JCSE Journal of Computing Science and Engineering

Regular Paper

Journal of Computing Science and Engineering,
Vol. 17, No. 4, December 2023, pp. 145-160

Research on Spider Fine-Grained Recognition Technology Based on Transfer Learning

Jianming Wang*
School of Mathematics and Computer Science, Dali University, Dali, China;
Yunnan Provincial Key Laboratory of Entomological Biopharmaceutical R&D, Dali University, Dali, China
wjm@dali.edu.cn

Longfeng Deng, Chenyang Shi, and Guosheng Ye
School of Mathematics and Computer Science, Dali University, Dali, China
stri-fun@outlook.com, scy@stu.dali.edu.cn, ygs@stu.dali.edu.cn

Zizhong Yang
Yunnan Provincial Key Laboratory of Entomological Biopharmaceutical R&D, Dali University, Dali, China
yangzizhong@dali.edu.cn

Abstract
Few-shot image recognition represents a critical challenge in computer vision research. The scarcity of samples often results in inaccurate classification, limited generalization capabilities, and overfitted model recognition. To address these issues, the present study focuses on spider image recognition utilizing transfer learning and data augmentation techniques in limited sample settings. First, the BasNet image segmentation model and background replacement algorithm are used to extract species image data from the foreground; data augmentation is then applied to address the scarcity of samples. Second, a layer-by-layer fine-tuned transfer learning strategy based on the ResNet-50 model is devised. Specifically, to mitigate overfitting in the few-shot image classification task, the first two residual blocks are frozen so that only the last two are trained. To enhance the model's representation and generalization abilities, the SSC-ResNet-50 optimization model is constructed by introducing symmetry techniques. This study aims to enhance the accuracy and performance of spider image recognition. The experimental results demonstrate that the improved SSC-ResNet-50 model achieves an average accuracy of 99.1% in recognizing five types of spiders, thereby surpassing the performance of traditional models. These findings offer valuable insights for the field of small-sample high-precision image recognition.

Category: Computer Graphics / Image Processing

Keywords: Deep learning; Data augmentation; Transfer learning; Fine-tuning; Image segmentation

I. INTRODUCTION

Deep learning is a crucial technique for recognizing and classifying species images. Conventional species image classification methods use manually designed features, which are both computationally complex and

Open Access http://dx.doi.org/10.5626/JCSE.2023.17.4.145 http://jcse.kiise.org

Received 19 August 2023; **Accepted** 13 December 2023
*Corresponding Author

 pISSN: 1976-4677 eISSN: 2093-8020

inefficient. By contrast, deep learning can automatically extract features from the original image, thus leading to efficient and error-free recognition and classification while significantly shortening processing times. Deep learning is currently used extensively in species image classification. Nonetheless, it remains difficult to obtain sufficient amounts of high-quality, live species image data, particularly for species that are hard to catch and tend to hunt at night. For instance, most species of spiders exhibit characteristics such as a tendency to escape at the slightest disturbance, a preference to hunt during nighttime, etc. This makes it difficult to acquire data resulting in reduced availability of high-quality spider species image samples for deep learning model training. Small-sample image datasets do not provide enough data patterns for the model to adequately capture. Moreover, applying the convolutional neural network (CNN) to deep learning requires a large number of labelled training samples. Therefore, it is imperative to focus on how technological means can be effectively utilized to improve the quantity and quality of spider species image data to better support relevant research on image recognition and classification based on CNNs. Progress has been made in spider recognition research worldwide. For example, Ticay-Rivas et al. [1] employed a least-squares vector support machine with a radial basis function as a classifier to identify and validate spiders. In other researches, Dolev and Nelson [2] studied the recognition and classification of innate patterns of jumping spiders, while Clark and Uetz [3] researched video image recognition of jumping spiders. However, these studies still could not resolve the issue of the scarcity of high-quality spider image data. There are several methods that can be used to rectify this problem. For instance, common data augmentation techniques [4] can generate additional pseudo-data by using a limited amount of labelled data. Such a technique could be used to supplement the under-labelled few-shot set. Fine-tuning approaches [5] can be used in transfer learning to address the lack of labelled data by training a model ahead of time on a vast dataset. From this, the model can gain a prior knowledge that can benefit the task at hand. Meta-learning [6] can optimize the model initialization parameters, thus leading to better performance on small-sized samples. Meta-learning can also enhance the model's rapid learning ability on a few-shot set by optimizing the model initialization parameters.

A. Main Contribution

Therefore, to address the above problems, the current paper designs a fine-grained spider images recognition method based on transfer learning. First, as a solution to the problem of a small amount of spider data, the spider foreground is extracted by using the BasNet image segmentation model, and an image background replacement scheme is designed to automatically replace the adaptive background. This increases the amount of available data, which we train based on the improved symmetric deep residual network on the spider dataset we constructed. The specific method involves using the parameters of the pre-trained ResNet-50 model on ImageNet as initial parameters, using the bottom layer of the frozen model to extract the primary features of the two residual blocks of the spider, and training only the last two residual blocks to extract the intermediate features of the spider. Finally, the original fully-connected layer of the model is removed and a symmetric strong convolution module is added. To extract the high-level features of the spider, the fully-connected layer is rebuilt; the primary, intermediate and high-level features of the spider are merged; global average pooling is added; and the neurons in the fully-connected layer are randomly discarded using dropout to reduce the training parameters and alleviate the overfitting problem that is often caused by few-shot learning.

The rest of this paper is organized as follows. Section II provides an explanation of the dataset setup and data enhancement. Section III focuses on the research methodology. Section IV details the experimental environment and also analyzes the experimental results. Section V describes the work related to model training. Section VI concludes this paper and suggests directions for future work.

II. DATA SOURCE AND PRE-PROCESSING

The aim here is to explore the effect of the fine-grained recognition method on transfer learning. The objective is to detect and identify five spider categories—namely *Trichonephila clavata*, *Thomisidae*, *Missulena*, *Araneidae*, and *Theraphosidae*—using spider images from the Yunnan Provincial Key Laboratory of Entomological Biopharmaceutical R&D at Dali University. These species are common in spider habitats, but the available sample data is limited. To address this issue, the study will employ data augmentation, pre-training, and fine-tuning techniques in transfer learning. The data will undergo three pre-processing steps: resizing, shuffling, and normalization [7]. Resizing involves adjusting the image size to suit different models. For the purposes of this study, the images will be resized to the standard input size of the ResNet-50 model, which is 224×224 pixels. Shuffling the training image data is beneficial because it prevents model overfitting and enhances its generalization capability. It is necessary to use normalization to scale the pixel values from the conventional range of [0, 255] to a range between 0 and 1 to prevent gradient-related issues, such as exploding or vanishing gradients. When utilizing pre-training model parameters, the data should be pre-processed according to the data pre-processing method used in the pre-training phase [8].

In this study, as the ImageNet dataset serves as the pre-training dataset, the RGB mean value of the ImageNet dataset should also be subtracted from the target domain

Fig. 1. Spider datasets.

dataset, thus following the same data pre-processing method that was used in the source domain. Fig. 1 shows an example of the spider dataset.

A. Data Augmentation

1) Traditional Data Augmentation

Traditional data augmentation methods are typically those used to solve the few-shot learning problem in which there are insufficient training samples. These methods encompass geometric and color transformations to enhance the diversity and volume of the dataset. Geometric transformations include flipping, rotation, cropping, scale deformation, and reflection, while color transformations involve techniques such as Gaussian noise, random erasure, blur, and the super pixel method [9]. In 2017, Zhong et al. [10] proposed an innovative data augmentation technique called random erasure for training CNNs. Random erasure selects a rectangular area within an image and randomly replaces the pixels within that area with random values. This process generates training images with varying degrees of occlusion, which helps reduce the risk of overfitting and improves the model's ability to handle occluded images. Random erasure serves as a complement to commonly used data augmentation methods like random cropping and flipping. It consistently enhances performance across different tasks such as image classification, object detection, and person re-identification. The transformation involved in random erasure is essentially an affine transformation applied to the original image:

$$y = wx + b\,. \tag{1}$$

In the formula, x represents the original image, y represents the transformed image, w represents the multiple by which the pixel value expands and contracts, and b represents the addition and subtraction of the pixel value.

Color conversion refers to the conversion of HSV (hue, saturation, value) parameters of the picture, or principal component conversion (PCA) on the RGB of the picture. The principal component transformation is performed on the three vectors and eigenvalues in the RGB space. The following transformations are generally performed on all pixels of the picture:

$$[p_1, p_2, p_3]\,[\gamma_1\lambda_1, \gamma_2\lambda_2, \gamma_3\lambda_3,]^T \tag{2}$$

HSV is composed of three components: hue, saturation, and value. H represents hue, which is expressed in degrees and where red is 0°, green is 120°, and blue is 240°. S stands for saturation, the proportion of pure color, which expresses the degree of lightness (0%–100%). V represents brightness (0%–100%). HSV color conversion refers to the random conversion of HSV parameters.

2) Data Augmentation based on Outstanding Prospects

Traditional image segmentation techniques aim to partition an image into distinct regions, often while focusing on segmenting the most salient target. However, further processing is typically needed to achieve accurate segmentation. In this context, using an image segmentation method based on salient object detection (SOD) can better solve such problems. This approach enables the rapid and precise identification of objects or regions of interest within an image. When confronted with a large number of images, SOD can be leveraged to effectively segment the essential parts of an image; in the present work, utilizing salient target detection for the spider dataset allows for quick extraction of the spider foreground.

BasNet [11], which is a renowned model for boundary augmentation, was introduced at the CVPR2019 conference by the Xuebin Qin team at the University of Alberta. This model comprises two modules, as depicted in Fig. 2. The prediction module builds upon the densely supervised encoding and decoding network of U-Net [12], which can extract underlying features from the input image and learns to generate a saliency map. The multi-scale residual refinement module (RRM) learns the saliency map and compares it with the real saliency map, and it optimizes the output of the final saliency map of the model to have a clearer boundary.

In this paper, the BasNet model is utilized as the image segmentation model, and it is retrained using the spider dataset to improve its ability to extract the foreground target from spider images. Considering that many spiders inhabit trees or leaves, images of which typically involve a green background, a data enhancement method is proposed to automatically replace the adaptive green background in the extracted spider foreground using a pixel background replacement algorithm.

B. Datasets Settings

The original datasets are acquired through field collection and web crawling. After expert manual classification, identification and annotation, and data expansion based

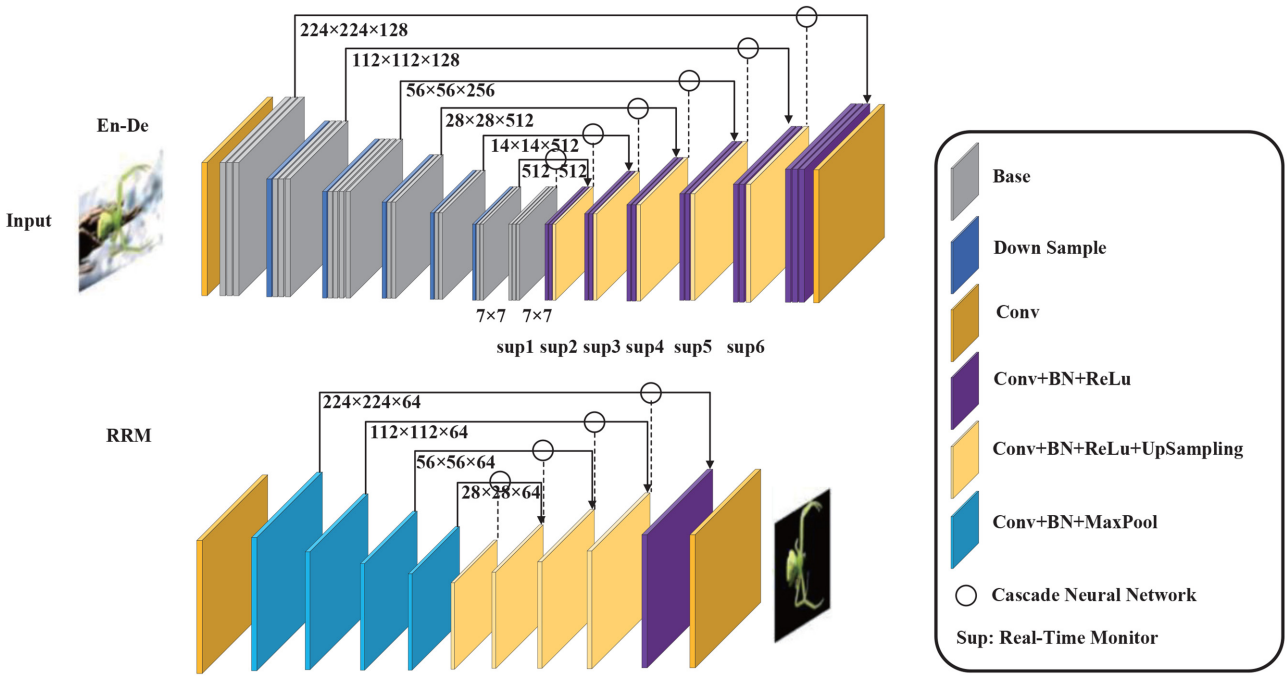


Fig. 2. BasNet model structure.

Table 1. Datasets

Dataset	Numbers	Description	Training	Test	Trainable
A	100	Original image	500	25	125
B	600	Original image + traditional data augmentation	3000	25	125
C	600	Original image + highlight the foreground	3000	25	125
D	600	Original image + highlight prospects + traditional data augmentation	3000	25	125

on foreground augmentation method and traditional method, four different spider training datasets were ultimately generated in this experiment.

As can be seen in Table 1, dataset A is the original dataset, which consists of 100 images for each category and 500 images in total. Dataset B is a combined dataset that includes the original images from dataset A and applies the traditional data augmentation method to expand the data. This dataset contains 600 images for each category, resulting in a total of 3,000 images. Similar to dataset B, dataset C combines the original images from dataset A with a data augmentation method that highlights the foreground to expand the data. Each category in this dataset consists of 600 images, comprising 3,000 images in total. Lastly, dataset D involves combining dataset C and the application of traditional data augmentation methods to expand the data further. The test set contains 25 images in each category, leading to a total of 125 additional images.

Since most spiders appear at night and will hide quickly after being stimulated, spider images are difficult to obtain, and training neural network models requires a large amount of image data [13], which makes it difficult to train neural network models for spider image recognition. To solve this problem, in the data pre-processing before training, traditional data enhancement techniques such as rotation and scaling are used to expand the training dataset [14], and transfer learning is employed to load the pre-training parameters and fine-tune the model to improve the recognition accuracy and generalization. The division and settings of the datasets are presented in Table 1.

III. RESEARCH METHODS

This paper mainly uses ResNet-50, a deep residual network pre-trained on ImageNet, to fine-tune the spider dataset. First, the BasNet saliency target detection model is used to extract the spider foreground, green is selected as the image background for data augmentation according to the RGB channel of the image, and four different

datasets are designed in combination with traditional data augmentation methods. Second, different fine-tuning methods are tested, and different layers of the model are frozen to find the model training structure that is most suitable for the few-shot data in this experiment. At this point, the original fully-connected layer of the model is deleted, a symmetric strong convolution module is built to extract the deep features of the spider, two fully-connected layers are added, and the dropout method is finally used to mitigate overfitting. The LeakyReLU function is used in the experiment to replace the original ReLU of ResNet-50.

A. Transfer Learning

Traditional machine learning tasks often require large amounts of data to achieve good performance. The quality and quantity of data both play crucial roles in determining the effectiveness of a neural network model. However, obtaining specific and relevant image data can often be challenging. Such a lack of data makes it difficult to effectively train neural networks. Transfer learning [15] has emerged as a powerful technique that addresses these challenges, and it has significantly improved the accuracy of few-shot learning tasks. It helps overcome the overfitting problem caused by limited data and accelerates the convergence of neural networks. Transfer learning involves transferring the knowledge or parameters learned from one task or domain to another task or domain. There are two important concepts in transfer learning: the Domain and the Task [16]. A domain refers to a range of data that exhibits different characteristics and spatial distributions. The source domain (Ds) corresponds to a large dataset used for pre-training, while the target domain (Dt) represents a smaller dataset that is specific to the task at hand.

After training the neural network, the pre-training model parameters are removed from the fully-connected layer, thus leaving only the bottom convolutional layer parameters to serve as a powerful feature extractor [17] to extract features such as data edges, textures, and spatial distribution to compensate for small samples overfitting caused by training and insufficient model feature extraction capabilities [18]. When using transfer learning, it is very important to consider the difference in the probability distributions between the source domain and the target domain. According to the difference between the feature space and the label space, transfer learning can be divided into two categories: homogeneous transfer learning and heterogeneous transfer learning [19]. When performing transfer learning, it is necessary to select a situation where the data distributions of the target domain and the source domain are similar for migration. This type of homogeneous transfer learning can achieve a multiplier effect. If the source domain and target domain distributions are too different from each other, they are considered to be heterogeneous. It is important to ensure that the gap between the datasets is narrowed for transfer learning; otherwise it is very likely to cause negative transfer [20]. ImageNet contains a large amount of animal and insect data and spider data, which is more suitable for the fine-grained classification of spider species in this study. Therefore, this article chooses ImageNet as the source domain. To sum up, this article focuses on the few-shot learning problem for the collection of spider datasets, and it uses the large dataset ImageNet containing 1,000 categories as the source domain to collect five types of spider sample datasets—*Trichonephila clavata*, *Thomisidae*, *Missulena*, *Araneidae*, and *Theraphosidae*—as the target domain, and ResNet-50 is used as the target model for transfer learning.

1) Pre-training and Fine-Tuning

In the category of transfer learning, there is a transfer based on shared parameters, the main methods involved in which are pre-training and fine-tuning [15, 16]. In general machine learning tasks, the parameters of the model are randomly initialized, after which the network begins being trained again. This allows the neural network to fully learn the characteristics of the differences between different categories for large datasets, but for small datasets, collectively speaking, the neural network cannot fully learn the underlying features. The law of neural network extraction features is that the shallow neural network extracts the primary features of the image, such as edges, textures, corners, etc.; the middle network extracts intermediate features such as color and shape, such as extracting the eyes, nose, mouth, and other features of a face in a face recognition task; and the deep neural network extracts high-level features, such as spatial position, relative direction, etc., and combines the local features extracted earlier. For the spider datasets, the law of extracting features during the training process of the model is to first extract the edge contour of the spider, after which the detailed information of the spider is extracted. As the network level deepens, the extracted features become increasingly more abstract, and the corresponding feature matrix is more blurred.

Pre-training plus fine-tuning is a simple and efficient method used in many transfer learning tasks, and it plays an important role in machine learning tasks. This method is mainly used to pre-train a neural network model on a source domain that has a similar feature distribution as the target domain, and then transfer the source model parameters to the new model.

In this paper, ResNet-50 [21] is used as the recognition model. The ResNet-50 model pre-trained on ImageNet is removed from three fully-connected layers, and only the parameters of the convolutional layer are retained as the initial parameters. During training using the spider datasets, the bottom parameters of the frozen model do not participate in training, and only the top parameters of the model are trained. Because the bottom layer of the model

extracts common features such as edge contours, the problem of insufficient data extraction at the bottom of the model caused by insufficient data can be solved using pre-trained parameters on large datasets with similar feature spaces in the source and target domains. The knowledge (parameters) of the source model is transferred to the new model for new classification tasks, thus improving the generalization ability of the model.

B. SSC-ResNet-50 and its Training Process

1) ResNet-50 Network

The ResNet series network was proposed by He et al. [21] in 2016 to solve the problem of model performance degradation caused by the deepening of the network layer, the disappearance of the gradient, or the explosion of the gradient. The ResNet-50 residual network can be considered as a stack of residual blocks, meaning the network can be designed very deep. The difference between the residual network and the general network is that it first copies and accumulates part of the data and then performs a nonlinear transformation. Ideally, as the depth of the network increases, the training error should gradually decrease, and ResNet can reduce the error generated by training the deep network. For general CNNs, gradient descent is a common optimization algorithm. The formula for the remaining function of ResNet network learning is shown in (3).

$$X_{l+1} = X_l + F(x_l, W_l). \tag{3}$$

The residual block is divided into two parts: the direct mapping part and the residual part. The direct mapping part is reflected in the curve structure on the right in Fig. 3, while the residual part is generally composed of two or three convolution operations; that is, the part on the left that contains convolution in Fig. 3 represents the characteristics of the residual block after processing. In ResNet-50, this residual structure also becomes the bottleneck residual structure, because the feature dimension of the input image is compressed from 256 to 64, then reduced to 256 by the 1×1 convolution kernel, thus forming two sides. The resulting bottleneck structure is thick and thin.

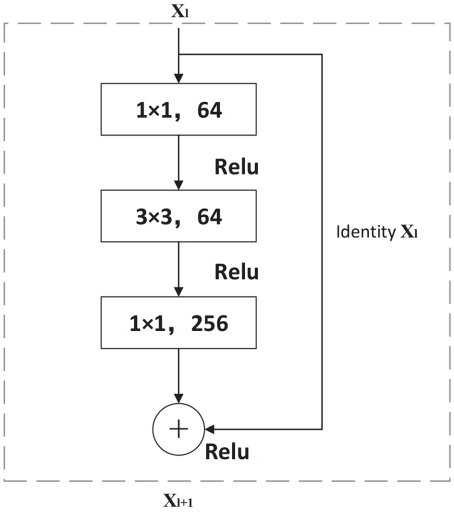


Fig. 3. ResNet residual block structure.

2) SSC-ResNet-50 Network

However, given the intricate architecture of ResNet, it consists of numerous model parameters, thus making it challenging to train with the limited spider dataset. Therefore, transfer learning is used to train the ResNet model for ImageNet. The initial parameters of the model are set by loading the pre-trained ResNet-50 model, and the fully-connected layer parameters are removed. Subsequently, the fully-connected layer network is reconstructed while incorporating techniques such as global average pooling and dropout to mitigate overfitting.

In few-shot learning [22], the primary features of edges and contours are considered to be less important. Hence, we opt to load the parameters pre-trained on ImageNet and freeze the two residual block parameters at the model's base, thereby preventing them from participating in further training. This approach allows us to fully utilize the pre-trained model's ability to extract primary features (common features) from a large dataset, thus compensating for the limitations of the few-shot dataset. Moreover, after unfreezing, the two residual blocks are trained to increase the model's familiarity with and extraction of characteristic information specific to spiders.

To further enhance the model's ability to extract advanced spider characteristics and integrate the spider's characteristic information, a symmetrical and robust convolution module is added to the model's end. This module effectively captures the intricate characteristics of spiders and facilitates the fusion of their characteristic information. The original ReLU activation function of the model is also replaced with LeakyReLU.

The newly added symmetric strong convolution module is designed to extract high-order features of spiders. It comprises three convolutional layers: a 1×1 convolution kernel with a size of 512 and a stride of 1, followed by a 3×3 convolution kernel with a dimension of 1024 and a stride of 1, and finally followed by a 1×1 convolution kernel with a dimension of 2048 and a stride of 1.

A batch normalization layer [7] is added after each convolution kernel, with momentum set to 0.9 and epsilon set to 1e-5. Batch normalization (BN) was proposed by Ioffe and Szegedy [7] in 2015 to solve the problem of internal covariate shift. Then, the BN becomes an important part of a neural network, just like convolution, pooling, and other layers. The function of the BN layer is to normalize the input of each layer so that the input value of the entire neural network can be controlled within a

stable range that is easy to calculate. This solves the internal covariate shift problem mentioned above. In the transfer learning process, it is necessary to control the BN layer that is not being trained. The realization formula of the BN layer is as follows:

$$\mu_B = \frac{1}{n}\sum_{i=1}^{n} x_i\,, \tag{4}$$

$$\sigma_B^2 = \frac{1}{n}\sum_{i=1}^{n}(x_i - \mu_B)^2\,, \tag{5}$$

$$x_i' = \frac{x_i - \mu_B}{\sqrt{\sigma_B^2 + \varepsilon}}\,, \tag{6}$$

$$y_i = \gamma_i \cdot x_i' + \beta_i\,, \tag{7}$$

where x_i represents the input of a batch of a certain layer of the neural network as a sample in $X = [x_1, x_2, ..., x_n]$; n is the batch size, which is the mean value of the elements of each batch input; σ_B^2 represents the variance of each mini-batch; x_i' represents the normalization process for each element; and y_i represents the final output of the network upon completion of the normalization process.

Then, a pooling layer and a stack of globally averaged pooling layers are added, dropout is to the fully-connected layers (with the random discard rate parameter set to 0.5), and LeakyReLU is used as the activation function throughout the model and added behind the convolutional layer and in front of the BN layer to form the model structure of Conv + LeakyReLU + BN. Here, the bias parameter of all convolution kernels is set to False, i.e., no bias is used, because if the convolution layer is followed by a normalization layer, then the normalization layer will normalize the output of the convolution layer and add its own bias. In other words, the bias (if any) of the convolutional layer is redundant. Among them, the first 1×1 convolutional kernel is mainly used for dimensionality reduction, while the second 3×3 convolutional kernel is mainly used to down-sample the size of the feature map of the input fully-connected layer in the original ResNet-50 from 7×7 to 2×2, with the aim of further extracting some tiny high-level detailed features of the spiders, and to make the model achieve a better classification result. The final 1×1 convolutional kernel restores the feature map to the original 2048 feature dimensions. The dimensions of the three convolutional kernels are gradually increased to gradually deepen the dimensions of the features, i.e., the depth of the model, to achieve the purpose of extracting the spider features through powerful convolution; this is shown in Fig. 4.

The newly introduced symmetric strong convolution module combines the convolution layer and the normalization layer in the model. This integration has the ability to address the irregularity in the original data, promote a more regularized data distribution, and accelerate the convergence speed of the network. The reasoning process behind this combination is demonstrated in the following formulas.

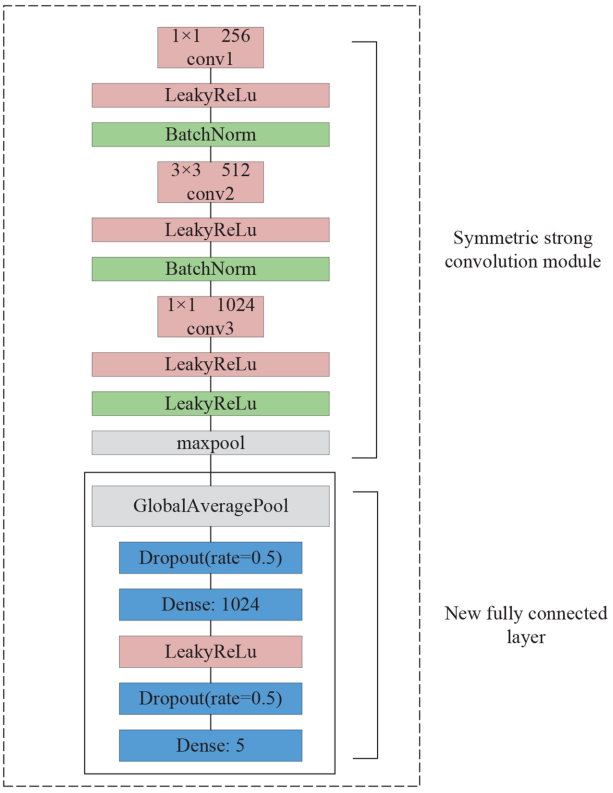


Fig. 4. Newly symmetric strong convolution module added to the model.

$$y_{conv} = w \cdot x\,, \tag{8}$$

$$y_{bn} = \gamma \cdot \left(\frac{y_{conv} - E[x]}{\sqrt{Var[x] + \varepsilon}}\right) + \beta\,, \tag{9}$$

$$\hat{\omega} = \frac{\gamma}{\sqrt{Var[x] + \varepsilon}} \cdot w\,, \tag{10}$$

$$\hat{b} = \frac{\gamma}{\sqrt{Var[x] + \epsilon}} \cdot E[x] + \beta\,, \tag{11}$$

$$y_{bn} = \hat{w} \cdot x + \hat{b}\,. \tag{12}$$

Here, y_{conv} represents the output of the convolutional layer and w represents the weight of the convolution kernel. Further, pre-training weights are used, and the convolutional layer does not use bias. $E[x]$ is the sliding mean, $Var[x]$ is the sliding variance, y_{bn} is the output of

the BN layer, $\hat{w}$ is the weight of the BN layer, $\hat{b}$ is the bias of the BN layer, and β is the training parameter.

The ReLU [23] activation function is used to solve the problem of the disappearance of the gradient. The ReLU function activates the weight x of the feature matrix. When x¡0, it is difficult to saturate, and the weight is reset to 0. When x¿0, the derivative is 1. The gradient does not decay, and the corresponding eigen matrix weight remains unchanged, thus alleviating the problem of gradient disappearance. The ReLU activation function formula (13) is as follows:

$$\mathrm{Re}LU\,(x) = \max(0, x) = \begin{cases} 0, x < 0 \\ x, x > 0 \end{cases}. \tag{13}$$

However, ReLU also has disadvantages. Despite its simple and efficient structure, which improves network convergence speed, it suffers from what is known as the "dying ReLU" problem. This occurs when the input value is negative, thus causing the gradient to be 0. As a result, some neurons cannot perform gradient descent, so their parameters remain unchanged, leading to an infinite cycle. This renders the neurons ineffective for the remaining training tasks, ultimately causing their "death." To address this issue, LeakyReLU was introduced as a solution. LeakyReLU compensates for the deficiency of the ReLU function by introducing a small negative value for any input that is less than 0, typically with a slight gradient slope of 0.01. Unlike ReLU, which sets the value to 0 for all inputs less than 0, LeakyReLU preserves a non-zero value for the negative input range.

This ensures that the gradient does not vanish entirely, as a small gradient helps maintain a non-zero partial derivative during gradient descent. This concept is illustrated in Fig. 5. The overall structure and fine-tuning methods of the SSC-ResNet-50 model are depicted in Fig. 6, while Table 2 presents the parameter settings for each component of the model.

$$Leaky\mathrm{ReLU}\,(x_i) = \begin{cases} x_i, x_i \geqslant 0 \\ ax_i, x_i < 0 \end{cases}. \tag{14}$$

After pre-processing the data, it is fed into the model for training. First, hyperparameters such as learning rate [24] (the learning rate), batch size [25] (the sample size used in one iteration), and epoch [26] (the number of iterations) need to be set according to the dataset size and the chosen optimizer [27]. Upon initiation of model training, the ResNet-50 model consists of four bottleneck

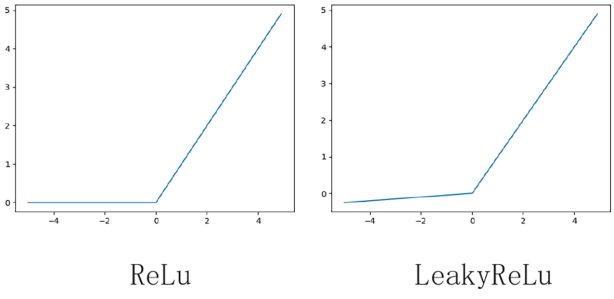


Fig. 5. Comparison of two activation functions' ranges.

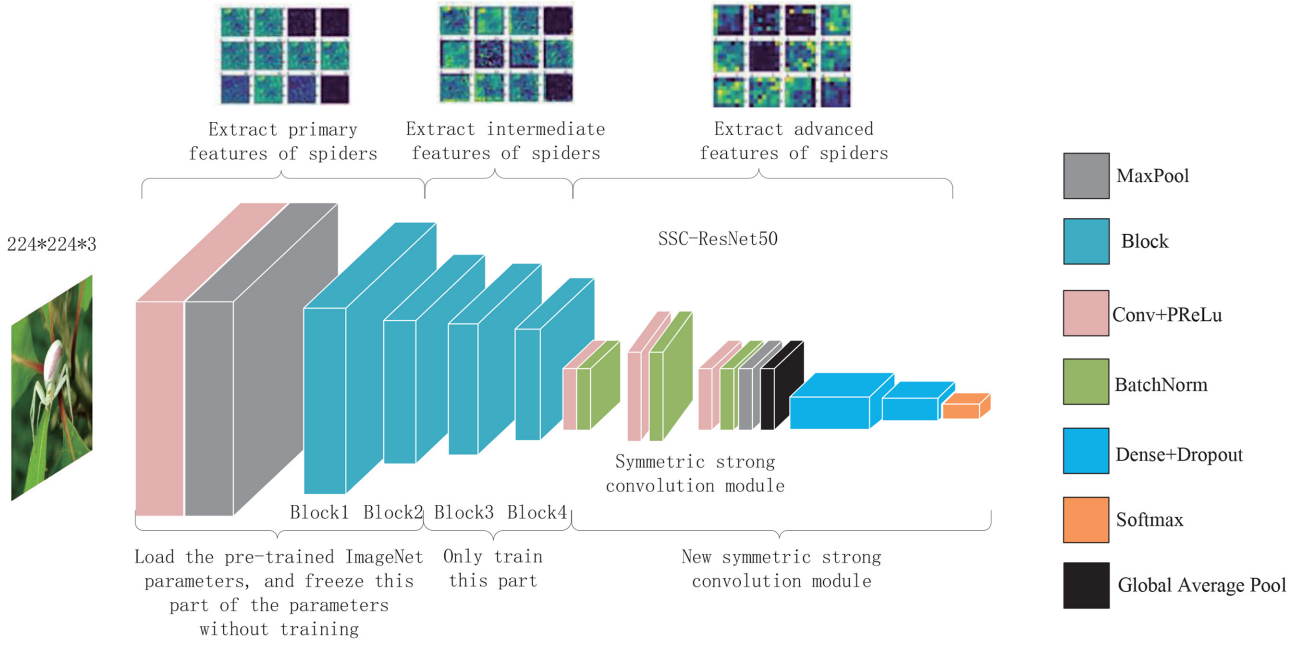


Fig. 6. Fine-tuning of SSC-ResNet-50 model.

Table 2. Model parameter settings

Model parameter settings	Parameters	Active	Padding	Output shape	Trainable
Input_1(Layer)	0	-	-	(224, 224, 3)	False
Conv_1(Conv2d)	9408	LeakyReLU	Same	(112, 112, 64)	False
Conv1/BatchNorm	256	-	-	(112, 112, 64)	False
LeakyReLU	0	-	-	(112, 112, 64)	False
Max_Pooling	0	-	Same	(56, 56, 64)	False
Block_1	218624	LeakyReLU	Same	(56, 56, 256)	False
Block_2	1226752	LeakyReLU	Same	(28, 28, 512)	False
Block_3	7118848	LeakyReLU	Same	(14, 14, 1024)	True
Block_4	14987264	LeakyReLU	Same	(7, 7, 2048)	True
SSC_Block	2234112	LeakyReLU	Same	(2, 2, 2048)	True
Dropout	0	-	-	(None, 1024)	True
Dense	1049600	LeakyReLU	-	(None, 1024)	True
Dropout	0	-	-	(None, 1024)	True
Dense	5125	-	-	(None, 5)	True
Softmax	0	-	-	(None, 5)	True

residual structures and two fully-connected layers. In the first four residual blocks, the input image undergoes convolution using the stacked bottleneck residual structure with three convolution kernels. This operation, combined with down-sampling [28], reduces the image size while simultaneously increasing the feature map's dimensions and extracting more channel information. Finally, the local features are merged into global features through the fully-connected layer, and the softmax [29] function is applied for multi-classification, thereby obtaining the predicted probability for each class [30]. The sum of probabilities across all classes approximates to 1. During the forward and backward propagation of model training, the network nodes (convolution kernels) in the first layer are represented by the convolution kernel corresponding to the *p* channel of the first layer and the *q* channel of the first layer. This symbolizes the weights of the first layer's fully-connected network and represents the forward input of the *l*-th layer without the activation function. The variables *u* and *v* represent the coordinate position of the convolution operation. Eq. (16) represents the convolution operation formula of the *l*-th layer, and the maximum pooling formula is as expressed in Eq. (17):

$$Z_p^l(i,j) = \sum_{q=1}^{n^{l-1}} \sum_{u=-1}^{1} \sum_{v=-1}^{1} \alpha_q^{l-1}(i-u, j-v)\, k_{p,q}^l(u,v), \tag{15}$$

$$\alpha_p^l(i,j) = LeakyReLU\left(z_p^l(i,j)\right), \tag{16}$$

$$z_p^l(i,j) = max\left(\alpha_p^{l-1}(2i-u, 2j-v)\right) \;\; u,v \in \{0,1\}. \tag{17}$$

The convolution and pooling operations of the model can continuously reduce the dimensionality of the input image while extracting image features. After passing through the first four residual blocks of the model and the newly-added symmetric strong convolution module, the image dimension is reduced to (3×3×1024), where the size of the image is reduced to 3×3 and the thickness of the feature map is increased to 1024 layers; it is then converted into a 2234112-dimensional vector as the input of the fully-connected layer. The output after the layer is Eq. (18), the formulas for forward propagation are shown in Eqs. (19) and (20), and the output layer is activated by the softmax function as expressed in Eq. (21). In calculating back propagation, an intermediate variable is introduced as the error of the first layer. The gradient of the forward propagation of the first layer can be expressed as, and formulas (22) and (23) are used to calculate the gradient.

$$\alpha^{dense} = F\left(\left\{z_p^{dense}\right\}_{p=1,2,\ldots,512}\right), \tag{18}$$

$$z^{l+1} = W^{l+1}\alpha^{*l} + \beta, \tag{19}$$

$$\alpha^{l+1} = LeakyReLU\left(z^{l+1}\right), \tag{20}$$

$$\alpha_i^L = softmax\left(z_i^l\right) = \frac{e^{z_i^l}}{\sum_{k=1}^{n^l} e^{z_k^l}}, \tag{21}$$

$$\frac{\partial L}{\partial W^l} = \frac{\partial L}{\partial z^l}\frac{\partial z^l}{\partial b^l} = \delta^l\left(\alpha^{l-1}\right)^T, \tag{22}$$

$$\frac{\partial L}{\partial b^l} = \frac{\partial L}{\partial z^l}\frac{\partial z^l}{\partial b^l} = \delta^l. \tag{23}$$

When using pre-training and fine-tuning in transfer learning for training, the pre-training weights should first be downloaded on ImageNet corresponding to ResNet-50 from the TensorFlow official website, after which the fully-connected layer parameters of the weights are removed, and then the weights are loaded into the model. Next, three different fine-tuning methods are used to train the model. These three different fine-tuning methods are described in Table 3. The entire model training and recognition flow chart is shown in Fig. 7. To compare the impacts of transfer learning and not using transfer learning, different fine-tuning methods, and different datasets on

Table 3. Fine-tuning methods

Fine-tuning method label	Pre-training source domain	Description
I	ImageNet	Freeze all convolutional layer parameters and train only the fully-connected layer
II	ImageNet	Use pre-training parameters as initial parameters to train from scratch
III	ImageNet	Freeze the parameters of Block1 and Block2, only train Block3 and Block4

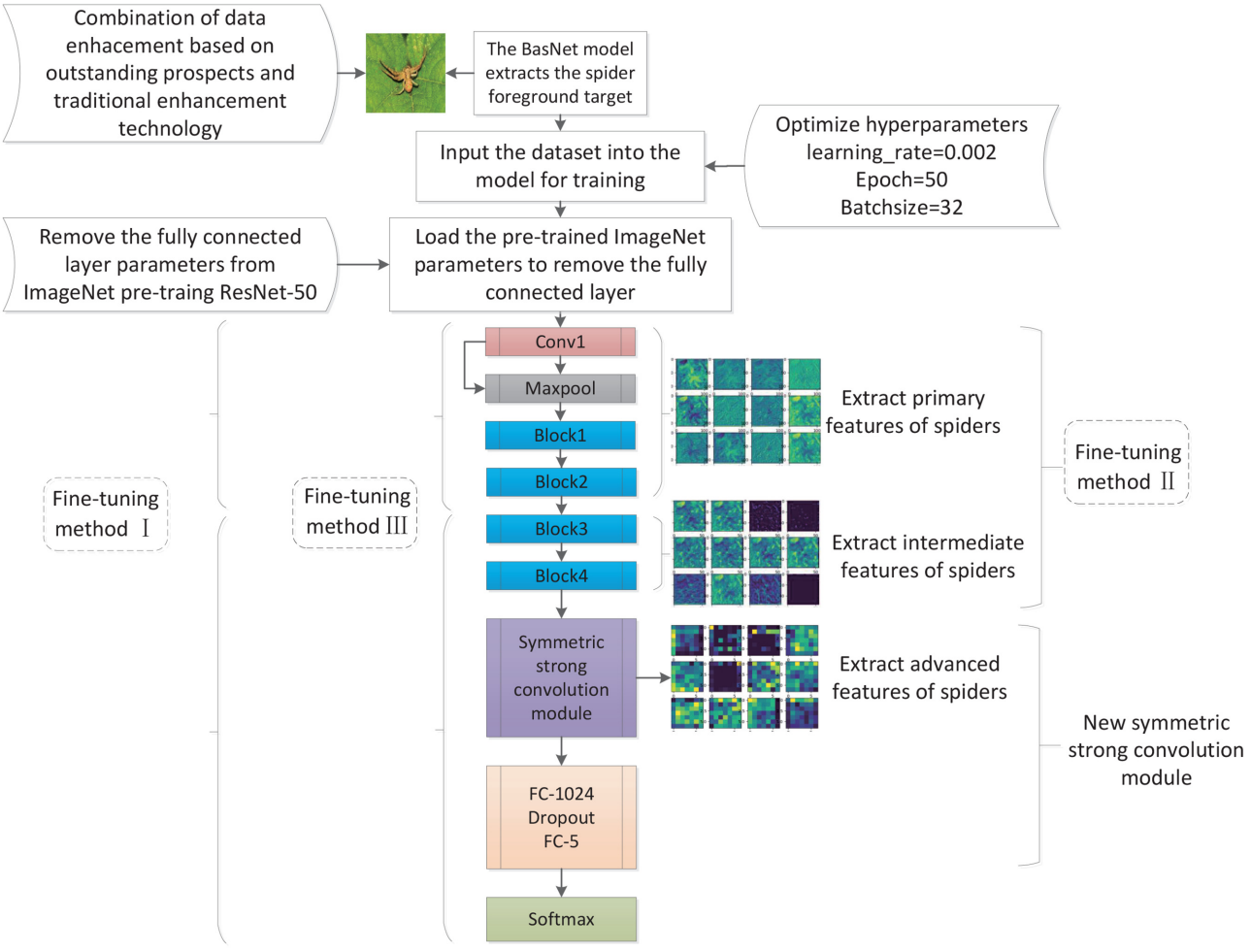


Fig. 7. Model training process of three different fine-tuning methods.

the accuracy of the model, this paper designs two sets of comparative experiments. In the experiment that does not use transfer learning, the number of training rounds is set to 100 epochs, the learning rate is 0.002, and the batch size is 16. Meanwhile, in the experiment that uses transfer learning, the number of training rounds is set to 50 epochs, the learning rate is 0.0002, and the batch size is 16. The Adam optimization algorithm is used as an optimizer, which integrates the advantages of AdaGrad and RMSProp algorithms. Adam not only calculates the adaptive parameter learning rate based on the mean value of the first-order moments like the RMSProp algorithm, but it also makes full use of the mean value of the second-order moments of the gradient.

IV. EXPERIMENTAL RESULTS AND ANALYSIS

A. Lab Environment

The experimental hardware configuration has the following specifications: RTX 2080Ti GPU×1, memory 32 GB×2, processor Intel E5-2603×2. The model is implemented using Python 3.7 and TensorFlow 2.1 deep learning framework under the Windows 10 operating system.

B. Experimental Results

1) Without using Transfer Learning

First, the ResNet network and the SSC-ResNet network are used in four different types of datasets: (i) Original image, (ii) original image + traditional data augmentation, (iii) original image + foreground data augmentation, and (iv) original image + highlight the prospects + training under traditional data augmentation. At this time, the pre-training parameters are not used, and the training is restarted. As can be seen in the results in Table 4, when transfer learning is not used, on dataset D, after training, the test accuracy of SSC-ResNet reaches 83.48%, which is 3.68% higher than that of the original ResNet-50 network, and higher than those of datasets B, C, and A, which were expanded with different data augmentation methods. The accuracy is the highest in dataset D, and the overall performance is in descending order of D>C>B>A. It can be seen that SSC-ResNet reduces the dimension of the feature map of the input fully-connected layer and increases the trainable parameters. In the case of insufficient data, the performance of dataset A (below the original image) is not as good as that of the original ResNet network.

2) Training based on Transfer Learning Method

To examine the impact of transfer learning on recognition accuracy, this study conducts an experimental analysis that combines transfer learning with data augmentation techniques. Four different types of datasets are designed: (i) original image, (ii) original image + traditional data augmentation, (iii) original image + prominent foreground data augmentation, and (iv) original image + prominent foreground + traditional data augmentation. Three different fine-tuning methods are used to train the few-shot spider data: method I, all convolutional layer parameters are frozen and only the fully-connected layer is trained; method II, pre-trained parameters are used as initial parameters to train from scratch; and method III, residual blocks Block1 and Block2 are frozen, and only residual blocks Block3 and Block4 are trained. To begin, in the most conventional pre-training and fine-tuning method, where all convolutional layer parameters are frozen and only the fully-connected layer is trained, four models (VGG-16, ResNet-50, Vision Transformer, and GoogleNet) are trained using dataset A, which contains the original images. The results, as shown in Table 5, indicate that the ResNet-50 model achieves the highest accuracy rate of 98.2%, while the VGG-16 model performs the worst with an accuracy rate of 91.6%. Previous reports have stated that the performance of Vision Transformer surpasses

Table 4. Comparison of results between the two models

Model	Dataset				Trainable parameter	Non-trainable parameter
	A	B	C	D		
SSC-ResNet-50	0.7529	0.7614	0.8260	0.8348	33,486,405	60,288
ResNet-50	0.7614	0.7710	0.8073	0.7980	25,611,333	53,120

Table 5. Comparison of four models

Model	Fine-tune	Dataset	Val accuracy	Trainable parameter	Non-trainable parameter
ResNet-50	I	A	0.982	2,102,276	23,561,152
VGG-16	I	A	0.916	55,588,869	14,714,688
Vison Transformer	I	A	0.964	3,845	85,798,656
GoogleNet	I	A	0.971	4,456,079	5,504,256

that of CNN models on large datasets. However, due to its extensive parameters, the Vision Transformer's performance on few-shot spider data falls short compared to that of the classic CNN model, ResNet. Therefore, the ResNet-50 model is chosen for the classification task and further optimized and enhanced specifically for the few-shot spider dataset. This enables it to better adapt to fine-grained spider recognition.

Further, the enhanced ResNet model, SSC-ResNet-50, is trained on the four datasets (A, B, C, D) using a combination of three different fine-tuning methods. According to the data presented in Tables 6, 7, and Fig. 8, it can be observed that the model achieves the highest accuracy of 99.1% on the test set when using dataset D and fine-tuning method III.

This accuracy is 0.9% higher than the average accuracy of the original ResNet-50 network, thus indicating good robustness. While many studies suggest that traditional data augmentation methods enhance model generalizability, it is important to note that the spider dataset differs from other large animal datasets. Spider images possess intricate edge contour features and complex shapes. In fine-grained recognition tasks, the differences between different types of spiders are often subtle, and there are many common features shared among different species. Thus, traditional data augmentation methods such as rotation, cropping, and tone transformation do not significantly benefit model training on the few-shot spider dataset; they may even have negative effects. However, when combined with data augmentation techniques that emphasize the outstanding aspects of spider features, these methods can effectively augment the dataset and contribute positively to model training.

Finally, based on the experimental findings, it is observed that, when using the same fine-tuning method, the different data augmentation methods have varying effectiveness. When training on a dataset B with traditional data augmentation techniques and a dataset C with foreground data augmentation techniques, with both datasets containing six times the amount of data compared to the original image, the results prove to be better than when training with only the original image. Further, the traditional data augmentation method yields slightly better results than the foreground data augmentation method. Moreover, the combination of the original image with both traditional and foreground data augmentation techniques outperforms the foreground enhancement method alone, with an average improvement of 1.8% compared to the traditional data augmentation method's improvement of 1.9%. Moreover, under the three fine-tuning methods, when the parameters of Block1 and Block2 are frozen while only training Block3 and Block4, and when training on dataset D, the training effect surpasses that achieved by the traditional

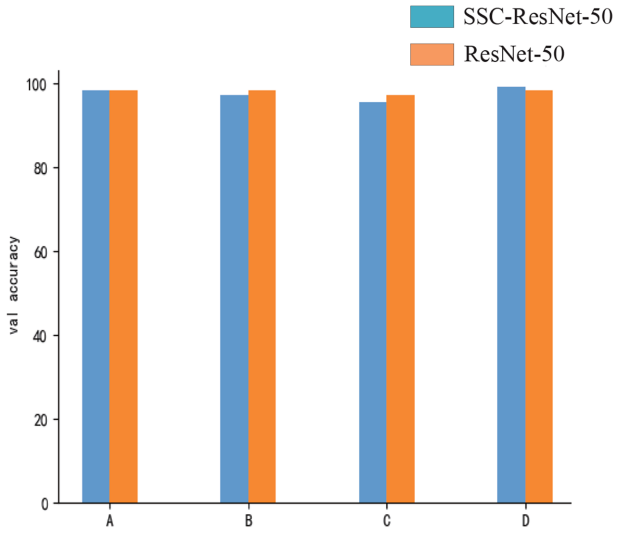


Fig. 8. Comparison of accuracy between the two models.

Table 6. SSC-ResNet model results

Model	Dataset				Trainable parameter	Non-trainable parameter
	A	B	C	D		
SSC-ResNet-50-I	0.982	0.972	0.954	0.973	9,978,373	23,568,320
SSC-ResNet-50-II	0.982	0.972	0.936	0.954	33,486,405	60,288
SSC-ResNet-50-III	0.971	0.972	0.954	0.991	32,041,477	1,505,216

Table 7. ResNet model results

Model	Dataset				Trainable parameter	Non-trainable parameter
	A	B	C	D		
ResNet-50-I	0.982	0.982	0.971	0.982	2102276	23561152
ResNet-50-II	0.973	0.945	0.963	0.954	25,611,333	53,120
ResNet-50-III	0.971	0.963	0.963	0.963	24,166,405	1,498,048

fine-tuning method.

From the above conclusions, it is evident that the model's new symmetrical strong convolution module, combined with fine-tuning method III, has a positive impact on extracting high-level features from spiders. The combination of data augmentation techniques, which are specifically based on prominent prospects and traditional methods, contributes to enhancing the data diversity, thereby improving the model's generalization ability.

Freezing the model's underlying pre-trained parameters can make full use of its powerful primary feature (public feature) extraction capabilities pre-trained on large networks, and the spider datasets can then be used to train the middle and top layers of the model to better extract the unique features of the spider. Compared to the traditional freezing of all layer parameters, aside from the fully-connected layer, this method has more training parameters, so the model has higher robustness and higher confidence when predicting spider images. The method of pre-training and fine-tuning can also greatly improve the training speed of the model. Using the transfer learning method to train, the model stands at a higher starting point, and the initial accuracy rate on the test set can reach more than 80%. It can be seen that the source domain ImageNet is well adapted to the spider datasets of the target domain, and that fine-tuning the parameters of the pre-training model substantially contributes to the spider species identification task.

When using transfer learning, the model loads pre-trained parameters from the ImageNet dataset. Compared to the parameters of the same first convolution kernel and batch normalization layer without using transfer learning, the distribution of pre-training parameters is relatively concentrated, and the parameter sizes are significantly different from each other. When no transfer learning is used and the parameters are randomly initialized, the parameter distribution is relatively uniform.

Comparing the transfer learning method with the retrained ResNet-50 and SSC-ResNet-50 in Fig. 9, it can be seen that the transfer learning method achieves an average improvement of 22.9% compared to retraining, and that significant improvements can be observed across the different datasets. This indicates that the various pre-training and fine-tuning methods designed in this experiment

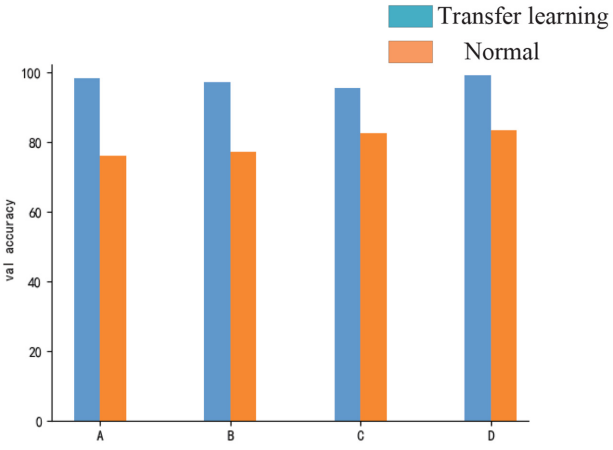


Fig. 9. Comparison of two training methods.

greatly contribute to improving the training accuracy of the model.

The following discusses the test results of the SSC-ResNet-50 model trained on dataset D using the fine-tuning method III, while comparing the precision, recall, and F1-scores for the manual recognition of five spider categories (*Thomisidae*, *Trichonephila clavata*, *Theraphosidae*, *Missulena*, and *Araneidae*). These test results are shown in Table 8. Moreover, the classifier performance of the SSC-ResNet-50 model trained on dataset D using fine-tuning method III is well explained by the ROC curves and AUC values in Fig. 10.

Accuracy: The ratio of the number of correctly classified samples to the total number of samples, as shown in formula (24):

$$Accuracy = \frac{TP + TN}{ALL}. \tag{24}$$

Precision: The ratio of the number of samples correctly identified by the model to the total number of samples retrieved, as shown in formula (25):

$$Precision = \frac{TP}{TP + FP}. \tag{25}$$

Table 8. Results of the system (unit: %)

Spider categories	Transfer learning + data augmentation			Data augmentation only		
	Precision	Recall	F1-score	Precision	Recall	F1-score
Theraphosidae	92.3	100	95.9	91.55	88.34	89.92
Missulena	100	100	100	88.73	89.12	88.92
Thomisidae	100	100	100	92.79	92.10	92.44
Trichonephila clavata	100	99.10	99.6	92.30	96.10	91.16
Araneidae	100	100	100	86.74	95.90	91.09

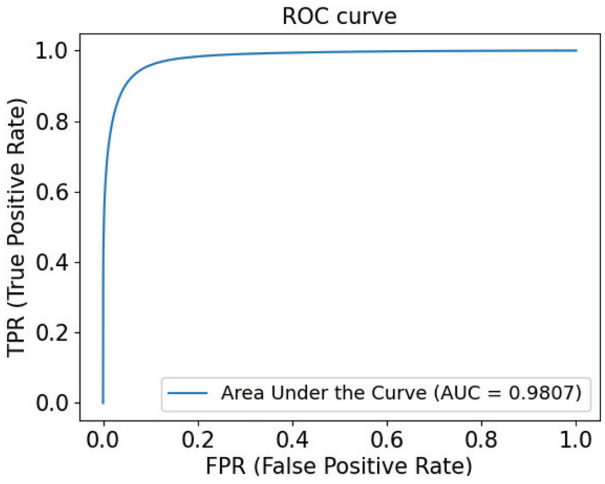


Fig. 10. ROC curves for the SSC-ResNet-50 model using the fine-tuning method III.

Recall: The ratio of the number of correctly identified samples to the number of samples that should be retrieved, as shown in formula (26):

$$Recall = \frac{TP}{TP + FN}. \tag{26}$$

F1-score: The harmonic average of model precision and recall rate, as shown in formula (27):

$$F1\text{-}Score = \frac{2 \times TP}{2 \times TP + FP + FN}. \tag{26}$$

Among them, TP represents the number of positive samples correctly recognized by the model as positive samples; FP represents the number of negative samples incorrectly recognized by the model as positive samples; FN represents the number of negative samples incorrectly recognized by the model as positive samples; and TN represents the number of negative samples correctly recognized by the model as negative samples.

In summary, using the transfer learning method and the ResNet-50 model to conduct fine-grained recognition of spider images, the recognition accuracy of the above five types of spider test sets has reached 99.1%, thus achieving high recognition accuracy. It can be seen that the model has high robustness and is set to be applied in actual production practice.

V. CONCLUSION

With the aim of addressing the problem of spider few-shot classification, this paper optimizes and improves upon the ResNet-50 network model as a foundation, and it builds an SSC-ResNet-50 model that is particularly suitable for few-shot spider datasets. On the one hand, it uses the method of pre-training and fine-tuning in transfer learning. Through comparative experiments, the best solution for selecting the residual blocks is found to involve freezing Block1 and Block2 at the bottom of the model and training only Block3, Block4, and the fully-connected layer. While effectively using the powerful public feature extraction capabilities of the pre-trained network model, the model can fully extract high-level features of spiders. On the other hand, a data augmentation method that combines traditional methods and data augmentation technology that highlights the foreground is used to grow the amount of data and improve the generalization performance of the model. Ultimately, the following conclusions can be obtained through the experimental results:

- The SSC-ResNet-50 network model has an accuracy of 99.1% on the test set of five types of spiders, which is 0.9% higher than that of the ResNet-50 network.
- Using outstanding prospects or traditional data augmentation methods alone will cause disturbances to the model, and may even reduce the accuracy of the model, while combining the two to expand the data is helpful to improve the accuracy of the model.
- Compared to the conventional transfer learning method that freezes all convolutional layer parameters and only trains the fully-connected layer, this paper freezes the bottom residual blocks of the model Block1 and Block2, and it only trains Block3, Block4, and the fully-connected layer. The effect of this solution on dataset A is the best among all groups of experiments.

ACKNOWLEDGEMENTS

This research was funded by the National Natural Science Foundation of China (No. 32001313), Yunnan Fundamental Research Projects (No. 202201AT070006), Yunnan Post-doctoral Research Fund Projects (No. ynbh20057), and Fundamental Research Joint Special Youth Project of Local Undergraduate Universities in Yunnan Province (No. 2018FH001-106).

Conflict of Interest(COI)

The authors have declared that no competing interests exist.

REFERENCES

1. J. R. Ticay-Rivas, M. del Pozo-Banos, W. G. Eberhard, J. B. Alonso, and C. M. Travieso, "Spider specie identification and verification based on pattern recognition of it cobweb,"

Expert systems with Applications, vol. 40, no. 10, pp. 4213-4225, 2013. https://doi.org/10.1016/j.eswa.2013.01.024
2. Y. Dolev and X. J. Nelson, "Innate pattern recognition and categorization in a jumping spider," *PLoS One*, vol. 9, no. 6, article no. e97819, 2014. https://doi.org/10.1371/journal.pone.0097819
3. D. L. Clark and G. W. Uetz, "Video image recognition by the jumping spider, Maevia inclemens (Araneae: Salticidae)," *Animal Behaviour*, vol. 40, no. 5, pp. 884-890, 1990. https://doi.org/10.1016/S0003-3472(05)80990-X
4. H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, "mixup: beyond empirical risk minimization," 2017 [Online]. Available: https://arxiv.org/abs/1710.09412.
5. W. Y. Chen, Y. C. Liu, Z. Kira, Y. C. F. Wang, and J. B. Huang, "A closer look at few-shot classification," 2019 [Online]. Available: https://arxiv.org/abs/1904.04232.
6. N. Mishra, M. Rohaninejad, X. Chen, and P. Abbeel, "Meta-learning with temporal convolutions," 2017 [Online]. Available: https://arxiv.org/abs/1707.03141v2.
7. S. Ioffe and C. Szegedy, "Batch normalization: accelerating deep network training by reducing internal covariate shift," 2015 [Online]. Available: https://arxiv.org/abs/1502.03167.
8. J. Wang, X. Chen, Z. Yang, C. Shi, Y. Zhang, and Z. Qian, "Influence of different data augmentation methods on model recognition accuracy," *Computer Science*, vol. 49, no. 6A, pp. 418-423, 2022. https://doi.org/10.11896/jsjkx.210700210
9. E. D. Cubuk, B. Zoph, D. Mane, V. Vasudevan, and Q. V. Le, "Autoaugment: learning augmentation policies from data," 2018 [Online]. Available: https://arxiv.org/abs/1805.09501.
10. Z. Zhong, L. Zheng, G. Kang, S. Li, and Y. Yang, "Random erasing data augmentation," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 7, pp. 13001-13008, 2020. https://doi.org/10.1609/aaai.v34i07.7000
11. X. Qin, Z. Zhang, C. Huang, C. Gao, M. Dehghan, and M. Jagersand, "BASNet: boundary-aware salient object detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Long Beach, CA, USA, 2019, pp. 7479-7489. https://doi.org/10.1109/CVPR.2019.00766
12. O. Ronneberger, P. Fischer, and T. Brox, "U-Net: convolutional networks for biomedical image segmentation," in *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015*. Cham, Switzerland: Springer, 2015, pp. 234-241. https://doi.org/10.1007/978-3-319-24574-4_28
13. V. Anitha and S. Murugavalli, "Brain tumour classification using two-tier classifier with adaptive segmentation technique," *IET Computer Vision*, vol. 10, no. 1, pp. 9-17, 2016. https://doi.org/10.1049/iet-cvi.2014.0193
14. M. Elhenawy, K. Young, A. Rakotonirainy, N. Haworth, R. Grazbieta, and A. Williamson, "Detecting driver distraction in the ANDS data using pre-trained models and transfer learning," in *Proceedings of the Australasian Road Safety Conference*, Adelaide, Australia, 2019.
15. M. Long, "Transfer learning: problems and methods," Ph.D. dissertation, Tsinghua University, Beijing, China, 2014.
16. K. Yao and W. Cao, "Trans-Net: stick figure recognition based on transfer learning" *Computer Engineering and Applications*, vol. 57, no. 3, pp. 182-188, 2021. https://doi.org/10.3778/j.issn.1002-8331.1911-0117
17. H. Ren, N. Kanhabua, A. Mogelmose, W. Liu, K. Kulkarni, S. Escalera, X. Baro, and T. B. Moeslund, "Back-dropout transfer learning for action recognition," *IET Computer Vision*, vol. 12, no. 4, pp. 484-491, 2018. https://doi.org/10.1049/iet-cvi.2016.0309
18. Y. Zhu, "Research and implementation of multi-view facial expression recognition based on generative adversarial networks," Master's thesis, Nanjing University of Posts and Telecommunications, Nanjing, China, 2009. https://doi.org/10.27251/d.cnki.gnjdc.2019.000597.
19. L. Chen, C. Song, Y. Fan, and H. Zhang, "Research on target location algorithm of multi-sensor based on distributed information fusion," *Command Control & Simulation*, vol. 42, no. 2, pp. 28-33, 2020. https://doi.org/10.3969/j.issn.1673-3819.2020.02.006
20. Y. Zhou, X. Zhang, Y. Wang, and B. Zhang, "Transfer learning and its application research," *Journal of Physics: Conference Series*, vol. 1920, no. 1, article no. 012058, 2021. https://doi.org/10.1088/1742-6596/1920/1/012058
21. K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Las Vegas, NV, USA, 2016, pp. 770-778. https://doi.org/10.1109/CVPR.2016.90
22. Y. He, T. Li, R. Ge, J. Yang, Y. Kong, J. Zhu, H. Shu, G. Yang, and S. Li, "Few-shot learning for deformable medical image registration with perception-correspondence decoupling and reverse teaching," *IEEE Journal of Biomedical and Health Informatics*, vol. 26, no. 3, pp. 1177-1187, 2022. https://doi.org/10.1109/JBHI.2021.3095409
23. V. Nair and G. E. Hinton, "Rectified linear units improve restricted Boltzmann machines," in *Proceedings of the 27th International Conference on Machine Learning (ICML)*, Haifa, Israel, 2010, pp. 807-814.
24. X. Ma, P. Tang, L. Zhao, and Z. Zhang, "A review of research on deep learning image data augmentation methods," *Journal of Image and Graphics*, vol. 26, no. 3, pp. 487-502, 2021. https://doi.org/10.11834/jig.200089
25. X. Chen and L. Jiang, "A method for improving generalization of neural network based on multi-heads attention," *Software Guide*, vol. 20, no. 5, pp. 34-38, 2021. https://doi.org/10.11907/rjdk.201854
26. Y. Huang, X. Wen, H. Ren, and J. Wang, "Application of deep transfer learning in zither classification," *Computer Engineering and Applications*, vol. 57, no. 10, pp. 218-224, 2021. https://doi.org/10.3778/j.issn.1002-8331.2009-0357
27. Y. Zeng, T. Dai, B. Chen, S. T. Xia, and J. Lu, "Correlation-based structural dropout for convolutional neural networks," *Pattern Recognition*, vol. 120, article no. 108117, 2021. https://doi.org/10.1016/j.patcog.2021.108117
28. S. Gidaris and N. Komodakis, "Dynamic few-shot visual learning without forgetting," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Salt Lake City, UT, USA, 2018, pp. 4367-4375. https://doi.org/10.1109/CVPR.2018.00459
29. S. Wang, Z. Hou, and J. Sun, "Difference detection method of adversarial samples oriented to deep learning," *Journal of Computer Applications*, vol. 41, no. 7, pp. 1849-1856, 2021. https://doi.org/10.11772/j.issn.1001-9081.2020081282
30. Z. Wang, A. Jiang, and O. Muhammad, "Unsupervised feature selection method based on regularized mutual representation," *Journal of Computer Applications*, vol. 40, no. 7, pp. 1896-1900, 2020. https://doi.org/10.11772/j.issn.1001-9081.2019122075

Jianming Wang

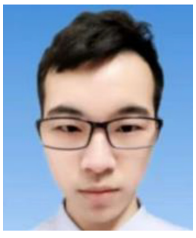
Jianming Wang received B.S. and M.S. degrees from the Department of Computer and Information Science and the School of Resources at Southwest Forestry University in 2010 and 2013, respectively, and received Ph.D. degree from the School of Information at Beijing Forestry University in 2017. Since July 2017, he has been an associate professor for the Department of Mathematics and Computer Science at Dali University, Dali, China. His research focuses on artificial intelligence, image recognition, geographic information systems, and forestry statistical modeling.

Longfeng Deng

Longfeng Deng is a master's student majoring in computer technology at the Department of Mathematics and Computer Science, Dali University since 2022. His research interests are in deep learning and forestry statistical modeling.

Chenyang Shi

Chenyang Shi received M.S. degree in Computer technology from the Department of Mathematics and Computer Science of Dali University in 2021. His research interests are in deep learning and computer vision such as image classification.



Guosheng Ye

Guosheng Ye is a master's student majoring in computer technology at the Department of Mathematics and Computer Science, Dali University since 2021. His research interests are in deep learning and computer vision.

Zizhong Yang

Zizhong Yang received B.S. and Ph.D. degrees in Biology Education and Zoology from the College of Life Sciences of Yunnan Normal University in 1994 and 2006, respectively. Since 2005, he has been a professor at the School of Pharmacy, Dali University, Dali, China. His research focuses on the construction of medicinal insect resources in Southwest China, the investigation and classification of spider resources in Yunnan, and the systematics, toxins, and evolution of the spiders of the Macrotheloidae family in China.