AI CUP 2020 Mango Image Recognition Challenge: Grade Classification and Defective Classification

1. The Mango Defective Sub-Challenge

For the competition of mango image classification, dataset was created at three fruit collection facilities in Fangshan, Pingtung, containing tens of thousands of images. The images were made in well-lighted spaces using smart phones and the video recordings with HDR camera. Each of the image contained a shot of one mango that was put on data collectors' hand or the conveyor. Furthermore, each image was labeled with mango grade and defective types by professionals. Defective classification includes five grades '乳汁吸附', '機械傷害', '炭疽病', '著色不佳', '黑斑病'. These labels are used as the gold standard of the AI CUP 2020 Mango Image Recognition Challenge.

2. Experiments and Results

For the Defective Classification task, dataset was divided into train, development and test sets. The following Table 1 shows the number of samples in each set. In the stage 1 competition, we first released train and development sets and the test set at a later time.

Table 1: *Database: Number of image data per class (grade) in the train/development/test splits: Test splits distributions are blinded during the ongoing challenge and will be given in the final.*

ML / Grade	Train	Dev	Test	Total
乳汁吸附	2579	391	blinded	blinded
機械傷害	502	90	blinded	blinded
炭疽病	23587	3504	blinded	blinded
著色不佳	15045	1997	blinded	blinded
黑斑病	1657	298	blinded	blinded

For all of these data, the images were preprocessed before passing into our baseline models. The steps were shown as below:

I. Resize image as (224, 224)
II. Rotate the image by angle (degree = 15)
III. Horizontally flip the image with given probability (p=0.5)
IV. Normalize the image with mean [0.485, 0.456, 0.406] and standard deviation [0.229, 0.224, 0.225]

2.1 The Image Feature Extractors

The official baseline feature set is extracted using a well-known convolutional neural network which have been used extensively in tasks of image classification, object detection, image pixelwise segmentation and instance segmentation. We selected the following pretrained models as our feature extractor: AlexNet [1], VGG16 [2] which are available on PyTorch[1] pretrained models subpackage[2]. After preprocessing the image, the image tensor was fed into each pretrained model to obtain 1000-dimension image representations.

2.2 Defective Classification

For the sake of transparency and reproducibility of baseline competition, we used an open-source implementation of Support Vector Machines (SVM) on scikit-learn[3] with linear kernel and balanced class mode. There are five classes on each mango image, resulting in a multi-label classification task. We separated it into five binary classification problem. For each binary classification task, the complexity parameter *C* was optimized during the testing phase. In this Sub-Challenge, we provided Feature-level Fusion model to defective classification. During training phase, we balanced two-class' data by removing non-defective sample due to sparse number of defective samples. Then we performed ANOVA univariate feature selection with a tuning parameter of percentage *P* of concatenated outputs of AlexNet and VGG16 to be input to the classifier, each five classifier's parameters, macro recall, macro precision, macro f1 score are shown in Table 2. Finally, we obtained the results of macro F1 on development set is ***63.92%***.

	Classifier-D1 乳汁吸附	Classifier-D2 機械傷害	Classifier-D3 炭疽病	Classifier-D4 著色不佳	Classifier-D5 黑斑病
C	1.0	0.1	0.1	0.1	1.0
P	50	10	20	10	20
Recall	56.11	55.61	77.19	76.79	60.32
Precision	54.1	50.38	77.19	76.75	54.93
F1 score	54.58	40.43	77.19	76.63	56

$$precision_{ma} = \frac{\sum_{i=1}^{5} precision_i}{5} = 62.67 \quad recall_{ma} = \frac{\sum_{i=1}^{5} recall_i}{5} = 65.21$$

$$\boldsymbol{F_{1,ma}} = 2\frac{recall_{ma} \times precision_{ma}}{recall_{ma} + precision_{ma}} = 63.92\ \%$$

[1] PyTorch version: 1.4.0

[2] PyTorch pretrained model: https://pytorch.org/docs/stable/torchvision/models.html

[3] scikit-learn version: 0.22.1

3. References

[1] Krizhevsky, Alex. "One weird trick for parallelizing convolutional neural networks." arXiv preprint arXiv:1404.5997 (2014).

[2] Simonyan, Karen, and Andrew Zisserman. "Very deep convolutional networks for large-scale image recognition." arXiv preprint arXiv:1409.1556 (2014).