Fast Recognition of Pig Faces Based on Improved Yolov3

Guangbo Li¹, Jun Jiao²*,Guolong Shi¹, Huimin Ma¹, Lichuan Gu¹, Liang Tao¹

¹School of Information and Computer Science, Anhui Agricultural University, Hefei, Anhui, China
²email: jiaojun2000@ahau.edu.cn

Abstract. The traditional ear labels are easy to get off and cause infection in the intelligent managements of live pigs. Therefore, a noninvasive pig face recognition method based on improved YOLOv3 was designed to carry out simultaneous recognition of multiple live pigs. The YOLOv3 model was used to introduce the dense network into the Darknet53 feature extractor, forming a new backbone network in combination with down-sampling, and the improved SPP unit was added to the YOLOv3 model, constructing the YOLOv3_DB_SPP model. The pig face data set used in the test was divided into 10 categories. After data enhancement, the number of samples was 8 512, and the ratio of training set to test set was about 9:1. The test results showed that: 1) Under different classification probability thresholds, the mean average precision of the YOLOv3_DB_SPP model for detecting pig face data set was higher than that of the YOLOv3 model; 2) When the IOU threshold was 0.5 and the classification probability threshold was 0.1, the mean average precision of YOLOv3_DB_SPP model was 9.87% higher than that of the YOLOv3 model; 3) When detecting long-distance covered small target samples, the mean average precision was also higher in the YOLOv3_DB_SPP model. Thus, the YOLOv3_DB_SPP model improved the feature extraction ability of the basic feature extractor and the precision of the detector.

1. Introduction

In recent years, with the development of intelligent pig breeding industry, precise management of pigs has become particularly important, and individual pig recognition is the key to precision breeding. Traditional pig recognition methods include color marking, wearing radio frequency identification (RFID) ear tags and so on. Although color marking is intuitive, it is limited to occasions where the number of live pigs is small; wearing ear tags may cause problems such as damage and off-label of ear tags, and infection of pig ears with parasites.

Up to now, there are few studies on pig face recognition at home and abroad, mainly including the following: with Eigen space method, Naoki et al. [1] manually segmented features to achieve individual recognition of live pigs, and the recognition rate achieved 97.9% on the data sets of 16 categories; Hansen et al. [2] built a CNN model based on convolution, maximum pooling, and dense block, which also achieved a good pig face recognition effect; Qin Xing et al. [3] used bilinear convolutional neural network to extract live pig facial features, and conducted outer product integration to the features of different levels, finally forming individual features, of which the recognition accuracy reached 95.73% on the test image set. When solving the problem of pig face recognition, the above research mainly considers that a single individual is affected by factors such as illumination, posture and scenes; however, these methods are difficult to complete the recognition tasks when there are multiple individuals in the image with many other things.

With the extensive use of deep learning technology in the field of computer vision [4-6], deep convolutional neural networks have achieved fruitful results in the field of face recognition.
Theoretically, face recognition methods applied to non-intrusive scenes also have a certain reference value for pig face recognition. The research on deep learning technology in the field of face recognition mainly includes the following: the Deep-Face method proposed by Taigman et al. [7] improved face alignment and face recognition by training the deep convolutional neural networks on the largest facial dataset which involved 2 million facial images belonging to 4 000 identities, and the method reached an accuracy of 97.35% on the Labeled Faces in the Wild (LFW) dataset; the DeepID series proposed by Sun et al. [8-9] used 25 network models and considered both classification loss and verification loss in the network structure, which significantly improved the face recognition effect; the DeepID2 neural network structure proposed by Sun et al. [10] used face recognition signals to extract different facial features, which increased the inter-personal variations extracted from different identities, and also used the signals to extract the same face, which reduced the intra-personal variations, thereby learning the features with strong distinguishing ability; with a total of 27 network layers, the FaceNet network structure proposed by Florian et al. [11] introduced the Triplet loss function to extract the feature information of the face, making the learning ability of the model more efficient; Omkar et al. [12] introduced cross-layer connections in the network structure and used a large-scale data set of 2.6 million, which achieved good results on the LFW and YTF data sets.

Based on the above research, this study intended to introduce DenseBlock into the backbone network and improved SPP unit into the detector, in order to design a pig face recognition algorithm that could detect targets of different sizes on multiple scale feature maps and could realize multiple identity recognition tasks in a targeted manner.

2. Materials and methods

2.1. Sample collection
Live pig images were collected from Jinghuimeng Breeding Farm in Mengcheng, Anhui. Logitech C920 Pro camera was used as a real-time acquisition tool. The NanoPc-T4 demoboard was used to transfer the images collected by the USB connected camera to the I/O buffer of the board. The image data was compressed and packaged on the graphics computing unit of the demoboard, and then transmitted to the server via the Internet. In the test, the houses where the cameras were installed had sufficient illumination, and the camera collection device could be remotely controlled to collect the image information of the pig faces from different angles.

In order to ensure that the collected live pig images were continuous and clear enough to identify each pig face, the sampling time interval of the image acquisition device was set to 0.5 s. Too short a time interval would lead to high similarity or even repetition between the collected images, so the collected images needed to be filtered. The structural similarity index method (SSIM) was used to compare consecutively acquired images. After a series of comparison tests on the samples, 2 test samples with SSIM values <0.78 were selected. If SSIM was greater than or equal to 0.78, the test samples with lower serial number were selected, and the sample image was 1 920 pixels×1 080 pixels.

2.2. YOLOv3
Through the use of continuous 1×1 and 3×3 convolutions and residual modules, a 53-layer feature extractor Darknet53 was constructed in the YOLOv3 model, which contained 52 convolutional layers and a fully connected layer (Fig.1). In the meantime, the input picture was 256 pixels × 256 pixels. In the prediction, in order to make full use of the up-sampling features and fine-grained features in the feature map, the multi-scale prediction in the YOLOv3 model was used, which went as the following [13]: first remove the fully connected layer in the feature extraction structure on the YOLOv3 model, add several convolutional layer sets with nuclear sizes of 1×1 and 3×3 behind the last residual block, and make predictions with the last 1×1 convolutional layer, which was scale 1. Then perform an up-sampling operation on the feature map extracted by this scale, use element-wise to splice it with the 26×26 feature extracted from the last residual block, and put the spliced feature image behind the subsequent multi-layer convolution set for prediction, which was scale2. Finally, up-sample the features extracted in scale2 to 52×52, which was then combined with the features output from the residual block before the last to form a feature map with a size of 52×52, which was input to the
The YOLOv3 model provided 3 different sizes of prior boxes, which were clustered to 9 different sizes using K-means method according to different down-sampling scales. According to the distribution principle of large feature maps with small size boxes, the YOLOv3 model predicted the target score of each prior box using linear regression[14]. The network model assigned only 1 prior box to each ground truth. During training, the error sum of squares was used as the loss function of the frame prediction. The relationship between the prior and predict boxes on the feature map is shown in Fig.2. The center coordinates ($b_x$, $b_y$), width $b_w$ and height $b_h$ of the predict box were calculated as formula (1):

$$
\begin{align*}
  b_x &= \sigma(t_x) + C_x \\
  b_y &= \sigma(t_y) + C_y \\
  b_w &= p_w e^{t_w} \\
  b_h &= p_h e^{t_h}
\end{align*}
$$

Where, $\sigma()$ is the sigmoid function; $t_x$, $t_y$ are the center offset of the predict box; $C_x$, $C_y$ is the abscissa and ordinate of the target unit relative to the upper left corner of the feature map, respectively; $p_w$, $p_h$ respectively is the width and height of the prior box; $t_w$, $t_h$ is the scaling of the width and height of the predict box, respectively.

The YOLOv3 model classified each box with multiple labels, which replaced the softmax classification with a separate logistic regression classifier. In the training process, Binary Cross Entropy was used to predict the category.
2.3. DenseNet

DenseNet is a cross-connected convolutional neural network, and it includes dense blocks and transition layers for connecting the blocks. The advantage of DenseNet is that it can effectively solve the problem of gradient vanishing of deep neural networks, and reduce parameters while achieving feature reuse [15-16].

For a DenseNet with $L$ layers, the $l$th layer has $l$ inputs, and the feature map of the $l$th layer will be passed to the subsequent $L-l$ layers to realize feature reuse. The network has a total of $\frac{L(L+1)}{2}$ connections. The network connection mode is:

$$x_j = H_l([x_0, x_1, \ldots, x_{j-1}])$$ (2)

Where, $X_l$ is the output of the $l$th layer; $H_l(\cdot)$ is the batch normalization layer (BN Layer), a compound function composed of ReLU function and 3×3 convolutional layers.

3. YOLOv3_DB_SPP model

Due to the narrow breeding environment of the pig farm, the face of the pig is easily covered by the dirt or other pigs. A deep neural network is required to extract more characteristic features for recognition. However, due to the high time cost of labeling pig face sample images, it is impossible to obtain a large number of pig face samples in a short period of time to deal with the problem of vanishing gradients that may occur in deep neural networks. In this study, the YOLOv3_DB_SPP model was proposed based on the original YOLOv3 model. First, this model introduced DenseNet to the basic feature extractor. Secondly, in order to integrate multi-scale information without introducing too many parameters, an improved SPP unit was added after the backbone network to improve the recognition precision of detecting pig faces that were covered by other pigs or dirt.

3.1. Feature extractor

The feature extractor of YOLOv3_DB_SPP model included the Convolutional unit and DenseBlock module (Fig.3). The Convolutional unit was composed of BN Layer, 7×7 or 3×3 convolutional layer, and Leaky ReLU activation function (Fig.4). Among them, BN Layer was used for adaptive
reparameterization, which played a role in avoiding the deviation of network parameter distribution. It could alleviate the problems of deep network overfitting and gradient vanishing to a certain extent, and had a relatively small impact on parameter initialization. The practices of ResNet [17] and GoogLeNet [18] were referred to more fully retain the original image information and improve the detection precision of small targets. For the feature extractor, the nuclear size of the first convolutional layer was set to 7×7, and the first down-sampling was implemented in this layer, while the remaining 4 down-samplings in the 3×3 convolutional layer.

| Type     | Filters | Size     | Output |
|----------|---------|----------|--------|
| Convolutional | 64     | 7×7/2    | 208×208|
| DenseBlock1 |        |          |        |
| Convolutional | 128    | 3×3/2    | 104×104|
| DenseBlock2 |        |          |        |
| Convolutional | 256    | 3×3/2    | 52×52  |
| DenseBlock3 |        |          |        |
| Convolutional | 512    | 3×3/2    | 26×26  |
| DenseBlock4 |        |          |        |
| Convolutional | 1024   | 3×3/2    | 13×13  |
| DenseBlock5 |        |          |        |

**Fig. 3 Feature extractor**

The improved feature extractor replaced the original Residual unit with DenseBlock, in which the number of feature maps output by each convolutional layer was k. The value of k was much smaller than the residual module, making the network narrower. For the 5 DenseBlocks in the feature extractor (Fig.3), except for Denseblock1 (Fig.5a), the remaining 4 DenseBlocks had the same level (Fig.5b). The nonlinear conversion $H_{i}$ was activated by BN Layer, 1×1 convolutional layer, Leaky ReLU activation function and the convolutional layer with the size of 3×3. At the same time, the Leaky ReLU function was used to replace the ReLU function used in the original DenseNet network to prevent the negative value of the parameter from being set to 0, which made the neuron to fail to learn. Before the 3×3 convolutional layer, a 1×1 convolution was still added as the bottleneck layer, which was used to reduce the dimension of the features input to the 3×3 convolutional layer and reduce the amount of calculation.
In DenseBlock1, the number of output channels of each convolutional layer was set to 32, using a 3-layer structure. After the repeated use of feature by DenseBlock1, the levels for the subsequent DenseBlock2, 3, and 4 were deepened, and the number of output channels of DenseBlock2, 3 were reduced. The $k$ values of these 3 dense blocks were set to 16, 16, 32, and each contained 8 convolutional layers. In order to output richer characteristic information, a larger number of channels of 64 were given to the last dense block, which adopted the 4-layer structure. Since there were more convolutional layers in DenseBlock than in Residual, the number of superimposed uses was reduced to 1, 2, 4, 2, and 1, respectively, so as to avoid the network level being too deep caused by the overuse, which could affect the efficiency of the model. The feature extractor of YOLOv3_DB_SPP model is shown in Fig. 6.
### 3.2. Detector

The detector of model YOLOv3_DB_SPP added a Convolutional Set after the feature extractor, which included 3 sets of 1×1 and 3×3 convolutional layers, and then an improved SPP unit was introduced behind the unit (Fig.7). The improved SPP unit performed maximum pooling on the feature maps extracted by the convolutional layer with sizes of 7×7, 5×5 and 3×3. In order to keep the input and output sizes consistent, padding was performed on the output of the feature extractor before the pooling operation, and the pooling step was set to 1.

![Fig. 6 The structure of YOLOv3_DB_SPP](image)

![Fig. 7 Improved SPP unit](image)
The improved SPP unit performed *concat* operations on the pooling results of 3 different scales. Multiple maximum pooling operations retained the most responsive features at different scales. The features extracted by the convolutional layer were retained using the output of the splicing feature extractor and the output of the improved SPP unit, and then the features were input into the 1×1 convolutional layer of the first-scale detector after processing by a 3×3 convolutional layer, realizing the detection on the feature map with a size of 13×13.

Compared with the traditional SPP, the improved SPP unit merged the channels of the input feature map with the pooled feature map, so that more features were captured, which greatly improved the recognition precision of large and general targets. Among them, the smaller-scale pooling was used to extract representative features of long-distance small targets.

The second scale of the network performed up-sampling on the feature map obtained by splicing the output of the improved SPP unit and the output of the backbone network, and the output feature map size was 26×26. Unlike the YOLOv3 model, the YOLOv3_DB_SPP model had a large number of dense blocks. If the 26×26 feature map obtained by upsampling was still spliced with the feature map output by DenseBlock4, the number of output channels would become very large, and performing 1×1 dimensionality reduction here may lose some features, so in the second scale, the output of the fourth down-sampling layer size of 26×26 was used for splicing with it. After the Convolutional Set and 3×3 convolution, the 1×1 convolutional layer was input to complete the second detection. The last scale repeated the operation of the second scale, and the output of the third down-sampling layer was spliced with the up-sampling feature of the second-scale spliced feature. After a series of 1×1 and 3×3 convolutions, the prediction was completed on the last 1×1 convolution.

4. Test data set

In the test, the input of the network model was RGD color pig face images of 416 pixel × 416 pixel, and there were 2 128 images in the original data set. A total of 8 512 sample images were generated after horizontal flipping, random cropping, mirror flipping and random shifting, and the ratio of training set to test set samples was approximately 9:1.

The 10 live pig identities included in the data set used for the test were individually numbered as pig1, pig2,..., pig10. labelImg was used to label the boxes manually and assign the label name, and the interface is shown in Fig.8. The XML file generated by labelImg contained information such as the size of the sample image and the coordinates of the upper left and lower right corners of the sample box.
The operating system used in this test was Ubuntu 18.04.3 with the CPU of Intel i5 9400F 2.9 GHz, memory of 64 G, GPU of NVIDIA GEFORCE RTX 2080Ti, and video memory of 32 G; the deep learning framework was Tensorflow, and the version number was 1.13.0.

The test was divided into 3 stages, which trained 15, 25, 40 epoch, respectively. The batch size was 32. The maximum learning rate was set as $lr_{max}$, the minimum learning rate as $lr_{min}$, and the number of iteration steps referred to the number of batches. The loss value may be NaN in the initial stage of network training, and in order to avoid this problem, the learning rate of the first stage was set to the product of $lr_{max}$ and the ratio of the current iteration steps to the total iteration steps in the first stage, which was used for stabilizing training, and the learning rate gradually increased to $lr_{max}$. The parameters of all layers outside the detector were frozen and removed from the detector at both the first and second stages, which was used to train only the detector layer. The initial parameters of the frozen layer were obtained by training on the PASCAL VOC2007 data set using the model, and the final learning rate of the second stage was the median value of $lr_{max}$ and $lr_{min}$. In the last stage, the whole network was trained, and the final learning rate was $lr_{min}$.

Since the improved YOLOv3_DB_SPP model and the YOLOv3 model were different in the number of convolutional layers, parameters and structure, and thus, during training, $lr_{max}$ and $lr_{min}$ of YOLOv3 and YOLOv3_DB were set to $10^{-4}$, $10^{-6}$, and $10^{-5}$, $10^{-7}$ according to the convergence of the previous test model.

The loss function of the network model consisted of three parts. The target category loss $L_{ch}$ and confidence loss $L_{conf}$ used the binary cross-entropy function, and the positioning loss $L_{loc}$ used the square sum of the difference between the predicted offset and the true offset as the loss function. The sum of the three was the final loss total loss, and details are as follows:

$$L_{ch} = -\sum_{i=0}^{k} \sum_{j=0}^{n} I_{obj} \left[ p_i(c) \log(\hat{p}_i(c)) + (1 - p_i(c)) \log(1 - \hat{p}_i(c)) \right]$$ (3)

$$L_{conf} = -\sum_{i=0}^{k} \sum_{j=0}^{n} I_{obj} \left[ C_i \log(\hat{C}_i) + (1 - C_i) \log(1 - \hat{C}_i) \right] - \sum_{i=0}^{k} \sum_{j=0}^{n} I_{obj} \left[ C_i \log(\hat{C}_i) + (1 - C_i) \log(1 - \hat{C}_i) \right]$$ (4)
\[ L_{\text{loc}} = \sum_{i=0}^{S^2} \sum_{j=0}^{B} I_{ij} \left( 2 - w_i \times h_i \right) \left[ \left( x_i - \hat{x}_i \right)^2 + \left( y_i - \hat{y}_i \right)^2 \right] + \sum_{i=0}^{S^2} \sum_{j=0}^{B} I_{ij} \left( 2 - w_i \times h_i \right) \left[ \left( w_i - \hat{w}_i \right)^2 + \left( h_i - \hat{h}_i \right)^2 \right] \tag{5} \]

\[ \text{total_loss} = L_{\text{class}} + L_{\text{conf}} + L_{\text{loc}} \tag{6} \]

Where, \( S \) means dividing the image into the grid of \( S \times S \); \( B \) is the number of boxes predicted for each grid unit; \( C \) is the confidence value; \( \hat{C} \) is the output of the predicted confidence value after inputting into the Sigmoid function; \( c \) is the category; \( P_i(c) \) is the probability of the target in the \( i^{th} \) grid belonging to category \( c \); \( \hat{P}_i(c) \) is the output after inputting the predicted probability into function Sigmoid; \( I_{ij} \) values 1, 0 respectively according to whether the \( i^{th} \) grid unit has a target in the \( j^{th} \) box, while the value of \( I_{ij}^{\text{no}} \) takes the opposite; \( x_i, y_i, w_i, h_i \) are the center coordinates, width, and height of the ground truth box; \( \hat{x}_i, \hat{y}_i, \hat{w}_i, \hat{h}_i \) are the center coordinates, width, and height of the predict box.

6. Test Results and Analysis
In the test, YOLOv3, YOLOv3_DB and YOLOv3_DB_SPP were used to train the live pig data set. After 20 315 iteration steps, the total loss of the three models all converged to around 0.15. As shown in Fig.9, the training time of the three models was all around 3.0-3.5 h.
In the test, IOU threshold and classification probability threshold \( T \) were used as the conditions of the evaluation model, where the classification probability referred to the probability value predicted when the model classified the detection target. Table 1 shows the detection results of the three models on the pig face dataset when the IOU threshold was 0.5 and the classification probability threshold \( T=0.1 \), including the mean Average Precision (mAP) and the Average Precision (AP). The YOLOv3_DB model was the detector without an improved SPP unit. As shown in Table 1, the mAP values of the YOLOv3_DB and the YOLOv3_DB_SPP were higher than those of the YOLOv3, and the latter had a significant improvement over the former, indicating that the method proposed in this study had a significant effect on improving the detection precision. Although the AP value of YOLOv3_DB improved on the first 7 types of samples, the detection results were not good enough for pig8, pig9, and pig10, which were live pig samples reared in captivity, and compared with the first 2 models, YOLOv3_DB_SPP not only showed better detection results for the first 7 types of samples, but also had better detection results for the samples read in captivity, indicating that the improved SPP unit was helpful for small target objects with longer detection distances and more coverage.

Table 1 Results of the three models detected on pig dataset while \( T=0.1 \) //%

| Model            | Average precision | Mean average precision |
|------------------|-------------------|------------------------|
|                  | Pig1  | Pig2  | Pig3  | Pig4  | Pig5  | Pig6  | Pig7  | Pig8  | Pig9  | Pig10 |            |
| YOLOv3           | 66.52 | 76.37 | 73.86 | 75.27 | 78.84 | 93.49 | 83.25 | 97.52 | 79.47 | 78.46 | 80.31      |
| YOLOv3_DB        | 79.19 | 89.68 | 85.96 | 83.08 | 77.95 | 95.03 | 93.48 | 80.18 | 70.89 | 72.39 | 82.87      |
| YOLOv3_DB_SPP    | 86.01 | 92.09 | 91.47 | 89.39 | 81.52 | 97.83 | 95.29 | 98.84 | 79.85 | 89.49 | 90.18      |

In order to further illustrate the performance of the model, Table 2 shows the mAP values and detection speed of the three models under different classification probability thresholds when detecting the pig face data set, when the IOU threshold was still 0.5. As shown in Table 2, with the continuous increase of the classification probability threshold, the mAP of model YOLOv3 dropped significantly, while the results of the improved 2 models were relatively less affected. The results showed that the detection results of the YOLOv3 model were generally low-probability. Once the threshold increased, results with low probability were eliminated, and thus mAP decreased accordingly. And when performing target detection on these 10 types of samples, the mAP of the improved network model improved to different degrees when compared to model YOLOv3, but the detection speed of YOLOv3_DB and YOLOv3_DB_SPP was inferior to model YOLOv3.
Tab. 2 mAP and speed of the three models detected on pig dataset with various threshold T

| Model          | Mean average precision /% | Speed/frames/s |
|----------------|----------------------------|----------------|
|                | \(T=0.1\)                | \(T=0.2\)    | \(T=0.3\)    | \(T=0.4\)    | \(T=0.5\)    |
| YOLOv3         | 80.31                      | 69.54         | 61.69         | 52.34         | 37.85         | 65             |
| YOLOv3_DB      | 82.76                      | 82.38         | 81.73         | 79.68         | 68.34         | 60             |
| YOLOv3_DB_SPP  | 90.18                      | 89.62         | 82.37         | 82.23         | 78.64         | 56             |

Fig. 10 Results of YOLOv3 (a), YOLOv3_DB (b) and YOLOv3_DB_SPP (c) detected on the image of pigs

As shown in Fig.10, compared with model YOLOv3, model YOLOv3_DB could detect the live pigs covered with shadows in the bottom right corner. Compared with the other 2 models, model YOLOv3_DB_SPP realized the detection of small samples with much coverage in the corner.

Fig. 11 Predict boxes of YOLOv3 (a), YOLOv3_DB (b) and YOLOv3_DB_SPP (c) detected on the image of pigs

As shown in Fig.11, compared with model YOLOv3, the detecting target of model YOLOv3_DB was more complete. Compared with the other 2 models, model YOLOv3_DB_SPP, could detect small targets with more coverage in longer distance, but the positioning of boxes with small samples were not precise enough.

7. Conclusion

In this study, DenseBlock was introduced into the feature extractor Darknet53 of model YOLOv3, obtaining the detection model YOLOv3_DB, and combined with the improved SPP unit, a new pig face recognition model YOLOv3_DB_SPP was proposed and applied to pig recognition. When the IOU threshold was 0.5 and the classification probability threshold was 0.1, the mean average precision of model YOLOv3_DB was 82.76% when recognizing the pig face dataset, and the recognition speed was 60 frame/s. Compared with model YOLOv3, its mean average precision was 2.76% higher, while the speed difference was only 5 frame/s; when using model YOLOv3_DB_SPP to detect the pig face dataset, the mean average precision was 90.18%, and the detection speed was 56 frame/s. Compared with model YOLOv3, its mean average precision was 9.87% higher, while the speed was 9 frame/s.
slower. The test results showed that the YOLOv3_DB_SPP model proposed in this study improved the recognition precision and realized the recognition of long-distance covered small targets with little loss of recognition speed, but the positioning precision of the predict box still needed to be improved.

Acknowledgments
This work was supported by the National Natural Science Foundation of China (No.31671589); Major scientific and technological projects in Anhui Province, China (No:201903a06020009), the Key Research and Development Program of Anhui Province, China(No:1804a07020130); the Key Research and Development Program of Hefei City, China (J2018G14), and the Open Fund of the Key Agriculture E-Commerce Laboratory, Ministry of Agricultural and Rural Affairs, China (AEC2018010, AEC2018003).

About the author: Guangbo Li, Anhui Agricultural University, email: 2711036892@qq.com

About the author: Lichuan GU(1974-), Male, P.R.China, Professor, Doctoral supervisor, Devoted to research about big data.

Corresponding author: Jun JIAO,(1964-), male, P.R.China, professor, master supervisor, devoted to research about robot control.

References
[1] Naoki W, Mikio S and Michio S 2013 Pig face recognition using eigenspace method ITE Transactions on Media Technology and Applications 1(4) pp 328-332
[2] Hansen M F, Smith M L, Smith L N, Salter M G, Baxter E M, Farish M and Grieve B 2018 Towards on-farm pig face recognition using convolutional neural networks Computers in Industry 98 pp 145-152
[3] Qin X and Song G F 2019 Pig face recognition algorithm based on bilinear convolution neural network Journal of Hangzhou Dianzi University 39(2) pp 12-17 (in Chinese)
[4] Jiao J, Wang W Z, Hou J B, et al. 2019 Freshness identification of iberico pork based on improved residual network Transactions of the Chinese Society for Agricultural Machinery 35(2) pp 364-371 (in Chinese)
[5] Qin F, Liu D X, Sun B D, et al. 2017 Image recognition of four different alfalfa leaf diseases based on deep learning and support vector machine Journal of China Agricultural University 22(7) pp 123-133 (in Chinese)
[6] Fu L S, Song Z Z, Zhang X, et al. 2020 Applications and research progress of deep learning in agriculture Journal of China Agricultural University 25(2) pp 105-120 (in Chinese)
[7] Taigman Y, Yang M, Ranzato M, et al. 2014 DeepFace: closing the gap to human-level performance in face verification 2014 IEEE Conference on Computer Vision and Pattern Recognition (IEEE) pp 1701-1708
[8] Sun Y, Wang X G and Tang X O 2014 Deep learning face representation from predicting 10,000 classes 2014 IEEE Conference on Computer Vision and Pattern Recognition. (IEEE) pp 1891-1898
[9] Sun Y, Liang D, Wang X G and Tang X O 2015 DeepID3: face recognition with very deep neural networks Computer Science - Computer Vision and Pattern Recognition
[10] Sun Y, Wang X G and Tang X O 2014 Deep learning face representation by joint identification-verification Advances in neural information processing systems (The MIT Press) pp 1988-1996
[11] Florian S, Dmitry K and James P 2015 FaceNet: a unified embedding for face recognition and clustering IEEE Conference on Computer Vision and Pattern Recognition (IEEE) pp 815-823
[12] Omkar P M, Andrea V and Andrew Z 2015 Deep face recognition British machine vision conference (BMVA Press)
[13] Wang K, Jiang W H, Lv Y, et al. 2019 Real-time detection method for cage laying hens behavior based on deep learning Journal of China Agricultural University 24(11) pp 123-133 (in Chinese)
[14] Sun Y C, Pan S G, Zhao T, Gao W and Wei J S 2020 Traffic light detection based on YOLOv3
optimization algorithm Acta Optica sinica (in Chinese)

[15] Jiang B, Tu W X, Yang C, et al. 2019 Semantic segmentation method for complex traffic scene based on DenseNet Pattern Recognition and Artificial Intelligence 32(5) pp 472-480 (in Chinese)

[16] Shi Y X and Xv X Q 2019 Improved DesenNet network for human pose estimation Control and Decision 36(5) pp 1206-1212(in Chinese)

[17] He K M, Zhang X Y, Ren S Q and Sun J 2016 Deep residual learning for image recognition 2016 IEEE Conference on Computer Vision and Pattern Recognition (Las Vegas) (IEEE) pp 770-778

[18] Szegedy C, Liu W, Jia Y Q, Sermanet P, Reed S, Anguelov D, Erhan D, Vanhoucke V, Rabinovich A 2015 Going deeper with convolutions 2015 IEEE Conference on Computer Vision and Pattern Recognition (Boston) (IEEE) pp 1-9