmentation method, which relied on the manual feature extraction. Threshold calculation errors occur when illumination varies so that fish cannot be counted accurately in complex environments. Our solution replaces the manual recognition of target individuals with extracting features. The advantages of deep learning technology in visual tasks is that there is no need for complex process of image and feature processing. It extracts image features automatically, and learns features automatically.

Compared with the traditional algorithm, deep learning has won big success in many fields including speech, natural language, visual tasks (Lecun et al., 2015). Counting with deep learning has recently became a widely used technology in artificial intelligence. Researches (Liu et al., 2017; Zhang et al., 2016b) described the use of a detector to identify multiple objectives and produced counts from the detection frame, but this method is only suitable for low-density and limited perspective scenes and does not perform well in situations with a lot of shading and overlapping objects, and the spatial distribution information provided is limited. In recent work, artificial neural networks were designed to estimate density (Zhang et al., 2016b; Zhang et al., 2017; Kumagai et al., 2017). Local density maps had been used to detect indistinctly or partially obscure targets (Ma et al., 2015), the detection of small animals (seagulls, fish, flies, and bees) is better than that of larger ones.

However, the application of this method is limited to small objects counting. Plus its complex procedures, this method is not suitable for pigs counting problems.

To solve this problem, based on Counting CNN model (Lempitsky and Zisserman, 2010; Onoro-Rubio and Lopez-Sastre, 2016), we design a modified Counting CNN model for automated pig counting in farms by incorporating the advantage of both Counting CNN and ResNeXt architectures, and the pre-trained parameters in some other network models are taken to fine-tune experimental parameters to guarantee performance.

3. Our approach

Our solution is to modify a CNN using Counting CNN and ResNeXt for pig counting, and both network architectures are CNN architectures. In the following subsections, we are going to review the Counting CNN and ResNeXt architectures before we discuss our proposed deep learning architecture.

3.1. Counting CNN and ResNeXt architecture

As shown in Fig. 3, Counting CNN is a regression model. It's a network that maps the extracted patches to the corresponding density map through learning, and a scale-aware model that converts image patches to object density maps without the need of perspectives.

Some of the existed research works reference the ResNeXt (Hitawala et al., 2018; Zhang et al., 2018; Han et al., 2018), and result in boosted accuracy with fixed number of parameters, so we combine Counting CNN with ResNeXt. As an improved version of ResNet (He et al., 2015), ResNeXt (Xie et al., 2016) is one of the best models in the public domain. It replaces ResNet's three-layer convolution block with a parallel stack of blocks with the same topology. Comparing to ResNet, ResNeXt performs better without adding model complexity in image classification, and it does not need to modify many parameters in other datasets and is more scalable.

3.2. The proposed deep learning architecture

As illustrated in Fig. 4, we proposed a new deep learning architecture, which combines Counting CNN model with the ResNeXt architecture (see Fig. 3) to make it more suitable for pig counting. It consists of thirteen convolutional layers. The first and second layers use kernel size 7×7 with a depth of 32, followed by max-pooling layers with stride equals 2 and 1 respectively. After that, it combines with an inception module used in ResNeXt to expand the convolutional layer of the model. The plus sign represents that the feature maps are summed up. The ResNeXt retains ResNet's stacking block and introduces group convolution. It takes the same convolution parameters, with fewer hyperparameters, resulting in improved generalizability and precision.

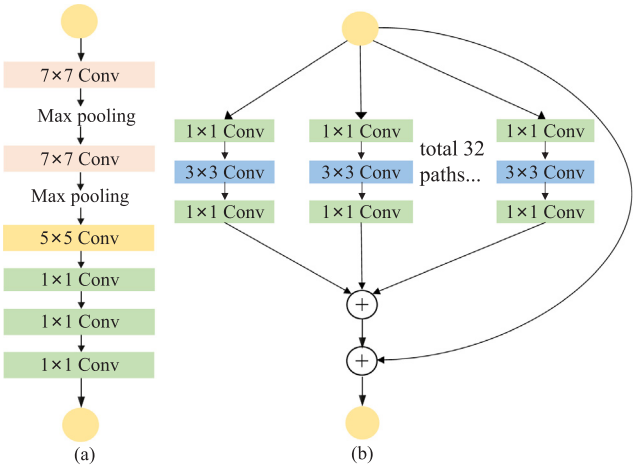


Fig. 3. (a) Counting CNN architecture; (b) ResNeXt architecture.

The first pooling layer is followed by 1×1 Resx1_conv1 layer, the second pooing layer is followed by 2×2 Resx1_match_conv layer, and we merge the two output feature maps from the Resx1_match_conv layer and Resx1_conv3 layer, and as the input to the next layer. From the Resx1_conv1 to Resx1_conv3 layer, and from the Resx2_conv1 to Resx2_conv3 layer, they are grouped convolutions with 32 groups, and concatenate input and output channels, and we change the output size of 128, 128, 256 in ResNeXt into 36, 36, 18 in Resx1_conv1~conv3 layer, and 18, 18, 18 in Resx2_conv1~conv3 layer. The stride is 2 in the Resx1_conv3 layer and Resx1_match_conv layer, and all other layers use stride 1.

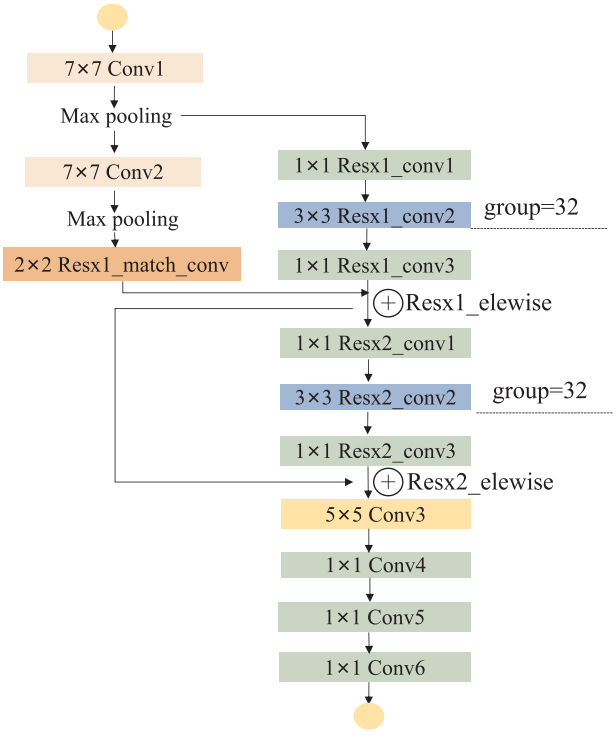
The following layer is 5×5 convolutional layer with a depth of 64. The last three convolutional layers have 1×1 filters with a depth of 1000, 400 and 1, respectively. These output feature map sizes are shown in the table in Fig. 4. Every convolutional layer of the entire structure is followed by a batch normalization (BN) layer (Kingma and Ba, 2014) which linearly transforms the input of each layer, keeping the normalized value distribution constant and achieving parameter regularization. This method reduces gradient dispersion, making the training process more robust. All the other layers are followed by rectified linear units (ReLU) except for the last layer introduced by Glorot et al. (2010). L2 regularization is used to increase the generalization performance of the model.

3.3. Counting from density map

In order to estimate the number of objects in an image, there are generally two methods: one is to input the images and output the estimated count; the other is to input the image, regress the distribution density map, and get the number of objects by summing the number of the density distribution. Counting CNN uses the second one. The reasons are as follows:

- Different degrees of perspective distortion, different postures and occlusion, resulting in counting the total number of direct regression images not accurate enough, because the group information provided is quite limited. For example, a large number of candidate windows need to be detected during the detection process, which reduces the efficiency of the algorithm and is not suitable for scenes with multi-perspective and multi-objective overlapping. In recent years, to our best knowledge, most of the researchers (Lempitsky and Zisserman, 2010; Zhang et al., 2016b; Kang et al., 2018) had adopted regression-based density maps, and calculated the total number of objects in the image by integrating the density map. This is because the density map contains more abundant spatial distribution information and can estimate the number of any area of the image. The density map gives the spatial distribution of objects in a given image relative to the total number of objects, which helps us better understand the scene information. It can be counted by spatial integration, and local area analysis can be performed to produce more accurate numbers based on the density map. It is also more suitable for any input image with different perspectives.
- As for images with different perspective, by learning from density map, CNN model could learn features with more semantic information so that counting accuracy is improved.

Therefore, following Counting CNN, our solution counts the number of pigs based on the density map algorithm, and the model needs to learn the density distribution of individuals in the image. Given the annotated position of each pig in image, the ground-truth density map is obtained by Gaussian kernel convolution (Lempitsky and Zisserman, 2010; Onoro-Rubio and Lopez-Sastre, 2016). We set the gradient of heat map according to the radius of the point. If two points intersect, the



Layer Name/Stride	Output Size
Input	3@72×72
7×7 Conv1/1	32@72×72
Max pooling/2	32@36×36
7×7 Conv2/1	32@36×36
Max pooling/1	32@36×36
1×1 Resx1_conv1/1	32@36×36
3×3 Resx1_conv2/1	32@36×36
1×1 Resx1_conv3/2	32@18×18
2×2 Resx1_match_conv/2	32@18×18
Resx1_elewise	32@18×18
1×1 Resx2_conv1/1	32@18×18
3×3 Resx2_conv2/1	32@18×18
1×1 Resx2_conv3/1	32@18×18
Resx2_elewise	32@18×18
5×5 Conv3/1	64@18×18
1×1 Conv4/1	1000@18×18
1×1 Conv5/1	400@18×18
1×1 Conv6/1	1@18×18

Fig. 4. Our proposed model architecture.

heat of this intersection area is superimposed. True value density distribution function is estimated by calculating kernel density as

$$D_I(p) = \sum_{\mu \in A_I} N(p;\mu, \Sigma), \tag{1}$$

where $A_I = \{A_1, \cdots, A_{C(I)}\}$, $\Sigma = \sigma^2 I_{2\times 2}$ and I is the image, A_I is a set of annotated 2D points of image I, $C(I)$ represents the number of target annotations in the I image, $N(p;\mu, \Sigma)$ is the evaluation of a normalized 2D Gaussian kernel, p is the position of a pixel, with the mean dot μ and isotropic covariance matrix Σ. The size of Σ is σ, which controls the smoothness of the 2D Gaussian kernel and we set $\sigma = 15$ in this paper.

According to the density map generated by Gaussian convolution, we sum up all the pixel values in the density map to get the final number as

$$S_I = \sum_{p \in I} D_I(p) \tag{2}$$

In this way, we can generate the density map for the original image, shown in Fig. 5 (①). Although there are overlapping among objects, all Gaussian is additive together, which keeps the total number of objects. Since the output density map of the model is down-sampled to 18 × 18, the true density map is also resized to 18 × 18.

To achieve accurate generation of density map in our model, we need to develop learning criteria for the neural network, which measures the distance between the density map and the truth density map at the training stage. According to the related research (Kang et al., 2018; Zhang et al., 2016b; Zhang and Shi, 2018), the Euclidean distance is mostly used to estimate the difference between the two density maps, the loss function is defined as

$$L(\theta) = \frac{1}{2N} \sum_{i=1}^{N} \|D(x_i;\theta) - D_i\|^2 \tag{3}$$

where θ represents parameters that can be learned from the network model, N is the number of training images, and x_i represents the input image, $D(x_i)$ is the model predicted density map, D_i represents the truth density map of the input image x_i, $L(\theta)$ is the loss between the estimated density map and the truth density map. As illustrated in Fig. 5 (②), the predicted density map and the truth density map carry on Back Propagation (BP) network optimizing the whole network structure by iterative training according to the loss function.

At testing stage, as shown in Fig. 5 (③), given a testing image, we extract patches by sliding the window, and feed them to our objects counting model. Due to the dense extraction, the image patches are overlapped, and it is possible that one pig exists in multiple patches. Finally, we average all predicted overlapped image patches and combine them to get the complete predicted distribution density map, which is the sum up of all the pixel values in the density map to obtain the total number of pigs. It is a commonly adopted method in present research (Boominathan et al., 2016; Han et al., 2017; Zhang et al., 2015).

3.4. Implementation details

We apply L2 regularization to avoid overfitting. We initial some parts of parameters with the pre-trained models, and optimize all the parameters using Adam with a learning rate of 0.001. We set the first and second order moment calculation as 0.9 and 0.99. The weights of each layer of the deep network are initialized with the Gaussian initializer (Krahenbuhl et al., 2015).

4. Experiment

We conduct experiments to verify the effectiveness of the proposed approach on our self collected dataset for pig counting. For measurement metrics, we use the Mean Absolute Error (MAE) and Root Mean Square Error (RMSE) to evaluate the performance of our proposed method. The MAE represents the average difference between the predicted result and the actual result, characterizes the accuracy of the algorithm. While the RMSE represents the degree of dispersion in the differences, and exams the robustness of models. In general, the smaller the MAE is, the higher accuracy of the estimated value is, and the smaller the RMSE is, the higher the robustness is. They are defined as

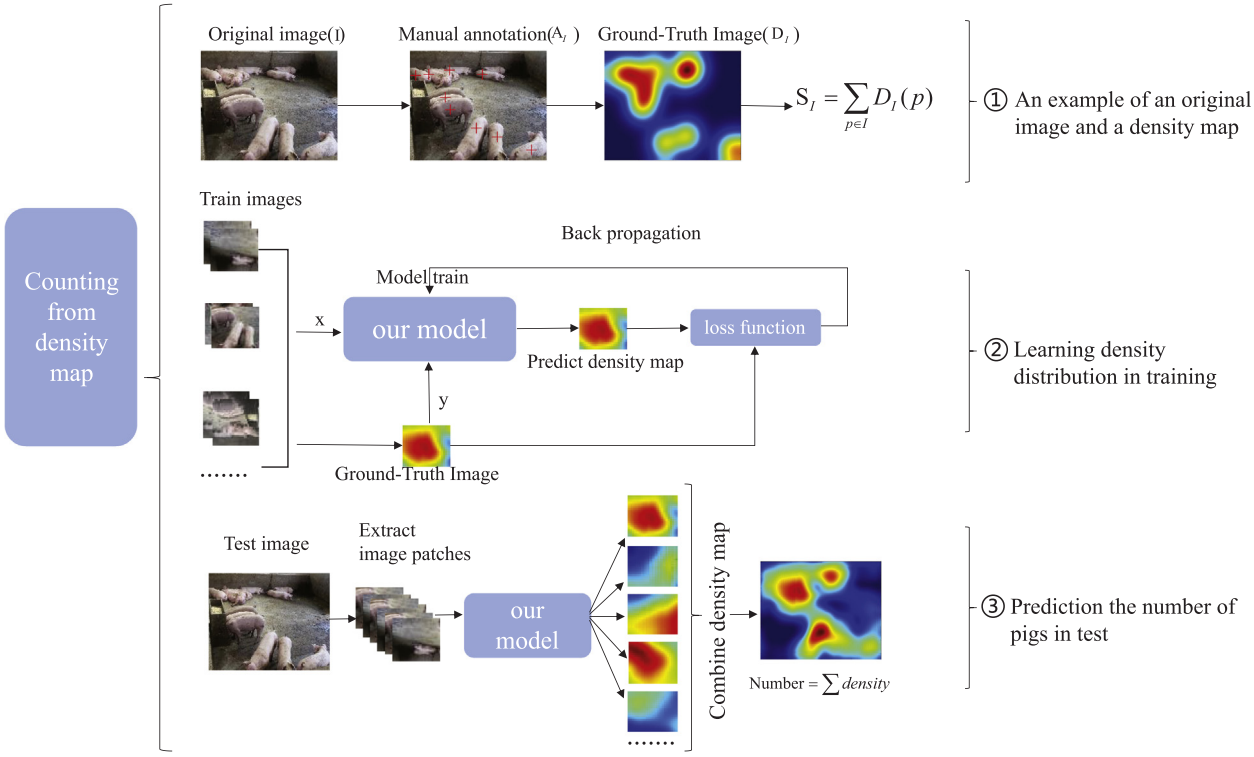


Fig. 5. Visualization of pig counting from density map.

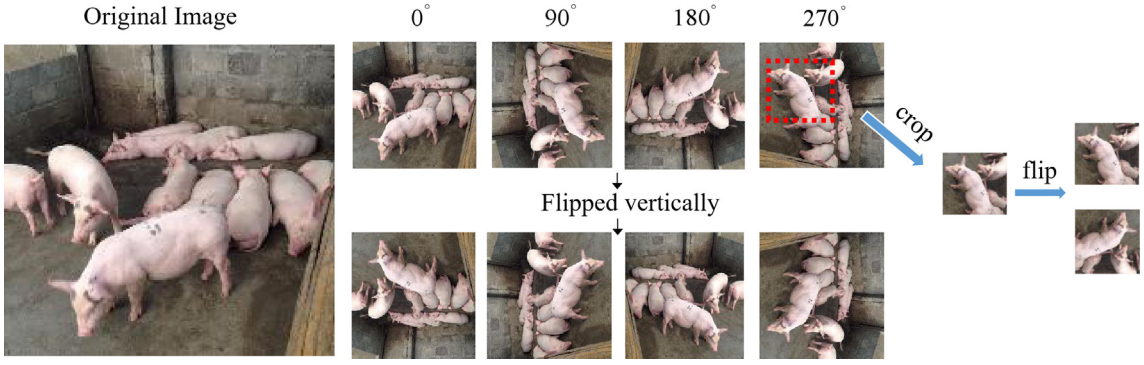


Fig. 6. Data augmentation by rotation, vertical flipping and cropping.

Table 1
Comparison between internet-derived and real-world pig datasets.

Dataset	Multi screen	Data format	Annotation format	Counting support	Average pig count per image	Range of pig count per image	Resolution
Internet data	✓	Image	Mark with dots	✓	10.5	3–40	High
Real-world data	✓	Video sequence and image	Mark with dots	✓	15.0	3–21	Low

Table 2
Number of images and patches for deep neural networks.

Usage	Number of images	Number of patches
Training datasets (internet)	1918	3 068 800
Validation datasets (internet)	581	929 600
Test dataset (internet)	485	N/A
Test dataset (real life)	417	N/A
Total	3401	3 998 400

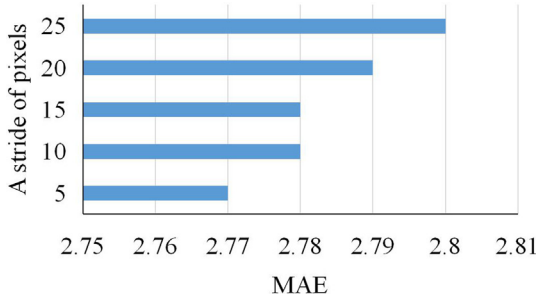


Fig. 7. Performance of pig counting in term of MAE with different stride values.

Table 3
Running times for a set of stride values.

Stride value (pixels)	5	10	15	20	25
Run time (seconds)	184	95	50	58	48

follow:

$$MAE = \frac{1}{N}\sum_{i=1}^{N}|y_i - \hat{y}_i| \quad (4)$$

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2} \quad (5)$$

where N is the number of test images, y_i represents the actual number of pigs in the ith image, and $\hat{y}_i$ represents the predicted number of pigs in the ith image.

4.1. Dataset

We collected some images from internet websites, containing different scenes and different perspectives, and different density distributions of pig population. To obtain annotations, we wrote a GUI-based annotating tool to obtain the (x, y) coordinates of center points, each of which represents one pig. Each image was annotated three times by ourselves separately, and the results of three annotations were cross-validated manually by us. If the annotated images similarity were within the predefined threshold, then we put them in dataset. The different annotated images were re-annotated, repeating it until more than five times, if still different, we discarded this image. In this way, we can control the annotation noise to some extents. Finally, we acquired the 373 valid images, but it is too small and easy to result in over-fitting, so data augmentation technique is introduced.

At the training stage, note that we applied data augmentation by rotating with 90, 180 and 270 degrees, vertical flipping and cropping (see Fig. 6). We extracted patches with the size of 72×72 from training image and validation images for the purpose of training. The data splitting information is summarized in Table 2. We got a pig herd dataset of total 2984 images by data augmentation, and randomly selected 1918 images as training, 581 images as validation and the resting 485 images as testing. To verify whether our model is suitable for practical applications, we also collected 417 images from a real farm for testing (see in Table 2). Among these real images, pigs are well distributed in each image. Table 1 summarizes the difference between the internet-derived and real-world datasets for pig counting. The data obtained from the internet contains only multi-perspective images, whereas the data observed from the farm includes not only the captured images but also video sequences, which contains pigs motion and tracking information. We collected images in perspective of overlooking to observe each object as much as possible, the number of pigs within per image ranges from 3 to 21.

Table 5
Effects of different network architectures on model performance.

Network architecture	Datasets	
	Validation (internet) MAE	Test (internet) MAE
Complete network	2.74	2.78
Without scale parameters in scale layer	2.94	2.93
Without BN layer	3.07	3.04

4.2. Choosing stride values for pig counting

A stride with an appropriate number of pixels must be set to scan the images and to combine the density sub-maps obtained into the density map as the estimated density map. We chose the stride value 5, 10, 15, 20, and 25 to run the experiments and the results are summarized in Fig. 7. We also provided the runtime of each stride value, to test and select the appropriate stride value in Table 3. As we can see, with the stride value 15, our approach can achieve MAE 2.78 and the runtime is 50 s.

4.3. Effectiveness of the proposed network architecture

To verify the effectiveness of our proposed network architecture, we design two baselines. One is to remove the scale parameters in scale layer, and the other one is to remove BN layer and scale layer in Table 4. From the table, “lr_mult” represents the coefficient of learning rate, and “decay_mult” represents the weight attenuation coefficient; “filler” represents the initialization for the learned scale parameter in scale layer, and “bias_filler” represents bias initialization in scale layer. We used the same data setting with the stride value 15 to train these networks and evaluated them on the same testing dataset. The results are summarized in Table 5.

As we can observe, adjusting the network structure does not greatly change the mean error of validation set or test set, but the lower mean error shows that our network combines with BN layer and scale parameters in scale layer, which improve the network learning ability and the performance of model obviously. According to the performance of the validation set, we chose early stopping training, and selected the model with best performance. Fig. 8 shows the learning curve of the training data in adjusted network architectures. The number of steps is the number of training iterations (a total of 50 000 steps), and the ordinate represents the Euclidean loss between the true density map and the predicted density map. As seen from Fig. 8, we used a simple moving average to smooth the curve, it shows that the adjusted complete network has faster converging time and smaller losses than the other two networks.

Table 4
Hyperparameters of the proposed network architectures.

Common hyperparameters	Network architecture	Network architecture hyperparameters		
		BN layer	Scale layer Without filler	Scale layer With filler
Base learning rate:0.0001 Learning rate policy:inv	Without BN layer and scale layer	–	–	–
Solver type:Adam Momentum:0.9 Weight decay:0.001	Without scale parameter in scale layer	lr_mult: 0 decay_mult: 0	lr_mult: 0.1 decay_mult: 0	–
Batch size:128 Regularization_type:L2	Complete network	lr_mult: 0 decay_mult: 0	lr_mult: 0.1 decay_mult: 0	filler: value = 1 bias_filler: value = 0

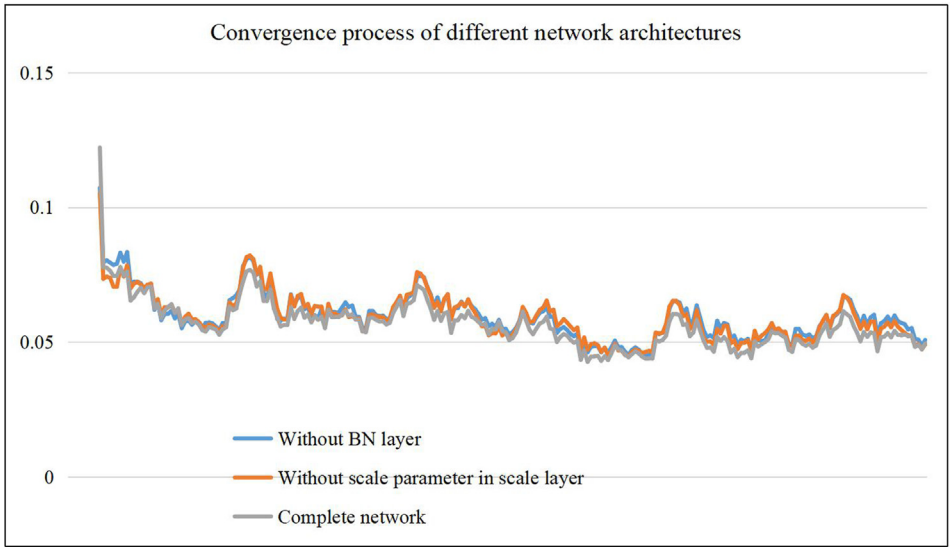


Fig. 8. Different network architectures of three instances in training.

Table 6
Summarization of competing methods.

Methods	Loss function	Prediction	Feature methods	Computer vision methods
Regression forest	Frobenius norm	Density patch map	Random regression	Traditional machine learning
Counting based on image processing	—	Crop image	Gradient map	Traditional image processing
Counting CNN	Euclidean loss	Density patch map	CNN	Deep learning

Table 7
Comparison of MAE and RMSE in methods (the result of our model is marked in bold).

Methods	Test images (internet)		Test images (Real life)	
	MAE	RMSE	MAE	RMSE
Counting based on image processing	4.79	6.48	6.16	6.90
Regression forest	2.90	3.79	2.44	2.96
Counting CNN	3.23	4.50	7.15	7.55
Our CNN model	**2.78**	**3.66**	**1.67**	**2.13**

Table 8
Average time of counting for each image (real-world).

Methods	Average time for one image (s)
Our CNN model	0.042
Counting based on image processing	0.025
Regression forest	0.183

4.4. Comparison of the state-of-the-art methods

We compared our proposed CNN model with three existing methods: (1) counting based on image processing, (2) regression forest method(Fiaschi et al., 2012), and (3) Counting CNN. Among these competing methods, the first two solutions take the advantages of traditional computer vision methods, the third one applies the deep learning technique. The last two methods and our model uses the integral function for calculation. To facilitate the readers to understand the competing methods, we summarize the description of each method in Table 6.

We evaluated all the competing methods on the test images collected from internet websites and from the real farm and summarized the performances in term of MAE and RMSE in Table 7. As we can see, the solution designed in this paper has achieved the best MAE and RMSE results in both test dataset (internet) and test dataset (real life), the mean error of the algorithm in the test dataset (internet) is 2.78, and in the test dataset (real life) is 1.67. Especially in real-world datasets, our solution is most effective, because that the test dataset (internet) is in different density distributions with different degrees of occlusion, and more complex environment and different perspectives. It worths mentioning that the distribution density of pigs in each image collected from the real farm is relatively homogeneous and fewer changes, and our solution is significantly better than other three algorithms. Compared to original Counting CNN model, the depth of our neural network is much deeper, and therefore able to extract more useful feature information, which makes our model fit better with the reality and get better estimates for the number of pigs.

Table 8 summarizes the average running time for pig counting in an image using different methods. From the table, we can observe that our CNN model is faster than the regression forest, slower than image processing, but has the lowest error. This is because the optimization process of our solution is simple and robust, while traditional image processing algorithm has challenging factors like illumination changes, occlusion and overlapping in scenes, which may cause some areas to be misidentified as pigs (such as drains, pig pens, trough). These factors increase the noise of the images and difficulty in recognition and incomplete counting. With high accuracy and low computation cost, our proposed solution meets the demand of agricultural technology.

To better explain why our proposed CNN model is able to outperform the competing algorithm, we compared prediction maps obtained by the competing methods in Fig. 9. As mentioned in Section 3, both ground truth and the prediction map for our proposed model is density map. The regression forest method is based on the density histogram. For counting based on image processing method, we use region growing method. Fig. 9 (e) and Fig. 9 (j) indicates that some background areas are misidentified as pigs. Fig. 9 (c) and Fig. 9 (h) show that our proposed model can also generate high-quality pig herd density map and count estimation, and can effectively distinguish between background

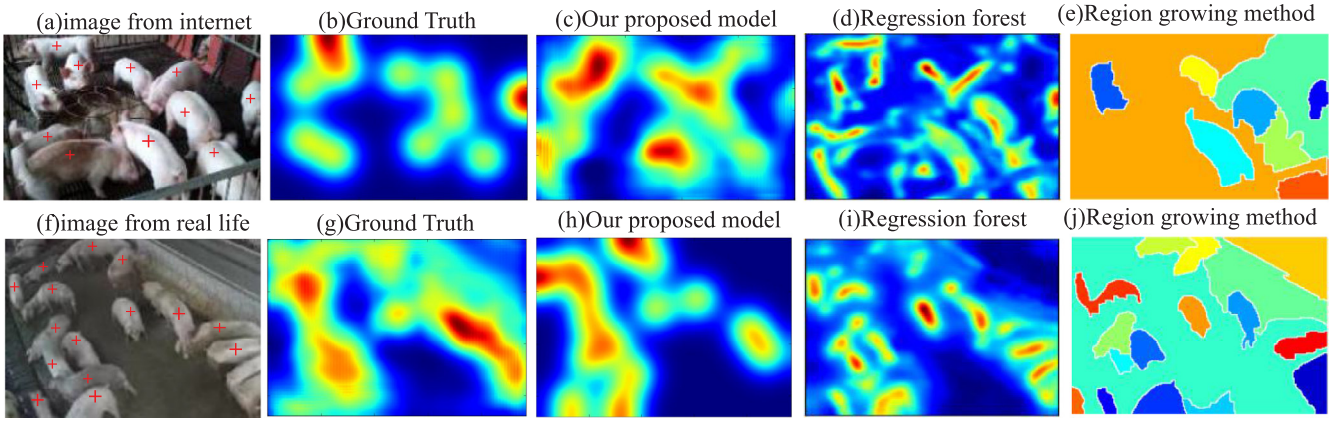


Fig. 9. The first four columns of instances in comparison between the regression forest and our model density prediction maps, the final column of instances for traditional image processing, (a) and (f) image dots marked by red cross are the ground-truth pigs. Except for images (e, j) in the final column, images of the top line are obtained from the internet and images of the last line are obtained from real life. The color of the density map is mainly represented by the density of the herd represented by this point. In the figure, the red region represents the region with larger pixel value that is the region with higher density value, and for the region with lower density, the color is blue. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

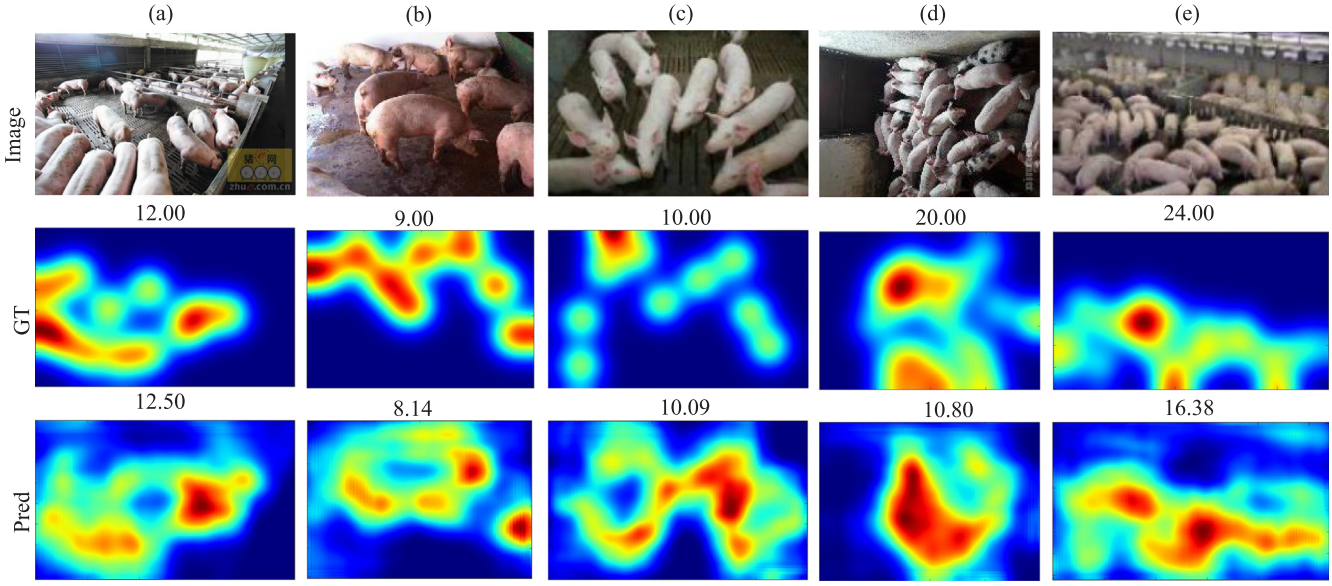


Fig. 10. Density map for test images, images are obtained from the internet. From top to bottom are testing images, GT and our prediction.

areas and pig herd goals, without pre-processing or human intervention.

4.5. Visualization

We visualize five representative images and their predictions and actual counts in two test datasets, as shown in Fig. 10 for the internet images and Fig. 11 for images collected from the real farm, respectively. The second row of each figure (GT) shows the true density map; the third row in each figure (Pred) shows the predicted density map, the actual and predicted pig counts are above these images.

For the internet images, the first three columns in Fig. 10(a–c) show that the difference between our predicted counts and actual counts is small. The results show that our proposed model accurately counts pigs regardless of the camera perspective. Fig. 10(d–e) shows a large difference between the true value and the model-predicted value because there are overcrowding and overlapping among pigs, which results in lower prediction accuracy.

Regarding real images from the farm, as shown in Fig. 11, the results in the fifth column are not as good as those in the first four columns due to the effect of the perspective. We observe that our model is successful and unaffected by changes in illumination, so the specific environment does not need to be controlled, and the model is more robust than the original model. Our solution is thus fully applicable to pig counting in the agricultural industry.

5. Conclusion

In this paper, a new solution for pig counting on the farm using deep learning is proposed. Our network is based on combination of Counting CNN and ResNeXt model to improve high-accuracy, low computational cost to high accuracy and high efficiency. The results demonstrate that in real-world data, our method gets a mean absolute error of 1.67, regardless of pigs with shadow, overlapping or different perspectives. This method could help to improve the management of current farms and agricultural production. However, for objects with greatly overlapped, with a decrease in counting accuracy. In future work, we will improve the counting accuracy in complex images by designing a multiscale input model, which uses different sizes of receptive fields. Besides, more images with complex situations will be captured in our future work.

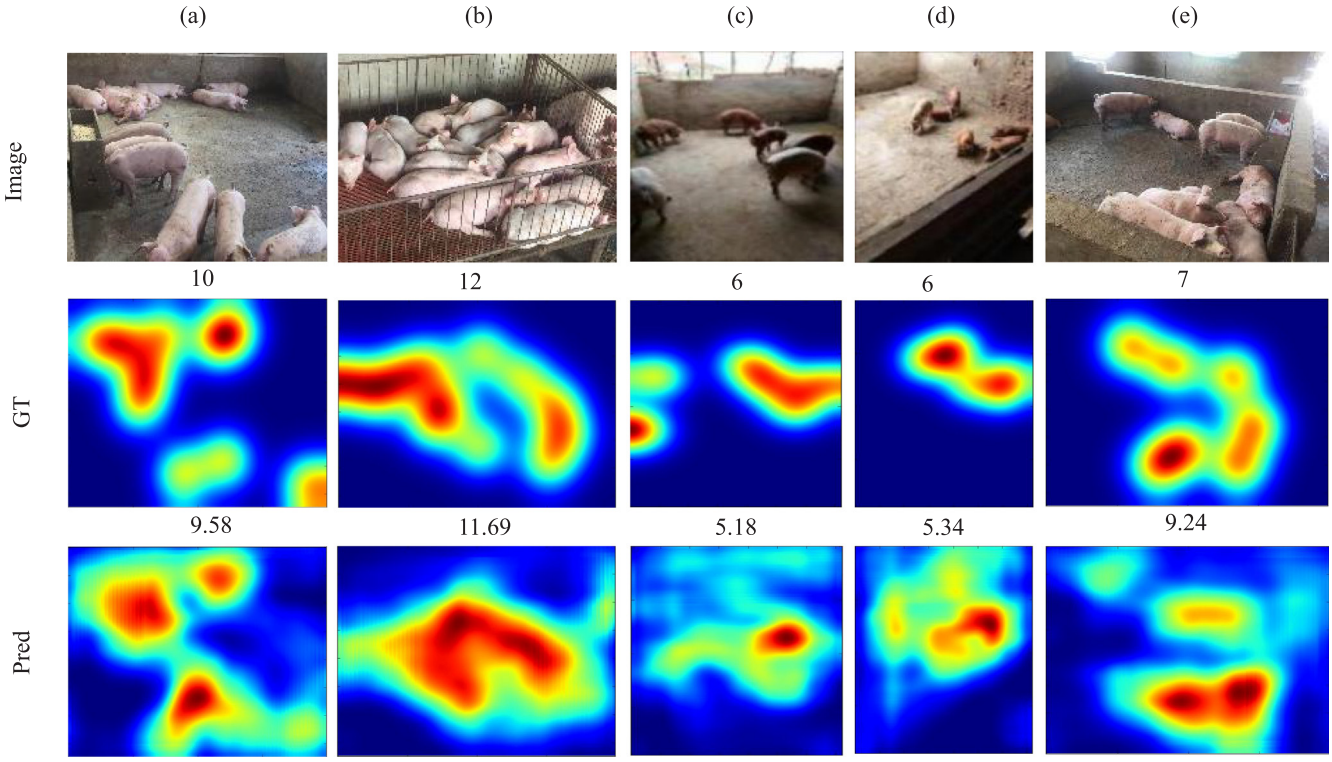


Fig. 11. Density map for test images, images are obtained from real life. From top to bottom are testing images, GT and our prediction.

Acknowledgement

This work was supported by National Natural Science Foundation of China [Grant No. 41601491]; the Fundamental Research Funds for the Central Universities [Grant No. 2019TC117]. Thanks to ShangDong WeiHai swine-breeding center of DA BEI NONG GROUP they provide us materials for experiment. We also thank Daniel and Roberto for making their code available (https://github.com/gramuah/ccnn) for our research to build project.

References

Amara, J., Bouaziz, B., Algergawy, A., Amara, J., Bouaziz, B., Algergawy, A., Amara, J., Bouaziz, B., Algergawy, A., Amara, J., 2017. A deep learning-based approach for banana leaf diseases classification. In: Datenbanksysteme Fur Business, Technologie Und Web, pp. 79–88.

Boominathan, L., Kruthiventi, S.S.S., Babu, R.V., 2016. Crowdnet: A deep convolutional network for dense crowd counting, pp. 640–644.

Chen, S.W., Skandan, S.S., Dcunha, S., Das, J., Okon, E., Qu, C., Taylor, C.J., Kumar, V., 2017. Counting apples and oranges with deep learning: a data driven approach. IEEE Robot. Automat. Lett. PP, 1.

Dyrmann, M., Karstoft, H., Midtiby, H.S., 2016. Plant species classification using deep convolutional neural network. Biosyst. Eng. 151, 72–80.

Fiaschi, L., Nair, R., Koethe, U., Hamprecht, F.A., 2012. Learning to count with regression forest and structured labels, pp. 2685–2688.

Gemert, J.C.V., Verschoor, C.R., Mettes, P., Epema, K., Lian, P.K., Wich, S., 2015. Nature conservation drones for automatic localization and counting of animals. In: European Conference on Computer Vision, pp. 249–259.

Glorot, X., Antoine, B., Bengio, Y., 2010. Deep sparse rectifier networks. Learn./Statist. Optim.

Han, K., Wan, W., Yao, H., Hou, L., 2017. Image crowd counting using convolutional neural network and markov random field. J. Adv. Computation. Intell. Intell. Inform. 21 (4), 632–638.

Han, W., Feng, R., Wang, L., Gao, L., 2018. Adaptive spatial-scale-aware deep convolutional neural network for high-resolution remote sensing imagery scene classification. In: IGARSS 2018–2018 IEEE International Geoscience and Remote Sensing Symposium. IEEE, pp. 4736–4739.

He, K., Zhang, X., Ren, S., Sun, J., 2015. Deep residual learning for image recognition, pp. 770–778.

Hitawala, S., Li, Y., Wang, X., Yang, D., 2018. Image super-resolution using vdsr-resnext and srcgan.

Hodgson, J.C., Baylis, S.M., Mott, R., Herrod, A., Clarke, R.H., 2016. Precision wildlife monitoring using unmanned aerial vehicles. Scient. Rep. 6, 22574.

Kamilaris, A., Prenafeta-Bold, F.X., 2018. Deep learning in agriculture: a survey. Comput. Electron. Agric. 147, 70–90.

Kang, D., Ma, Z., Chan, A.B., 2018. Beyond counting: comparisons of density maps for crowd analysis tasks – counting, detection, and tracking. IEEE Trans. Circuits Syst. Video Technol, PP, 1.

Kashiha, M., Bahr, C., Ott, S., Moons, C.P.H., Niewold, T.A., Berckmans, D., 2013. Automatic identification of marked pigs in a pen using image pattern recognition. Comput. Electron. Agric. 93, 111–120.

Kingma, D., Ba, J., 2014. Adam: A method for stochastic optimization. Comput. Sci.

Krahenbuhl, P., Doersch, C., Donahue, J., Darrell, T., 2015. Data-dependent initializations of convolutional neural networks. Comput. Sci.

Kumagai, S., Hotta, K., Kurita, T., 2017. Mixture of counting cnns: adaptive integration of cnns specialized to specific appearance for crowd counting.

Lecun, Y., Bengio, Y., Hinton, G., 2015. Deep learning. Nature 521, 436.

Lempitsky, V.S., Zisserman, A., 2010. Learning to count objects in images. In: International Conference on Neural Information Processing Systems, pp. 1324–1332.

Liang, Y., Zhang, T., Zhiyi, H.E., 2017. A remote image acquisition and target counting system for livestock farm. J. Guilin Univ. Electron. Technol.

Liu, J., Gao, C., Meng, D., Hauptmann, A.G., 2017. Decidenet: Counting varying density crowds through attention guided detection and density estimation.

Ma, Z., Yu, L., Chan, A.B., 2015. Small instance detection by integer programming on object density maps. Comput. Vis. Pattern Recogn. 3689–3697.

Mohanty, S.P., Hughes, D.P., Salathe, M., 2016. Using deep learning for image-based plant disease detection. Front. Plant Sci. 7.

Onoro-Rubio, D., Lopez-Sastre, R.J., 2016. Towards perspective-free object counting with deep learning. pp. 615–629.

Park, K., Hong, Y.K., Kim, G.H., Lee, J., 2018. Classification of apple leaf conditions in hyper-spectral images for diagnosis of marssonina blotch using mrmr and deep neural network. Comput. Electron. Agric. 148, 179–187.

Picon, A., Alvarez-Gila, A., Seitz, M., Ortiz-Barredo, A., Echazarra, J., Johannes, A., 2018. Deep convolutional neural networks for mobile capture device-based crop disease classification in the wild. Comput. Electron. Agric.

Rahnemoonfar, M., Sheppard, C., 2017. Deep count: fruit counting based on deep simulated learning. Sensors 17.

Sa, I., Ge, Z., Dayoub, F., Upcroft, B., Perez, T., Mccool, C., 2016. Deepfruits: A fruit detection system using deep neural networks. Sensors 16, 1222.

Shen, Y., Zhou, H., Li, J., Jian, F., Jayas, D.S., 2018. Detection of stored-grain insects using deep learning. Comput. Electron. Agric. 145, 319–325.

Sirmacek, B., Wegmann, M., Reinartz, P., Dech, S., 2012. Automatic population counts for improved wildlife management using aerial photography. In: Iemss, pp. 1–8.

Sladojevic, S., Arsenovic, M., Anderla, A., Culibrk, D., Stefanovic, D., 2016. Deep neural networks based recognition of plant diseases by leaf image classification. Comput. Intell. Neurosci. 1–11 (2016-6-22).

Thanapongtharm, W., Linard, C., Chinson, P., Kasemsuwan, S., Visser, M., Gaughan, A.E., Epprech, M., Robinson, T.P., Gilbert, M., 2016. Spatial analysis and characteristics of pig farming in thailand. Bmc Vet. Res. 12, 218.

Uzal, L.C., Grinblat, G.L., Namas, R., Larese, M.G., Bianchi, J.S., Morandi, E.N., Granitto,

P.M., 2018. Seed-per-pod estimation for plant breeding using deep learning. Comput. Electron. Agric. 150, 196–204.
Xie, S., Girshick, R., Dollar, P., Tu, Z., He, K., 2016. Aggregated residual transformations for deep neural networks. pp. 5987–5995.
Zhang, C., Li, H., Wang, X., Yang, X., 2015. Cross-scene crowd counting via deep convolutional neural networks. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 833–841.
Zhang, H.W., Yuan, G.L., Zhang, Y., Wen-Chi, Y.U., 2013. The method research of counting fish spawns based on image processing. Electron. Des. Eng. 27, 6754–6762.
Zhang, L., Shi, M., 2018. Crowd counting via scale-adaptive convolutional neural network. In: IEEE Winter Conference on Applications of Computer Vision, pp. 1113–1121.
Zhang, S., Wu, G., Costeira, J.P., Moura, J.M.F., 2017. Fcn-rlstm: Deep spatio-temporal neural networks for vehicle counting in city cameras. In: IEEE International Conference on Computer Vision, pp. 3687–3696.
Zhang, T., Liang, Y., He, Z., 2016a. Applying image recognition and counting to reserved live pigs statistics. Comput. Appl. Softw.
Zhang, Y., Zhou, D., Chen, S., Gao, S., Ma, Y., 2016b. Single-image crowd counting via multi-column convolutional neural network. Comput. Vis. Pattern Recogn. 589–597.
Zhang, Z., Zhang, X., Chao, P., Xue, X., Jian, S., 2018. Exfuse: Enhancing feature fusion for semantic segmentation.
Zheng, C., Zhu, X., Yang, X., Wang, L., Tu, S., Xue, Y., 2018. Automatic recognition of lactating sow postures from depth images by deep learning detector. Comput. Electron. Agric. 147, 51–63.