Hindawi
Journal of Electrical and Computer Engineering
Volume 2017, Article ID 9240407, 8 pages
https://doi.org/10.1155/2017/9240407

Research Article

Multi-Input Convolutional Neural Network for Flower Grading

Yu Sun, Lin Zhu, Guan Wang, and Fang Zhao

School of Information Science and Technology, Beijing Forestry University, Beijing 100083, China

Correspondence should be addressed to Fang Zhao; fangzhao@bjfu.edu.cn

Received 21 April 2017; Revised 11 July 2017; Accepted 12 July 2017; Published 31 August 2017

Academic Editor: Sos Agaian

Flower grading is a significant task because it is extremely convenient for managing the flowers in greenhouse and market. With the development of computer vision, flower grading has become an interdisciplinary focus in both botany and computer vision. A new dataset named BjfuGloxinia contains three quality grades; each grade consists of 107 samples and 321 images. A multi-input convolutional neural network is designed for large scale flower grading. Multi-input CNN achieves a satisfactory accuracy of 89.6% on the BjfuGloxinia after data augmentation. Compared with a single-input CNN, the accuracy of multi-input CNN is increased by 5% on average, demonstrating that multi-input convolutional neural network is a promising model for flower grading. Although data augmentation contributes to the model, the accuracy is still limited by lack of samples diversity. Majority of misclassification is derived from the medium class. The image processing based bud detection is useful for reducing the misclassification, increasing the accuracy of flower grading to approximately 93.9%.

1. Introduction

Flower grading means dividing flowers into several grades according to the quality based on the appearance. *Gloxinia* is a kind of flower which is beneficial for mental and physical health of humans. Our research is focused on the flower grading, taking the *Gloxinia* as an example. Quality grading of flowers is significant because it is extremely convenient for greenhouse and market. Flower grading has a very important effect in handling and marketing the flowers after cultivation. In addition, it can be used in our house to judge which grade the flowers belong to, making our daily life more intelligent. With the development of computer vision, flower grading becomes automatic and intelligent based on images identification.

Flower grading is considered as a challenging task because the differences between each grade are not obvious, as illustrated in Figure 1. In particular, the shape and color of medium grade flowers are very similar to high or low grade flowers. In addition, classifying quality of flowers is a challenging task also considering the lack of dataset which contains different quality grades of the flowers.

Many researchers pay attention to the quality grading. Arakeri and Lakshmana [1] proposed a computer vision based automatic system for tomato grading using ANN (artificial neural network). Wang et al. [2] also proposed an automatic grading system of diced potatoes based on computer vision and near-infrared lighting. Although these systems are successful, both of them focused on binary classification. In addition, these researches still need complex preprocessing such as extracting features from cleaned background. Al Ohali [3] proposed a date fruit grading system which classifies dates into three quality categories using back propagation neural network (BPNN) algorithm with only 80% accurately.

In computer vision, deep learning has made great breakthrough in the last few years, especially the convolutional neural networks (CNNs). CNNs are very successful in ImageNet Large Scale Visual Recognition Challenge (ILSVRC) [4]. Several researchers use CNNs to identify plant images. Lee et al. [5] presented system that utilizes CNN to automatically learn discriminative features from leaf images. Reyes et al. [6] fine-tuned a CNN model for plants identification achieving a great success. References [5, 6] are both based on the CNN where the architecture was firstly proposed by Krizhevsky [7].

This paper presents a deep learning model for flower grading. Each flower may not be fully described by one image.

Label	Number	Photo 1	Photo 2	Photo 3
Good	1			
	2			
	3			
Medium	4			
	5			
	6			
Bad	7			
	8			
	9			

Figure 1: Typical samples of the dataset.

Figure 2: Image acquisition and equipment setup.

At least, three images are requested. Plants should not be graded just based on partial regions. Thus, traditional CNN is not appropriate for our research. A new deep learning model named three-input convolutional neural network is proposed by us for flower grading. Differing from traditional CNN, three-input CNN takes three images as the input. Each sample which is inputted to three-input CNN model contains three images rather than single image. Empirically, our method achieves a satisfactory accuracy on the dataset. A new *Gloxinia* dataset named BjfuGloxinia consisting of 321 *Gloxinia* samples belonging to three grades is also proposed by us for training and validating our model.

The rest of the paper is organized as follows: Section 2 reviews the concept of CNN and then gives an overview of dataset and approach that we proposed. The experimental results are presented in Section 3. Section 4 introduces bud detection to improve the performance of flower grading. Section 5 draws the conclusions.

2. Proposed BjfuGloxinia Dataset and Multi-Input CNN Model

2.1. The Gloxinia Grade Dataset. The BjfuGloxinia (BG) dataset collected at a greenhouse in Beijing Forestry University, Beijing, China, is employed in the experiment. This is the first image dataset for flower grading. It can be downloaded from ftp://iot.bjfu.edu.cn/. The dataset containing 321 samples of *Gloxinia* is divided into three grades by expert according to the relevant rules. In these rules, the plant which has more than two high-quality flowers belongs to the good class. The plant which just has buds or only one flower belongs to the medium class. The plant with no flowers belongs to the bad class. Each grade contains 107 samples and each sample consists of three images. Typical samples of the BjfuGloxinia are illustrated in Figure 1. It is obvious that the dataset is challenging because plants from different grades have very similar appearance, especially the samples in the medium class which are easily confused with the good or the bad class.

In order to obtain the training set and the testing set, some equipment and materials are needed, including a digital single lens reflex (DLSR) camera, a tripod, a timing switch, and an electric turntable disk. The datasets are collected by a series of processes as follows. Put each flower on the electric turntable disk and keep the vertical distance between the bottom of flowerpot and the ground at 49 cm. The electric turntable disk is connected to the timing switch. The horizontal distance between the center of the turntable and the center of the tripod is 70 cm. The DSLR camera is fixed on the tripod. The tilt angle is 25 degrees relative to the vertical direction. The vertical distance between the tripods to the ground is 92 cm. The image acquisition and equipment setup are depicted in Figure 2. Flowers are rotated by an electric turntable whose speed is fixed at 30 s a lap. The disk is set to rotate every 120 degrees and pause 5 seconds for image acquisition.

2.2. Three-Input Convolutional Neural Network. One image cannot cover the whole plant. Every sample in our research is described by taking at least three images. Therefore, traditional single-input CNN architecture is not suitable for our research. We designed a new CNN model to accept three images as input.

2.2.1. Convolutional Neural Network. Convolutional neural networks [8, 9], originally proposed by LeCun et al. for handwritten digit recognition, have been recently succeeded in image identification, detection, and segmentation tasks [10–15]. CNN is proved to have a strong ability in large scale image classification. It is mainly composed of three types of layers: convolutional layers, pooling layers, and full-connection layers. Convolutional and pooling layers are the most important layers. The convolutional layers are used to extract features by convolving image regions with multiple filters. As the layers increase, the CNN understands an image

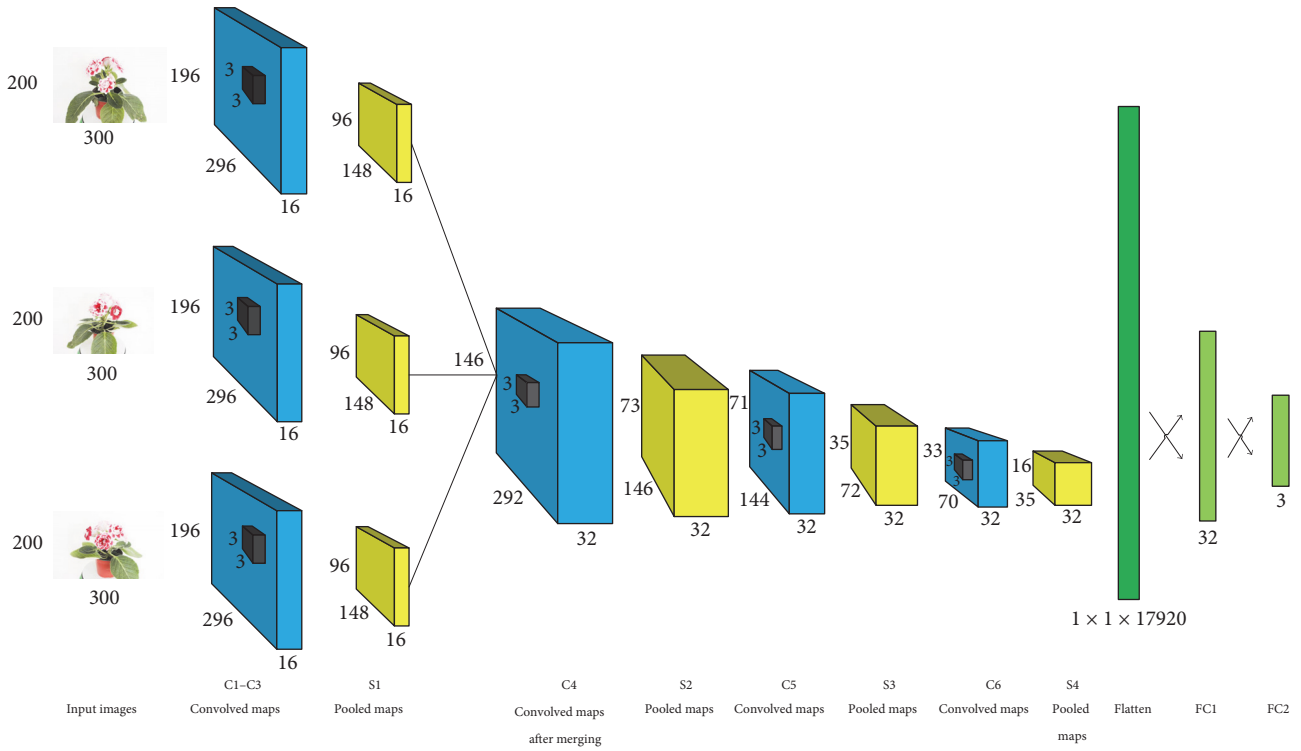

Figure 3: An architecture of 3-input CNN model for *Gloxinia* grading. This architecture has the best effect on testing set.

(a) (b)

Figure 4: Extract features by cutting image. (a) The original image and (b) the cropped image.

progressively. The pooling layers reduce the size of output maps from convolutional layers and prevent overfitting. Through these two layers, numbers of neurons, parameters, and connections are much fewer in CNN models. Therefore, CNNs are more efficient than BP neural networks with similarly sized layers.

2.2.2. Architecture of Three-Input Convolutional Neural Networks. Based on the traditional CNN architecture, a new model named three-input CNN is proposed by us. The model is employed to perform *Gloxinia* grading and achieve a preferable result on the dataset. The full model of our CNN architecture is depicted in Figure 3. The convolutional layers C1–C3 filter three $300 \times 200 \times 3$ input images with 32 kernels of size $7 \times 7 \times 3$ with stride of 1 pixel. The stride of pooling layer S1 is 2 pixels. Then, the three convolutional layers are merged into one. C4 has 16 kernels of size $3 \times 3 \times 3$ with stride of 1 pixel. S2 pools the merged features with a stride of 4. Both C5 and C6 have 32 kernels with size of $3 \times 3 \times 3$ with stride of 1 pixel. The dropout is applied to the output of S4 which has been flattened. The fully connected layer FC1 has 32 neurons and FC2 has 3 neurons. The activation of the output layer is softmax function.

3. Experiments and Results

3.1. Dataset Augmentation. The performance of models is limited by mini-scale dataset due to the lack of samples. To augment the dataset, images are flipped horizontally and vertically, shifted, and rotated. Besides these traditional methods for dataset augmentation, the main region which contains the important features is cut from the whole image. The operation is shown in Figure 4.

3.2. Implementation and Preprocess. 80% of BjfuGloxinia dataset is randomly selected for training and 20% of the dataset is for testing. The model is implemented in "Keras" which is a high-level neural networks API [16]. All the experiments were conducted on a Ubuntu Kylin 14.04 server with a 3.40 GHz i7-3770 CPU (16 GB memory) and a GTX 1070 GPU (8 GB memory). Our model is evaluated on BjfuGloxinia dataset which is detailed in Section 2. The size of an original image is 4288 × 2848 pixels, which should be reduced to fit the GPU memory. All the original images are resized to 300 × 200 pixels and then per-pixel value is divided by 255. The images should also be normalized and standardized before being inputted to models for fast convergence. The inputted images are shuffled to avoid the model influenced by inputting order. Both the sequence of samples and the three images belonging to each sample should be shuffled.

3.3. Training Algorithm. Training algorithm of convolutional neural network is divided into two stages. The one is forward propagation and the other is backward propagation.

3.3.1. Forward Propagation. Data are transferred from the input layer to the output layer by a series of operations including convolution, pooling, and fully connected. Each convolutional layer uses trainable kernels to filter the result of previous layer followed by an activation function to form the output feature map. In generally, the operation is shown as follows:

$$x_j^\ell = f\left(\sum_{i\in M_j} x_i^{\ell-1} * k_{ij}^\ell + b_j^\ell\right), \tag{1}$$

where M_j represents the set of input maps that we selected, b is a bias added to every output map, k represents the kernels, and k_{ij}^ℓ is the weight of the row "i" and column "j" in each kernel. The operation of pooling layer is downsample which summarizes the outputs of surrounding neurons by a kernel map [7]

$$x_j^\ell = f\left(\beta_j^\ell \operatorname{down}\left(X_j^{\ell-1}\right) + b_j^\ell\right), \tag{2}$$

where β is the multiplicative bias and b is an additive bias and "down" is a subsampling function adopted max-pooling [17]. The reason why we select max-pooling rather than mean-pooling is because with the latter it is difficult to find the important information such as the edge of objects while the former selects the most active neuron of each region in feature maps [18]. Therefore, with max-pooling, it is easier to extract useful features. The fully connected layer is equal to hidden layer of multilayer perceptron. The activation of output layer is softmax function [19] applied for multiclassification, which is given by

$$\sigma(z)_j = \frac{e^{z_j}}{\sum_{k=1}^K e^{z_k}} \quad \text{for } j = 1, \ldots, K, \tag{3}$$

where Z is a K-dimensional vector and in the range $(0, 1)$. In this paper, K is 3.

3.3.2. Backward Propagation. Backward propagation updates parameters to minimize the discrepancy between the desired output and the actual output by stochastic gradient descent (SGD). The discrepancy is given by the categorical cross-entropy loss function:

$$\text{Loss}_i = -\log\left(\frac{e^{f_{yi}}}{\sum_j e^{f_j}}\right) \quad \text{for } j = 1, 2, 3, \tag{4}$$

where f_j is probability of sample i which is classified to class j. L_1 and L_2 regularization are adopted to prevent overfitting. L_1 is given by

$$C = C_0 + \frac{\lambda}{n}\sum_w |w|, \tag{5}$$

where C_0 is the loss in formula (4). L_2 is given by

$$C = C_0 + \frac{\lambda}{2n}\sum_w w^2. \tag{6}$$

In this paper, weight of L_1 and L_2 regularization is 0.0001. Dropout [20] is also adopted to prevent overfitting and it is set to 0.1. SGD algorithm computes the gradients and updates the coefficient or weights. It can be expressed as follows:

$$\begin{aligned} \delta_x &= w_{x+1}\left(\sigma'\left(w_{x+1}\cdot c_x + b_{x+1}\right)\circ \operatorname{up}\left(\delta_{x+1}\right)\right), \\ \Delta w_x &= -\eta\cdot\sum_{i,j}\left(\delta_x \circ \operatorname{down}\left(S_{x-1}\right)\right), \end{aligned} \tag{7}$$

where δ_x denotes sensitivities of each unit with respect to perturbations of the bias b, $\circ$ denotes element-wise multiplication, up( ) represents an upsampling operation, down( ) represents subsampling operation, w is the updated weight, and η represents the learning rate.

3.4. Results and Failure Analysis. Large quantities of experiments are conducted to find the best-performing models for flower grading. The architectures of models varied by changing the size of filter kernels, number of feature maps, and convolutional layers. These models are depicted in Tables 1 and 2. As is shown in Table 1, when the number of convolutional layers after merging is in the range of one to two, change the number of layers before merging and observe the effect of models. As is shown in Table 2, the number and the size of filter kernels of the convolutional layers are varying when the number of convolutional layers in every branch before merging is fixed to one.

Top ten best-performing models are selected eventually. The accuracy evolution of 10 models on *Gloxinia* grading is shown in Figure 5.

The result of Table 1 shows that 1-2 layers before merging are better than more. Table 2 shows that 2-3 convolutional layers after merging is the best. As the number of layers increases, the accuracy tends to decline. The change of accuracy is not obvious when varying the size of kernels. The size of 5 × 5 is slightly better than 3 × 3. M4 is the best model with the highest accuracy of 0.89 on testing set.

Table 1: Architecture of models with different number of layers before merging.

Name of models	Convolutional layers								
	C1*	C2*	C3*	C4*	C5	C6	FC1	FC2	ACC
M1	3 × 3,16		—	—	3 × 3,32		32	3	76.2% ± 0.3
M2	3 × 3,16		—	—	3 × 3,32	3 × 3,32	32	3	81.6% ± 0.3
M3	3 × 3,16	3 × 3,16	—	—	3 × 3,32	—	32	3	85.5% ± 0.3
M4	3 × 3,16	3 × 3,16	—	—	5 × 5,64	5 × 5,32	32	3	84.6% ± 0.3
M5	3 × 3,16	3 × 3,32	—	—	3 × 3,64	3 × 3,64	32	3	88.8% ± 0.3
M6	5 × 5,16	5 × 5,16	—	—	5 × 5,32	5 × 5,32	32	3	84.7% ± 0.3
M7	3 × 3,16	3 × 3,32	—	—	5 × 5,32	5 × 5,32	32	3	80.6% ± 0.3
M8	3 × 3,16	3 × 3,32	3 × 3,32	—	3 × 3,64	—	64	3	82.7% ± 0.3
M9	3 × 3,16	3 × 3,16	3 × 3,32	—	3 × 3,64	3 × 3,64	64	3	82.1% ± 0.3
M10	5 × 5,16	3 × 3,16	3 × 3,32	3 × 3,32	3 × 3,32	—	32	3	81.6% ± 0.3
M11	5 × 5,16	5 × 5,16	3 × 3,32	3 × 3,32	3 × 3,32	3 × 3,32	32	3	80.6% ± 0.3

∗ represents the convolutional layers in each branch before merging. Each branch has only one convolutional layer. "3 × 3,32" represents the size of filter kernels which is 3 × 3 and the number of kernels is 32. All the strides of kernels are set to 1 × 1.

Table 2: Architecture of models with one convolutional layer before merging.

Name of models	Convolutional layers								
	C1–C3	C4	C5	C6	C7	C8	FC1	FC2	ACC
M12	3 × 3,32	3 × 3,32	3 × 3,32	3 × 3,64	3 × 3,64	3 × 3,64	32	3	33.3% ± 0.3
M13	3 × 3,32	3 × 3,32	3 × 3,32	3 × 3,64	3 × 3,64	—	32	3	68.4% ± 0.3
M14	3 × 3,32	3 × 3,32	3 × 3,32	3 × 3,64	—	—	32	3	87.8% ± 0.3
M15	5 × 5,16	3 × 3,32	3 × 3,32	3 × 3,32	—	—	32	3	89.6% ± 0.3
M16	7 × 7,16	3 × 3,32	3 × 3,32	3 × 3,32	—	—	32	3	83.7% ± 0.3
M17	3 × 3,32	3 × 3,48	3 × 3,48	3 × 3,48	—	—	32	3	85.7% ± 0.3
M18	3 × 3,16	3 × 3,32	3 × 3,32	—	—	—	32	3	81.6% ± 0.3
M19	3 × 3,16	3 × 3,32	3 × 3,16	—	—	—	16	3	81.6% ± 0.3
M20	5 × 5,16	3 × 3,32	3 × 3,16	—	—	—	16	3	84.7% ± 0.3
M21	5 × 5,16	3 × 3,64	3 × 3,32	—	—	—	32	3	87.8% ± 0.3
M22	7 × 7,16	3 × 3,32	3 × 3,16	—	—	—	16	3	87.8% ± 0.3
M23	3 × 3,16	5 × 5,32	3 × 3,32	—	—	—	32	3	87.8% ± 0.3

C1–C3 represent the layers in three branches before merging. Pooling layers are ignored in this table. Generally, every convolutional layer is followed by a pooling layer. All the sizes and strides of pooling layers are set to 2 × 2.

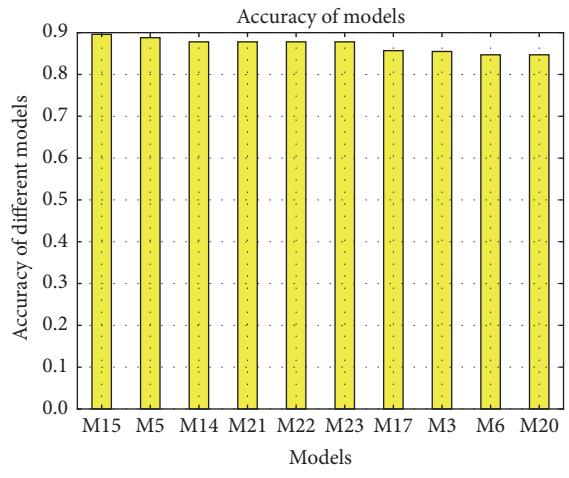

Figure 5: Average accuracy of top ten models on testing set. It is obvious that M15 is the best model with the highest accuracy 0.89 on testing set.

The process of flower grading by single-input CNN is divided into two steps. Firstly, each image of a sample is classified separately. Secondly, the majority of the categories are selected as a result of sample classification. As is shown in Table 3, comparing to the single-input CNN, multi-input CNN is much better than single-input CNN for flower grading. The single-input CNN cannot grade flowers well. The probable reason is that when the three image classification results are inconsistent, it is very difficult to draw the conclusion about which grades the sample is belonging to. For example, a sample contains three images. The first image is classified to the good class, the second image is classified to the medium class, and the third image is classified to the bad class. Therefore, the sample cannot be classified to any grades without additional rules. In this paper, the sample is considered to be misclassified in the case of inconsistent result. Comparing to the single-input CNN, multi-input CNN not only improves the accuracy, but also reduces the number of predictions. Multi-input CNN predicts a sample

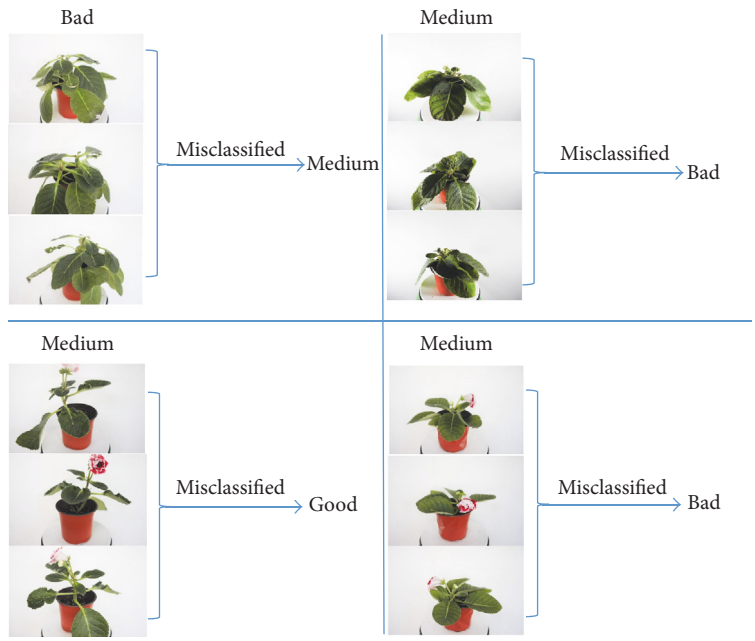

Figure 6: Examples of misclassified classes. This figure shows four misclassified samples. For example, the first sample in this figure which belongs to the bad class is misclassified to the medium class by our CNN model.

Table 3: Architecture of single-input models.

Name of models	Convolutional layers							
	C1	C2	C3	C4	C5	FC1	FC2	ACC
M24	3 × 3,32	3 × 3,32				32	3	84.7% ± 0.3
M25	3 × 3,32	3 × 3,32	3 × 3,32	—	—	32	3	77.6% ± 0.3
M26	3 × 3,32	3 × 3,32	3 × 3,32	3 × 3,64	—	32	3	75.5% ± 0.3
M27	3 × 3,16	3 × 3,32	3 × 3,32	3 × 3,64	3 × 3,64	32	3	75.5%± 0.3

C1–C3 represent the layers in three branches before merging. Pooling layers are ignored in this table. Generally, every convolutional layer is followed by a pooling layer. All the sizes and strides of pooling layers are set to 2 × 2.

Table 4: Confusion matrix of our CNN model for flower grading.

Class		Predicted		
		Bad	Medium	Good
Actual class	Bad	30	2	0
	Medium	7	25	0
	Good	0	1	31

just once while single-input CNN needs to predict three images of a sample. The confusion matrix is depicted in Table 4. From the confusion matrix we can observe that with the model it is easier to classify the good class and the bad class. It is very difficult to classify the medium class (7 misclassified). The error rate of the medium class is near to 0.3.

From our investigation as illustrated in Figure 6, samples which were misclassified are probably caused by two reasons. One is that these plants have almost similar appearance to other classes. The other is that the proportion of features in the image is still very small, though the important region has been cut out from the whole image. For example, the bud is the most important feature which could distinguish the medium class from the bad class. But it is very small and difficult to be found in the image. It will be worse if the neurons which contain the bud information are thrown away after the dropout operation. Furthermore, due to the shortage of plants, although the dataset enlarged by several methods, it is still very small and lack samples diversity, limiting the accuracy of our models.

4. Bud Detection

Bud detection is based on PlantCV [21] which is an open source package. The buds were detected by image processing. The main idea of the detection is finding an appropriate threshold in training set which can separate the target region

(a) (b)

Figure 7: Example of bud detection. This figure shows the operation of bud detection. The buds in (b) are marked by aquamarine circle.

Table 5: Select samples for bud detection.

Number	Bad class	Medium class	Good class	Selected
1	$9.16e-01$	$8.08e-02$	$2.46e-03$	False
2	$9.88e-01$	$3.86e-03$	$7.65e-03$	False
3	$6.92e-01$	$3.02e-01$	$4.63e-03$	True
4	$9.46e-01$	$1.39e-02$	$3.96e-02$	False
5	$4.53e-01$	$4.34e-01$	$1.10e-01$	True

Number 3 and number 5 are selected for bud detection because $6.92e-01$ is close to $3.02e-01$ and $4.53e-01$ is close to $4.34e-01$.

of image from others. The binary threshold is expressed as follows:

$$\mathrm{dst}(x, y) = \begin{cases} \max \text{value} & \text{if } \mathrm{src}(x, y) > \text{threshold} \\ 0 & \text{otherwise,} \end{cases} \quad (8)$$

where the max value is set to 256 and threshold is 190 and $\mathrm{src}(x, y)$ is the g channel value of (x, y) in the image using RGB color space.

The result shows that almost all of the errors derived from the medium class are misclassified to the bad class. The most problem probably is difficulty to extract the small important feature. In order to solve this problem and improve the accuracy of classifying the medium class, we focus on bud detection. At first, our model is used to predict the probability that every sample belongs to each class. The samples whose probability belonging to the bad class is close to the medium class are selected for bud detection. Sample selection is shown in Table 5. A sample is classified to the medium class if it contains buds. The accuracy of our model on testing set is lifted to 93.9% after detection. Bud detection is shown in Figure 7.

5. Conclusion

This paper presents a three-input convolutional neural model for grading every three images of a flower. This paper also presents a new *Gloxinia* dataset named BjfuGloxinia which consists of three grades, containing 107 samples and 321 images of each quality grade. After dataset augmentation, the number of plants in dataset are increased to 760 samples and 2780 images in training set. The experimental results show that learning the features through three-input CNN can make good performance on *Gloxinia* grading with the highest accuracy of 89.6% on the testing set after dataset augmentation. This accuracy is increased by 8 percentage points compared to using the original dataset. The result demonstrates that the method of dataset augmentation is effective and three-input CNN is the promising model for large scale flower grading. Bud detection is proposed to improve the accuracy of classifying the medium class. It lifts the accuracy on testing set to 93.9%.

In the future work, BjfuGloxinia will be enlarged by more quality grades and more plants. The performance of the model should also be improved. Application of the model will be extended from flower grading to more plant species grading even to other fields, such as plant disease detection and segmentation.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Authors' Contributions

Yu Sun and Lin Zhu contributed equally to this work.

Acknowledgments

This work was supported by the Fundamental Research Funds for the Central Universities: 2017JC02 and TD2014-01. The authors thank Jie Chen, Ying Liu, Ke Ma, and YingJie Liu for collecting dataset with them.

References

[1] M. P. Arakeri and Lakshmana, "Computer vision based fruit grading system for quality evaluation of tomato in agriculture industry," *Procedia Computer Science*, vol. 79, pp. 426–433, 2016.

[2] C. Wang, W. Huang, B. Zhang et al., "Design and Implementation of an Automatic Grading System of Diced Potatoes Based on Machine Vision," in *Computer and Computing Technologies in Agriculture IX*, vol. 479 of *IFIP Advances in Information and Communication Technology*, pp. 202–216, Springer, Cham, 2016.

[3] Y. Al Ohali, "Computer vision based date fruit grading system: Design and implementation," *Journal of King Saud University - Computer and Information Sciences*, vol. 23, no. 1, pp. 29–36, 2011.

[4] A. Berg, J. Deng, and L. Fei-Fei, Large scale visual recognition challenge 2010, http://www.image-net.org/challenges/LSVRC.

[5] S. H. Lee, C. S. Chan, P. Wilkin, and P. Remagnino, "Deep-Plant: Plant Identification with convolutional neural networks," *Computer Science*, 2015.

[6] A. K. Reyes, J. C. Caicedo, and J. E. Camargo, Fine-tuning Deep Convolutional Networks for Plant Recognition,.

[7] A. Krizhevsky, Convolutional Deep Belief Networks on CIFAR-10.

[8] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2323, 1998.

[9] Y. LeCun, B. Boser, J. S. Denker et al., "Handwritten digit recognition with a back-propagation network," *in Advances in Neural Information Processing Systems*, pp. 396–404, 1990.

[10] R. Vaillant, C. Monrocq, and Y. L. Cun, "An original approach for the localization of objects in images," in *Proceedings of the in International Conference on Artificial Neural Networks*, pp. 26–30, 1993.

[11] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. Lecun, *OverFeat: Integrated Recognition, Localization and Detection using Convolutional Networks, Eprint Arxiv*, 2013.

[12] S. J. Nowlan and J. C. Platt, "A Convolutional Neural Network Hand Tracker," *in Advances in Neural Information Processing Systems 7*, pp. 901–908, 1995.

[13] S. Lawrence, C. L. Giles, A. C. Tsoi, and A. D. Back, "Face recognition: a convolutional neural-network approach," *IEEE Transactions on Neural Networks*, vol. 8, no. 1, pp. 98–113, 1997.

[14] B. Hariharan, P. Arbelaez, R. Girshick, and J. Malik, "Object Instance Segmentation and Fine-Grained Localization Using Hypercolumns," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 4, pp. 627–639, 2017.

[15] C. Garcia and M. Delakis, "Convolutional face finder: A neural architecture for fast and robust face detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, no. 11, pp. 1408–1423, 2004.

[16] Google, François Chollet, keras, 2015, https://github.com/fchollet/keras.

[17] M. Riesenhuber and T. Poggio, "Hierarchical models of object recognition in cortex," *Nature Neuroscience*, vol. 2, no. 11, pp. 1019–1025, 1999.

[18] D. Ciregan, U. Meier, and J. Schmidhuber, "Multi-column deep neural networks for image classification," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR '12)*, pp. 3642–3649, June 2012.

[19] C. M. Bishop, *Pattern Recognition and Machine Learning*, Springer, New York, NY, USA, 2006.

[20] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[21] N. Fahlgren, M. Feldman, M. A. Gehan et al., "A versatile phenotyping system and analytics platform reveals diverse temporal responses to water availability in Setaria," *Molecular Plant*, vol. 8, no. 10, pp. 1520–1535, 2015.