HW3 Pseudo Code

TA

Data description

- Training file: 6600 pictures have format of [type]_[ID].jpg
- Validation file: 2200 pictures have format of [type]_[ID].jpg
- Testing file: 2200 pictures have format of [ID].jpg
- Train the CNN model with Training data and Validation data.
- Make prediction of Testing file and output a csv file of result.
- The correspond food type numbers are [Back ribs: 0, Bibimbap: 1, Cheesecake: 2, Chicken wings: 3, Sandwich: 4, Cup cake: 5, Donuts: 6, Dumplings: 7, Edamame: 8, Fried rice: 9, Hamburger: 10]

Environment

```
import os
import numpy as np
import cv2
#If it shows No module named 'cv2'
#Activate Anaconda(tenserflow) as administrator
#Enter: pip install opencv-python
```

Load image files

Code

```
1  | def readfile(path, label):
2  |     image_dir = sorted(os.listdir(path))
3  |     x = np.zeros((len(image_dir), 128, 128, 3), dtype=np.uint8)
4  |     y = np.zeros((len(image_dir)), dtype=np.uint8)
5  |     for i, file in enumerate(image_dir):
6  |         img = cv2.imread(os.path.join(path, file))
7  |         x[i, :, :] = cv2.resize(img,(128, 128))
8  |         if label:
9  |             y[i] = int(file.split("_")[0])
10|     if label:
11|         return x, y
12|     else:
13|         return x
```

Preprocess

Code

```
1  | x_img_train_normalize = train_x.astype('float32') / 255.0
2  | x_img_val_normalize = val_x.astype('float32') / 255.0
3  | x_img_test_normalize = test_x.astype('float32') / 255.0
4  | from keras.utils import np_utils
5  | y_label_train_OneHot = np_utils.to_categorical(train_y)
6  | y_label_val_OneHot = np_utils.to_categorical(val_y)
```

Build Model

Code #The setting of layers here are for reference only

```
1  | from keras.models import Sequential
2  | from keras.layers import Dense, Dropout, Activation, Flatten
3  | from keras.layers import Conv2D, MaxPooling2D, ZeroPadding2D
4  | model = Sequential()
5  | model.add(Conv2D(filters=32,kernel_size=(3,3), #First convolutional layer
                    input_shape=(128, 128,3), #The shape have to match the read in size
                    activation=*...*,
                    padding='same'))
6  | model.add(MaxPooling2D(pool_size=(2, 2)))
7  | *Add 2nd or more conv layers, you may add dropout to prevent overfitting*
```

Build Model

```
8  | model.add(Flatten())
9  | model.add(Dense(1024, activation=*…*))
10| model.add(Dense(11, activation=*…*)) # Correspond 11 food types
11| model.compile(loss=*…*,
          optimizer=*…*,metrics=['accuracy'])
12|  train_history=model.fit(x_img_train_normalize, y_label_train_OneHot,
                  validation_data=(x_img_val_normalize, y_label_val_OneHot),
                  epochs=10, batch_size=128, verbose=1)
```

Attention!!

To answer the question in report, you can use print(model.summary)

Predict the result

```
1| prediction=model.predict_classes(x_img_test_normalize)
2| *Save answer(test_ID, prediction) in csv file*
```