File - /Users/surfacebook/PycharmProjects/pythonProject1/tasks/predict.py

```python
import os
import tensorflow as tf
import cv2
import shutil
import numpy as np
#####################################################
#                任务三 7 开始                       #
#####################################################

model = tf.saved_model.load('任务三/panda_elephant.model')

#####################################################
#                任务三 7 结束                       #
#####################################################

#####################################################
#                任务三 8 开始                       #
#####################################################
def preprocess(image):
    image = cv2.fastNlMeansDenoisingColored(image,None,20,20,3,21)
    image = cv2.resize(image,(128,128))
    image = cv2.normalize(image,None,0,255,cv2.NORM_MINMAX)
    return image

id2label={0:'elephant',
          1:'panda'}


def predict(in_path,):
    test_data = []
    # model = tf.saved_model.load('任务三/panda_elephant.model')
    for path in os.listdir(in_path):
        full_in_path = os.path.join(in_path,path)
        img = cv2.imread(full_in_path)
        img = preprocess(img)
        img = np.array(img,dtype=float)
        inputs = tf.convert_to_tensor(img)
        inputs = tf.expand_dims(img,0)
        result = np.argmax(model(inputs).numpy())
        print(f'{full_in_path}<----->{id2label[result]}')


if __name__ == '__main__':
    predict('任务三/测试数据一',model)
    predict('任务三/测试数据二',model)
    predict('任务三/测试数据三',model)

#####################################################
#                任务三 8 结束                       #
#####################################################
```

File - /Users/surfacebook/PycharmProjects/pythonProject1/tasks/Data_name.py

```python
import os
import shutil

def rename(in_path,out_path):
    if not os.path.exists(out_path):
        os.mkdir(out_path)
    for path in os.listdir(in_path):
        if os.path.isdir(os.path.join(in_path,path)):
            continue
        full_in_path = os.path.join(in_path,path)
        if 'panda' in in_path:
            full_out_path  = os.path.join(out_path,'panda'+path)
        else:
            full_out_path  = os.path.join(out_path,'elephant'+path)
        shutil.copy(full_in_path,full_out_path)

if __name__=='__main__':
    rename('elephant_2','elephant_3')
    rename('panda_2','panda_3')
```

File - /Users/surfacebook/PycharmProjects/pythonProject1/tasks/Model_train.py

```python
#####################################################
#               任务三 1 开始                       #
#####################################################
import tensorflow as tf
from Data_Pretreatment import read_data
from tensorflow.keras.models import Sequential
from keras.layers import Conv2D, Activation, BatchNormalization,MaxPooling2D,Flatten,Dense,Dropout
from keras.initializers.initializers_v2 import TruncatedNormal
# from tensorflow.python.keras import optimizers


#####################################################
#               任务三 1 结束                       #
#####################################################


#####################################################
#               任务三 4 开始                       #
#####################################################

test_ratio = 0.2         #------任务三 4题 设置测试集比例为0.2, 训练集比例为 0.8
class_num = 2            #------任务三 4题 设置分类个数为 2
batch_size = 32          #------任务三 4题 设置训练批大小为 32
max_train_epoch = 20     #------任务三 4题 设置最大训练迭代次数 20
lr = 1e-3                #------任务三 4题 设置学习率 0.001
loss='categorical_crossentropy' #------任务三 4题 使用分类交叉熵作为损失函数
metrics=['accuracy']     #------任务三 4题 使用准确度作为模型评估验证的指标
earlystop_monitor = 'val_accuracy'#------任务三 4题 设置早停策略中 评估模型效果的指标为准确度
earlystop_patience=15              #------任务三 4题 设置早停策略中 可以忍受模型准确度不提升的迭代数量
#####################################################
#               任务三 4 结束                       #
#####################################################



#####################################################
#               任务三 2 开始                       #
#####################################################

panda_train_data, panda_test_data = read_data('任务三/数据集/panda',test_ratio = test_ratio)
elephant_train_data, elephant_test_data = read_data('任务三/数据集/elephant',test_ratio = test_ratio)

train_data = panda_train_data.concatenate(elephant_train_data)
train_data = train_data.shuffle(len(train_data)).batch(batch_size)
test_data = panda_test_data.concatenate(elephant_test_data).batch(batch_size)


#####################################################
#               任务三 2 结束                       #
#####################################################



#####################################################
#               任务三 3 开始                       #
#####################################################
class SimpleVGGNet:

    @staticmethod
    def build(width, height, depth, classes):
        model = Sequential()
        inputShape = (height, width, depth) ###---- 3 err 1
        chanDim = -1

        # CONV => RELU => POOL
        model.add(Conv2D(32, (3, 3), padding="same",
                         input_shape=inputShape, kernel_initializer=TruncatedNormal(mean=0.0, stddev=0.01)))
        model.add(Activation("relu"))
        model.add(BatchNormalization(axis=chanDim))
        model.add(MaxPooling2D(pool_size=(2, 2))) ###---- 3 err 2

        # (CONV => RELU) * 2 => POOL
        model.add(Conv2D(64, (3, 3), padding="same", kernel_initializer=TruncatedNormal(mean=0.0, stddev=0.
01)))
        model.add(Activation("relu"))
        model.add(BatchNormalization(axis=chanDim))
        model.add(Conv2D(64, (3, 3), padding="same", kernel_initializer=TruncatedNormal(mean=0.0, stddev=0.
01)))
        model.add(Activation("relu"))
        model.add(BatchNormalization(axis=chanDim)) ###---- 3 err 3
        model.add(MaxPooling2D(pool_size=(2, 2)))

        # (CONV => RELU) * 3 => POOL
        model.add(Conv2D(128, (3, 3), padding="same", kernel_initializer=TruncatedNormal(mean=0.0, stddev=0.
01)))
        model.add(Activation("relu")) ### 3 err 4
        model.add(BatchNormalization(axis=chanDim))
        model.add(Conv2D(128, (3, 3), padding="same", kernel_initializer=TruncatedNormal(mean=0.0, stddev=0.
01)))
```

File - /Users/surfacebook/PycharmProjects/pythonProject1/tasks/Model_train.py

```
 86         model.add(Activation("relu"))
 87         model.add(BatchNormalization(axis=chanDim))
 88         model.add(Conv2D(128, (3, 3), padding="same", kernel_initializer=TruncatedNormal(mean=0.0, stddev=0
    .01)))
 89         model.add(Activation("relu"))
 90         model.add(BatchNormalization(axis=chanDim))
 91         model.add(MaxPooling2D(pool_size=(2, 2)))  ###---- 3 err 5
 92         # FC
 93         model.add(Flatten())
 94         model.add(Dense(512, kernel_initializer=TruncatedNormal(mean=0.0, stddev=0.01)))
 95         model.add(Activation("relu"))
 96         model.add(BatchNormalization())
 97         model.add(Dropout(0.6))
 98 
 99         # softmax
100         model.add(Dense(classes, kernel_initializer=TruncatedNormal(mean=0.0, stddev=0.01)))
101         model.add(Activation("softmax"))
102 
103         return model
104 
105 
106 model = SimpleVGGNet.build(128,128,3,classes=class_num)
107 model.compile(optimizer = tf.keras.optimizers.Adam(learning_rate=lr),
108             loss=loss,metrics=metrics)
109 
110 ####################################################
111 #               任务三 3 结束                      #
112 ####################################################
113 
114 ####################################################
115 #               任务三 5 开始                      #
116 ####################################################
117 
118 earlystop_callback = tf.keras.callbacks.EarlyStopping(
119     monitor='val_accuracy', patience=earlystop_patience)
120 
121 
122 model.fit(train_data,
123           epochs=max_train_epoch, callbacks=[earlystop_callback],
124           validation_data=test_data,
125           validation_freq=1,
126           batch_size=batch_size,
127           shuffle=True)
128 ####################################################
129 #               任务三 5 结束                      #
130 ####################################################
131 
132 
133 
134 ####################################################
135 #               任务三 6 开始                      #
136 ####################################################
137 
138 tf.saved_model.save(model,'任务三/panda_elephant.model')
139 
140 ####################################################
141 #               任务三 6 结束                      #
142 ####################################################
```

File - /Users/surfacebook/PycharmProjects/pythonProject1/tasks/Data_reformat.py

```python
import os
from PIL import Image

def img2jpeg(in_path,out_path):
    if not os.path.exists(out_path):
        os.mkdir(out_path)
    for path in os.listdir(in_path):
        if os.path.isdir(os.path.join(in_path,path)):
            continue
        full_in_path = os.path.join(in_path,path)
        img = Image.open(full_in_path).convert('RGB')
        full_out_path = os.path.join(out_path,path.replace('jpg','jpeg'))
        print(f'img2jpeg : {full_out_path}')
        img.save(full_out_path)


if __name__=='__main__':
    img2jpeg('elephant_1','elephant_2')
    img2jpeg('panda_1','panda_2')

```

File - /Users/surfacebook/PycharmProjects/pythonProject1/tasks/Data_Denoising.py

```python
import os
import cv2

def denosing(in_path,out_path):
    if not os.path.exists(out_path):
        os.mkdir(out_path)
    for path in os.listdir(in_path):
        if os.path.isdir(os.path.join(in_path,path)):
            continue
        full_path = os.path.join(in_path,path)
        img = cv2.imread(full_path)
        # img = cv2.GaussianBlur(img,(5,5),1.5)
        img = cv2.fastNlMeansDenoisingColored(img, None, 20, 20, 3, 21)
        full_out_path = os.path.join(out_path,path)
        cv2.imwrite(full_out_path,img)
        print(f'denosing : {full_out_path}')

if __name__=='__main__':
    denosing('elephant_3','elephant_4')
    denosing('panda_3','panda_4')
```

File - /Users/surfacebook/PycharmProjects/pythonProject1/tasks/Data_Pretreatment.py

```python
#####################################################
#               任务二 1 开始                       #
#####################################################
import tensorflow as tf
import numpy as np
import cv2
import os
#####################################################
#               任务二 1 结束                       #
#####################################################


def read_data(in_path,test_ratio=0.1):
    data_X = []
    data_Y = []
    #####################################################
    #               任务二 2 开始                       #
    #####################################################
    for path in os.listdir(in_path):
        full_path = os.path.join(in_path,path)
        if 'panda' in path:
            data_Y.append(np.eye(2)[1])
        else:
            data_Y.append(np.eye(2)[0])
        img = cv2.imread(full_path)
        data_X.append(img)
    #####################################################
    #               任务二 2 结束                       #
    #####################################################


    #####################################################
    #               任务二 3 开始                       #
    #####################################################
    for img_idx in range(len(data_X)):
        data_X[img_idx] = cv2.resize(data_X[img_idx],(128,128))
    #####################################################
    #               任务二 3 结束                       #
    #####################################################


    #####################################################
    #               任务二 4 开始                       #
    #####################################################
    for img_idx in range(len(data_X)):
        data_X[img_idx] = cv2.normalize(data_X[img_idx],None,0,255,cv2.NORM_MINMAX)
    #####################################################
    #               任务二 4 结束                       #
    #####################################################


    #####################################################
    #               任务二 5 开始                       #
    #####################################################
    data_X = np.array(data_X,dtype=float)
    data_Y = np.array(data_Y,dtype=float)
    data = tf.data.Dataset.from_tensor_slices((data_X,data_Y))


    data_size = len(data_X)
    # print(f'X{data_size},Y{data_Y.shape}')
    num_test = int(test_ratio * data_size)

    test_data = data.take(num_test)
    train_data = data.skip(num_test)
    print(f'测试集数据量:{len(test_data)}')
    print(f'训练集数据量:{len(train_data)}')
    #####################################################
    #               任务二 5 结束                       #
    #####################################################


    #####################################################
    #               任务二 6 开始                       #
    #####################################################
    def agument_random_flip_left_right(image,label):
        image = tf.image.random_flip_left_right(image)
        return image,label
    def agument_rot90(image,label):
        image = tf.image.rot90(image)
        return image,label
    def agument_random_up_down(image,label):
        image = tf.image.flip_up_down(image)
        return image,label

    train_agument_random_flip_left_right = train_data.map(agument_random_flip_left_right)
    train_agument_rot90 = train_data.map(agument_rot90)
    train_agument_random_up_down = train_data.map(agument_random_up_down)

```

```
 90
 91     train_data = train_data.concatenate(train_agument_random_flip_left_right)
 92     train_data = train_data.concatenate(train_agument_rot90)
 93     train_data = train_data.concatenate(train_agument_random_up_down)
 94     print(f'数据增强后的训练集数据量:{len(train_data)}')
 95     ##################################################
 96     #               任务二 6 结束                     #
 97     ##################################################
 98     return train_data,test_data
 99
100 if __name__=='__main__':
101     data1,data2 = read_data('数据集')
```