A New Approach Towards Waste Container Detection in Smart Cities

Miguel Valente
Instituto Politécnico de Castelo Branco
Castelo Branco, Portugal
mvalente@ipcbcampus.pt

João M. L. P. Caldeira
Instituto de Telecomunicações
Instituto Politécnico de Castelo Branco
Castelo Branco, Portugal
jcaldeira@ipcb.pt

Vasco N. G. J. Soares
Instituto de Telecomunicações
Instituto Politécnico de Castelo Branco
Castelo Branco, Portugal
vasco.g.soares@ipcb.pt

Hélio Silva
EVOX Technologies
Castelo Branco, Portugal
helio.silva@evox.pt

Pedro D. Gaspar
Universidade da Beira Interior
Covilhã, Portugal
dinis@ubi.pt

***Abstract*—This paper presents a new approach to help redesigning waste management for the cities of the future. The current state of tracking waste containers is rigid, inefficient and hard to oversee. Although attempts have been made in the past using radio-frequency identification for waste container detection, it has shown problems like flexibility, cost and environmental impact. We propose and demonstrate a solution based on the use of computer vision techniques, for object detection and classification, towards the differentiation between different types of waste containers.**

Keywords—smart cities, waste management, waste container detection, computer vision

I. Introduction

There has been a growing interest in smart cities in the last decade and many international initiatives in the field started to appear. The idea behind smart cities is to combine the use of information and communications technology to make better decisions and deliver a better quality of life. It is a diverse topic of discussion with several application areas. The work presented on this paper tackles the issue of waste management for smart cities.

Waste management plays an important role in a city by maintaining it clean, hygienic and promoting environmental consciousness. The current state of technology, in the field of smart waste management makes use of sensors and Internet of Things (IoT) to gather data to help make decisions for trash collection and optimized truck routes, being one of the key aspects, the waste container identification. It allows to extract information such as; frequency of waste collection, rates of waste production by type of waste, frequency of containers washups, collection time and routes used during waste collection for specific containers. Then in turn, the new data can be analysed in conjunction with city statistics (population density, waste containers per km^2) to discover which areas of a city are; getting dirtier faster, that often overfill before trash collection, rate of trash production and unoptimized routes used by trucks. Stressing, that all the data is automatically generated and presented in forms of graphs or statistics allowing for city planners to focus on the optimization and improvement of the waste management process.

Currently, the detection of waste containers is primarily done using radio-frequency identification (RFID), Fig. 1 illustrates a typical solution. However, RFID carries a limitation that it is impossible to circumvent, more concretely the necessity of tags; a core aspect of the technology. From this point, it is possible to point out some flaws:

- *Locality*: the pitfall of using this technology and where all the other flaws stem from. Every single waste container and waste collector truck must have a transponder/antenna combination;
- *Maintenance/Installation*: must be done locally and failing units can only be discovered when used;
- *Cost of implementation*: it is necessary to collect, wash and modify the containers in use, which is a long and costly process;
- *Complexity*: a conjugation of the previous flaws that results in a complex system. It is a system that relies on the communication of a series of components in different locals.

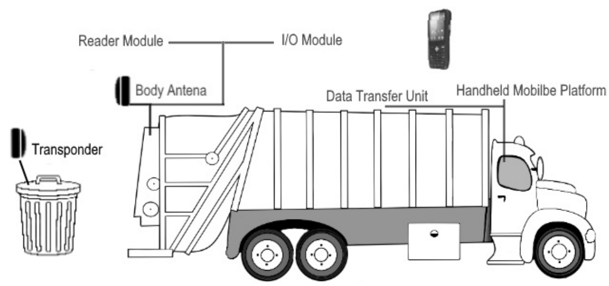

Fig. 1. Ilustration of a system for waste container detection using RFID.

The approach presented in this article focuses on the mitigation of these disadvantages through the use of computer vision techniques, more specifically a convolutional neural network [1]. This approach solves the *locality* constraint, resulting on a simpler solution as illustrated on Fig. 2. Even though the capacity of precisely distinguishing between each individual container is lost, it is not needed. The information comes from the interaction of environment with the object and from the type object, not from the object itself; the correct identification of type the container is all that it is required.

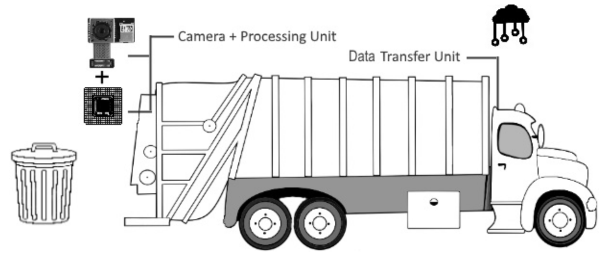

Fig. 2. Ilustration of a system for waste container detection using computer vision.

XXX-X-XXXX-XXXX-X/XX/$XX.00 ©20XX IEEE

The remainder of this paper is organized as follows. Section II introduces the proposed solution with a brief description of the different modules. Section III presents examples of experiments conducted in this stage of initial prototyping for proof of concept. Section IV presents the conclusion and suggest future work directions.

II. Proposed Solution

In the proposed solution, the main idea is to assume the architecture presented in Fig. 3. This architecture is based on the following four key components:

- *Computing Unit*: is responsible for running a convolutional neural network;
- *Web Service*: is the bridge between the Computing Unit and the Internet;
- *Server*: is used to aid the detection process;
- *Database*: is where all the information generated from the Computing Unit and Server is stored.

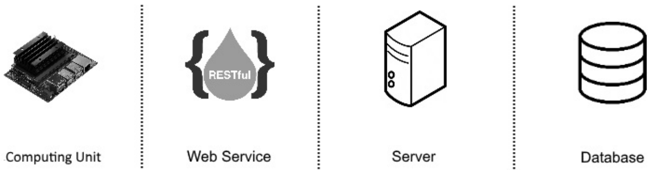

Fig. 3. Overview of the proposed solution.

Starting off with the “on the field” hardware of the proposal, the key components are: (1) processing capacity and (2) a camera. The processing hardware must have enough processing power to run convolutional neural networks. Real-time video streams are not necessary, meaning that the framerate can be lower than 1 frame per second. An acceptable framerate wold be 0.25, 1 frame every 4 seconds. Another feature to keep in mind is size, a smaller form factor is required for ease installation and convenience.

The camera, although essential, doesn’t need to follow strict guidelines, being the most important features the ability to see the entire light spectrum (colours) and the resolution. Nowadays, most small form factor cameras offer both of these features making it easy to select one among all the existing options. Besides the essential components to perform detection, secondary components are required to enhance the usability and to maintain a network connection. This requires the use of a global positioning system (GPS) module and a network communication module. Recent computing technologies such as NVIDIA’s Jetson Nano [2] and Raspberry Pi Compute Model 3 [3] respect both of these key components and can be thought as examples for the proposed solution.

Regarding software the key component is the convolutional neural network, the cornerstone of this proposal, an emergent technology from the past decade with several different architectures for different use cases. The convolutional neural network used on this proposal is You Only Look Once (YOLO) [4], which has been proven to produce good results with low processing power [1], [4].

The convolutional network runs in TensorFlow’s framework [5] and produces an output in JavaScript Object Notation (JSON) format with the following values; probability of the prediction, associated identification label and detected frame. This data is sent to a server via a RESTful webservice; with the capacity of communicating in JSON format. All of the extra, non-essential computation is done server-sided to maximize the processing power available in the unit. This architecture proposal has no user-oriented interface besides the standard interaction with the server and the computing unit via command line interface.

The server aids the detection phase with the aim of increasing detection accuracy through the use of contextual data stored in the database such as the position and type of containers in the area. While performing a detection there are two possible scenarios when consulting the database; no containers on the current location and containers on the current location. If the latter occurs, the stored data can be used to double-check the detection and increase the accuracy, otherwise, the information from the detection is stored in the database with the correspondent location. This implementation allows for a generation of new data containing positive detections, false-positives, failed detections and corresponding images which can be used to train a new iteration of a network.

III. Performance Assessment

This section presents examples of experiments conducted in this stage of prototyping for proof of concept of the use of a convolutional neural network [1], more precisely YOLOv2 [4], to identify different types of waste containers.

The use of YOLOv2 needs to follow three main steps; collection of data, labelling of data and the convolutional neural network training. In the first step, collection of data, on this application means collecting images of waste containers. All images were taken specifically for this application, given that waste containers are region specific and that there is no available dataset with examples for Portugal.

TABLE I. contains the distribution of collected data per use and lighting conditions. The data was taken using a Xiaomi Pocophone F1 and subsequently resized to a 960x720 resolution. All of other computation, training or use of software were done on a standard ASUS NJ750 with Ubuntu 18.04.01 LTS.

TABLE I. Distribution of Images per Use and Lighting Conditions.

	Night	*Day*	*Total*
Multiple Containers per image ***(training YOLO)***	60	84	144
Multiple Containers per image ***(testing YOLO)***	9	22	31
Total	69	106	175

The second step is labelling data. Although this step comes after the collection process, it can be done simultaneously after each collection of data. This process must be done without errors. Mislabelled data reduces the accuracy of the neural network considerably. Thus, the labels must correspond to the type of containers. Fig. 4 presents an example of this step; for this stage it was used an opensource software, labelImg [6].

Fig. 4. Labelling data.

The third step is the training process. It makes use of the training and testing data simultaneously, allowing the user to accompany the evolution in accuracy and the decrease in the learning rate simultaneously. To achieve this step, it was used *darknet* [7], an opensource implementation of YOLO. To take benefit of the graphical processing unit (GPU) computational power it is required to install CUDA 9.0 [8] and cuDNN 7.1.4 [9]; although, secondary, the installation of OpenCV 3.2.0 [10] is required for *darknet* to be able show results and produce graphs. Fig. 5 shows this process in detail. Accuracy is represented by the red line and learning rate is represented in a blue line.

Observing the image, it is possible to see that as soon the average loss hits values between 0.5 and 0.6 it starts to stabilize. Following the same logic, it is possible to see the accuracy rising while the average loss decreases until the end of the training, floating around the 87% mark. The metric used to evaluate a convolutional neural network makes use of mean average precision (mAP) [11], referred as accuracy.

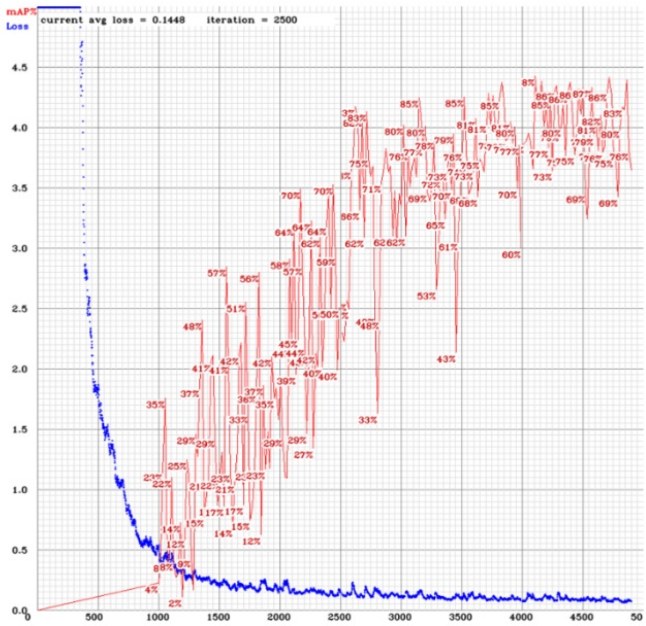

Fig. 5. Results from the trainning process.

The fourth and final step is seeing the results – the actual detections. This step is secondary since accuracy is already known. Nevertheless, it allows for a clear vision of what is happening during detection. Fig. 6 and Fig. 7 show examples of detection during daylight and night-time.

As it can be observed, in these experiments the accuracy reached 87%; a lower accuracy when compared with the one achieved using RFID, nevertheless, the increase in flexibility and a correct engineering of the system can make up for the missing 12%.

The interested reader can refer to our previous work [12], [13] for further details on the rationale, evaluation, and comparison of feature-based methods and convolutional neural networks for object detection and classification, towards the differentiation between types of waste containers.

Fig. 6. Daylight detection.

Fig. 7. Nighttime detection.

IV. Conclusions

This paper discussed the application of computer vision techniques, more specifically a convolutional neural network – YOLOv2, to substitute the identification of waste containers via radio-frequency identification. The obtained results confirm that the approach is capable to identify different types of waste containers, using images, video or real-time video capture.

The proposed approach may play an important role for an improved smart waste management solution in the context of a smart city. However, some improvements should be done in the hardware component, to optimize computation resources. For instance, building a specific solution that has the basic needs for running a neural network and to transmit information, instead of, relying on prebuilt hardware with unnecessary functions and background processes. This should improve computational resources available and reduce the stress on the hardware.

Future work consists of two main directions. One aims at having a larger dataset with more data - different types of waste containers - to be able to produce new iterations of neural networks that have a broader detection capacity. The iterations will also be trained with the generated data from the application, this to improve the accuracy of the convolutional neural network. The other, is developing a fully working solution based on the proposed architecture and test it under a real world scenario.

ACKNOWLEDGMENTS

The authors would like to acknowledge the companies EVOX Technologies and InspiringSci, Lda for their interest in and valuable contributions to the successful development of this work. This work was funded by FCT/MEC through national funds and, when applicable, co-funded by the FEDER–PT2020 partnership agreement under the project UID/EEA/50008/2019.

REFERENCES

[1] Z.-Q. Zhao, P. Zheng, S. Xu, and X. Wu, "Object detection with deep learning: A review," arXiv Prepr. arXiv1807.05511, 2018.

[2] NVIDIA Corporation, JETSON NANO DEVELOPER KIT. NVIDIA Corporation, 2019.

[3] Raspberry Pi, Raspberry Pi Compute Module 3 - NEC Edition. Datasheet, 2017.

[4] J. Redmon and A. Farhadi, "YOLO9000: better, faster, stronger," arXiv Prepr., 2017.

[5] Mart\'\in~Abadi et al., "{TensorFlow}: Large-Scale Machine Learning on Heterogeneous Systems." 2015.

[6] T. Lin, "labelImg," GitHub repository. GitHub, 2019.

[7] A. Bezlepkin, "darknet," GitHub repository. GitHub, 2019.

[8] NVIDIA Corporation, NVIDIA CUDA Compute Unified Device Architecture Programming Guide. NVIDIA Corporation, 2007.

[9] S. Chetlur et al., "cudnn: Efficient primitives for deep learning," arXiv Prepr. arXiv1410.0759, 2014.

[10] G. Bradski, "The OpenCV Library," Dr. Dobb's J. Softw. Tools, 2000.

[11] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes (voc) challenge," Int. J. Comput. Vis., vol. 88, no. 2, pp. 303–338, 2010.

[12] M. Valente et al., "Detection of Waste Containers Using Computer Vision," Appl. Syst. Innov., vol. 2, no. 1, p. 11, Mar. 2019.

[13] M. Valente et al., "Técnicas de Visão Computacional para a Deteção de Contentores de Residuos," Conferência Ibérica de Sistemas e Tecnologias de Informação 2019 (CISTI), AISTI, Coimbra, Portugal, 2019 (accepted for publication)