Identification and classification of orchid species

line 1: 1st ZhenChen,Bian
line 2: *Institute of Data Science*
National Cheng Kung University,
line 4: Taiwan
line 5: RE6103013@gs.ncku.edu.tw

line 1: 2nd YunZhong,Jiang
line 2: *Institute of Data Science*
National Cheng Kung University,
line 4: Taiwan
line 5: yunzhong1105@gmail.com

I. INTRODUCTION

Taiwan has a long history of orchid cultivation and has a wide variety of varieties, and its output and quality are internationally recognized. Taiwan has the world's leading orchid breeding research and development, and has the most phalaenopsis species in the world. Ninety percent of the phalaenopsis is used for export, making it the most exquisite domestic product. Agricultural amount first. However, due to the advancement of agricultural biotechnology, the propagation of a large number of tissue seedlings has affected the research and development of new varieties, and other countries have actively invested in breeding and production. Most breeding manufacturers have their own varieties that they focus on cultivating. Professionals are needed to distinguish them. At present, there is no software and technology for identifying phalaenopsis species in the world. We like to train a high-resolution image recognition model for orchids through deep learning.

TASK&DATASET

A. Task

The task is image classification. Given an image with a single class label, ask to predict the image class for a set of unseen test images and measure the accuracy of the prediction. Usually, an image is input and a text class is output.

B. Datasets

There are 2190 images and 219 classes in the dataset. That means we only have 10 images in each class. The size of every image is 640x480. Some of pictures are 640 at vertical side, others are at horizontal side.

C. Criteria

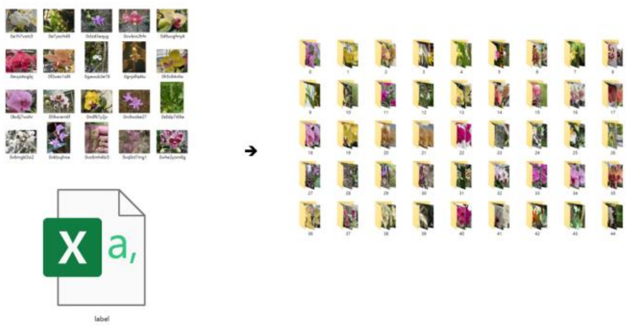
Criteria is a principle or standard to let us judge and evaluate. So, we use accuracy as our criteria.

II. DATA PREPROCESSING

A. Regroup the figures

In the beginning, the dataset is made up of images and label.csv. To fit our model, we must regroup it first. We establish 219 folders with the label name as folder name. There are train and validation set in the branches of main dataset, which conclude 8 images inside each class folder in train set and 2 in validation set. The dataset structure is shown below.

```
├──Dataset
│ ├──train
│ │ ├──0
│ │ │ ├──image1.jpg
│ │ │ ├──image2.jpg
│ │ │ │ …
│ │ ├──1
│ │ │ …
│ ├──val
│ │ ├──0
│ │ │ ├──image1.jpg
│ │ │ ├──image2.jpg
│ │ │ │ …
│ │ ├──1
```



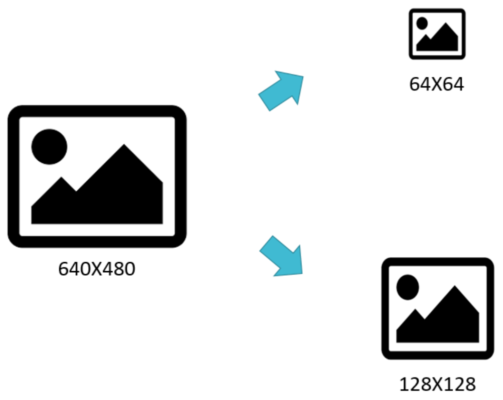
B. Resize the figure

In the application process based on convolutional neural network, image resize is an essential step. Usually, the original image size is relatively large. For example, common surveillance cameras come out with 1080P high-definition or 720P quasi-high-definition images, while the network model input is generally not so large. For example, the network model input size of Yolo series target detection is generally 608*608 or 512* 512 etc.

Due to the limitation of the network structure, the input dimension into the fully connected layer must be fixed. So one of the simplest solutions is to normalize the input image to a fixed size, get the output and inverse transform it back.

This resize approach is a way to accommodate convolutional neural networks.

For reasons outlined above, we decide to resize the image to 64x64 and 128x128. Note that we only use resized image on VGG-16, EfficientNet-B4, Bilinear VGG-16, without Swin Transformer.



C. OHE on Labels

After we read data, the form of labels is saved in an array, which sorted by the class. We translate it to table shape via One-Hot Encoding (OHE).

In the begin, our train y is 1demension, and we need make train y become 2 dimension and train x become 4 dimension, which can easily to train model, so we need to translates it by OHE.

Training label
[0 , 0 , 0 , 0 , 0 , 0 , 0 , 1 , 1 , 1 ,
1 , 1 , 1 , 1 , 2 , 2 , 2 , 2 , 2 ,2 , 2
, 3 , 3 , 3 , 3 , 3 , 3 , 3 , 4 , 4 , 4 ,
4 , 4 ,4 , 4 , 5 , 5 , 5 , 5 , 5 , 5 , 5
, 6 , 6 , 6 , 6 , 6 , 6 , 6 , 7 , 7 , 7
....
....
....
218 , 218 , 218 , 218 , 218 , 218]

OHE ➔

	0	1	2	...	217	218
pic1	1	0	0	...	0	0
pic2	1	0	0	...	0	0
pic3	0	1	0	...	0	0
pic4	0	0	1	...	0	0
...	...	...	...	...	...	...
...	...	...	...	...	...	...

III. Method

Here we use 5 methods Swin Transformer, VGG, EfficientNet-B4, ResNet, and test the best accuracy in this dataset

A. Swin Transformer(Hierarchical Vision Transformer using Shifted Windows)

With the establishment of Transformer's mainstream status in the NLP field, many researchers have begun to try to apply Transformer to the CV field. The development of CV Transformer mainly experienced the introduction of the Attention mechanism in CNN, and then began to replace CNN with a complete Transformer model. The current mainstream research is the optimization of CV Transformer details.

Before the Swin Transformer, ViT and iGPT, both of which used small-sized images as input, this direct resize strategy will undoubtedly lose a lot of information. Unlike them, the input of Swin Transformer is the original size of the image, such as 224*224 of ImageNet, and Swin Transformer uses the commonly hierarchical network structure in CNN. A particularly important point in CNN is that with the deepening of the network level, the receptive field of the node is also expanding. This feature is also satisfied in Swin Transformer. Swin Transformer proposes a total of 4 network frameworks, which are Swin-T, Swin- S, Swin-B and Swin-L. The simplest architecture is Swin-T, where the pre-trained model we used in the training eperiment is also based under this network.

You can regard Patch Partition with Linear Embedding in the figure as Patch Merging, which is equivalent to downsampling the picture. This procedure is very similar to the structure of pooling. The Swin Transformer Block is the core point of the algorithm. It consists of window multi-head self-attention and shifted-window multi-head self-attention, followed by Normalization and MLP processing.

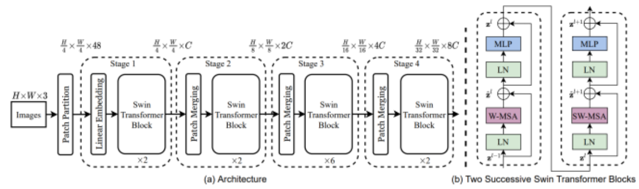
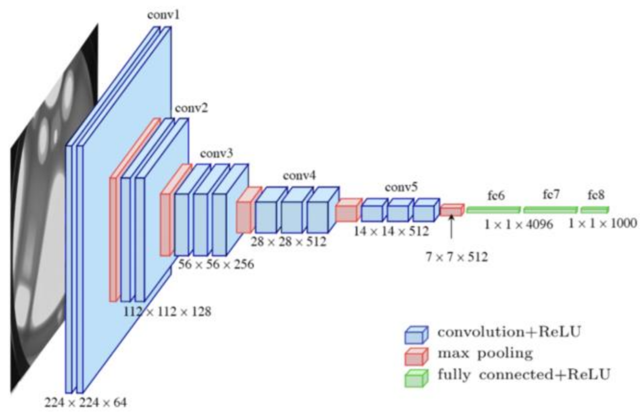


Figure 3. (a) The architecture of a Swin Transformer (Swin-T); (b) two successive Swin Transformer Blocks (notation presented with Eq. (3)). W-MSA and SW-MSA are multi-head self attention modules with regular and shifted windowing configurations, respectively.

B. VGG(Very Deep Convolutional Networks for Large-Scale Image Recognition)

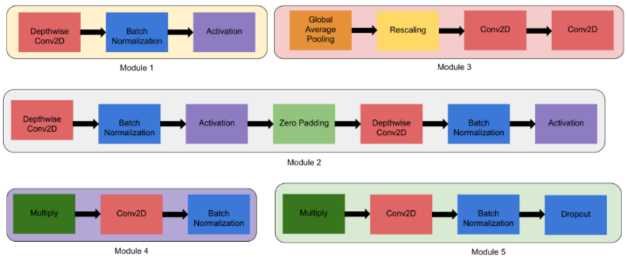
VGG is a classic model in deep learning. His main contribution is to stack CNN through smaller convolution to make the model deeper.

VGG-16 has a total of 16 layers, 13 convolutional layers and 3 fully connected layers. After the first two convolutions with 64 convolution kernels, a pooling is used, and the second time through two 128 convolution kernel convolutions. After that, pooling is used again, and three 512 convolution kernels are repeated twice, and then pooling is performed, and finally three full connections are made. As you can see in the structure graph, the blue block means convolution with ReLU, red one stands for max pooling, and the green block is the fully connected layer to flatten the results.

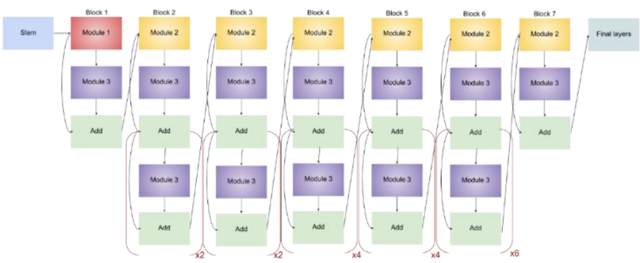


C. EfficientNet B4(Rethinking Model Scaling for Convolutional Neural Networks)

In general, models are designed to be too wide, too deep, or too high-resolution. In the beginning, adding these features is useful, but it saturates very quickly, and then the model has a lot of arguments and is therefore not very efficient. In EfficientNet, these features are augmented by the suite in a more principled way, that is, everything is incrementally added, and since the number of arguments is rather low, this family of models is very efficient and provides better results.



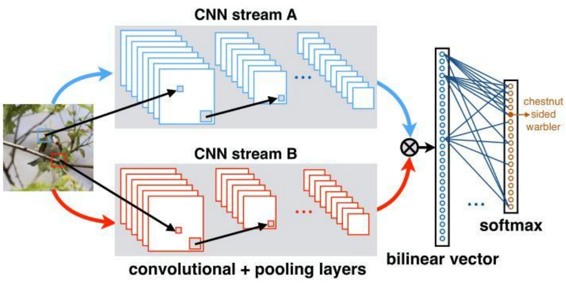
EfficientNet has 8 different network architecture. Here we use the B4 type. It constructs by the stem in the head and final layers at the end, and the backbone of the network is composed of 5 modules. You can see the explanation below.

- Module 1: The starting point of the subblock.
- Module 2: This is used as the starting point of the first subblock of all 7 main mods except the first one.
- Module 3: It is wired to all child blocks as a jump.
- Module 4: Used to merge jump lines into the first subblock.
- Module 5: Each sub-block is wired to the previous sub-block in a skip-wire manner and combined using this module.
- 

-

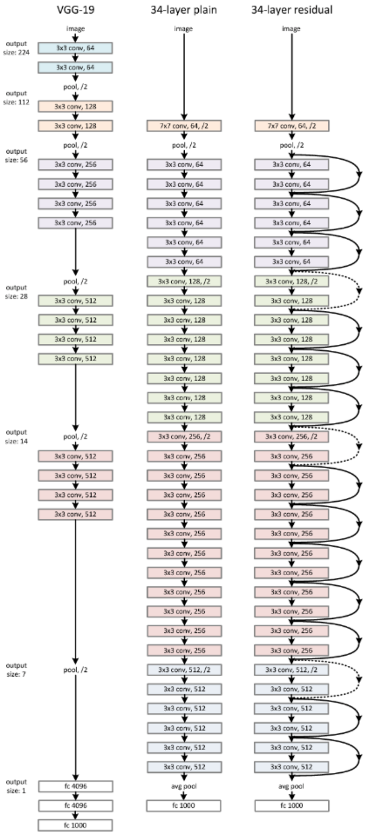
D. Bilinear CNNs for Fine-grained Visual Recognition

In the original paper, author use two CNNs as two feature extractors, and then combined them to a network. We replace the CNNs with VGG-16 because we have tried VGG-16. We want to see how result will be if we assemble two VGG-16.



E. Resnet

It' also modified network. Unless ResNet's 101 layers, we only construct 7 layers here. The main purpose is simplifying network layer and getting great performance. The structure is shown below.



EXPERIMENT OUTCOME

The preferred spelling of the word "acknowledgment" in America is without an "e" after the "g". Avoid the stilted expression "one of us (R. B. G.) thanks ...". Instead, try "R. B. G. thanks...". Put sponsor acknowledgments in the unnumbered footnote on the first page.

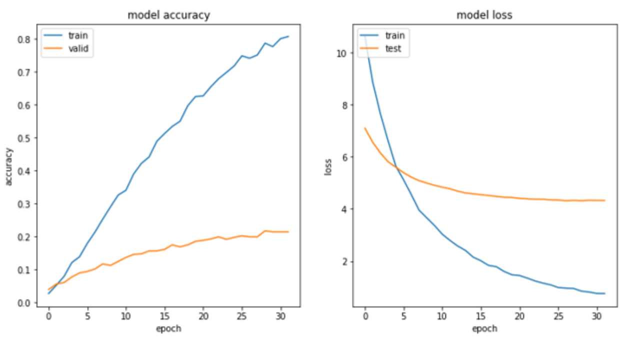
A. Accuracy of models

As you can see, the table display the accuracy and loss. The best model is Swin Transformer, which up to 88% acc@1 and 98% acc@5. Besides, It only get 2.19 on loss score. We use the basic pre-trained model Swin-T, which have already trained on ImageNet-1K

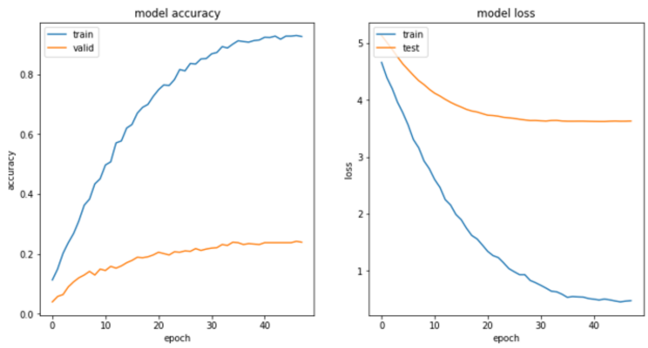
	Val ACC@1	Val ACC@5	Avg Loss	Pre-trained	epoch
Swin Transformer	0.88	0.98	2.1980		300
	Train ACC	Train Loss	Val ACC	Val Loss	Stop epoch
VGG16-resize64*64	0.81	0.76	0.21	4.33	32
VGG16-resize128*128	0.97	0.16	0.37	2.91	49
Bilinear VGG16-resize64*64	1	0.04	0.28	2.95	81
Bilinear VGG16-resize128*128	1	0.01	0.42	2.33	100
EfficientNetB4-resize64*64	0.81	0.76	0.21	4.33	48
EfficientNetB4-resize128*128	0.98	0.2	0.37	2.7	40
ResNet	0.91	0.53	0.42	3.89	49

B. Train history(64*64)

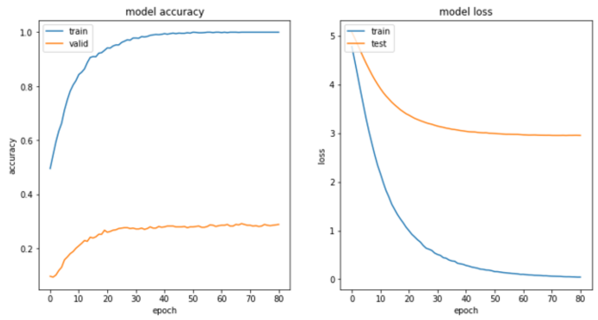
VGG16



EfficientNet-B4

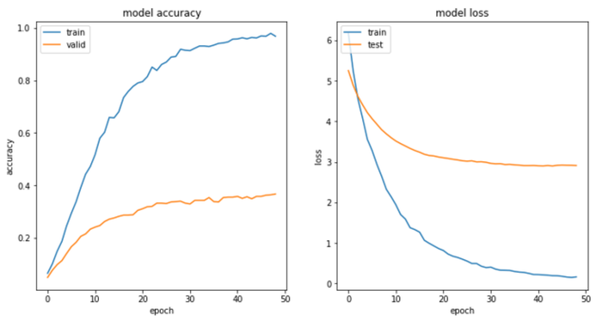


Bilinear VGG-16

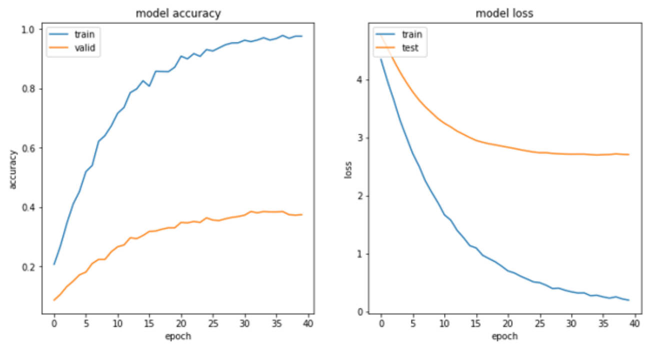


*C. Train history(128*128)*

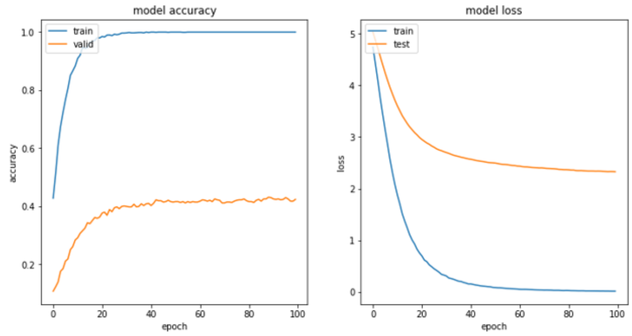
VGG16



EfficientNet-B4



Bilinear VGG-16



IV. Conclusion

1.VGG based model seems not perform well on this dataset.

2.EfficientNet with complex layer didn't perform greater than other models.

3.If we do more augmentation, VGG16, Bilinear-VGG16,

EfficientNet-B4 will have better performance according to our experimental result.

4.We guess that all models have overfitting because the lack of images in each class.

5.Swin Transformer has the best score, which is up to 87% on validation set.

References

[1] Tutorial on Keras flow_from_dataframe----------------------------------- https://vijayabhaskar96.medium.com/tutorial-on-keras-flowfrom-dataframe-1fd4493d237c

[2] Trains a ResNet on the CIFAR10 dataset---------------------------------- https://keras.io/zh/examples/cifar10_resnet/

[3] Self-Promoted Supervision for Few-Shot Transformer------------------- https://arxiv.org/abs/2203.07057

[4] Swin Transformer: Hierarchical Vision Transformer using ShiftedWindows------------------- https://arxiv.org/abs/2103.14030

[5] ResNet's residual learning-- https://medium.com/@hupinwei/%E6%B7%B1%E5%BA%A6%E5%AD%B8%E7%BF%92-resnet%E4%B9%8B%E6%AE%98%E5%B7%AE%E5%AD%B8%E7%BF%92-f3ac36701b2f