Hindawi
Journal of Control Science and Engineering
Volume 2021, Article ID 2518837, 19 pages
https://doi.org/10.1155/2021/2518837

Research Article

Deep Transfer Learning for Biology Cross-Domain Image Classification

Chunfeng Guo,[1] **Bin Wei**,[2,3] **and Kun Yu**[1]

[1]*Shandong Foreign Trade Vocational College, Qingdao 266100, China*
[2]*The Affiliated Hospital of Qingdao University, Qingdao 266071, China*
[3]*Shandong Key Laboratory of Digital Medicine and Computer Assisted Surgery, Qingdao 266071, China*

Correspondence should be addressed to Chunfeng Guo; qdgcf@163.com

Received 25 April 2021; Revised 10 November 2021; Accepted 22 November 2021; Published 15 December 2021

Academic Editor: Radek Matušů

Automatic biology image classification is essential for biodiversity conservation and ecological study. Recently, due to the record-shattering performance, deep convolutional neural networks (DCNNs) have been used more often in biology image classification. However, training DCNNs requires a large amount of labeled data, which may be difficult to collect for some organisms. This study was carried out to exploit cross-domain transfer learning for DCNNs with limited data. According to the literature, previous studies mainly focus on transferring from ImageNet to a specific domain or transferring between two closely related domains. While this study explores deep transfer learning between species from different domains and analyzes the situation when there is a huge difference between the source domain and the target domain. Inspired by the analysis of previous studies, the effect of biology cross-domain image classification in transfer learning is proposed. In this work, the multiple transfer learning scheme is designed to exploit deep transfer learning on several biology image datasets from different domains. There may be a huge difference between the source domain and the target domain, causing poor performance on transfer learning. To address this problem, multistage transfer learning is proposed by introducing an intermediate domain. The experimental results show the effectiveness of cross-domain transfer learning and the importance of data amount and validate the potential of multistage transfer learning.

1. Introduction

Building accurate knowledge of the identity, taxonomy, the geographic distribution, and the evolution of living species are essential for a sustainable development of humanity as well as for biodiversity conservation.

In terrestrial ecosystems, plants are extremely complex and diverse, and there are millions of different plant species [1, 2]. For us, plants must be classified into identifiable groups in order to have a clear, organized way of identifying the diverse array of plants and some specific applications such as weed control [3, 4].

Besides, the study of marine ecosystems is vital for global climate and environment protection [5–8]. There are many kinds of organisms in the marine worth studying, such as fish and plankton, which play an important role in the ecosystem [9] and the marine food chain [10].

At the very beginning, species classification was usually implemented on morphological diagnoses provided by taxonomic studies [11] in a manual identification process. However, for some species like weed plants and plankton, only experts such as taxonomists and trained technicians can identify taxa accurately. Furthermore, one expert may only identify a limited number of species in a specific domain (such as only species of weeds or phytoplankton) because it requires special skills acquired through extensive experiences [3, 12]. At the same time, there is an increasing shortage of skilled taxonomists [13]. The declining and partly nonexistent taxonomic knowledge within the general public has been termed "taxonomic crisis" [14], making great challenges to the future of biological study and conservation [11].

Using computer-based multimedia identification tools with computer vision and machine learning techniques

have been considered as promising solutions to classify organisms, and a lot of work has been done on this topic [15, 16].

1.1. Traditional Image Classification. The traditional image classification process can be generally divided into three steps: image preprocessing, feature extraction/description, and classification [17]. Some preprocessing techniques are often used in the image classification system for producing a suitable enhanced image for the next feature extraction step, such as image denoising, image enhancement, image segmentation, and so on [18]. Feature extraction refers to taking measurements, geometric or otherwise, of possibly segmented, meaningful regions in the image [19]. To characterize and describe some properties of the organism image by a set of values, computer vision experts have handcrafted a lot of features. In previous studies, some general features like size [20], color, shape context [21–24], invariant moments, granulometric features, co-occurrence matrix, Fourier descriptor, Gabor filters, local binary pattern (LBP) [25], histograms of oriented gradients (HOG), scale invariant feature transform (SIFT) etc., have been used commonly. There are also some features that have been designed for some specific species [26–28].

In the classification step, all extracted features are concatenated into a feature vector and then fed into the subsequent classifiers. Several kinds of traditional classifiers have been employed in previous studies, including k-nearest neighbor (kNN) [22], decision tree (DT) [22], random forest (RF) [29, 30], neural network (NN) [22, 23], support vector machines (SVM), and ensemble learning methods [12, 22, 31, 32].

However, the handcrafted features are usually lack of robustness and cannot represent the complex biomorphic characteristics of some organisms [12]. Besides, some features are elaborately handcrafted for specific organisms [33], which often perform poorly after being extended to other organisms. These traditional classifiers usually have not high prediction accuracy on different datasets [12]. Especially when the datasets are big or contain more than 20 categories, these classifiers may be limited by the "curse of dimensionality" [34], so that they are hard to be directly applied for ecological studies.

1.2. Deep Convolutional Neural Networks. In recent years, DCNNs [35–42] have become a mainstay of computer vision community due to their record-shattering performance in the ImageNet large-scale visual recognition challenge (ILSVRC) [43]. ImageNet is a large-scale image dataset with 1000 classes, containing 1.3 million training images, 50,000 validation images, and 100,000 testing images. DCNNs consist of a stack of learned convolution filters that extract hierarchical contextual image features, thus are high-capacity classifiers. With the high capability, DCNNs can find the relevant contextual image features in classification problems intelligently and are less likely to be restricted by the "curse of dimensionality. " Moreover, unlike traditional methods, DCNNs do not need to divide the training process into several steps but use end-to-end learning mechanism, which is more suitable for real applications. The outstanding performance of DCNNs in image classification and other problems has received unprecedented attention, prompting scholars to apply them to various practical problems including biology image classification [3, 44–48]. Nevertheless, the very large number of parameters in DCNNs requires large-scale annotated training data. For some organisms inhabiting a complex environment, such as some marine and even microscopic organisms, it is very difficult to collect their images. For another thing, the collected data can only be used after being precisely classified by experienced experts. While the experienced experts are often scarce and one expert can only identify a limited number of species in a specific domain (such as only species of weeds or phytoplankton) [12], the data available in practical studies may be insufficient to fully exploit the potential of DCNNs.

1.3. Transfer Learning with DCNNs. Transfer learning aims to transfer knowledge between the source domain and the target domain [49]. In biology image classification or some other scenarios, obtaining training data might be difficult and expensive. However, transfer learning can overcome the deficit of training examples in some domains by adapting classifiers trained on another domain [50]. There are two ways to apply transfer learning with DCNNs. One is treating the DCNN as a big feature extractor and utilizing the pretrained network with learning weights to extract features that would be subsequently used in a new domain. The outputs of the DCNN are considered as high-level features and are then fed into the following classifier. Another is to fine-tune the network weights by training the network with the data from the new domain. In this case, the dimension of the output layer must be changed to match the number of classes in the new domain dataset. There are some studies about biology image classification using transfer learning. Ge et al. [51] learned a domain-generic DCNN for the task of plant classification, by applying transfer learning on the parameters of the GoogLeNet [37] model (pretrained on the large-scale ImageNet dataset) using all of the training data for the plant classification task. Lee et al. [52] incorporated transfer learning by pretraining DCNN with class-normalized data and fine-tuning with original data.

Orenstein and Beijbom [53] built on the insights from Kaggle's National Data Science Bowl (NDSB) and investigated how DCNNs perform on several datasets of *in situ* plankton images, and their study suggests that weights from a highly tuned network for one planktonic image set could be used effectively in another plankton domain. Ge and Yu [54] introduced a source-target selective joint fine-tuning scheme for improving the performance of deep learning tasks with insufficient training data. Their idea is to identify and use a subset of training images from the original source learning task whose low-level characteristics are similar to those from the target learning task and jointly fine-tune shared convolutional layers for both tasks.

Previous studies about transfer learning with DCNNs mainly focused on the tasks which transfer from ImageNet

to a specific domain or transfer between two closely related domains [53]. Only a few studies exploited the transfer learning between two domains that are not directly related. When applying transfer learning to biology image classification, the different distance between species in the source domain and the target domain may have different effects on the performance. Although there is a certain biological distance between the two domains, they may share some common patterns in the view of DCNNs.

In this paper, inspired by the analysis of the literature and practical applications, deep transfer learning for biology cross-domain image classification is explored. By analyzing the experimental results on image datasets in different biology domains, including flowers, plant seedlings, plankton, and fish, some interesting conclusions are drawn. The main contributions of this paper can be listed as follows:

(1) A multiple transfer learning scheme is designed to explore deep transfer learning for biology cross-domain image classification. Following this scheme, deep transfer learning is applied among datasets in multiple domains.

(2) Even there is no clear relationship between the species from the source domain and the target domain, deep transfer learning can also be applied to improve the DCNNs performance. The features learned by DCNNs in one domain are high-level and can be transferred to another domain with a large distance.

(3) Fine-tuning on ImageNet usually gets a better result than training from scratch. However, for some datasets which have huge differences from ImageNet, using ImageNet as the source domain dataset may not get the results improved. Based on multiple transfer learning, multistage transfer learning is proposed to address this issue. In the first stage, the DCNNs on ImageNet is pre-pretrained; in the second stage, the DCNNs on an intermediate domain is pretrained, aiming to make the features adapted to the target domain; finally, the DCNNs on the target domain is fine-tuned to get a possible better result.

The experimental results show the potential of cross-domain transfer learning and may provide some ideas for other people who use transfer learning to study biology image classification or other related issues.

2. Methods

To exploit deep transfer learning for biology cross-domain image classification, multiple transfer learning scheme and propose multistage transfer learning are designed to train DCNNs with several datasets from different domains.

2.1. Deep Convolutional Neural Networks (DCNNs). Several DCNNs were trained on datasets from different domains, including AlexNet [35], VGG-16 [36, 55], GoogLeNet v3 [39], ResNet [40, 56, 57] with 18, 34, 50, 101 and 152 layers. Table 1 shows their depths, parameter numbers and performances on ImageNet dataset.

AlexNet [35] consists of five convolutional layers and three fully connected layers. There are three max-pooling layers of 3×3 after layers 1, 2, and 5. In the first layer, the 3 channels in the filters correspond to the red, green and blue components of the input image. The local response normalization (LRN) [35] was dropped in our implementation, which was introduced in AlexNet but was no longer used in subsequent DCNNs as it was replaced with batch normalization [38].

VGG-16 [36] consists of 13 convolutional layers and 3 fully-connected layers. In order to increase the depth of the network, the small (3×3) convolution filters are used in all convolutional layers.

GoogLeNet [37] has 22 layers, which consist of three convolutional layers, nine inception layers (each of which is two convolutional layers deep), and one fully connected layer. The inception layer is composed of parallel connections with different sized filters, including 1×1, 3×3, and 5×5, along with 3×3 max-pooling, are used for each parallel connection. The outputs of each connection in the inception module are concatenated together as the inception output. Using multiple filter sizes has the effect of processing the input at multiple scales. In order to reduce the number of weights, 1×1 filters are applied as a "bottleneck" to reduce the number of channels for each filter. GoogLeNet has multiple versions while batch normalization was introduced in the second version, and the most popular version, as known as GoogLeNet v3, is used in this paper. GoogLeNet v3 decomposes the convolutions by using smaller 1-D filters to reduce the number of weights to go deeper.

As the error back-propagates through the network, the gradient shrinks, which affects the ability to update the parameters in the earlier layers for very deep networks. To deal with the vanishing gradient problem, ResNet uses residual connections. ResNet introduces a "shortcut" module which contains an identity connection so that the "weight" layers (the layers that contain parameters) can be skipped. Rather than learning the function for the weight layers, the shortcut module learns the residual mapping. The "bottleneck" approach used in GoogLeNet, which uses 1×1 convolution to reduce the number of weight parameters, is also used in ResNet. The ResNet can be implemented with different layers. In this paper, ResNet with 18, 34, 50, 101, and 152 is built.

2.2. Rectified Linear Unit. Rectified Linear Unit (ReLU) activation function is applied to the output of every convolutional layer in all DCNNs used in this paper. The ReLU activation function can be described by the following equation:

$$f(z) = \max(0, z), \tag{1}$$

where z indicates the input of ReLU activation function. The ReLU activation function can make DCNNs more sparse. For example, in a randomly initialized network, only about 50% of hidden units $(z < 0)$ are activated (having nonzero

Table 1: The depth and the number of parameters of DCNNs.

Model	Depth	No. of parameters (M)	Top-1 accuracy (%) on ImageNet
AlexNet	8	57.5	56.43
VGG-16	16	134.8	71.64
GoogLeNet-v3	48	25.4	77.29
ResNet-18	18	11.2	70.14
ResNet-34	34	21.3	73.55
ResNet-50	50	23.8	76.00
ResNet-101	101	42.7	77.44
ResNet-152	152	58.4	78.43

output) simultaneously. Another benefit of ReLU is that it reduces the likelihood of vanishing gradient. This arises when $z < 0$, the gradient has a constant value, which results in faster learning of the DCNNs.

2.3. Dropout. Dropout is a technique to reduce overfitting, which sets the output of each hidden neuron to zero with a probability. The neurons which are "dropped out" in this way do not contribute to the forward pass and do not participate in back-propagation of the training process. Every time an input is presented, the neural network samples a different architecture, but all these architectures share weights. Dropout in the fully-connected layers of AlexNet and VGG is employed.

2.4. Batch Normalization. Batch Normalization [38] speed up the training process and improve accuracy by controlling the input distribution across layers. To this end, the distribution of the layer input activations (μ, σ) are normalized such that it has a zero mean and a unit standard deviation, which can be described as

$$y = \frac{x - \mu}{\sqrt{\sigma^2 + \epsilon}} \gamma + \beta, \tag{2}$$

where μ and σ indicate the mean and standard deviation of the distribution of layer input activations, γ and β are parameters that can be learned from training, and ϵ is a small constant to avoid numerical problems.

2.5. Softmax. Softmax function is employed after the output layer, which is a fully connected layer with K units. Here K indicates the number of classes in the image classification task, which has the same meaning in equation (3). The output of the softmax can represent a probability distribution over all the predicted classes, which is computed by

$$f(x_i) = \frac{e^{x_i}}{\sum_{i=0}^{K-1} e^{x_i}}, \tag{3}$$

where x_i represents the output of the i-th unit in the last fully-connected layer and i ranges from 0 to $K - 1$.

2.6. Data Augmentation. By enlarging the dataset using label-preserving transformations [35, 39] artificially, data augmentation is the easiest way to reduce overfitting on image data. There are three forms of data augmentation in our classification system: feature normalization, image resizing/cropping, and image horizontal flipping. It has been proved that feature normalization can make the gradient descent converge faster [38]. During both the training phase and the test phase, when image data are fed into the system, the system will do feature normalization for each channel of the image, respectively,

$$x_c' = \frac{x_c - \mu_c}{\sigma_c}, \tag{4}$$

where x_c indicates the c-th channel of the input image; μ_c and σ_c indicate the mean and standard deviation in the c-th channel among all the images in the training set, respectively; x_c' indicates the c-th channel of the normalized input image.

2.7. Pipeline and Experiment Details. All the DCNNs in this paper are implemented with PyTorch deep learning framework. For GoogLeNet v3 network, firstly the input image will be resized to 342×342 and then be cropped into 299×299; for other networks, the input image will be resized to 256×256 and then be cropped into 224×224. To prevent substantial overfitting [35], different methods of cropping are employed during the training phase and the test phase. During training phase, randomly cropping are employed by extracting random 224×224 patches (for GoogLeNet v3 network, it is 299×299) from the 256×256 images (for GoogLeNet v3 network, it is 342×342). Then, randomly horizontally flip these patches and feed them into the network for training. During the test phase, for each image in the test set, only need to predict once, the foreground organisms in the image are more likely to appear in the center. So, only center cropping is employed.

All the first convolutional layers of the DCNNs in this paper have three channels, corresponding to the three channels of an RGB image. Except for GoogLeNet v3, all the inputs to the DCNNs are fixed-sized $224 \times 224 \times 3$ images. While for GoogLeNet v3, the input image size is fixed to $299 \times 299 \times 3$. If a single-channel gray image is an input to the DCNN, it will be converted to an RGB image with three same channels, whose values are copied from the single-channel image.

To get more details of our experiments, please visit our open-sourced repository BioTL [58] on GitHub.

2.8. Training from Scratch. The DCNNs training procedure generally follows Krizhevsky et al. [35]. The initialization of the network weights is important because the bad initialization can

stall learning due to the instability of gradients in DCNNs [36]. The biases with zero and the weights are initialized in all the convolutional layers with $\mathcal{N}(0, 2/n)$, where n is the product of the size and the number of channels of the filters in the layer. A weight decay of 10^{-4} and a minibatch size of 16 is used. The learning rates of AlexNet and VGG-16 are both initialized to 10^{-3}, while the learning rates of all other DCNNs are initialized to 10^{-2}. With the initial learning rate, all DCNNs are trained up to 300 epochs, during which every 100 epochs divide the learning rate by 10.

2.9. Cross-Domain Transfer Learning

2.9.1. Fine-Tuning on ImageNet. To fully utilize the potential of DCNNs with small amounts of data, we use ImageNet as the source domain and apply transfer learning to transfer the knowledge learned from ImageNet to the target domain. The operations of data augmentation are the same with training from scratch. Instead of initializing all the weights randomly, they are initialized them (except the last fully-connected layer) with the weights learned from ImageNet dataset. Because the number of classes in the target task may differ from the ImageNet's 1000 classes, which corresponds to the output dimension of the last fully-connected layer, the weights of the last fully-connected layer in the pretrained modeled were dropped.

2.9.2. Multiple Transfer Learning. To exploit the deep transfer learning for biology image classification, a multiple transfer learning scheme is designed.

The multiple transfer learning scheme is designed to apply transfer learning several times on multiple source domains to observe the effect of cross-domain. For example, at first, a DCNN model on the Flowers17 dataset is trained, which is considered as the source domain. Secondly, all the weights of the trained model are used except the last fully-connected layer to initialize a new model with the same architecture. This is because the dimension of the output in the last fully-connected layer corresponds to the number of classes in the classification task. While the number of classes in the source domain is often different from that in the target domain, so the last fully-connected layer needs to be rebuilt to fit the new task. At last we train the new model with initialized weights on the target domain dataset, such as QUT Fish.

In practice, first of all, ImageNet dataset is used as the source domain and then fine-tune the pretrained models on the five target domain datasets (Flowers17, Flowers102, Plant Seedlings, PlanktonSet 1.0, and QUT Fish). After that, to exploit the effects of different distance between species from the source domain and the target domain, different combinations from the five datasets to apply transfer learning is chosen.

2.9.3. Multistage Transfer Learning. There may be a huge difference between the source domain dataset and the target domain dataset, causing the knowledge learned from the source domain cannot be well-transferred. If the data in the intermediate domain can adapt the learned features to fit the target domain, the hindering effect will not be particularly noticeable or the performance may be improved. To make the knowledge learned from the source domain more transferable, the multistage transfer learning is proposed. To perform multistage transfer learning, to add an intermediate domain between the source domain and the target domain is needed.

In Figure 1, a diagram is used to demonstrate the multistage transfer learning framework. In Figure 1, "CONV 1" to "CONV N" blocks indicate N convolutional layers in the DCNN model, "FC" block indicates the fully connected layer in the DCNN model. As shown in Figure 1, the proposed multistage transfer learning consists of three stages: prepretrain the models on ImageNet which is considered as the source domain; pretrain the models in an intermediate domain; and fine-tune the models in the target domain. We do not know how to find the best intermediate domain dataset, so followed multiple transfer learning scheme with a grid search to try different datasets as the intermediate domain. Considering on the computational cost consideration, only multistage transfer learning on ResNet-18, ResNet-34, and ResNet-50 three models are explored, which have the similar structures but different depths.

3. Datasets

In this paper, to exploit cross-domain transfer learning, several datasets that come from different domains are choosed, including Oxford Flowers, Plant Seedlings, PlanktonSet 1.0, and QUT Fish.

3.1. Oxford Flowers. There are two versions of Oxford Flowers datasets, Oxford Flowers 17 (Flowers17) and Oxford Flowers 102 (Flowers102). Flowers17 contains 17 classes of flowers, with 80 images in each class which was chosen to be indistinguishable solely by color. Flowers102 dataset consists of 102 classes represented by 40 to 258 images per class and 8189 images in total. There are about 45% of the Flowers17 images are also part of the Flowers102, so Flowers17 is not simply a subset of Flowers102. The image examples of these two datasets are shown in Figures 2 and 3, in which the images in the same row come from the same class and images from different rows come from different classes. According to the recommendation in the official datasets documents, the datasets are splitted into the training set, validation set, and test set, respectively.

3.2. Plant Seedlings. Plant Seedlings [4] dataset contains images of approximately 960 unique plants belonging to 12 species at several growth stages. There are three versions of Plant Seedlings dataset: original raw images, automatically segmented plants, and single plants that are not segmented. The version of non-segmented single plants, which contains 4750 images from 12 species, was used in the Kaggle competition of Plant Seedlings Classification in 2017. In this paper, the version of the nonsegmented single plants is used with 5-fold cross-validation. Figure 4 shows some image examples from Plant Seedlings dataset, in which the images in the same rectangle belong to the same class.

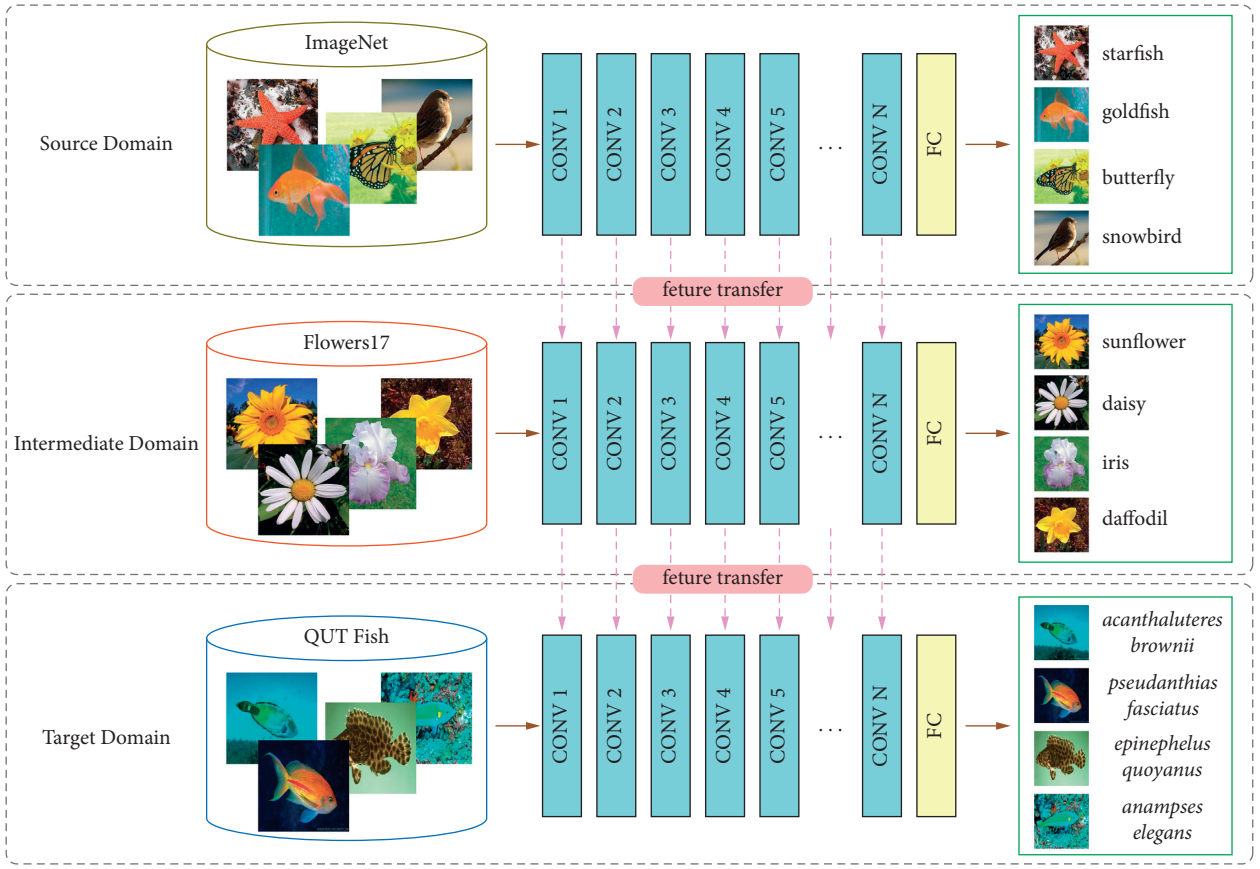

Figure 1: The framework of multistage transfer learning.

Figure 2: Image examples from Flowers17 dataset.

Figure 3: Image examples from Flowers102 dataset.

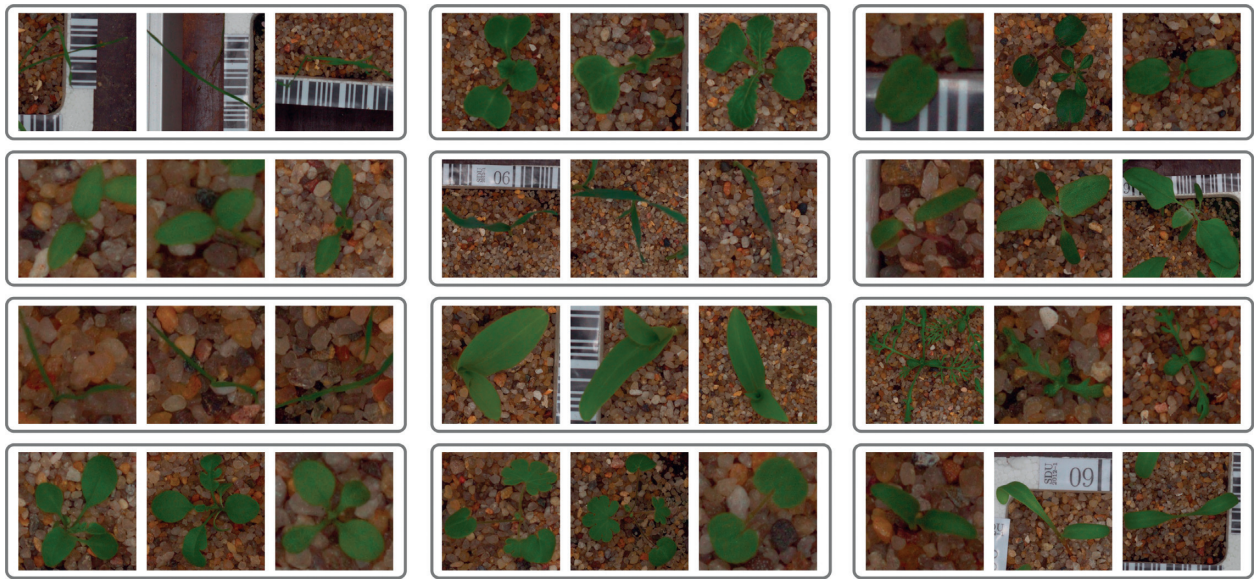

Figure 4: Image examples from Plant Seedlings dataset.

3.3. PlanktonSet 1.0. PlanktonSet 1.0 [59] is a medium-sized dataset with a fair amount of complexity, which was used in the National Data Science Bowl hosted by Kaggle in 2015. Image segments extracted from the raw data contains 60 736 images in total are sorted into 121 plankton classes and split into a training dataset and test dataset with a ratio of 1 : 1. The images obtained using the camera were already processed by a segmentation algorithm to classify and isolate individual organisms and then cropped accordingly, which can be seen in Figure 5. The image samples demonstrate that there are high intraclass variance and small interclass variance among some plankton species.

3.4. QUT Fish. QUT Fish [60] consists of 3960 images collected from 482 fish species. The data contain real-world

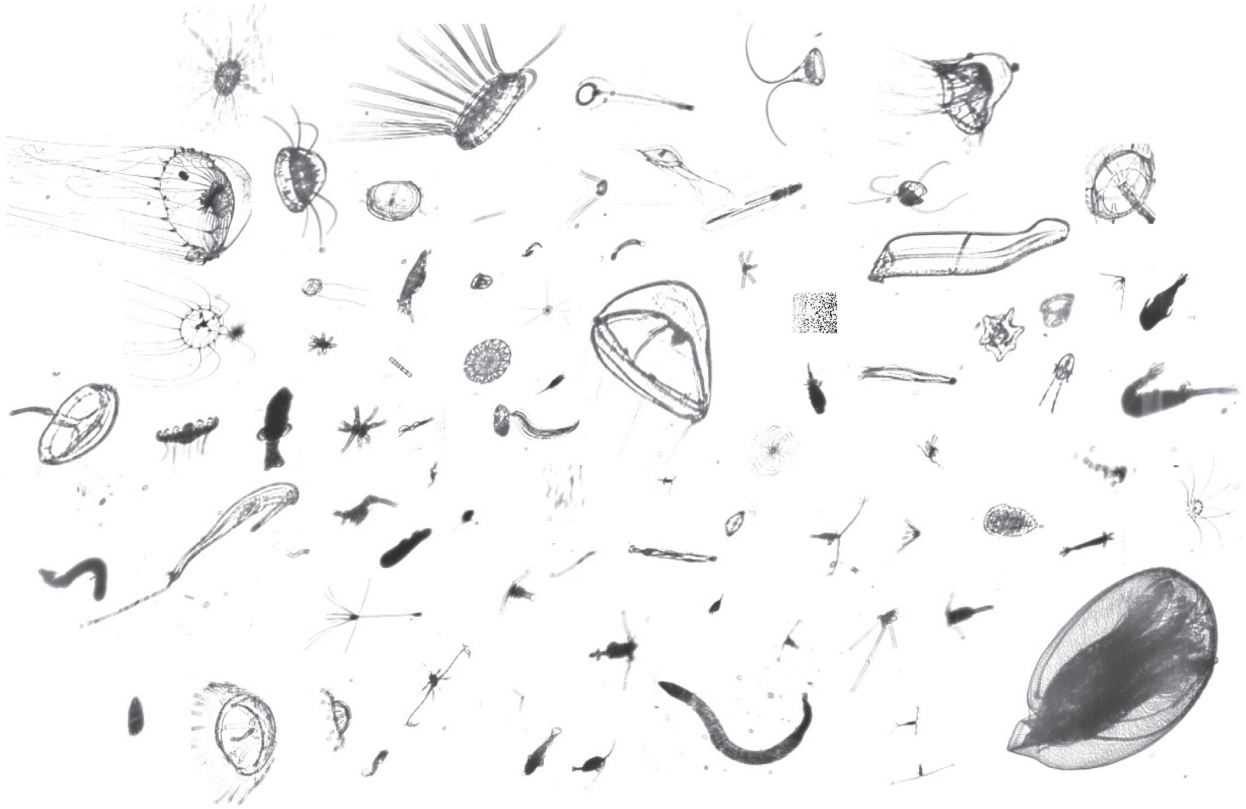

Figure 5: Image examples from PlanktonSet 1.0 dataset.

images of fish captured in conditions defined as "controlled," "*in situ*," and "out-of-the-water" shown in Figure 6. Since "controlled" images are captured with a controlled background and high quality, when splitting the dataset, to split "controlled" images into training set while splitting "*in situ*" and "out-of-the-water" images with low-quality and pose variations are tended into the test set. At last, QUT Fish dataset is splitted into the training set and the test set with a ratio 1 : 1. As a result of there are some classes in this dataset that only contain two image examples, only 2-fold cross-validation on it can be applied.

Because the amount of training data plays a crucial rule in training DCNNs, the number of all training examples and the average number of training samples are listed in each class for above datasets in Table 2. From Table 2, the scales of all the datasets are small compared to ImageNet, which contains more than one million training image examples. For QUT Fish, the data is extremely scarce since on average there are only 4 training samples in each class. In Table 2, it is obvious that with the increase of the number of layers (depth), the performance of the DCNN on ImageNet is getting better and better. At the same time, the number of parameters is also increasing along with the layers.

4. Evaluation

In this paper, accuracy and F_{measure} as the evaluation metrics are used.

Accuracy is the most intuitive and frequently-used 2 performance measure of the classification task. Accuracy is simply a ratio of correctly predicted samples to the total samples so it can be easily calculated. Accuracy is a good measure if the datasets are symmetric, however, for some imbalanced datasets, accuracy may not reflect the real performance of the classifier. Most of the datasets used in this paper are imbalanced, such as Flowers102, Plant Seedlings, PlanktonSet 1.0, and QUT Fish. The distributions of these datasets can be seen in Figure 7. To evaluate the classification performance on imbalanced datasets, F_{measure} as another metric is used.

Both accuracy and F_{measure} can be calculated by the confusion matrix, which is a table containing information about actual and predicted classifications. As shown in Table 3 (Refer to Table 1 in Ref. [12]), each row of the confusion matrix represents the instances in a predicted class while each column represents the instances in an actual class. For a binary classifier, according to the true condition and predicted condition, the confusion matrix consists of four parts: true positive (TP), true negative (TN), false positive (FP), and false negative (FN). In this way, several measures can be derived from a confusion matrix:

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}, \tag{5}$$

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}, \tag{6}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}. \tag{7}$$

F_{measure} is the harmonic mean of precision and recall

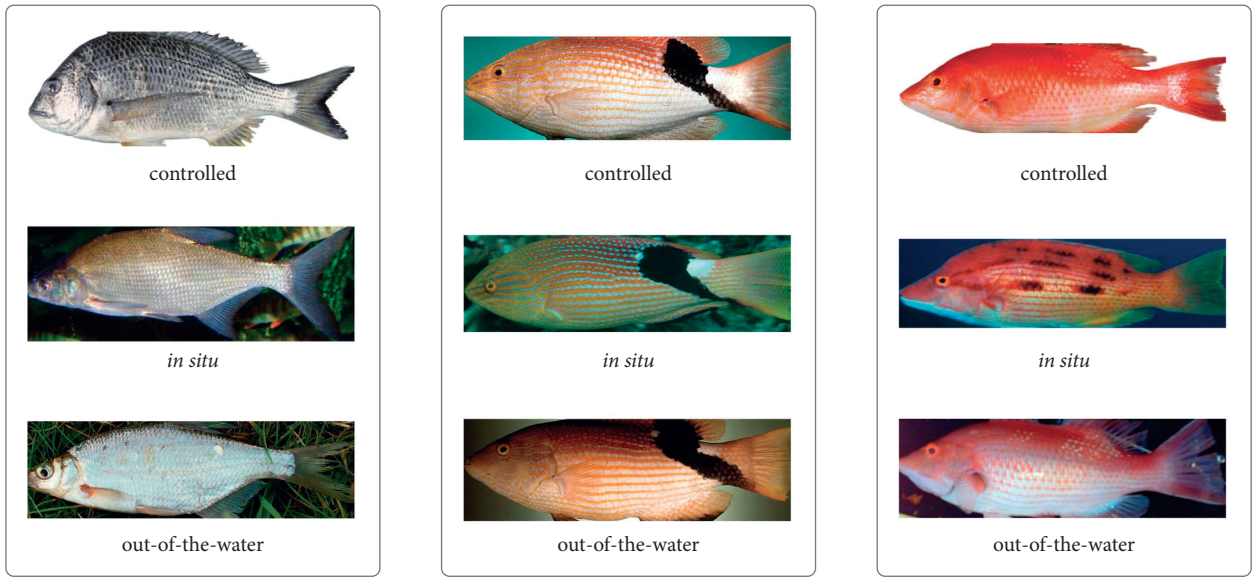

Figure 6: Image examples from QUT Fish dataset.

Table 2: Number of training examples in data sets.

Dataset	No. of classes	No. of training examples	Average no. of training examples in each class
Flowers17	17	680	40
Flowers102	102	1020	10
Plant Seedlings	12	3800	317
PlanktonSet 1.0	121	30 336	251
QUT Fish	482	1980	4
ImageNet	1000	1.3M	1300 (approx.)

$$\mathrm{F}_{\mathrm{measure}} = 2 \times \frac{\mathrm{precision} \times \mathrm{recall}}{\mathrm{precision} + \mathrm{recall}}. \quad (8)$$

Therefore, F_{measure} takes both *FP* and *FN* into account and is more useful than accuracy when we have an uneven class distribution.

5. Results

The multiple transfer learning scheme is designed to exploit deep transfer learning on Flowers17, Flowers102, Plant Seedlings, PlanktonSet 1.0, and QUT Fish datasets. When performing transfer learning, to make a comparison, DCNN models are also pretrained on ImageNet and then fine-tuned the models on the five datasets. For multiple transfer learning, one dataset from the above five datasets is chosen as the source domain and chosen another as the target domain. In the table of experimental results, *source domain* ⇒ *target domain* are used to illustrate the transfer process from *source domain* to *target domain*. To handle the problem of extremely insufficient data, multistage transfer learning is proposed, which introduces an intermediate domain between the source domain and the target domain. Then in the table of experimental results, *source domain* ⇒ *intermediate domain* ⇒ *target domain* are used to illustrate this multistage transfer process.

5.1. Training from Scratch. The classification results of all DCNNs training from scratch on the five datasets are listed in Tables 4 and 5. From the results, when training from scratch, ResNet-18 achieved the best performance on Flowers17 with 90.29% accuracy and 0.903 1 F_{measure}. Meanwhile, ResNet-18 achieved the best performance on the Plant Seedlings with 98.02% accuracy and 0.977 8 F_{measure}. The best performance on Flowers102 is achieved by ResNet-34, with 57.16% accuracy and 0.551 3 F_{measure}. Similarly, the best performance on QUT Fish is also achieved by ResNet-34, with 36.51% accuracy and 0.281 1 F_{measure}. There are relatively more data in PlanktonSet 1.0 so the DCNNs achieved better results on PlanktonSet 1.0 tend to be deeper. On PlanktonSet 1.0, the best accuracy 77.40% is achieved by ResNet-152 and the best F_{measure} 0.659 3 is achieved by ResNet-101.

5.2. Cross-Domain Transfer Learning. To make a comparison, experiments of fine-tuning the pretrained on ImageNet model are performed. The multiple transfer learning scheme is designed to apply transfer learning on several cross-domain datasets. Similar to fine-tuning on ImageNet, the weights from the model pretrained on the source domain is used to initialize a new DCNN and then fine-tune it on the target domain dataset. To adapt the features in pretrained on ImageNet models to fit the target domain well, multistage transfer learning is proposed by adding an intermediate

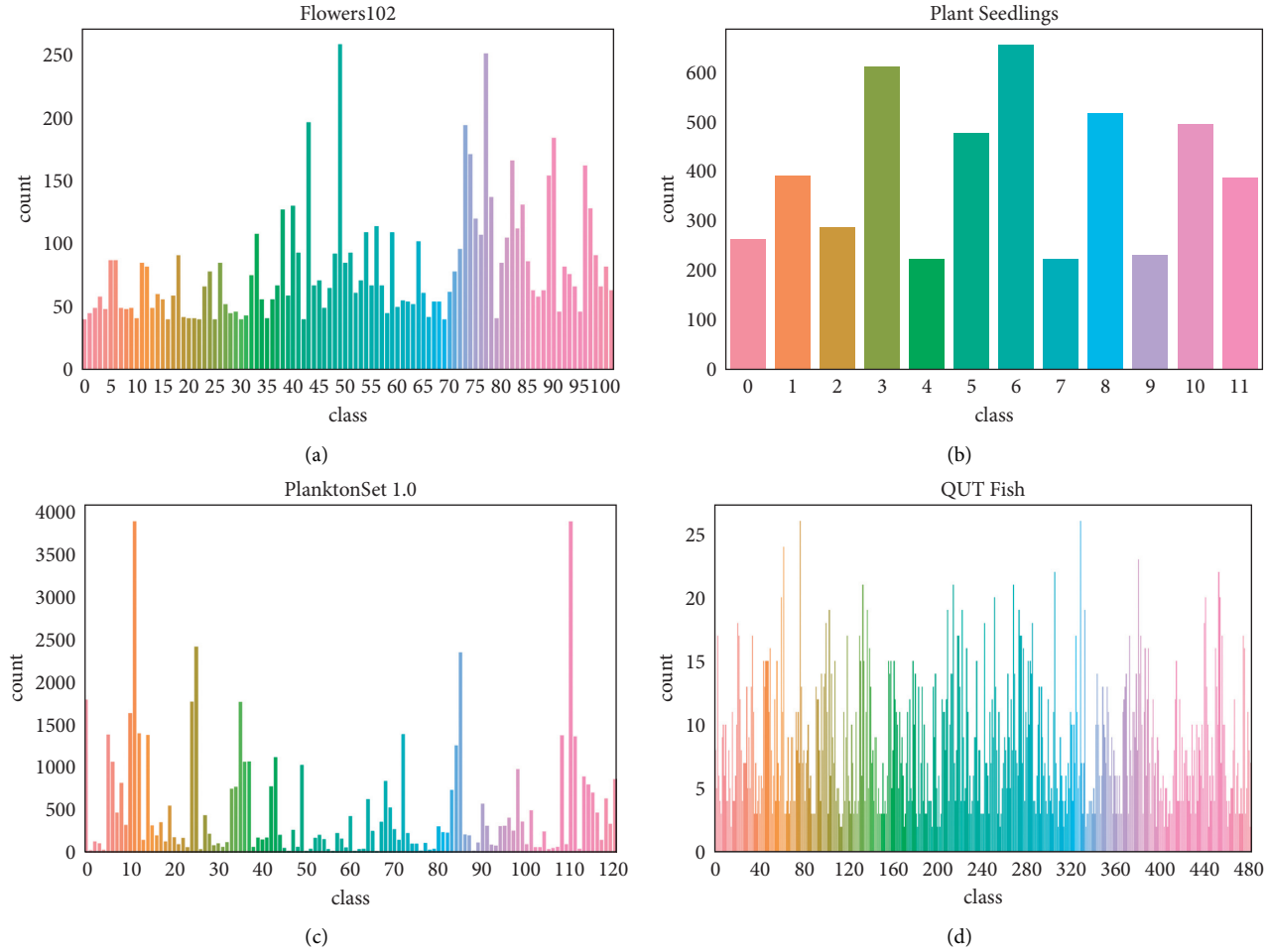

Figure 7: The distributions of four imbalanced datasets where (a) shows the Flowers102, (b) shows the Plant Seedings, (c) shows PlanktonSet 1.0, and (d) shows QUT Fish.

Table 3: Confusion matrix.

	Total population	Predicted condition	
		Prediction positive	Prediction negative
True condition	Condition positive	True positive (TP)	False negative (FN)
	Condition negative	False positive (FP)	True negative (TN)

domain between the source domain and the target domain. The multiple transfer learning results are shown in Tables 6 and 7, where the best source domain and intermediate domain datasets under the same transfer learning conditions are highlighted. To form a sharp contrast, in Table 8, how much gains the cross-domain transfer learning methods get compared with training from scratch is listed.

In the "transfer process" column, the entries in boldface indicate the source domain dataset or intermediate dataset with best classification results; in the "Accuracy (%)" column, the entries in boldface indicate the best performance with the highest accuracy; in the "F_{measure}" column, the entries in boldface indicate the best performance with the highest F_{measure}.

In the "Transfer process" column, the entries in boldface indicate the source domain dataset or intermediate dataset with best classification results; in the "Accuracy (%)" column, the entries in boldface indicate the best performance with the highest accuracy; in the "F_{measure}" column, the entries in boldface indicate the best performance with the highest F_{measure}.

5.3. Fine-Tuning on ImageNet. The results of fine-tuning on ImageNet are shown in Tables 9 and 10. Comparing with the training from scratch results in Tables 4 and 5, for Flowers17, Flower102, and QUT Fish dataset, every single model achieves a better performance after fine-tuning on ImageNet. For Flowers17, after fine-tuning on ImageNet, 9.52% accuracy and 0.091 1 F_{measure} were gained on average among all models; for Flowers102, there is a much better result, with 35.65% accuracy and 0.369 5 F_{measure} gain on average after fine-tuning on ImageNet; For QUT Fish, there is also a better result, with 17.96% accuracy and 0.174 6 F_{measure} gain on

Table 4: Accuracy (%) results of training from scratch.

Model	Flowers17	Flowers102	Plant Seedlings	PlanktonSet 1.0	QUT Fish
AlexNet	83.82	47.02	96.21	75.09	25.84
VGG-16	84.12	44.35	97.08	76.19	25.21
GoogLeNet-v3	87.94	55.44	97.48	77.12	32.92
ResNet-18	**90.29**	55.60	**98.02**	76.36	34.96
ResNet-34	89.71	**57.16**	97.68	76.73	**36.51**
ResNet-50	83.53	52.77	97.58	77.23	30.55
ResNet-101	83.24	50.76	97.47	77.33	30.06
ResNet-152	81.18	51.24	97.05	**77.40**	30.64

The entries in boldface indicate the best classification results with the highest accuracy.

Table 5: F_{measure} results of training from scratch.

Model	Flowers17	Flowers102	Plant Seedlings	PlanktonSet 1.0	QUT Fish
AlexNet	0.839 8	0.458 3	0.957 8	0.616 2	0.202 2
VGG-16	0.841 2	0.425 9	0.966 8	0.643 0	0.196 7
GoogLeNet-v3	0.879 2	0.533 9	0.974 8	0.658 1	0.256 8
ResNet-18	**0.903 1**	0.533 3	**0.977 8**	0.651 6	0.271 7
ResNet-34	0.897 9	**0.551 3**	0.974 0	0.658 5	**0.281 1**
ResNet-50	0.837 1	0.506 1	0.973 5	0.659 0	0.238 0
ResNet-101	0.832 7	0.486 1	0.971 6	**0.659 3**	0.235 7
ResNet-152	0.817 3	0.493 8	0.966 8	0.659 0	0.237 0

The entries in boldface indicate the best classification results with the highest F_{measure}.

Table 6: Results of cross-domain transfer learning on Flowers17.

		Transfer process			Model	Accuracy (%)	F_{measure}
		Flowers102	⇒	Flowers17	GoogLeNet-v3	90.00	0.9002
		Plant Seedlings	⇒	Flowers17	GoogLeNet-v3	88.529 4	0.8832
		PlanktonSet 1.0	⇒	Flowers17	GoogLeNet-v3	**92.06**	**0.9198**
		QUT Fish	⇒	Flowers17	GoogLeNet-v3	89.12	0.8912
ImageNet	⇒	**Flowers102**	⇒	Flowers17	GoogLeNet-v3	**97.94** †	**0.9795** †
ImageNet	⇒	Plant Seedlings	⇒	Flowers17	GoogLeNet-v3	95.88	0.9589
ImageNet	⇒	PlanktonSet 1.0	⇒	Flowers17	GoogLeNet-v3	96.47	0.9648
ImageNet	⇒	QUT Fish	⇒	Flowers17	GoogLeNet-v3	95.00	0.9500
		Flowers102	⇒	Flowers17	ResNet-18	88.82	0.8908
		Plant Seedlings	⇒	Flowers17	ResNet-18	90.88	0.9077
		PlanktonSet 1.0	⇒	Flowers17	ResNet-18	**92.06**	**0.9198**
		QUT Fish	⇒	Flowers17	ResNet-18	89.12	0.8912
ImageNet	⇒	**Flowers102**	⇒	Flowers17	ResNet-18	**97.06**	**0.9703**
ImageNet	⇒	Plant Seedlings	⇒	Flowers17	ResNet-18	93.82	0.9381
ImageNet	⇒	PlanktonSet 1.0	⇒	Flowers17	ResNet-18	93.53	0.9356
ImageNet	⇒	QUT Fish	⇒	Flowers17	ResNet-18	94.41	0.9436
		Flowers102	⇒	Flowers17	ResNet-34	90.88	0.9088
		Plant Seedlings	⇒	Flowers17	ResNet-34	89.41	0.8937
		PlanktonSet 1.0	⇒	Flowers17	ResNet-34	**92.65**	**0.9265**
		QUT Fish	⇒	Flowers17	ResNet-34	88.82	0.8884
ImageNet	⇒	**Flowers102**	⇒	Flowers17	ResNet-34	**97.65** †	**0.9763** †
ImageNet	⇒	Plant Seedlings	⇒	Flowers17	ResNet-34	95.00	0.9496
ImageNet	⇒	PlanktonSet 1.0	⇒	Flowers17	ResNet-34	93.53	0.9353
ImageNet	⇒	QUT Fish	⇒	Flowers17	ResNet-34	94.41	0.9441
		Flowers102	⇒	Flowers17	ResNet-50	84.41	0.8456
		Plant Seedlings	⇒	Flowers17	ResNet-50	89.41	0.8927
		PlanktonSet 1.0	⇒	Flowers17	ResNet-50	**92.06**	**0.9203**
		QUT Fish	⇒	Flowers17	ResNet-50	87.06	0.8689
ImageNet	⇒	Flowers102	⇒	Flowers17	ResNet-50	96.47	0.9650
ImageNet	⇒	Plant Seedlings	⇒	Flowers17	ResNet-50	96.18	0.9614
ImageNet	⇒	PlanktonSet 1.0	⇒	Flowers17	ResNet-50	94.12	0.9405
ImageNet	⇒	**QUT Fish**	⇒	Flowers17	ResNet-50	**96.76**	**0.9675**

†indicates the results outperform the corresponding results of training from scratch and fine-tuning on ImageNet.

Table 7: Results of cross-domain transfer learning on QUT Fish.

Transfer process					Model	Accuracy (%)	F_{measure}
		Flowers17	⇒	QUT Fish	GoogLeNet-v3	38.99	0.3093
		Flowers102	⇒	QUT Fish	GoogLeNet-v3	37.10	0.2899
		Plant Seedlings	⇒	QUT Fish	GoogLeNet-v3	39.36	0.3110
		PlanktonSet 1.0	⇒	QUT Fish	GoogLeNet-v3	**50.33**	**0.4139**
ImageNet	⇒	Flowers17	⇒	QUT Fish	GoogLeNet-v3	57.74	0.4725
ImageNet	⇒	**Flowers102**	⇒	QUT Fish	GoogLeNet-v3	**58.09**	**0.4828**
ImageNet	⇒	Plant Seedlings	⇒	QUT Fish	GoogLeNet-v3	54.17	0.4451
ImageNet	⇒	PlanktonSet 1.0	⇒	QUT Fish	GoogLeNet-v3	53.51	0.4515
		Flowers17	⇒	QUT Fish	ResNet-18	40.77	0.3178
		Flowers102	⇒	QUT Fish	ResNet-18	36.34	0.2807
		Plant Seedlings	⇒	QUT Fish	ResNet-18	38.08	0.2974
		PlanktonSet 1.0	⇒	QUT Fish	ResNet-18	**45.46**	**0.3637**
ImageNet	⇒	**Flowers17**	⇒	QUT Fish	ResNet-18	**51.99** †	**0.4191** †
ImageNet	⇒	Flowers102	⇒	QUT Fish	ResNet-18	51. 51†	0. 4130†
ImageNet	⇒	Plant Seedlings	⇒	QUT Fish	ResNet-18	48.99	0.3951
ImageNet	⇒	PlanktonSet 1.0	⇒	QUT Fish	ResNet-18	49.42	0.4051
		Flowers17	⇒	QUT Fish	ResNet-34	40.93	0.3223
		Flowers102	⇒	QUT Fish	ResNet-34	37.16	0.2882
		Plant Seedlings	⇒	QUT Fish	ResNet-34	38.92	0.3004
		PlanktonSet 1.0	⇒	QUT Fish	ResNet-34	**46.86**	**0.3782**
ImageNet	⇒	**Flowers17**	⇒	QUT Fish	ResNet-34	**52.65** †	0.4220
ImageNet	⇒	**Flowers102**	⇒	QUT Fish	ResNet-34	52.25	**0.4235**
ImageNet	⇒	Plant Seedlings	⇒	QUT Fish	ResNet-34	49.39	0.3956
ImageNet	⇒	PlanktonSet 1.0	⇒	QUT Fish	ResNet-34	49.54	0.4012
		Flowers17	⇒	QUT Fish	ResNet-50	35.74	0.2793
		Flowers102	⇒	QUT Fish	ResNet-50	31.61	0.2443
		Plant Seedlings	⇒	QUT Fish	ResNet-50	35.77	0.2721
		PlanktonSet 1.0	⇒	QUT Fish	ResNet-50	**39.98**	**0.3151**
ImageNet	⇒	**Flowers17**	⇒	QUT Fish	ResNet-50	**52.34** †	**0.4242**
ImageNet	⇒	Flowers102	⇒	QUT Fish	ResNet-50	51.96	0.4222
ImageNet	⇒	Plant Seedlings	⇒	QUT Fish	ResNet-50	49.22	0.3979
ImageNet	⇒	PlanktonSet 1.0	⇒	QUT Fish	ResNet-50	43.25	0.3477

†indicates the results outperform the corresponding results of training from scratch and fine-tuning on ImageNet.

average after fine-tuning on ImageNet. For Plant Seedlings, there is only improvement with 0.48% accuracy and 0.005 4 F_{measure} on average after fine-tuning on ImageNet, possibly because the original performances are good enough. For PlanktonSet 1.0, some of the results are improved while some of them declined, with 0.02% accuracy decrease and 0.000 2 F_{measure} gain on average after fine-tuning on ImageNet. The results reflect the huge difference between PlanktonSet 1.0 and ImageNet.

5.4. Multiple Transfer Learning. From Tables 6, 11 and 7, it shows that all the multiple transfer learning experiments can get make a better result than training from scratch on Flowers17, Flowers102, and QUT Fish. For these three datasets, using PlanktonSet 1.0 as the source domain can get better results than using other datasets. Specifically, on average, for Flowers17, using PlanktonSet 1.0 as the source domain can get the gain with 4.34% accuracy and 0.042 3 F_{measure} than training from scratch; for Flowers102, using PlanktonSet 1.0 as the source domain can get the gain with 14.84% accuracy and 0.148 9 F_{measure} than training from scratch; for QUT Fish, using PlanktonSet 1.0 as the source domain can get the gain with 11.92% accuracy and 0.105 8 F_{measure} than training from scratch. For the Flowers17 dataset, using Flowers102 as the source domain dataset gets the gain with 0.66% accuracy and 0.007 0 F_{measure} on average, which is much poorer than using PlanktonSet 1.0 as the source domain dataset; for Flowers102, using Flowers17 as the source domain gets the gain with 8.44% and 0.087 1 F_{measure} on average, which is also poorer than using PlanktonSet 1.0 as the source domain dataset.

In the "Transfer process" column, the entries in boldface indicate the source domain dataset or intermediate dataset with best classification results; in the "Accuracy (%)" column, the entries in boldface indicate the best performance with the highest accuracy; in the "F_{measure}" column, the entries in boldface indicate the best performance with the highest F_{measure}.

For Plant Seedlings, Table 12 shows that using Flowers17 as the source domain dataset can get the gain with 0.21% accuracy and 0.001 7 F_{measure} on average. Using Flowers102, PlanktonSet 1.0 and QUT Fish as the source domain dataset all get the result decreased.

In the "Transfer process" column, the entries in boldface indicate the source domain dataset or intermediate dataset with best classification results; in the "Accuracy (%)" column, the entries in boldface indicate the best performance

Table 8: Transfer learning results gains (average on GoogLeNet-v3, ResNet-18, ResNet-34 and ResNet-50) compared with the results of training from scratch.

Transfer process					Gain of accuracy (%)	Gain of $F_{measure}$
		ImageNet	⇒	Flowers17	9.42	0.0911
		Flowers102	⇒	Flowers17	0.66	0.0070
		Plant Seedlings	⇒	Flowers17	1.69	0.0150
		Plankton	⇒	Flowers17	4.34	0.0423
		QUT Fish	⇒	Flowers17	0.66	0.0053
ImageNet	⇒	Flowers102	⇒	Flowers17	9.41	0. 0935†
ImageNet	⇒	Plant Seedlings	⇒	Flowers17	7.35	0.0727
ImageNet	⇒	Plankton	⇒	Flowers17	6.54	0.0647
ImageNet	⇒	QUT Fish	⇒	Flowers17	7.28	0.0720
		ImageNet	⇒	Flowers102	34.83	0.3642
		Flowers17	⇒	Flowers102	8.44	0.0871
		Plant Seedlings	⇒	Flowers102	3.57	0.0367
		Plankton	⇒	Flowers102	14.84	0.1489
		QUT Fish	⇒	Flowers102	6.31	0.0591
ImageNet	⇒	Flowers17	⇒	Flowers102	33.97	0.3541
ImageNet	⇒	Plant Seedlings	⇒	Flowers102	27.62	0.2870
ImageNet	⇒	Plankton	⇒	Flowers102	21.84	0.2230
ImageNet	⇒	QUT Fish	⇒	Flowers102	27.67	0.2851
		ImageNet	⇒	Plant Seedlings	0.53	0.0053
		Flowers17	⇒	Plant Seedlings	0.21	0.0017
		Flowers102	⇒	Plant Seedlings	−0.01	−0.0007
		Plankton	⇒	Plant Seedlings	−0.35	−0.0044
		QUT Fish	⇒	Plant Seedlings	−0.40	−0.0051
ImageNet	⇒	Flowers17	⇒	Plant Seedlings	0.42	0.0043
ImageNet	⇒	Flowers102	⇒	Plant Seedlings	0.51	0.0054†
ImageNet	⇒	Plankton	⇒	Plant Seedlings	−0.32	−0.0041
ImageNet	⇒	QUT Fish	⇒	Plant Seedlings	0.39	0.0041
		ImageNet	⇒	Plankton	−0.17	−0.0005
		Flowers17	⇒	Plankton	−0.06	−0.0006
		Flowers102	⇒	Plankton	−0.09	−0.0023
		Plant Seedlings	⇒	Plankton	0. 08†	0. 0002†
		QUT Fish	⇒	Plankton	−1.18	−0.0189
ImageNet	⇒	Flowers17	⇒	Plankton	−0.76	−0.131
ImageNet	⇒	Flowers102	⇒	Plankton	−0.02	0.0036†
ImageNet	⇒	Plant Seedlings	⇒	Plankton	−0.08	0.0010†
ImageNet	⇒	QUT Fish	⇒	Plankton	−0.18	−0.0005
		ImageNet	⇒	QUT Fish	20.03	0.1759
		Flowers17	⇒	QUT Fish	5.38	0.0453
		Flowers102	⇒	QUT Fish	1.82	0.0139
		Plant Seedlings	⇒	QUT Fish	4.30	0.0333
		Plankton	⇒	QUT Fish	11.92	0.1058
ImageNet	⇒	Flowers17	⇒	QUT Fish	19.95	0.1725
ImageNet	⇒	Flowers102	⇒	QUT Fish	19.72	0.1735
ImageNet	⇒	Plant Seedlings	⇒	QUT Fish	16.71	0.1465
ImageNet	⇒	Plankton	⇒	QUT Fish	15.20	0.1395

† indicates the results outperform the corresponding results of training from scratch and fine-tuning on ImageNet.

Table 9: Accuracy (%) results of fine-tuning on ImageNet.

Model	Flowers17	Flowers102	Plant Seedling	PlanktonSet 1.0	QUT Fish
AlexNet	91.18	77.23	96.80	75.17	39.55
VGG-16	96.76	80.94	98.15	75.85	49.27
GoogLeNet-v3	97.06	**93.06**	98.19	76.49	**59.15**
ResNet-18	**97.96**	88.21	98.00	76.44	51.12
ResNet-34	96.47	88.73	98.19	76.55	52.54
ResNet-50	97.65	90.29	**98.53**	77.27	52.24
ResNet-101	97.06	90.55	98.44	**77.48**	52.78
ResNet-152	97.06	90.54	98.48	77.19	53.25

The entries in boldface indicate the best classification results with the highest accuracy.

Table 10: F_{measure} results of fine-tuning on ImageNet.

Model	Flowers17	Flowers102	Plant Seedlings	PlanktonSet 1.0	QUT Fish
AlexNet	0.912 6	0.764 2	0.964 8	0.628 3	0.311 7
VGG-16	0.967 5	0.800 4	0.980 1	0.645 1	0.397 2
GoogLeNet-v3	0.970 4	**0.927 1**	0.979 8	0.655 3	**0.489 6**
ResNet-18	0.970 7	0.878 1	0.978 0	0.650 5	0.409 7
ResNet-34	0.964 4	0.877 1	0.979 9	0.652 3	0.427 2
ResNet-50	0.976 3	0.899 6	**0.983 7**	0.667 0	0.424 6
ResNet-101	0.970 4	0.898 4	0.983 0	**0.669 3**	0.425 0
ResNet-152	**0.979 3**	0.900 1	0.983 2	0.666 4	0.430 6

The entries in boldface indicate the best classification results with the highest F_{measure}.

Table 11: Results of cross-domain transfer learning on Flowers102.

Transfer process					Model	Accuracy (%)	F_{measure}
		Flowers17	⇒	Flowers102	GoogLeNet-v3	64.11	0.6228
		Plant Seedlings	⇒	Flowers102	GoogLeNet-v3	61.42	0.5948
		PlanktonSet 1.0	⇒	Flowers102	GoogLeNet-v3	**73.95**	**0.7212**
		QUT Fish	⇒	Flowers102	GoogLeNet-v3	61.98	0.5948
ImageNet	⇒	**Flowers17**	⇒	Flowers102	GoogLeNet-v3	**91.19**	**0.9054**
ImageNet	⇒	Plant Seedlings	⇒	Flowers102	GoogLeNet-v3	85.01	0.8407
ImageNet	⇒	PlanktonSet 1.0	⇒	Flowers102	GoogLeNet-v3	79.17	0.7788
ImageNet	⇒	QUT Fish	⇒	Flowers102	GoogLeNet-v3	86.21	0.8533
		Flowers17	⇒	Flowers102	ResNet-18	66.06	0.6421
		Plant Seedlings	⇒	Flowers102	ResNet-18	59.34	0.5675
		PlanktonSet 1.0	⇒	Flowers102	ResNet-18	**72.39**	**0.7061**
		QUT Fish	⇒	Flowers102	ResNet-18	61.81	0.5932
ImageNet	⇒	**Flowers17**	⇒	Flowers102	ResNet-18	**87.88**	**0.8698**
ImageNet	⇒	Plant Seedlings	⇒	Flowers102	ResNet-18	81.36	0.8074
ImageNet	⇒	PlanktonSet 1.0	⇒	Flowers102	ResNet-18	75.09	0.7308
ImageNet	⇒	QUT Fish	⇒	Flowers102	ResNet-18	79.92	0.7821
		Flowers17	⇒	Flowers102	ResNet-34	65.67	0.6400
		Plant Seedlings	⇒	Flowers102	ResNet-34	59.39	0.5773
		PlanktonSet 1.0	⇒	Flowers102	ResNet-34	**72.30**	**0.7013**
		QUT Fish	⇒	Flowers102	ResNet-34	63.34	0.6072
ImageNet	⇒	**Flowers17**	⇒	Flowers102	ResNet-34	**88.50**	**0.8788** †
ImageNet	⇒	Plant Seedlings	⇒	Flowers102	ResNet-34	95.00	0.9496
ImageNet	⇒	PlanktonSet 1.0	⇒	Flowers102	ResNet-34	76.24	0.7477
ImageNet	⇒	QUT Fish	⇒	Flowers102	ResNet-34	80.84	0.7943
		Flowers17	⇒	Flowers102	ResNet-50	58.89	0.5688
		Plant Seedlings	⇒	Flowers102	ResNet-50	55.10	0.5324
		PlanktonSet 1.0	⇒	Flowers102	ResNet-50	**61.70**	**0.5922**
		QUT Fish	⇒	Flowers102	ResNet-50	59.10	0.5662
ImageNet	⇒	**Flowers17**	⇒	Flowers102	ResNet-50	**89.28**	**0.8877**
ImageNet	⇒	Plant Seedlings	⇒	Flowers102	ResNet-50	83.22	0.8199
ImageNet	⇒	PlanktonSet 1.0	⇒	Flowers102	ResNet-50	77.85	0.7600
ImageNet	⇒	QUT Fish	⇒	Flowers102	ResNet-50	84.68	0.8359

† indicates the results outperform the corresponding results of training from scratch and fine-tuning on ImageNet.

with the highest accuracy; in the "F_{measure}" column, the entries in boldface indicate the best performance with the highest F_{measure}.

For PlanktonSet 1.0, Table 13 shows that there is no clear evidence proving that using multiple transfer learning can get the results improved all the time. On average, using Flowers17 as the source domain dataset gets a decrease with 0.06% accuracy and 0.000 6 F_{measure}; using Flowers102 as the source domain dataset gets a decrease with 0.09% accuracy and 0.002 3 F_{measure}; using Plant Seedlings as the source domain dataset gets the gain with 0.08% accuracy and 0.000 2 F_{measure}. In fact, even using ImageNet as the source domain dataset get the accuracy decreased with 0.17% accuracy and 0.000 5 F_{measure} on average.

In the "Transfer process" column, the entries in boldface indicate the source domain dataset or intermediate dataset with best classification results; in the "Accuracy (%)" column, the entries in boldface indicate the best performance with the highest accuracy; in the "F_{measure}" column, the entries in boldface indicate the best performance with the highest F_{measure}.

5.5. Multistage Transfer Learning. In Table 8, the gains of cross-domain transfer learning results compared with

Table 12: Results of cross-domain transfer learning on Plant Seedlings.

Transfer process					Model	Accuracy (%)	F_{measure}
		Flowers17	⇒	Plant Seedlings	GoogLeNet-v3	**97.75**	**0.9762**
		Flowers102	⇒	Plant Seedlings	GoogLeNet-v3	97.64	0.9750
		PlanktonSet 1.0	⇒	Plant Seedlings	GoogLeNet-v3	97.47	0.9724
		QUT Fish	⇒	Plant Seedlings	GoogLeNet-v3	97.32	0.9702
ImageNet	⇒	Flowers17	⇒	Plant Seedlings	GoogLeNet-v3	98.06	0.9784
ImageNet	⇒	**Flowers102**	⇒	Plant Seedlings	GoogLeNet-v3	**98.16**	**0.9806**†
ImageNet	⇒	PlanktonSet 1.0	⇒	Plant Seedlings	GoogLeNet-v3	97.58	0.9738
ImageNet	⇒	QUT Fish	⇒	Plant Seedlings	GoogLeNet-v3	97.85	0.9765
		Flowers17	⇒	Plant Seedlings	ResNet-18	**98.00**	**0.9775**
		Flowers102	⇒	Plant Seedlings	ResNet-18	97.77	0.9750
		PlanktonSet 1.0	⇒	Plant Seedlings	ResNet-18	97.79	0.9751
		QUT Fish	⇒	Plant Seedlings	ResNet-18	97.31	0.9700
ImageNet	⇒	Flowers17	⇒	Plant Seedlings	ResNet-18	97.96	0.9777
ImageNet	⇒	**Flowers102**	⇒	Plant Seedlings	ResNet-18	**98.11** †	**0.9791**†
ImageNet	⇒	PlanktonSet 1.0	⇒	Plant Seedlings	ResNet-18	97.52	0.9720
ImageNet	⇒	QUT Fish	⇒	Plant Seedlings	ResNet-18	98.00	0.9778
		Flowers17	⇒	Plant Seedlings	ResNet-34	**97.96**	**0.9771**
		Flowers102	⇒	Plant Seedlings	ResNet-34	97.79	0.9755
		PlanktonSet 1.0	⇒	Plant Seedlings	ResNet-34	97.90	0.9767
		QUT Fish	⇒	Plant Seedlings	ResNet-34	97.31	0.9701
ImageNet	⇒	Flowers17	⇒	Plant Seedlings	ResNet-34	98.11	0.9789
ImageNet	⇒	**Flowers102**	⇒	Plant Seedlings	ResNet-34	**98.27** †	**0.9813**†
ImageNet	⇒	PlanktonSet 1.0	⇒	Plant Seedlings	ResNet-34	97.75	0.9749
ImageNet	⇒	QUT Fish	⇒	Plant Seedlings	ResNet-34	98.19	0.9808†
		Flowers17	⇒	Plant Seedlings	ResNet-50	**97.90**	**0.9760**
		Flowers102	⇒	Plant Seedlings	ResNet-50	97.54	0.9720
		PlanktonSet 1.0	⇒	Plant Seedlings	ResNet-50	96.21	0.9581
		QUT Fish	⇒	Plant Seedlings	ResNet-50	97.24	0.9694
ImageNet	⇒	**Flowers17**	⇒	Plant Seedlings	ResNet-50	**98.32**	**0.9822**
ImageNet	⇒	Flowers102	⇒	Plant Seedlings	ResNet-50	98.25	0.9807
ImageNet	⇒	PlanktonSet 1.0	⇒	Plant Seedlings	ResNet-50	96.63	0.9630
ImageNet	⇒	QUT Fish	⇒	Plant Seedlings	ResNet-50	98.29	0.9814

† indicates the results outperform the corresponding results of training from scratch and fine-tuning on ImageNet.

training from scratch results are listed. For most of the results, fine-tuning on ImageNet can often get the best results. But for some datasets which have a huge difference from ImageNet like PlanktonSet 1.0, fine-tuning on ImageNet may hinder the performances of models.

In multistage transfer learning, after fine-tuning the model on ImageNet, the model is trained on an intermediate domain instead of the target domain. From Table 8, selecting different intermediate domains will have different effects on the final results. For Flowers17, Flowers 102, and Plant Seedlings and QUT Fish, the results of multistage transfer learning do not outperform fine-tuning on ImageNet. For PlanktonSet 1.0, selecting Flowers102 as the intermediate domain can get the best performance, with the gain of 0.003 6 F_{measure} on average.

6. Discussion

In this paper, the multiple transfer learning scheme and the multistage transfer learning method are introduced to exploit cross-domain transfer learning on biology image classification. Our aim is to address the problem that limited labeled data may not fully utilize the feature representation power of DCNNs. In order to achieve this, multiple transfer learning scheme is designed to explore cross-domain transfer learning and proposed multistage transfer learning to learn high-level patterns from different domains to get the learned features fitting the target domain.

Table 1 shows that, with the increase of the DCNN's depth, the performance on ImageNet can get better and better. But meanwhile, the parameters of the network also increase dramatically, which makes training the network more difficult especially when the amount of data is scarce. In order to compare the performances of different models on different datasets and observe their changes trend intuitively, the performances of different models in Figure 8 are normalized and translated. Added the depth, the number of parameters and the performance on ImageNet for each model to Figure 8, which have also been normalized and translated.

In Table 2, it can be seen that the scales of datasets in this paper are very small compared to ImageNet. It can be seen that after the depth of the network reaches a certain level, its performance will no longer improve as the depth of the network increases. Most of the best results on the datasets are achieved with ResNet-18 or ResNet-34.

DCNNs can learn some high-level patterns that are general, so transfer learning can be used to transfer these

Table 13: Results of cross-domain transfer learning on PlanktonSet 1.0.

Transfer process					Model	Accuracy (%)	$F_{measure}$
		Flowers17	⇒	PlanktonSet 1.0	GoogLeNet-v3	77. 22†	**0.6589** †
		Flowers102	⇒	PlanktonSet 1.0	GoogLeNet-v3	76.89	0.6593
		Plant Seedlings	⇒	PlanktonSet 1.0	GoogLeNet-v3	**77.24**	0.6580
		QUT Fish	⇒	PlanktonSet 1.0	GoogLeNet-v3	74.28	0.6059
ImageNet	⇒	Flowers17	⇒	PlanktonSet 1.0	GoogLeNet-v3	74.59	0.6091
ImageNet	⇒	**Flowers102**	⇒	PlanktonSet 1.0	GoogLeNet-v3	**76.82**	**0.6642**†
ImageNet	⇒	Plant Seedlings	⇒	PlanktonSet 1.0	GoogLeNet-v3	76.71	0.6549
ImageNet	⇒	QUT Fish	⇒	PlanktonSet 1.0	GoogLeNet-v3	76.71	0.6549
		Flowers17	⇒	PlanktonSet 1.0	ResNet-18	**76.47** †	**0.6489**
		Flowers102	⇒	PlanktonSet 1.0	ResNet-18	76.39	0.6470
		Plant Seedlings	⇒	PlanktonSet 1.0	ResNet-18	76.35	0.6482
		QUT Fish	⇒	PlanktonSet 1.0	ResNet-18	75.83	0.6450
ImageNet	⇒	Flowers17	⇒	PlanktonSet 1.0	ResNet-18	76.38	0.6505
ImageNet	⇒	**Flowers102**	⇒	PlanktonSet 1.0	ResNet-18	**76.53** †	**0.6520**†
ImageNet	⇒	Plant Seedlings	⇒	PlanktonSet 1.0	ResNet-18	76.39	0.6476
ImageNet	⇒	QUT Fish	⇒	PlanktonSet 1.0	ResNet-18	76.32	0.6482
		Flowers17	⇒	PlanktonSet 1.0	ResNet-34	76.30	0.6538
		Flowers102	⇒	PlanktonSet 1.0	ResNet-34	76.61	0.6535
		Plant Seedlings	⇒	PlanktonSet 1.0	ResNet-34	**76.96** †	**0.6614**†
		QUT Fish	⇒	PlanktonSet 1.0	ResNet-34	75.87	0.6381
ImageNet	⇒	Flowers17	⇒	PlanktonSet 1.0	ResNet-34	76.29	0.6475
ImageNet	⇒	**Flowers102**	⇒	PlanktonSet 1.0	ResNet-34	**76.60**	0.6505
ImageNet	⇒	**Plant Seedlings**	⇒	PlanktonSet 1.0	ResNet-34	76.56	**0.6571**
ImageNet	⇒	QUT Fish	⇒	PlanktonSet 1.0	ResNet-34	76.53	0.6547
		Flowers17	⇒	PlanktonSet 1.0	ResNet-50	77.20	**0.6630**
		Flowers102	⇒	PlanktonSet 1.0	ResNet-50	77.18	0.6581
		Plant Seedlings	⇒	PlanktonSet 1.0	ResNet-50	**77.21**	0.6602
		QUT Fish	⇒	PlanktonSet 1.0	ResNet-50	76.74	0.6628
ImageNet	⇒	Flowers17	⇒	PlanktonSet 1.0	ResNet-50	77.15	0.6679
ImageNet	⇒	Flowers102	⇒	PlanktonSet 1.0	ResNet-50	77. 41†	0.6747†
ImageNet	⇒	**Plant Seedlings**	⇒	PlanktonSet 1.0	ResNet-50	**77.45** †	**0.6714** †
ImageNet	⇒	QUT Fish	⇒	PlanktonSet 1.0	ResNet-50	77.17	0.6676

† indicates the results outperform the corresponding results of training from scratch and fine-tuning on ImageNet.

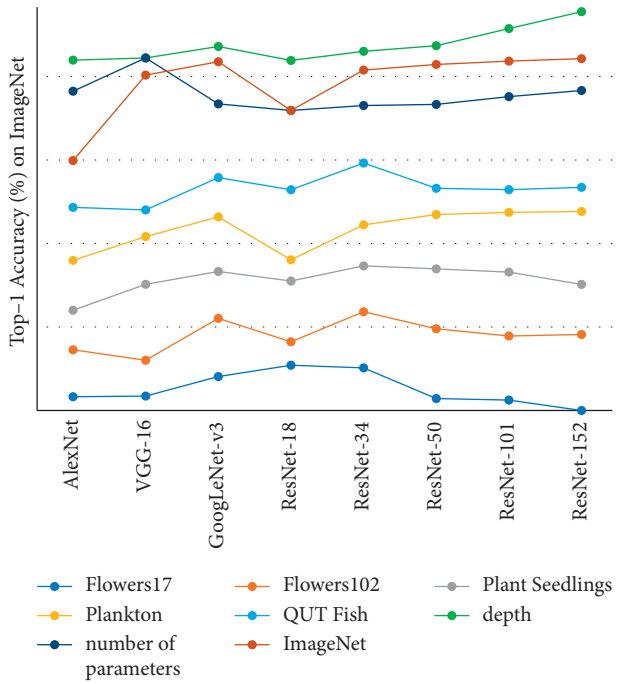

Figure 8: Performances comparison among DCNNs with different depth.

learned high-level patterns to the target domain with limited data. When the data amount in the target domain is small, the data amount in the source domain plays an important role in the transfer learning performance. For example, there are more data in PlanktonSet 1.0, so when make PlanktonSet 1.0 as the source domain dataset, the multiple transfer learning results tend to be better (Tables 6, 11 and 7). For example, in Table 11, although there is a closer biological distance between Flowers17 and Flowers102, the performance of using Flowers17 as the source domain dataset is worse than using PlanktonSet 1.0. When the data amount in the target domain is large, the effect of different biological distances between the species in the source domain and the target domain will be reflected (Table 12). In Table 12, although PlanktonSet 1.0 contains more data than all other datasets, using PlanktonSet 1.0 as the source domain dataset did not get the best result.

Multistage transfer learning is proposed to address the problem caused by the big gap between the source domain and the target domain. From Table 8, it can be seen that since there is a huge difference between ImageNet and PlanktonSet 1.0, multistage transfer learning with cross-domain datasets can improve the performance of fine-tuning on ImageNet. But when performing multistage transfer

learning, to select a dataset is needed as the intermediate domain which can adapt the learned features fitting to the dataset in the target domain. Otherwise, the performance may be hindered due to the big difference between the dataset in the source domain and dataset in the target domain.

7. Conclusions

In this paper, the multiple transfer learning scheme is designed to exploit deep transfer learning for biology cross-domain image classification. By pretraining the DCNN model in different source domains, the results on the target domain dataset can be improved significantly. It has been proved by the experimental results that even the out-of-domain data are effective when the target domain data is insufficient. Multistage transfer learning method is also proposed which can improve the performance of DCNNs when there is a huge difference between the source domain and the target domain. A limitation of multistage transfer learning is that the datasets in the intermediate domain should be carefully selected; otherwise, the final performance may be hindered. However, it is difficult to found the best way to search the optimal dataset as the intermediate domain and this need further study. In our view, searching the datasets which have similar low-level characteristics with the target domain may be a good choice. Since DCNNs can learn some high-level domain-independent features, the ideas of multiple transfer learning and multistage transfer learning can be widely applied to biology image classification or other fields.

Data Availability

The authors provide links to all datasets here: (1) Oxford Flowers (https://www.robots.ox.ac.uk/~vgg/data/flowers/), (2) Plant Seedlings (https://vision.eng.au.dk/plant-seedlings-dataset/), (3) PlanktonSet 1.0 (https://www.ncei.noaa.gov/access/metadata/landing-page/bin/iso?id=gov.noaa.nodc:0127422), and (4) QUT Fish (https://www.kaggle.com/sripaadsrinivasan/fish-species-image-data).

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This research was supported by the National Natural Science Foundation of China (61771440 and 41776113) and Qingdao Municipal Science and Technology Program (17-1-1-5-jch).

References

[1] A. Joly, H. Goëau, H. Glotin et al., "Multimedia life species identification challenges," in *Experimental IR MeetsMultilinguality, Multimodality, and Interaction*, pp. 286–310, Springer, Cham, Switzerland, 2016.

[2] F. I. Woodward, "Climate and plant distribution," *Quarterly Review of Biology*, vol. 69, no. 154, pp. 189–197, 1988.

[3] M. Dyrmann, H. Karstoft, and H. S. Midtiby, "Plant species classification using deep convolutional neural network," *Biosystems Engineering*, vol. 151, pp. 72–80, 2016.

[4] T. M. Giselsson, M. Dyrmann, R. N. Jørgensen, P. K. Jensen, and H. S. Midtiby, "A public image database for benchmark of plant seedling classification algorithms," arXiv preprint, 2017.

[5] G. C. Hays, A. J. Richardson, and C. Robinson, "Climate change and marine plankton," *Trends in Ecology & Evolution*, vol. 20, no. 6, pp. 337–344, 2005.

[6] M. Iain and D. R. Suthers, *Plankton: A Guide to Their Ecology and Monitoring for Water Quality*, CSIRO Publishing, Clayton, Victoria, 2009.

[7] C. Castellani and M. Edwards, *Marine Plankton: A Practical Guide to Ecology, Methodology, and Taxonomy*, Oxford University Press, Oxford, UK, 2017.

[8] d.M. J. Van, J. N. Aldridge, C. Coughlan, E. R. Parker, D. Stephens, and P. Ruardij, "Modelling marine ecosystem response to climate change and trawling in the north sea," *Biogeochemistry*, vol. 113, no. 1-3, pp. 213–236, 2013.

[9] A. Regaudie-De-Gioux and C. M. Duarte, "Temperature dependence of planktonic metabolism in the ocean," *Global Biogeochemical Cycles*, vol. 26, no. 1, 2012.

[10] H. F. Hoff and W. T. Snell, *Plankton Culture Manual*, Florida Aqua Farms Inc., Dade, FL, USA, 2007.

[11] M. Martineau, D. Conte, R. Raveaux, I. Arnault, D. Munier, and G. Venturini, "A survey on image-based insect classification," *Pattern Recognition*, vol. 65, pp. 273–284, 2017.

[12] H. Zheng, R. Wang, Z. Yu, W. Nan, Z. Gu, and Z. Bing, "Automatic plankton image classification combining multiple view features via multiple kernel learning," *BMC Bioinformatics*, vol. 18, no. 16, p. 570, 2017.

[13] N. Macleod, M. Benfield, and P. Culverhouse, "Time to automate identification," *Nature*, vol. 467, no. 7312, pp. 154-155, 2010.

[14] B. Dayrat, "Towards integrative taxonomy," *Biological Journal of the Linnean Society*, vol. 85, no. 3, pp. 407–415, 2015.

[15] J. Wäldchen and P. Mäder, "Plant species identification using computer vision techniques: a systematic literature review," *Archives of Computational Methods in Engineering*, vol. 25, pp. 1–37, 2017.

[16] E. J. Farnsworth, M. Chu, W. J. Kress et al., "Next-generation field guides," *BioScience*, vol. 63, no. 11, pp. 891–899, 2013.

[17] R. Szeliski, *Computer Vision: Algorithms and Applications*, Springer, New York City, NY, USA, 2011.

[18] C. R. Gonzalez and E. R. Woods, *Digital Image Processing*, Pearson, London, UK, 2017.

[19] M. Nixon, *Feature Extraction and Image Processing for Computer Vision*, Academic Press, Cambridge, MA, USA, 2012.

[20] H. M. Sosik and R. J. Olson, "Automated taxonomic classification of phytoplankton sampled with imaging-in-flow cytometry," *Limnology and Oceanography: Methods*, vol. 5, no. 2, pp. 204–216, 2007.

[21] X. Tang, F. Lin, S. Samson, and A. Remsen, "Binary plankton image classification," *IEEE Journal of Oceanic Engineering*, vol. 31, no. 3, pp. 728–735, 2006.

[22] H. Du Buf, *Automatic Diatom Identification*, Vol. 51, World Scientific, , Singapore, 2002.

[23] M. A. Mosleh, H. Manssor, S. Malek, P. Milow, and A. Salleh, "A preliminary study on automated freshwater algae recognition and classification system," *BMC Bioinformatics*, vol. 13, p. 25, 2012.

[24] N. Santhi, C. Pradeepa, P. Subashini, and S. Kalaiselvi, "Automatic identification of algal community from

microscopic images," *Bioinformatics and Biology Insights*, vol. 7, pp. 327–334, 2013.

[25] B. Fan, Z. Wang, and F. Wu, *Local Image Descriptor: Modern Approaches*, Springer, New York City, NY, USA, 2015.

[26] J. Chaki, R. Parekh, and S. Bhattacharya, "Recognition of whole and deformed plant leaves using statistical shape features and neuro-fuzzy classifier," in *Proceedings of the IEEE International Conference on Recent Trends in Information Systems*, pp. 189–194, Kolkota, India, July 2015.

[27] A. Aakif and M. F. Khan, "Automatic classification of plants based on their leaves," *Biosystems Engineering*, vol. 139, pp. 66–75, 2015.

[28] M. A. Mosleh, H. Manssor, S. Malek, P. Milow, and A. Salleh, "A preliminary study on automated freshwater algae recognition and classification system," *BMC Bioinformatics*, vol. 13, no. Suppl 17, p. 25, 2012.

[29] R. Faillettaz, M. Picheral, J. Y. Luo, C. Guigand, R. K. Cowen, and J. O. Irisson, "Imperfect automatic image classification successfully describes plankton distribution patterns," *Methods in Oceanography*, vol. 15, pp. 60–77, 2016.

[30] A. Verikas, A. Gelzinis, M. Bacauskiene, I. Olenina, and E. Vaiciukynas, "An integrated approach to analysis of phytoplankton images," *IEEE Journal of Oceanic Engineering*, vol. 40, no. 2, pp. 315–326, 2015.

[31] I. Dimitrovski, D. Kocev, S. Loskovska, and S. Džeroski, "Hierarchical classification of diatom images using ensembles of predictive clustering trees," *Ecological Informatics*, vol. 7, no. 1, pp. 19–29, 2012.

[32] M. Gönen and E. Alpaydın, "Multiple kernel learning algorithms," *Journal of Machine Learning Research*, vol. 12, no. Jul, pp. pp2211–2268, 2011.

[33] G. Gorsky, M. D. Ohman, M. Picheral et al., "Digital zooplankton image analysis using the zooscan integrated system," *Journal of Plankton Research*, vol. 32, no. 3, pp. 285–303, 2010.

[34] B. E. Richard, *Adaptive Control Processes: A Guided Tour*, Princeton University Press, Princeton, NJ, USA, 2015.

[35] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," *Advances in Neural Information Processing Systems*, vol. 60, no. 6, pp. 1097–1105, 2012.

[36] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *Proceedings of the International Conference on Learning Representations*, San Deigo, CA, USA, May 2015.

[37] C. Szegedy, W. Liu, Y. Jia et al., "Going deeper with convolutions," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–9, Boston, MA, USA, June 2015.

[38] S. Ioffe and C. Szegedy, "Batch normalization: accelerating deep network training by reducing internal covariate shift," in *Proceedings of the International Conference on Machine Learning*, Lille, France, July 2015.

[39] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2818–2826, Las Vegas, NV, USA, June 2016.

[40] 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, June 2016.

[41] G. Huang, Z. Liu, L. van der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Honolulu, HI, USA, July 2017.

[42] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," arXiv preprint arXiv:1709.01507, 2017.

[43] O. Russakovsky, J. Deng, H. Su et al., "ImageNet large scale visual recognition challenge," *International Journal of Computer Vision*, vol. 115, no. 3, pp. 211–252, 2015.

[44] I. Heredia, "Large-scale plant classification with deep neural networks," in *Proceedings of the ACM International Conference on Computing Frontiers*, pp. 259–262, Siena, Italy, May 2017.

[45] R. O. Q. Dias and D. L. Borges, "Recognizing plant species in the wild: deep learning results and a new database," in *Proceedings of the IEEE International Symposium on Multimedia*, pp. 197–202, Taichung, Taiwan, December 2017.

[46] S. A. Siddiqui, A. Salman, M. I. Malik et al., "Automatic fish species classification in underwater videos: exploiting pretrained deep neural network models to compensate for limited labelled data," *ICES Journal of Marine Science*, vol. 75, no. 1, pp. 374–389, 2018.

[47] P. Ouyang, H. Hu, and Z. Shi, "Plankton classification with deep convolutional neural networks," in *Proceedings of the IEEE Information Technology, Networking, Electronic and Automation Control Conference*, pp. 132–136, Chongqing China, May 2016.

[48] C. Wang, Z. Yu, H. Zheng, N. Wang, and B. Zheng, "Cganplankton: towards large-scale imbalanced class generation and fine-grained classification," in *Proceedings of the IEEE International Conference on Image Processing*, pp. 3713–3717, Phoenix, AZ, USA, September 2016.

[49] M. Alshehri, A. Kumar, A. Bhardwaj, and S. Mishra, "Deep learning based approach to classify saline particles in sea water," *Water*, vol. 13, no. 9, p. 1251, 2021.

[50] S. K. Punia, M. Kumar, T. Stephan, and R. Patan, "Performance analysis of machine learning algorithms for big data classification," *International Journal of E-Health and Medical Communications*, vol. 12, no. 4, pp. 60–75.

[51] Z. Ge, C. McCool, C. Sanderson, and P. Corke, "Content specific feature learning for fine-grained plant classification," CLEF (Working Notes), 2015.

[52] H. Lee, M. Park, and J. Kim, "Plankton classification on imbalanced large scale database via convolutional neural networks with transfer learning," in *Proceedings of the IEEE International Conference on Image Processing*, pp. 3713–3717, Phoenix, AZ, USA, September 2016.

[53] E. C. Orenstein and O. Beijbom, "Transfer learning and deep feature extraction for planktonic image data sets," in *Proceedings of the IEEE Winter Conference on Applications of Computer Vision*, pp. 1082–1088, Santa Rosa, CA, USA, March 2017.

[54] W. Ge and Y. Yu, "Borrowing treasures from the wealthy: deep transfer learning through selective joint fine-tuning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Honolulu, HI. USA, July 2017.

[55] K. Abhinav, K. S. Sanjay, S. Sonal, K. Lakshmanan, and K. Arun, "Deep feature learning for histopathological image classification of canine mammary tumors and human breast cancer," *Information Sciences*, vol. 508, pp. 405–421, 2020.

[56] K. Abhinav, K. S. Sanjay, K. Lakshmanan, S. Sonal, and S. Sameer, "A novel cloud-assisted secure deep feature classification framework for cancer histopathology images," *ACM Transactions on Internet Technology*, vol. 21, pp. 1–22, 2021.

[57] K. Abhinav, K. S. Sanjay, S. Sonal, K. S. Amit, and S. Sameer, "CoMHisP: a novel feature extractor for histopathological

image classification based on fuzzy SVM with within-class relative density," *IEEE Transactions on Fuzzy Systems*, vol. 29, pp. 103–117, 2021.

[58] A. Du, "BioTL," 2018, https://github.com/zhenglab/BioTL.

[59] Cowen, R. K., S. Sponaugle, K. L. R., Luo, J.: PlanktonSet 1.0: Plankton Imagery Data Collected from F.G. Walton Smith in Straits of Florida from 2014-06-03 to 2014-06-06 and Used in the 2015 National Data Science Bowl (NODC Accession 0127422). Accessed date: 1 November 2017.

[60] K. Anantharajah, Z. Ge, C. McCool et al., "Local inter-session variability modelling for object classification," in *Proceedings of the 2014 IEEE Winter Conference on Applications of Computer Vision (WACV)*, Steamboat Springs, CO, USA, March 2014.