ML Classification of Car Parking with Implicit Interaction on the Driver's Smartphone

Enrico Bassetti, Alessio Luciani, Emanuele Panizzi

ML Classification of Car Parking with Implicit Interaction on the Driver's Smartphone

Enrico Bassetti, Alessio Luciani, and Emanuele Panizzi

Department of Computer Science, Sapienza University of Rome, Italy
{bassetti, panizzi}@di.uniroma1.it, luciani.1797637@studenti.uniroma1.it

Abstract. On-street parking places parallel to the curb have variable lengths, depending on the size of the car that emptied the place. Thus, a Smart Parking system should publish such places only to drivers whose car is shorter than the available parking length. We developed a crowd-sourced Smart Parking app, based on implicit interaction, intending to publish an available parking spot only to drivers whose car can fit the existing place. This app detects the type of parking (parallel vs angle or perpendicular) using machine learning on the driver's smartphone and considers the length of the cars involved.

Keywords: smartphone · parking · sensing · implicit interaction · machine learning · curb · parallel · angle parking · smart city · context aware.

1 Introduction

Finding an on-street parking place is a problem, especially in big cities. Cars cruising for parking constitute 30% of urban traffic [7] and contribute significantly to air pollution. The average search time for parking is 8 minutes [10], which causes driver frustration.

Smart Parking solutions try to overcome this problem by providing the driver with information about the available parking spots' exact location. Unfortunately, this approach is sensitive to the quality of the information provided, and users may abandon the Smart Parking system if the information is incorrect.

In this work, we focus on one possible misinformation, i.e. the on the size of the parking place. In particular, we refer to parking places that are not delimited in length by marks on the ground, like the on-street parking places marked by a continuous line parallel to the curb (Fig. 1 top). These places have variable length, as it depends on the length of the car parked there before. For this reason, before providing a driver with the information of an available place, it should be evaluated if the place is suitable for the driver's car, as it will not host a car longer than the car which made the parking available.

If the Smart Parking system provides a driver with the wrong parking spot information, the driver will discover it only by driving to that place. This mistake prolongs cruising and reduces the Smart Parking system's perceived reliability,

Fig. 1. Parking place types: parallel (top), angle (bottom left), perpendicular (bottom right)

wasting the effort made. The more frequently these mistakes happen, the less the service is attractive and valuable, leading to user abandonment.

We refer to the crowd-sourced type of Smart Parking systems [4]. Data about the available parking spaces are collected on the drivers' smartphones, possibly using the smartphone's sensors, without any local physical infrastructures.

We approach evaluating the suitability of an on-street parking place for a given car by collecting and processing all the necessary data on the drivers' smartphones and then providing the Smart Parking server with more accurate information about available parking spaces to be published.

In the following sections, after reporting about related work, we describe the mobile app that we developed to collect and process the relevant information to perform available parking evaluation: we describe the driver's interaction with the app, which is both explicit and implicit; we then describe the app architecture and the smartphone sensors we used to allow for implicit interaction; we describe the machine learning model we trained for the evaluation of the parking type and, finally, we report on the results of a test we ran on 540 parking manoeuvres. We conclude with known limitations and ongoing and future work.

2 Related Work

Among the different Smart Parking solutions [4], we focus on crowdsourced Smart Parking systems. We tackled parking information quality and particularly the parking size evaluation in on-street parallel car parks. We did not find papers about this specific problem in the literature. Nevertheless, many inspiring works are available about crowdsourcing for Smart Parking systems and the use of a smartphone for this purpose.

Smartphone sensors have proved quite useful for solving car parking-related issues. Nandugudi et al. use crowdsourced park-unpark events from smartphone sensors to create an estimation of park availability [6]. Soubam et al. [11] use a hybrid approach (accelerometer + WiFi) to spot parking events. Cervantes et al. [3] and Castignani et al. [2] both address the car maneuvering detection using smartphone accelerometer, though not related to parking. Kim et al. [5] exploit IoT sensors, such as wireless beacons and NFC readers, in combination with mobile applications. Wahlström et al. leverage accelerometer and gyroscope samplings to cluster the possible positions of the smartphone in the vehicle [12]. Although we draw from these papers the idea of classifying car motion and smartphone position through smartphone sensors, we exploit sensor data using implicit interaction to address a problem not previously encountered in the literature. Furthermore, we experiment with a different ML approach (i.e. supervised learning).

3 User Interaction

We developed a mobile app to collect relevant data about cars and car parks. We also developed a Smart Parking service to which the app connects to get information about available car parks and to provide information when the driver leaves a parking place.

The driver can add their car to the app, providing some primary data, including car length. Once added, a car is associated with the car Bluetooth (generally available in the car radio or In-Vehicle Information System – IVIS).

The app tracks Bluetooth connections and disconnections to the IVIS when the driver enters and exits the car. Each disconnection is considered a parking operation, and the app records the smartphone GPS position. Each connection to the IVIS is, on the other hand, tracked as an unparking operation, and the app reports to the Smart Parking service an available parking place at the previously recorded location. This interaction is implicit, and in fact, it does not require user intervention, avoiding user fatigue and enhancing reliability on the information gathering. We monitor GPS speed after connection to confirm the unparking event. We call the *giver* a driver who unparked their car and is leaving a place.

The other main interaction in our app is explicit and corresponds to searching for an available parking place. We call the *taker* a driver who is looking for parking. The Smart Parking service provides only available places suitable for the taker's car size, if any. To do this, the giver and taker apps send four pieces of information to the Smart Parking server:

1. giver's GPS position
2. giver's car length
3. giver's parking type (see below)
4. taker's car length

In fact, as described, the GPS location alone is not sufficient for the reporting of an available parking place. The place's size determines the possibility of

parking the car: most of the time, the place width is large enough for any car model, while the place length may vary if strips do not delimit it. Generally, strips delimit the length and width of angle and perpendicular parking places (Fig. 1, bottom-left and bottom-right). On the other hand, parallel places are often delimited only in width, using a stripe parallel to the curb.

Thus, the type of parking of the giver (*parallel* vs *angle* or *perpendicular* [13]) is essential information (along with the length of both cars) to decide if the taker's car can be parked in the place just made available by the giver. Thus, the Smart Parking service will provide a giver's availability information only to takers whose car fits the place size[1].

The app detects the parking type automatically when the driver parks the car through a machine learning algorithm running on the mobile phone. Here the interaction is again implicit. To train the supervised machine learning model, we recorded the smartphone sensors data during parking, and then we asked the driver to label their parking. We used a simple explicit interaction after parking: an actionable notification appears on the driver's smartphone a few seconds after Bluetooth disconnection, asking to select one of the three types of parking.

4 Architecture and ML Model

To classify the parking in the three categories (angle, parallel, and perpendicular) using implicit interaction, we first trained a machine learning model using data collected from the smartphone sensors. After that, we deployed the ML model in the mobile app to run it every time the user parks their car.

Three drivers collaborated with us in collecting data for the model training, exploiting the implicit interaction that we designed. The app recorded the sensor data about their car parking automatically. They only had to report the type of parking by selecting an answer to the automatic notification they received after each parking. The dataset thus collected is composed of 540 parking samples that we doubled by mirroring the manoeuvres.

The app collects smartphone sensor data from the beginning to the end of the trip, using the framework in [8], and store them in a circular buffer that contains 2048 samples. The app collects samples at a rate of 10Hz, i.e. every $100ms$. At parking time, the buffer contains the last 204 seconds of trip data.

The data collected in the samples are the timestamp, the acceleration on the X, Y, and Z axes, the rotation rate on the three axes, the car speed, and the heading, i.e. the angle that the device forms with the geographical north.

The smartphone we used is an iPhone XS running iOS 13.

The collected samples are processed to clean up sensor signals from noise and spikes and remove any part of the trip after the parking manoeuvre. After normalization, we generated new samples by mirroring the collected parking manoeuvres longitudinally to obtain 540×2 samples to feed the ML model.

[1] We do not address the parking space reservation problem, which is ongoing work.

4.1 Trimming

First, we use gyroscope samples to detect if the smartphone is in an idle position in the car or is taken in the user's hands [12]. We observed that the smartphone's rotation rate is relatively stable when positioned in the car (it only undergoes some small fluctuations). However, it reaches high peaks when the smartphone is manually moved (for example, when the driver gets off the car). After doing some tests, we established a threshold of 85 deg/s, and we remove the final samples of all sensors from the first gyroscope peak to the end of the sample set.

Secondly, we trim the initial interval and leave only the last 30 seconds of data. We determined it as the average time for parking by visually analysing a subset of the parking samples that we collected.

4.2 Heading normalization

The heading is the first feature for the ML model (Fig. 2).

All the heading values undergo a normalization process to have the last sample always oriented at 180°. We do this because we want to train the model about manoeuvres independently from the actual car orientation. Also, heading values are allowed to go over degree bounds (0° and 360°) to indicate complete rotations. The final result is a signal as shown in Figure 3.

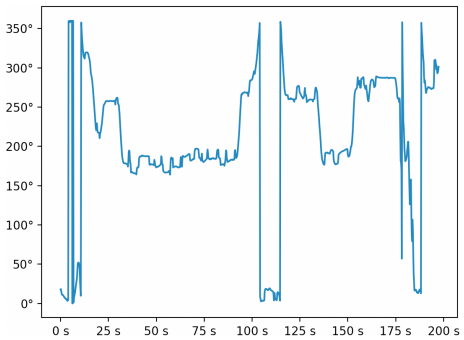

Fig. 2. Raw heading samples

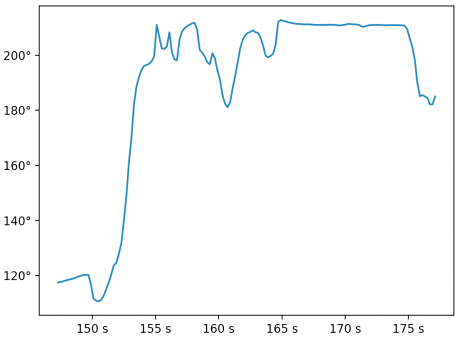

Fig. 3. Final heading signal

Once finished with the heading normalization, we compute a new feature column as the difference between the current heading value and the previous one for every sample. This data is helpful since it resembles the angular velocity on the horizontal plane.

4.3 Acceleration and rotation rate

The mobile framework that we use automatically removes the gravity component from the accelerometer data and the gyroscope data's bias. However, generally,

the smartphone's axes are not aligned with the car's ones. So we shift and rotate acceleration and rotation rate data in order to match the least-squares plane [9] with the horizontal plane (Figure 4). Thus, obtained values are similar to what we could collect if the smartphone were positioned horizontally in the car.

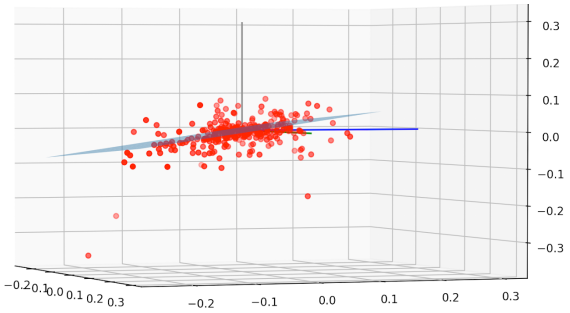

Fig. 4. 3D distribution of acceleration points. The least squares plane is drawn in blue (side view) - values are in m/s^2

4.4 Model creation

The classifier we use is the Create ML "Tabular" classifier [1], based on tree ensemble techniques, which does not need an extensive data set size. We chose it as we have not collected a sufficient number of parking samples yet; thus, we decided to opt for shallow learning. As an ensemble method, it provides good generalization capabilities even with little data. Due to the lack of data, we could not sufficiently train a more sequence-aware model (e.g. an RNN). Therefore, we handcrafted some features that the tree ensemble could interpret as time-related. To do this, we split the total sampling time frame into five contiguous intervals and computed the difference of the heading value between the extremes of those five intervals. This way, those five features formed a sort of temporal sequence of events. We compute other features on the entire time frame (e.g. the acceleration mean). The final feature vector is eventually ready to be input into the classifier (Figure 5).

5 Results and Limitations

We trained two versions of the model. In the first version, we defined the three classes described previously. The model needs to distinguish angle and perpendicular parking motions, although it is not relevant for our case. Results are encouraging, although there is confusion between parallel and angle classes (Fig. 7. However, this model's overall accuracy is 71.42% over three classes (see Table 1).

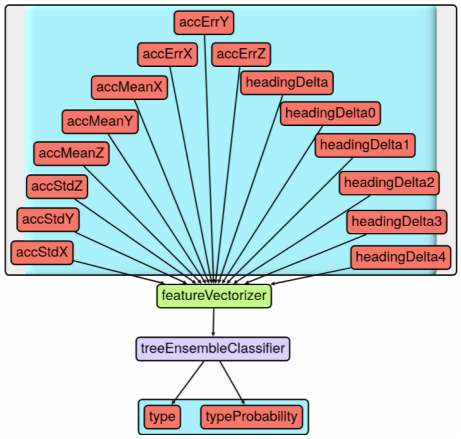

Fig. 5. Tree Ensemble Classifier model - generated by `coremltools` library by Apple™

Input		Result - accuracy	
Samples	**Classes**	**Training**	**Validation**
540 * 2	angle parallel	93.30%	85.99%
540 * 2	angle parallel perpendicular	92.41%	71.42%

Table 1. Results obtained with CreateML Tabular classifier (tree ensemble). The dataset was enhanced using generated data mirroring the samples

The second version simplifies the first one: we group angle and perpendicular parking samples since they make little difference in our case. The latter model's overall accuracy is higher since it is a binary classification, and it reaches 85.99% with significantly reduced confusion between classes (Fig. 6).

The described approach comes with an obvious limitation since it only relies on data from the smartphone's sensors. The aid of external sensors (e.g. car and street-based sensors) would help reach a better accuracy. Additionally, the collected data is manually labelled, and this can have lead to accuracy problems.

For instance, our users may have sometimes postponed labelling to a later moment, then forgetting the type of manoeuvre they did. Another possible error is that they may have had difficulties distinguishing a perpendicular and an angle parking. Last but not least, they may have tapped the wrong button inadvertently in the actionable notification. Therefore the training set can contain some wrongly labelled samples.

In case the user moves the smartphone during the parking manoeuvre, the data collection will be invalidated, as it would have a low probability of belonging to be classified correctly. During training, we assumed that drivers did not move the smartphone during parking, as we instructed them so.

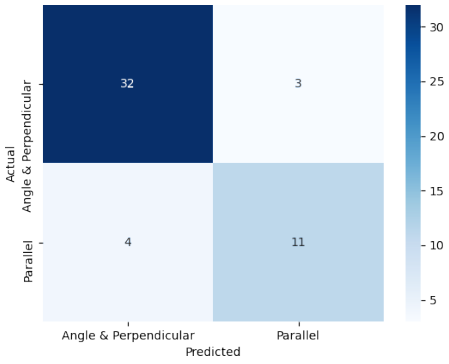

Fig. 6. Confusion matrix (2 categories)

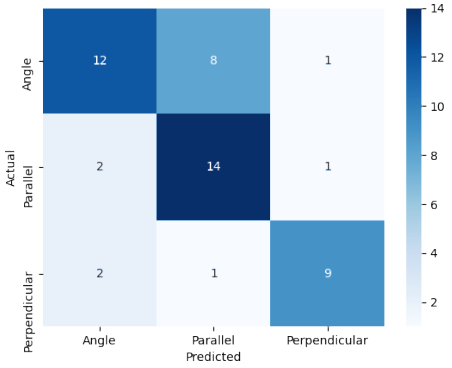

Fig. 7. Confusion matrix (3 categories)

6 Conclusions and Ongoing Work

Our application proved very effective in collecting parking data on the driver's smartphone and can detect parallel vs angle or perpendicular parking type, with an acceptable accuracy of 86%. We obtained this binary classification grouping angle and perpendicular samples in a single class. For our use case of crowd-sourced information for Smart Parking systems, it is unnecessary to distinguish between these two parking types. However complete classification in the three parking types yielded an accuracy value of 71.

Our app's interface successfully exploits implicit interaction to detect parking manoeuvres and unparking actions without the user's need to open the app and interact with it explicitly. Users can perform other tasks in an explicit-interaction style, like adding a car, looking for any available parking spaces, and labelling the parking type for our machine learning model training.

Currently, our collected dataset of parking manoeuvres does not allow us to train a model with our desired accuracy (95% or greater, as opposed to the current 71%-86% for the validation and testing sets). We plan to add more contributors to the data collection process and start filling the dataset at a higher rate. We will then test other neural network architectures designed for motion activity data that could handle time-series samples and, therefore, automatically extract information from the sequence of timestamp-indexed feature vectors (e.g. a recurrent unit).

7 Acknowledgements

This work was partially supported by the MIUR grant "Dipartimenti di eccellenza 2018-2022" and by the grant "Progetti di Ateneo 2019" of Sapienza University.

References

1. Apple: `Create ML` tabular classifier, https://developer.apple.com/documentation/createml/mlclassifier
2. Castignani, G., Derrmann, T., Frank, R., Engel, T.: Smartphone-based adaptive driving maneuver detection: A large-scale evaluation study. IEEE Transactions on Intelligent Transportation Systems **18**(9), 2330–2339 (2017)
3. Cervantes-Villanueva, J., Carrillo-Zapata, D., Terroso-Saenz, F., Valdes-Vela, M., Skarmeta, A.F.: Vehicle maneuver detection with accelerometer-based classification. Sensors **16**(10), 1618 (2016)
4. Diaz Ogás, M.G., Fabregat, R., Aciar, S.: Survey of smart parking systems. Applied Sciences **10**(11) (2020). https://doi.org/10.3390/app10113872, https://www.mdpi.com/2076-3417/10/11/3872
5. Kim, D., Park, S., Lee, S., Roh, B.: Iot platform based smart parking navigation system with shortest route and anti-collision. In: 2018 18th International Symposium on Communications and Information Technologies (ISCIT). pp. 433–437. IEEE (2018)
6. Nandugudi, A., Ki, T., Nuessle, C., Challen, G.: Pocketparker: Pocketsourcing parking lot availability. In: Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing. p. 963–973. UbiComp '14, Association for Computing Machinery, New York, NY, USA (2014). https://doi.org/10.1145/2632048.2632098, https://doi.org/10.1145/2632048.2632098
7. Nawaz, S., Efstratiou, C., Mascolo, C.: Parksense: A smartphone based sensing system for on-street parking. In: Proceedings of the 19th Annual International Conference on Mobile Computing & Networking. p. 75–86. MobiCom '13, Association for Computing Machinery, New York, NY, USA (2013). https://doi.org/10.1145/2500423.2500438, https://doi.org/10.1145/2500423.2500438
8. Panizzi, E., Calvitti, D.: A framework to enhance the user experience of car mobile applications. In: Proceedings of the 20th International Conference on Human-Computer Interaction with Mobile Devices and Services Adjunct. p. 245–252. MobileHCI '18, Association for Computing Machinery, New York, NY, USA (2018). https://doi.org/10.1145/3236112.3236146, https://doi.org/10.1145/3236112.3236146
9. Romero, L., Garcia, M., Suárez, C.: A tutorial on the total least squares method for fitting a straight line and a plane. REVISTA DE CIENCIA E INGENIERÍA DEL INSTITUTO TECNOLÓGICO SUPERIOR DE COATZACOALCOS **1**, 167–173 (12 2014)
10. Shoup, D.: Cruising for parking. Access Magazine **1**(30), 16–23 (2007)
11. Soubam, S., Banerjee, D., Naik, V., Chakraborty, D.: Bluepark: Tracking parking and un-parking events in indoor garages. In: Proceedings of the 17th International Conference on Distributed Computing and Networking. ICDCN '16, Association for Computing Machinery, New York, NY, USA (2016). https://doi.org/10.1145/2833312.2833458, https://doi.org/10.1145/2833312.2833458
12. Wahlström, J., Skog, I., Händel, P., Bradley, B., Madden, S., Balakrishnan, H.: Smartphone placement within vehicles. IEEE Transactions on Intelligent Transportation Systems **21**(2), 669–679 (2020)
13. Yousif, S., Purnawan, S.: On-street parking: Effects on traffic congestion. Traffic Engineering+ Control **40**(9), 424–7 (1999)