Improving Public Parking by Using Artificial Intelligence

L. Albuquerque[1,b)], C. Coelho[1,a)], M. Fernanda P. Costa[1,c)], L.L. Ferrás[1,d)] and A.J. Soares[1,e)]

[1]*Centre of Mathematics, University of Minho, Campus de Gualtar, 4710 - 057 Braga, Portugal*

[a)]Corresponding author: ceciliaeduarda58@gmail.com
[b)]lmiguelaa@gmail.com
[c)]mfc@math.uminho.pt
[d)]luislimafr@gmail.com
[e)]ajsoares@math.uminho.pt

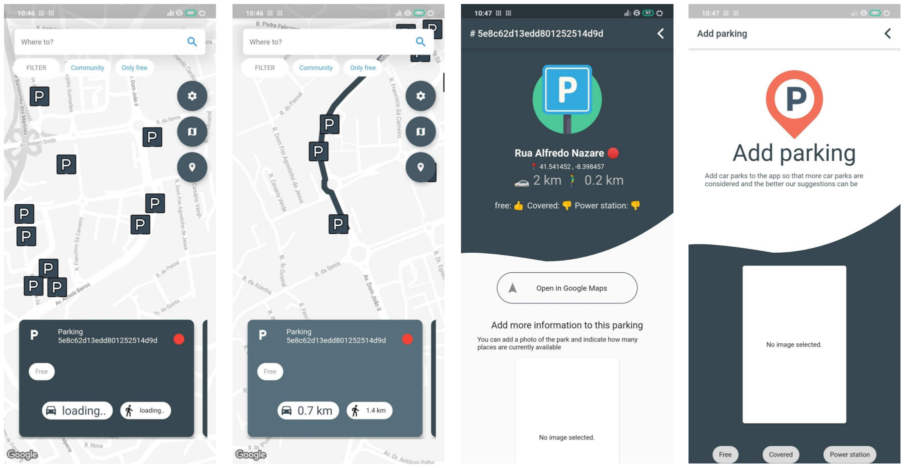
Abstract.
In this work, a generic solution for parking management is presented. The solution uses artificial intelligence and remote data reading to implement a detection system, a server, a mobile phone application and a back office. With the mobile phone application, users can retrieve information about available parking spaces, specify their preferences and destination on the map, and receive local parking suggestions from the server. The information about the parking space can come from sensors and cameras (in real time), but also from information communicated by other users about the capacity of the park. This information is collected by the server and used to train a neural network that suggests parking spaces. The neural network makes the appropriate predictions and increases the accuracy of the prediction model over time. An example of how the application works can be seen at: https://www.youtube.com/watch?v=G-MdzFcsBWw.

INTRODUCTION

The final product of this work is a mobile phone application that focuses on simplicity and practicality, so that anyone can use it and benefit from its help in finding a free parking space. To this end, a Neural Network (NN) accesses the user's needs and provides an answer that is constantly trained based on new data available through user feedback. The more people use the application, the better the results become. To make the application appealing, the parking spaces need to be clearly and intuitively presented and accessible from any device. For this purpose, the *Flutter framework* was chosen, which is known for its use in creating responsive *Web* pages and generating applications for a variety of platforms with just one code, i.e. **Android**, **IOS** , **Windows**, **Mac Os** , **Linux** and even **Web**.

In addition to disseminating information, the application allows users to add new parking spaces and information about the number of available spaces in a parking lot, which can improve the application's response. Both communication between users and the creation of more extensive databases will allow efficient parking behaviour to be developed over time (using NNs). The application can be used in two different ways: either simply as GPS, where the driver can select the parking space and follow the displayed route, or as a tool to find the parking space that best suits the user's needs, with the suggested parking spaces displayed on the map. In addition to a description of each parking space, information is provided about the possibility of finding a parking space and the estimated time until a space becomes available.

The application is simple and intuitive so that any person with minimal experience can naturally navigate and access all features. When users launch the application (Figure 1 (a)), they are presented with a fairly simple layout - a map that takes up the entire screen. At the top is a search bar with autocompleted suggestions for selecting the destination, and three buttons to filter the application's suggestions according to the user's preferences. On the right side are 3 buttons: Settings, Toggle Satellite View and Add New Parks. It is also possible to select the destination from the map by using gesture navigation (pinch to zoom, drag, press).



(a) (b) (c) (d)

FIGURE 1. Application's different screens: (a) main screen; (b) route planning; (c) parking information; (d) add new parking data.

Once the destination is selected, 10 suggestions appear at the bottom of the screen with some relevant parking information: address, price (free or paid), whether the parking lot is covered, availability of charging stations for electric vehicles, distance between the parking lot and the destination, and an indication of the (estimated) capacity of the parking lot. The latter is done in the form of a small coloured circle indicating whether there is a free parking space: red - low probability, orange - there might be one, yellow - most likely and green - most likely there is more than one. When the user selects a suggestion, he is shown the route from his location, Figure 1 (b).

It should be noted that sensors are an expensive solution, as we need one sensor per parking space. Cameras are cheaper and allow us to cover a large area with a single device. The main drawback is that most parking spaces are not easy to detect or delineate, so a third solution is needed, based on user feedback. For example, there may be non-official parking spaces that cannot be covered by cameras. Therefore, the solution proposed in this paper adapts to all free parking detection systems. Only the communication mode between the solution and the used free space detection system [1, 2] needs to be defined. The functionality *add park* tries to solve this problem. Users can enter a new parking space when they are on site by taking a picture of the parking space and indicating its status, as in Figure 1 (d). All the data collected from the users and the real-time systems (cameras, sensors, etc.) are sent to a server, which in turn feeds an NN that predicts the available parking spaces, Figure 2.

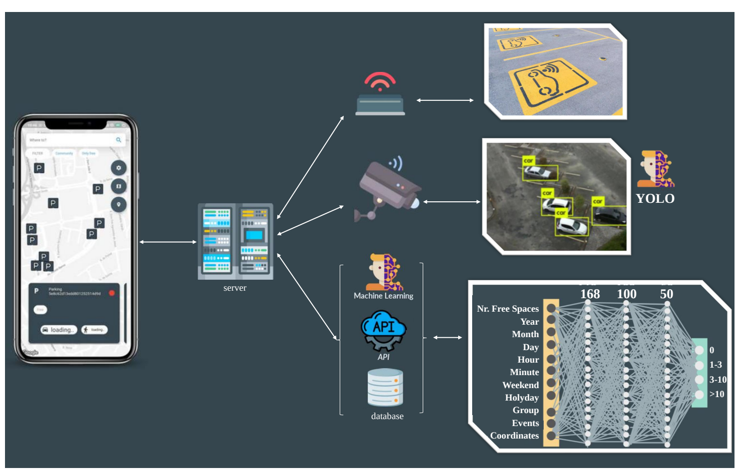


FIGURE 2. Data processing and neural networks: for car recognition (YOLO) and predicting the number of available parking spaces (fully-connected NN with 3 hidden layers of 168-100-50 neurons).

RESULTS and DISCUSSION

While the information collected by the sensors is immediately available to the parking recommendation system, the data from the cameras is in the form of images. Therefore, in order to retrieve relevant information, a NN specialised in object detection should be used to detect the parking spaces and determine their status (free or occupied).

To achieve this, the YOLO (You Only Look Once) NN was chosen because it is able to determine the status of parking spaces in real time with high accuracy. However, in order to train the model, a dataset is required. We tried to create our own dataset, but it was not suitable for the problem under study because the angle at which the images were taken was not ideal and had too many variations. To overcome this difficulty, cameras should be placed in high locations to capture as many vehicles as possible from the same angle (aerial views). Since cars and parking lots do not have a uniform appearance, the amount of data needed to train a model for this task must be huge, and such a dataset does not exist in the literature, and creating it from scratch is practically impossible.

Therefore, in the absence of a suitable training dataset, it was decided to use an existing trained model for vehicle detection - Darknet (an open source neural network framework), which provides two YOLO models that can solve our problem. One with greater emphasis on accuracy, whose model is more robust, and a lighter version for use on less powerful machines, where the model is less accurate. It should be noted that the model was tested without a GPU, which is most likely the case with the machine installed in the parking garage, and took about 20 seconds to detect. The YOLO model was tested on a small set of images representative of the problem under study. Figure 3 (a) shows that the model was able to identify all the vehicles in this image.

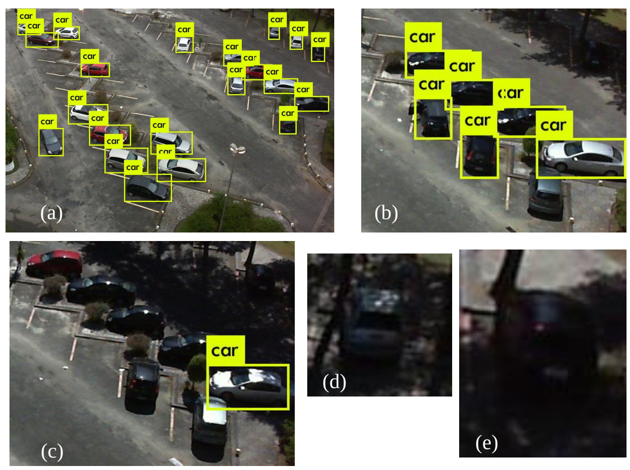


FIGURE 3. YOLO detection results in images of different quality.

Figure 3 (b) shows that YOLO could not detect one of the vehicles, although the picture has good light conditions. If we now look at the same park, but with dimmer lighting (see Figure 3 (c)), we can see that the model had problems, and only detected one vehicle. This shows that good lighting is crucial for the performance of the model. Figure 3 (d) is representative of a parking lot that is very poorly lit (shadows from trees). As you can see, the lines of the vehicle are not visible, so it is very difficult for the model to see them. Figure 3 (e) shows an even more extreme case, where even the human eye has difficulties to detect the vehicle. In summary, vehicle detection problems in dimly lit images can be solved with one of the following strategies: Train the model with more images that were challenging to test, or add samples with noise (if data is scarce); use a camera that allows better quality photos; pre-process the images to increase exposure and contrast so that lines become more distinct and shadows blur or even disappear. Knowing a priori the maximum capacity of a parking space and the exact number of parked cars, the total number of free spaces is the maximum capacity minus the number of occupied spaces.

Intelligent Prediction of Parking Spaces: Since not all parking lots have a system to detect vacant spaces, the number of vacant spaces must be predicted. The prediction does not refer to the exact number of free spaces, but intervals are defined (0 spaces, 0-3 spaces, 3-10 spaces and more than 10 spaces). As in the case of detecting vacant parking spaces by analysing camera images, a neural network (Figure 2) is used in this case to detect more complex patterns and identify relationships between attributes and different features.

The input layer corresponds to the following attributes: **identifier**, **coordinate x** and **coordinate y** of the parking place, **month**, **day**, **hour** and **minute** on which the registration was made, index of the group of **destinations**, that affect the capacity of the park, current **weather conditions**, intensity of upcoming **events**, whether it is a specific period, **weekend**, and, **capacity** interval. The network has 3 hidden layers with a distribution of 128-100-50 neurons and a ReLU activation function. The output layer has 4 neurons and uses Softamx to specify the interval range of free spaces. Based on information about the city of Braga, Portugal, we created a dataset containing both manually collected and scripted generated information, considering information about the city and various literature. The created dataset was previously mixed and divided into 80% training and 20% testing data. To evaluate the model predictions, we used the following metrics: *0* - the model predicted the correct interval; *1* - the model indicated that there were more parking spaces than the real value, which missed by one; *-1* - the model indicated that there were fewer spaces than actual, which missed by one. The same is true for *2*, *-2*, *3*, and *-3*.

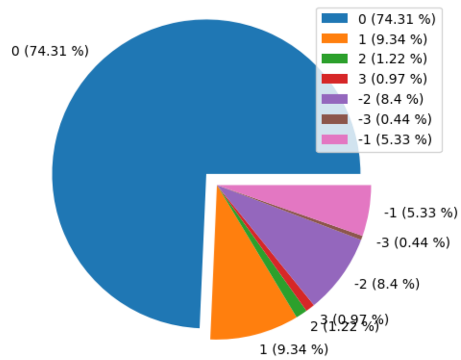


FIGURE 4. Results given by the performance metrics for the free space prediction model.

Figure 4 shows the results obtained when evaluating the distance of the prediction from the assigned value (0, 1, -1, 2, -2, 3, -3). It can be seen that in 74% of the cases the prediction of the parking space was correct in its interval. Of the 26% of errors, 56% correspond to the error of only one interval. In the binary analysis of whether a parking space is available or not, the model is correct 91% of the time, incorrectly indicates that a parking space is present 8% of the time, and incorrectly indicates that no parking space is present 1% of the time.

CONCLUSIONS

This paper presents the architecture of a complete parking search solution that can be implemented and adapted to any scenario. The project covers the entire process, from information gathering, to the dissemination of parking data, to the elaboration of suggestions using artificial intelligence, which in turn recognises patterns, determines the probability that a given parking space is available, and proposes a set of parking spaces based on user preferences.

ACKNOWLEDGMENTS

The authors acknowledge the funding by Fundação para a Ciência e Tecnologia (Portuguese Foundation for Science and Technology) through projects UIDB/00013/2020 and UIDP/00013/2020.

REFERENCES

[1] L. Albuquerque, *Improving Public Parking*, Master's thesis, University of Minho - Department of Mathematics, R. da Universidade, 4710-057 Braga (2021).
[2] A. K. Martin Treiber, *Traffic Flow Dynamics: Data, Models and Simulation* (Springer, 2013).