Introduction to Intelligent Computing – Final project report
Team10

1. *Methodology*

A. Deep Learning Classifiers

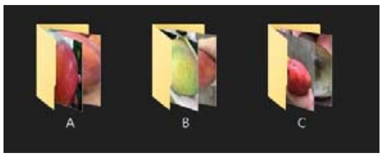
We have used two ways to build our deep learning model. First, we use transfer learning of **VGG16** and **VGG19**. Second, we build our own CNN model for image classification. We used the 5600 images in **C1-P1_Train** as training data, and 800 images in **C1-P1_Dev** as validation data. As for testing, we simply upload to the website and evaluate the result.

B. Ensemble Learning

We have trained seven different models by manipulating learning rate, model structure, optimizer, activation function, and method of data augmentation. Among these seven models, we perform majority voting on model-1, model-6, model-7. We chose these three models by iteratively try and error, and found out that the combination of these three models has the best performance. (See mango-pred.py, pred_models())

C. Data Augmentation

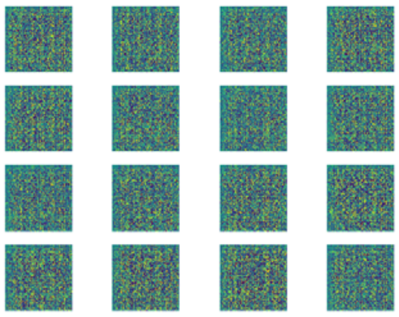
We used **ImageDataGenerator** in **tensorflow.keras.preprocessing.image** to load data and perform data augmentation. Before loading the images, we need to write a python code to put images under the folder of their class. That is, put all images labeled A under the folder A, put all images labeled B under the folder B, put all images labeled C under the folder C. It will be like this: (See classify.py)



Then, we can load images and perform data augmentation by **ImageDataGenerator**. We have done several augmentations to the training images: (See mango-train.py, get_data())

(a) Rotate images between 0 to 180 degree randomly.
(b) Shift left or right within 20% of width randomly.
(c) Shift up or down within 20% of height randomly.
(d) Manipulate brightness to be within 30% darker of brighter randomly.
(e) Zoom images within 20% randomly.
(f) Horizontal flip images randomly.
(g) Vertical flip images randomly.
(h) Normalize pixel values into [0, 1]

Besides, we also trained a GAN network to generate fake images of mangos. However, it was not successful, the generated images barely look like mangos. Here are the generated images by our GAN model.



We spent a few days trying to fix the error. Train for more epochs, replace random noise with random pixel value of mango image, check if generator or discriminator is much stronger than the other. However, after more than a week of testing, we decided to quit on GAN and spend our effort on ensemble learning and classification models. (See mango-gan.py)

2. *Model Detail*

A. Transfer learning

We used pretrained model **VGG16** and **VGG19** provided by **keras** in **tf.keras.applications**. For VGG16, we set learning rate to 1e-5 and fine tune at layer 5 (number of layers of VGG16 is 19). For VGG19, we used adaptive learning rate from 1e-4 to 5e-6 as the epochs goes on, and fine tune at layer 8 (number of layers of VGG19 is 22). We used Adam as our final optimizer. Accuracy of these two models are 76.06% and 75.37%, respectively. (See mango-train.py, build_model())

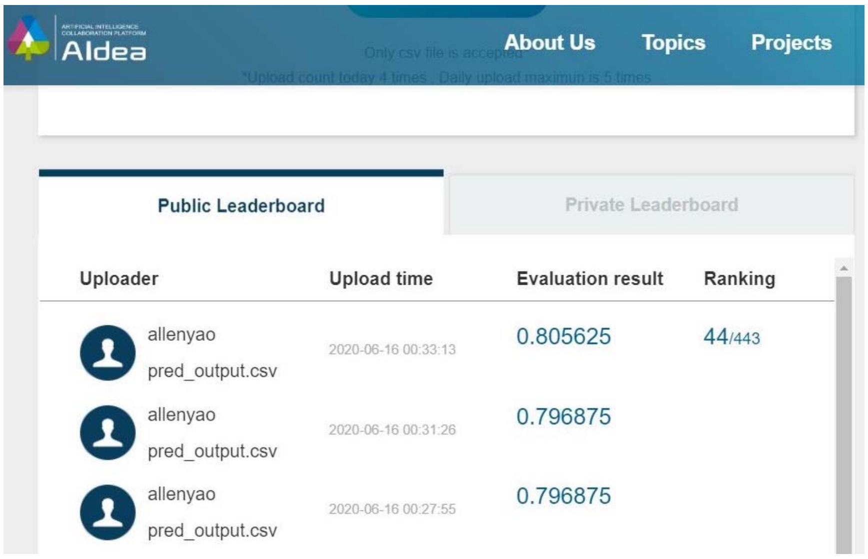
B. CNN model

That's divide into feature extraction and classification. For feature extraction, we used five blocks, each block consisted of one convolution layer and a max-pooling layer. In convolution layer, padding is set to same, size of filter is set to 3*3, and number of filters are 32, 64, 128, 256, 256. In max-pooling layer, pool size is set to 2*2. All activation functions are relu. Then add a flatten layer to flatten the features to make feature vectors. For classification, we used two blocks. Each of them consisted of a dense layer with 512 units, a batch normalization layer and dropout layer to prevent overfitting. We used Adam as our final optimizer. Finally, we used softmax as the activation function of output layer. (See mango-train.py, build_model_1())

Then, we use this model as base model and trained five classifiers by manipulating learning rate, model structure, optimizer, activation function, and method of data augmentation. (See mango-train.py, lr_schedule(epoch) for adaptive learning rate)

3. *Result*

We trained seven models in total. The accuracies of model-1 to model-7 evaluated by uploading to the website are 76.06%, 78.93%, 74.62%, 75.37%, 78.75%, 79.43%, 79.6%, respectively. We have tried performing majority voting on model-1, 2, 5, 6, 7, and the accuracy is 79.56%. Majority voting on model-1, 2, 6, accuracy is 80.12%. Majority voting on model-1, 6, 7, accuracy is 80.56%, which is the highest score we've got. Here is the screenshot of our best score.



Only csv file is accepted

*Upload count today 4 times , Daily upload maximun is 5 times

Uploader	Upload time	Evaluation result	Ranking
allenyao pred_output.csv	2020-06-16 00:33:13	0.805625	44/443
allenyao pred_output.csv	2020-06-16 00:31:26	0.796875	
allenyao pred_output.csv	2020-06-16 00:27:55	0.796875	

4. *More Information and Discussion*

First, we have to mention that we have tried to use **AdaBoostClassifier** and **BaggingClassifier** in **sklearn.ensemble**. Nevertheless, the result is not so good. The accuracy always can't pass 60%. We even tried to use features extracted by our trained model as input, but it was no any better. (See mango-adaboost.py). At the beginning, we used the 94 sample images as testing data instead of uploading to the website. However, we then realize that 94 images are too few for testing, and the images seems come from training data. (See mango-test.py). In the domain of machine learning, this problem is not really difficult to work out a solution. However, it is hard to come up with a perfect solution. In fact, we have come up with an idea that we can first classify the mangos into A and B, C. Then, we classify B and C. It may be a better solution since boundary between class B and C is not so clear comparing to A. Many mistakes occur on class B and C. Therefore, we think this will be a superior solution. However, we don't have time to redo this project. If we can get into the final in the competition, we will try this method.

5. *Conclusion*

Because all of us are not familiar with tensorflow and deep learning, it takes us plenty of time searching documents, googling, asking senior classmates. Meanwhile, we learned a lot during the project. Setting up environment, preprocessing images, load images, build model structure, and more, all of them are challenges to us at first. After finishing this project, we had better understanding on deep learning and we think that the application of deep learning is very wide. Due to the fact that we simply used an intuitive way to settle this problem, we are looking forward to see how other classmate deal with this problem.

6. *Teamwork Allocation*

106060011 姚瀚宇:	mango-train.py
	mango-gan.py
	mango-pred.py
	Report
	Video
106000213 王士亞:	mango-gan.py
	Presentation Slides
106062122 丁憶涵:	mango-test.py
	mango-adaboost.py
106062123 張子宜:	classify.py
	mango-adaboost.py