Laboratory 5: Machine Learning and Artificial Intelligence

Objective

1. Learn about the use of machine learning in robotics.
2. Use neural networks to classify different objects in an industrial robotics scenario.

The field of artificial intelligence (AI) has revolutionized the field of computer vision and robotics. The use of machine learning is vastly applicable to solve varied challenges in robotic assembly, packaging, and even customer service. Different machine learning algorithms are used widely in the field of industrial robotics, as they can provide real-time course correction in assembly operations, improve packaging processes by lowering cost and improving packaging accuracy and augment the efficiency of varied processes.

Commons uses of machine learning in robotics focus on implementing image recognition algorithms to detect the presence of objects and people, predict an object's position, inspect product quality, predict storage requirements and the like. In the robotic industry, machine learning is commonly used for decision making processes. For example, a robotic system can use AI and information from the environment to trigger key actions such as packaging an object, discarding a produced part, halting a process due to safety conditions and adapting their operation to the presence of other robot and human collaborators. In this assignment, we will explore the use of convolutional neural networks to improve the accuracy of robot part classification and packaging.

For this assignment you will learn:

- How to use a neural network as an image classifier
- How to test different network configurations and hyperparameters to see their effect in an image classification task

Deliverables

1. All your source codes in soft copy (the scene files and python scripts), upload all of the files in your submission.
2. A report explaining the solution to all the problems in the handout, including the code modifications used to accomplish the task and the required diagrams, pictures, and answers to the questions.
3. Videos of the solutions for problem 3.

.

Robot waste recycling system

In laboratory 5, we will study the use of convolutional neural networks to assist the operation of a robot waste recycling system. The main operation of the robot system is to sort objects distributed on a conveyor belt into suitable categories for proper material recycling.

In this laboratory, a camera system will be utilized to capture an image of an object on the conveyor belt and a convolutional neural network will be used to classify such object in one of 4 classes (cardboard, plastic, glass and metal). The classification of a neural network for a given object will be used to trigger a robot command to pick up the object from the conveyor belt and move it to the correct bin.

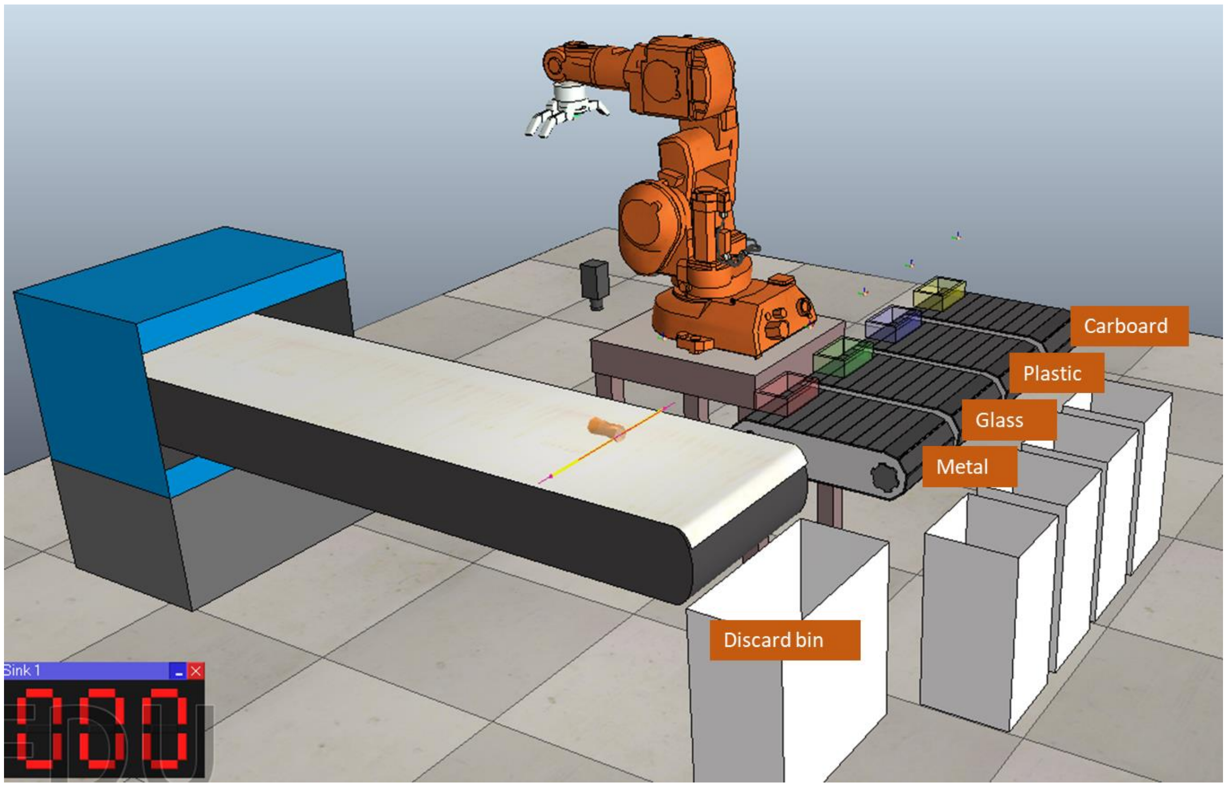


Figure 1. v-rep environment for laboratory 5

Setting up the environment

Before starting to solve the lab, follow the instructions in the installation_guide.pdf. This file will present the instructions to install the anaconda environment, the packages for using keras to train convolutional neural networks and the installation of the v-rep python API according to your computer OS. You will also find in this file the links for video tutorials to perform this installation for each operating system (Ubuntu, Windows and Mac).

After you follow the installation instructions you should be able to import the deep learning packages keras and tensorflow, as well as communicate with v-rep from a python script. Once you finish this installation instructions you can continue with the rest of the laboratory.

PROBLEM 1 – Use a Convolutional Neural Network for robot waste recycling on v-rep

In the previous section you created a conda environment called tensorflow_cpu. Now, make sure to activate this environment to have access to all the packages installed (all of this is explained in the installation video tutorial).

Step 1. To activate the environment, run in you operating system console the following command:

```
conda activate tensorflow_cpu
```

Inside the environment created (*tensorflow_cpu*) run the online_test_lab5_.py file (you should have the *laboratory5_2020_scene.ttt* open in vrep in parallel.

Step2. Now, run the following command from the command prompt:

```
python online_test_lab5.py
```

If your installation of the python API in your operating system was correct, the v-rep simulation should start, and the "parts producer" will start to produce objects. The parts producer creates different waste items of 4 different classes: Carboard, Glass, Metal and Plastic. Each type of waste item is composed of different items, among them: beer bottle, milk carboard, plastic bottle, plastic cup, tuna can, egg carton, etc.

A produced object will come through the conveyor belt in random orientation until they reach the vision sensor and proximity sensor at the pickup place. When the object reaches the proximity sensor the conveyor belt will stop and an image from the top camera will be retrieved from the python script. This image is then used to classify the object into the four categories using the provided network "pretrained_classifier.h5". At last, a command to pick and classify the object will be sent to the robot according to the detected object and deposit it into the correct bin.

Step3. In the main function of the *online_test_lab5.py* file, you will find a function that correctly classifies one object. Extend the code to perform the classification of 12 objects.

Step4. In your report, paste an image that shows your 12 classified items like the following image (your image should contain your 12 classified objects and display all the content of the bins).

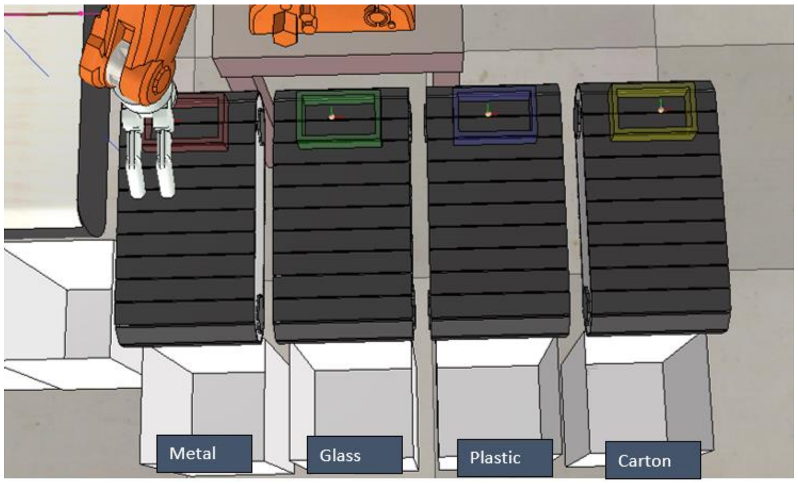


Figure 2. Image of classified object

The network provided to you has been trained over an image dataset for waste recycling. The dataset contains images of several objects from 4 recycling classes in random colors and orientations as shown in the following image. The dataset is divided into training, validation and testing sets as shown in Table 3. Inside the folder *material_lab5* you will find the dataset for recycling classification, inside the folder you will find organized the images for the sets: training, validation and testing.

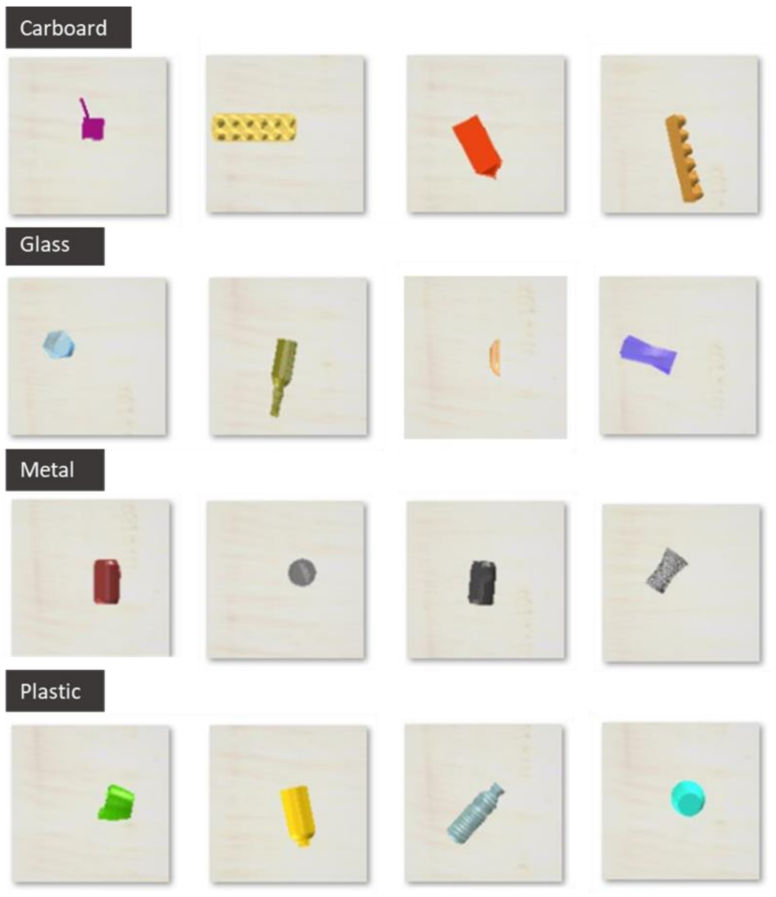


Figure 3. Example images of the recycling dataset.

Table 1. Dataset size for the 4 classes.

	Cardboard	*Glass*	*Metal*	*Plastic*
Training	400	400	400	400
Validation	100	100	100	100
Test	50	50	50	50

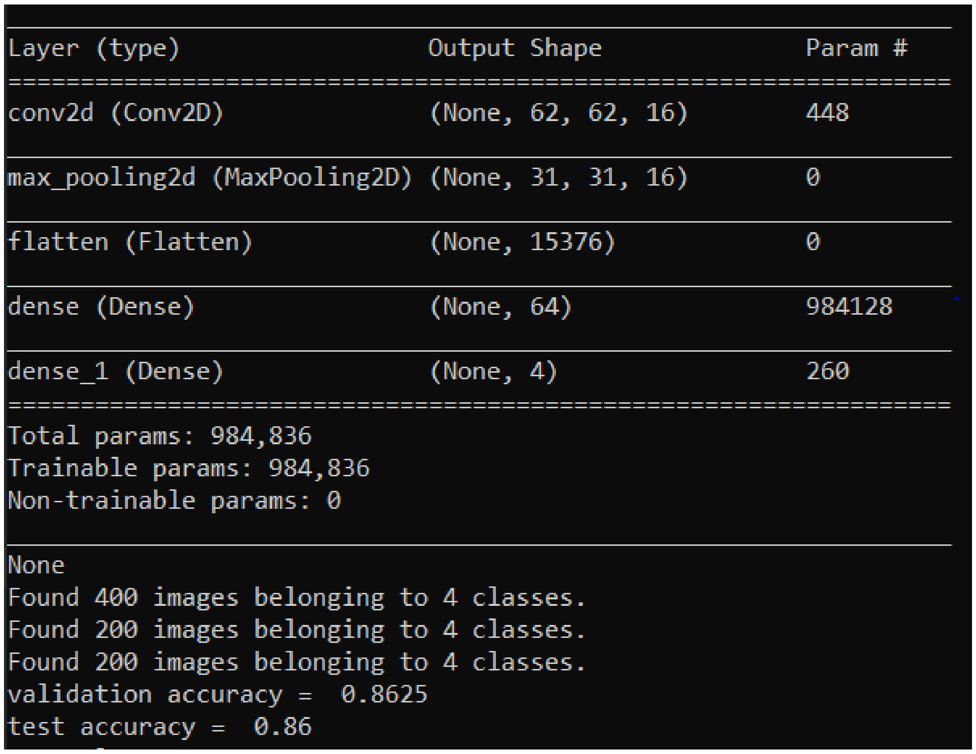
PROBLEM 2 - Train a Convolutional Neural Network for image classification

Now, let's explore the file train_network_lab5.py. This python file allows to train a convolutional neural network based on the dataset found in the folder "data". The lab 5 folder contains a pretrained neural network named “pretrained_classifier.h5”. Currently, if your run the file train_network_lab5.py from the console, the script will load the model saved as “pretrained_classifier.h5” and will test the pretrained model over the validation and testing set.

Step5. Run the file train_network_lab5.py from the console to test the network (remember to check that your environment “tensorflow_cpu” is activated):

```
python train_network_lab5.py
```

The code will print the basic architecture of the network and will also print the accuracy of the model over the training, validation and test (as shown in Figure4.**Error! Reference source not found.**).



```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 62, 62, 16)        448
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 31, 31, 16)        0
_________________________________________________________________
flatten (Flatten)            (None, 15376)             0
_________________________________________________________________
dense (Dense)                (None, 64)                984128
_________________________________________________________________
dense_1 (Dense)              (None, 4)                 260
=================================================================
Total params: 984,836
Trainable params: 984,836
Non-trainable params: 0
_________________________________________________________________
None
Found 400 images belonging to 4 classes.
Found 200 images belonging to 4 classes.
Found 200 images belonging to 4 classes.
validation accuracy =  0.8625
test accuracy =  0.86
```

Figure 4. Summary of the Architecture of example Convolutional Neural Network

The architecture of the network provided to you in the assignment can be further observed in Figure 5.

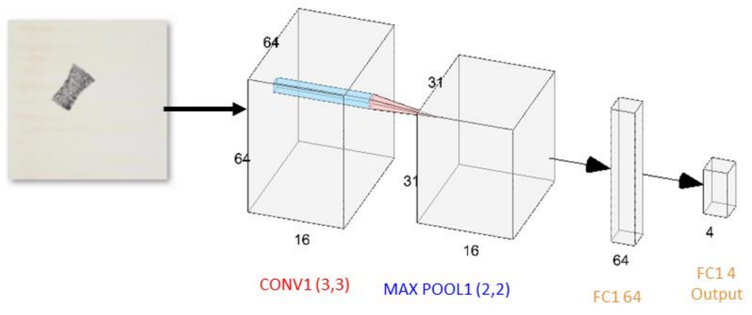
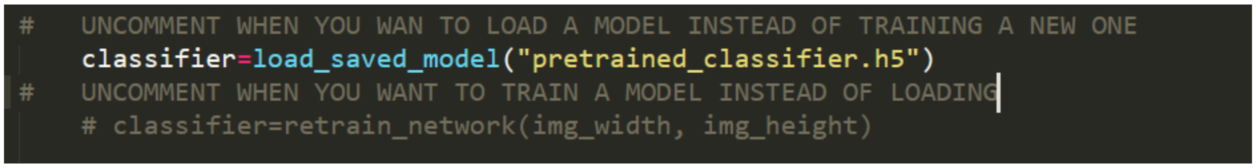


Figure 5. Illustration of the pretrained Convolutional Neural Network provided by the TA.

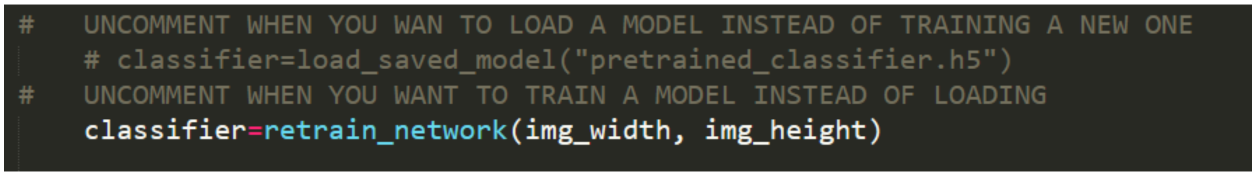
In this section, you need to retrain the network to improve the classification accuracy over 97% in the validation set.

Step6. Inside the file train_network_lab5.py you must uncomment the line to retrain the network and comment the line to load the training model. In python a # symbol is used to comment a line (so that it does not execute).



```
#   UNCOMMENT WHEN YOU WAN TO LOAD A MODEL INSTEAD OF TRAINING A NEW ONE
    classifier=load_saved_model("pretrained_classifier.h5")
#   UNCOMMENT WHEN YOU WANT TO TRAIN A MODEL INSTEAD OF LOADING
    # classifier=retrain_network(img_width, img_height)
```

*Figure 6. Original code in the train_network_lab5.py that will **load the network "pretrained_classifier.h5"** a network and then test it.*



```
#   UNCOMMENT WHEN YOU WAN TO LOAD A MODEL INSTEAD OF TRAINING A NEW ONE
    # classifier=load_saved_model("pretrained_classifier.h5")
#   UNCOMMENT WHEN YOU WANT TO TRAIN A MODEL INSTEAD OF LOADING
    classifier=retrain_network(img_width, img_height)
```

*Figure 7. Code in the train_network_lab5.py **to retrain the network** according to the parameters in the retrain_network function (instead of loading the model). This function will also save the new trained network as **"classifier.h5".***

Step7. Now change the parameters in the retrain_network() function to train the convolutional neural network. Experiment with different parameters until you obtain an accuracy of at least 97% in your validation set.

To retrain the network, consider changing some parameters in the network configuration. Some suggested parameters to change are shown in

Table 2. Suggested parameters to change in the network.

SUGGESTED PARAMETERS TO CHANGE IN THE NETWORK

CONVOLUTIONAL LAYERS	Number of CONV layers. To train in cpu, sensible values could be 1 or 2 CONV layers.	size of the CONV filter, sensible values could be (3,3), (5,5)	number of filters. Sensible values could be 8,16,32
FULLY CONNECTED LAYERS	Number of dense layers.	Number of hidden neurons in each layer. Common values: 32, 64,128,186,256	
TRAINING HYPERPARAMETERS	Training epochs: sensible values could be 50,100,200		
MAX POOLING PARAMETERS	Pool size (size of the max pooling) sensible values could be (2,2), (3,3)		

Step8. Report which parameters you changed to attempt to improve the performance of the network (architecture tuning and hyperparameter change). To do this, attach in your report a picture of the network architecture summary as printed during training (like the one in Figure 4). In addition to the architecture summary report in a table the parameters that you changed that do not show in the summary (such as number of training epochs, or the size of the convolution and polling layers).

Step9. Include in your report as well and image of the accuracy results of your network as printed after training. (after your network finished training it will output the accuracy values over the testing and training set).

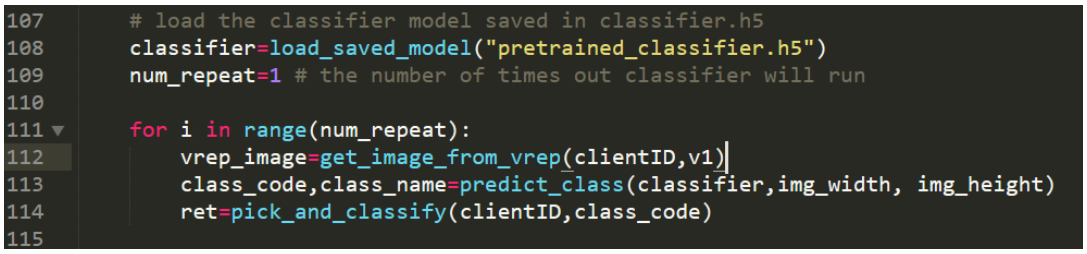
Step10. Compare the accuracy values of the original network provided by the TA and the network your trained over all the sets (training, validation and test). (you can create a table showing the accuracy of each network on the 3 datasets).

Remember, after you train your network it will be saved in the lab5 folder under the name “classifier.h5”.

PROBLEM 3

Now that you have trained your own network, you should test its performance in vrep as you did for problem 1 with the pretrained network provided by the TA.

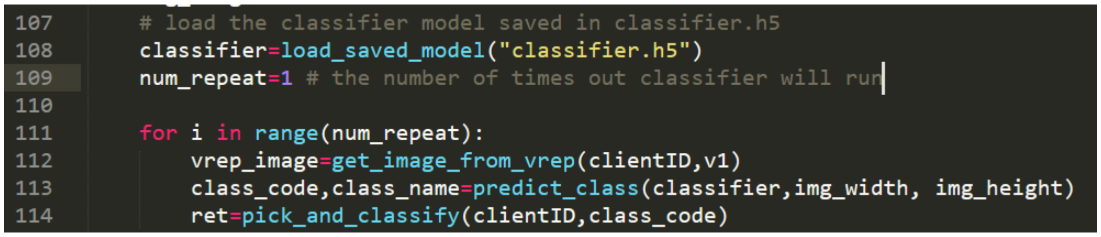
Step 11. Open the file online_test_lab5.py and change the name of the model to load. Previously, the code was reading the model named “pretrained_classifier.h5”. Now, change this file name to the name “classifier.h5”, which is the network that you trained.



```
    # load the classifier model saved in classifier.h5
    classifier=load_saved_model("pretrained_classifier.h5")
    num_repeat=1 # the number of times out classifier will run

    for i in range(num_repeat):
        vrep_image=get_image_from_vrep(clientID,v1)
        class_code,class_name=predict_class(classifier,img_width, img_height)
        ret=pick_and_classify(clientID,class_code)
```

Figure 8. Loading and testing the pretrained network in v-rep, on file online_test_lab5.py



```
    # load the classifier model saved in classifier.h5
    classifier=load_saved_model("classifier.h5")
    num_repeat=1 # the number of times out classifier will run

    for i in range(num_repeat):
        vrep_image=get_image_from_vrep(clientID,v1)
        class_code,class_name=predict_class(classifier,img_width, img_height)
        ret=pick_and_classify(clientID,class_code)
```

Figure 9. Code for loading and testing the network the student trained “classifier.h5”, on file online_test_lab5.py

Step 11. Now test the use of the network in the robot waste recycling system, as you did in step 4. Test the classification of 12 objects and report an image of the classified objects.

Step 12. Create a video showing the classification of the objects. Publish this video in youtube and attach a link to the video in your laboratory report.

Step 13. Report the statistics of classification accuracy per class (carboard, glass, metal, plastic) and the total accuracy. Your can create a table that shows the classification accuracy per class and the total accuracy over all items.

Attach to your submission a pdf file with your Laboratory 5 report, the scene file .ttt (Vrep scenes), and the python files. The code will be evaluated on the python files and on your report (you must include both!). Remember to write in your report the link to a video (posted publicly in yourtube) that shows the robot preforming the task in problem 3 - classifying the parts correctly.

Appendix - Material for the laboratory

Unzip the folder material_lab5.zip, inside you will find the following material:

Material_lab5/data

- Inside the folder material_lab5 you will find the dataset for recycling classification, inside the folder you will find organized the images for the sets: training, validation and testing.

Train_network_lab5.py

- The file train_network_lab5.py is the python file that allows to train a convolutional neural network based on the dataset found in the folder "data".
- In this file, the function **classifier=retrain_network(img_width, img_height)** will train a convolutional neural network and save it in the folder as "classifier.h5", it will also return the model (classifier) to be used to classify the images.
- In this file, the function **classifier=load_saved_model(model_name)** will load the model named as the model_name input.
- In this file, the function **class_code, class_name=predict_class_file(classifier, img_str, img_width, img_height)** will use the inputted classifier to predict the image in the location img_str.
  - For example, in the following line of code the classifier is being used to predict and image called "plastic.png" that is inputted to the network as size 64x64. This line of code returns the class of the object in the image "class_name" which (if correct) would be "plastic" and class_code would be 3.

```
class_code,class_name=predict_class_file(classifier,"plastic.png",64,64).
```

Table 3. Class code for the object classification task.

CLASS_CODE	CLASS_NAME
0	Carboard
1	Glass
2	Plastic
3	Metal

Online_test_lab5.py

- The file online_test_lab5.py is the python file that run the v-rep scene lab5_2020_scene.ttt, gets an image from the v-rep environment, loads the trained network "pretrained_classifier.h5", uses the network to predict an object class and sends a robot command to pick up the object and drop it its corresponding bin.

- In this file, the functions **classifier=load_saved_model(model_name)** and **predict_class_classified()** are the same as in the file train_network_lab5.py
- In this file, the function **get_image_from_vrep(ClientID, cam_handle)** gets an image from vrep by waiting until an object is detected by the camera and when an object is detected. When the image is detected in vrep, the function gets the image and saves it under the name "current_image.png".
- In this file, the function **pick_and_classify(clientID,class_code)** will send a request to the running vrep scene and ask to execute the pick_and_classify function on the vrep side. This function will make the robot approach the pickup position of the detected object and drop it at the correct bin depending on the class code inputted (as shown in Table 3**Error! Reference source not found.** -> 0 for carboard and so on).

Laboratory5_2020_scene.ttt

This file is the v-rep scene where the industrial setting of the robot waste recycling system