Parking space management through deep learning – an approach for automated, low-cost and scalable real-time detection of parking space occupancy

Michael René Schulte
Leuphana Universität Lüneburg

Lukas-Walter Thiée
Leuphana Universität Lüneburg

Jonas Scharfenberger
Leuphana Universität Lüneburg

Burkhardt Funk
Leuphana Universität Lüneburg

Parking space management through deep learning – an approach for automated, low-cost and scalable real-time detection of parking space occupancy

Michael René Schulte[1], Lukas-Walter Thiée[2],
Jonas Scharfenberger[2], and Burkhardt Funk[2]

[1] Leuphana University, Lüneburg, Germany
michael@web-schulte.de
[2] Leuphana University, Institute of Information Systems, Lüneburg, Germany
{lukas-walter.thiee,jonas.scharfenbereger,burkhardt.funk}@leuphana.de

Abstract. Balancing parking space capacities and distributing capacity information play an important role in modern metropolitan life and urban land use management. They promise not only optimal urban land use and reductions of search time for suitable parking, but also contribute to a lower fuel consumption. Based on a design science research approach we develop a solution to parking space management through deep learning and aspire to design a camera-based, low-cost, scalable, real-time detection of occupied parking spaces. We evaluate the solution by building a prototype to track cars on parking lots that improves prior work by using a TensorFlow deep neural network with YOLOv4 and DeepSORT. Additionally, we design a web interface to visualize parking capacity and provide further information, such as average parking times. This work contributes to camera-based parking space management on public, open-air parking lots.

Keywords: design science research, parking space management, object detection, deep learning

1 Introduction

With the increasing population in city centers and the advancement of metropolitan areas, not only the traffic situation is becoming an increasing challenge, but also the search for parking spaces. Every day, millions of people invest time searching for a suitable parking space near their destination. A large-scale study from 2017 by INRIX describes the "impact of parking pain" [1]. Asking 17.868 car drivers in the 30 largest cities in the USA, Great Britain and Germany, Cookson and Pishue [1] study the average search time and economic consequences. The study reveals that German car drivers spend an average of over 41 hours over the course of one year looking for parking space. This does not only lead to considerable time losses, but also causes a tremendous waste of fuel. The study estimated an economic impact of over 40 billion Euros, due to decreased productivity. The picture in the USA is even more alarming,

16th International Conference on Wirtschaftsinformatik,
March 2021, Essen, Germany

with economic costs of over 72 billion US Dollars [2]. However, Cookson and Pishue conclude that this is "a problem that technology can help fix" [3].

One technological approach to reduce "parking pain" is the implementation of parking space management systems and the distribution of parking capacity information. Conventional parking space management systems, such as barrier and gate systems, are installed to monitor entry and exit of parking lots and provide information of the total capacity, i.e. 40 of 80 parking spaces free. While this number is very intuitive and can be used for traffic routing through parking guidance systems, it does not provide information on individual parking spots, e.g. handicapped parking, and is usually not available online for public use. Other technologies, such as induction [4] or RFID sensors [5] can be installed on individual parking spots. Even though this approach provides accurate data on individual parking spots, it has two major disadvantages. First, it is difficult to install these sensors after construction and second, it is rather expensive in initial investment and in long-term maintenance compared to other approaches and therefore less scalable.

A promising approach, which is cheaper, and still provides detailed parking information is parking space surveillance with camera systems and image recognition. In this case, a single camera can capture and assess the occupancy of a large number of parking spaces. Both a subsequent installation of camera systems and the use of already installed cameras is possible, provided that these are connected to the internet. Whereas prior work [10-14] has been focusing primarily on the optimization of individual aspects of such camera systems, we derive the requirements of a parking lot surveillance system and design a ready-to-use artifact that can potentially help reduce the aforementioned problem. For this purpose, we propose a deep learning solution harnessing state-of-the-art frameworks and off-the-shelf hardware components, to ensure resource efficiency, low cost and scalability. This paper goes beyond the experimental test setup of a proof of concept and presents a fully applicable solution, that was tested under real-world conditions.[1] Additionally, in order to provide parking information online, we draft a web user interface. This work contributes to parking space management, focusing on public, open-air parking lots.

2 Research method

In this paper we apply a design science research (DSR) approach. DSR is a framework that develops, tests and finally evaluates potential solutions for a specific real-world problem [6], [7]. The goal of DSR is to address a well-defined general problem and to develop and design the optimal solution based on a systematic approach. The definition and evaluation of requirements for this solution play a pivotal role. DSR is therefore often described as an outcome driven research method that focuses not only on explaining a circumstance, but on creating a useful and feasible artifact. As a framework, design science research aims to create a better understanding for the application of such solutions and provide new insights through the documentation of the results. In our case we consider it very useful to apply DSR for two reasons. First,

[1] Real-world test on a webcam in New York: http://96.56.250.139:8200/mjpg/video.mjpg

DSR offers the necessary scientific and practical guidelines for such projects. And second, we want to present an applicable artifact [8]. Here, we follow Österle et al. [8] and go through the steps problem identification (chapter 3), design requirements (chapter 4), proposed solution (chapter 5), and finally evaluation (chapter 6). Based on the real-world challenges of parking space management mentioned in the introduction, we present related work in the field of parking space surveillance through camera systems and image recognition algorithms. The discussion and categorization of the related work serves the identification of open problems to be solved. We derive our design requirements from related work and real-world challenges and highlight the applicability. We narrow down the possible solutions within our problem identification, and therefore only present one solution to this problem. In order to provide real-time information to end-users, our proposed solution offers a web interface, utilizing state-of-the-art deep learning frameworks. The evaluation of our solution takes place in two ways. First, established machine learning evaluation metrics [9], then qualitative design requirements are discussed in the light of our proposed solution.

3 Related work

Deep learning object recognition in images is a growing field of research with a broad range of applications [10]. That is why different attempts to integrate deep learning object recognition algorithms in the field of camera-based parking lot surveillance have also been explored. This approach promises various benefits, such as cost reduction and time savings, as well as easy identification and higher security.

Conducting a keyword-based literature search on the AIS eLibrary and Google Scholar, we use the search terms "car object detection", "parking lot detection" and "parking lot deep learning". Since we only want to include most recent deep learning approaches, we limit the search to the past four years (2016-2020). Despite similar objectives, we find very different approaches in the literature. We generally distinguish these approaches into three categories 1) background subtraction, 2) image classification, and 3) object detection. In total, we identify five relevant papers which we describe and analyze with regard to the areas of software and hardware, user interface, and data. We summarize our comparison in a concept matrix (Table 1). The matrix is divided into three areas, namely software and hardware, data, and application. Finally, we derive the gap in research and practice and discuss the relevance of our proposed solution.

Soo [11] identifies regions of interest, i.e. image coordinates indicating potential cars, through background subtraction. Therefore, he uses an image of an empty parking lot and subtracts it from a current image of the same parking lot. By subtracting the numeric values of the pixels, only those places with significant differences remain. This method is computationally efficient, since both images are processed as a matrix and then calculated in a single step. However, this method only finds the pixel difference of two images and cannot distinguish objects per se. Thus, to detect cars, Soo has to apply a supplemental classifier. Combining background subtraction and a classifier, Soo is able to count cars on a given image. Yet, he is not able to assign these cars to parking spaces.

Instead of using background subtraction, Amato et al. [12] and Acharya et al. [13] try to achieve the assignment of cars to parking spots by manually defining image sections each showing exactly one parking spot. They use image classification, applying a customed-trained convolutional neural network (CNN) on these image sections. This serial approach yields an accurate assignment. Nevertheless, manually cropping and individually classifying image parts is slow in terms of process. A significant advantage, however, is that single image classification is less resource-intensive in terms of memory (RAM) and computing intensity, than object localization. For this reason, the possibility arises to run a classifier even on a computationally weak edge device. For instance, Amato et al. [12] employ a Raspberry Pi 2, which offers a cost-effective overall solution and high scalability compared to server-side computation.

Another approach for parking space evaluation is object detection. For object detection, it is not necessary to pre-define regions on the image or cropping the image. Instead, the neural network is trained to detect and mark all cars in the frame with bounding boxes. This enables the detection of cars, regardless of their position on the image. Therefore, Chen et al. [14] and Ordonia [15] use the object detection algorithm "You Only Look Once" (YOLO) [16] for car detection. YOLO is a single shot detector (SSD) which processes the entire image in a single step. Chen et al. [14] evaluate the occupation of the parking lot by comparing detected cars to a predefined map of the parking spots, utilizing an Nvidia Jetson edge device. However, the focus of the work is not on parking space surveillance per se. Instead, the authors use a miniature model of road side parking, to develop a smart street lighting system. Their solution is limited to a small number of parking spots.

Ordonia [15] focuses on the reliability of the car identification task with the detector, for which he compares different versions of YOLO on two public datasets. Ordonia focuses on the accuracy of object detection, only counting the cars present, from which he derives detailed capacity statistics. However, he does not map the detected cars to marked parking spaces. Since there is no information given about the hardware used, we assume that a desktop computer was running the demo system.

The papers discussed above only cover specific aspect of the whole system, such as the most reliable image recognition or optimization of computational resources. The authors do excellent work and reveal insights into new frameworks and system approaches. In the course of our research, however, we aspire to design a complete solution, that closely follows the DSR approach.

This paper extends the presented prior work, in particular the work of Ordonia [15]. Based on his findings, e.g. to analyze an entire image in a single step, we also decide to use object detection. Modern object detectors can run on edge devices and offer the possibility to track single objects, which is not possible with common classifiers. Tracking objects enables us to record more sophisticated data, such as parking duration. As we present the related work as basis for our proposed solution, our design science research is focused on three key issues, that we identify (Table 1). First, previous work does not use most recent versions and frameworks in their deep learning application, second, none of the presented solutions feature an end user interface, and third, the data are rarely visualized and forecasted to that extent, that an interpretable result, in the form of "parking space available or not" is given.

Table 1. Concept matrix of related work

Concept	*Sander Soo* [11]	*Giuseppe Amato et al.* [12]	*Debaditya Acharya et al.* [13]	*Lun-Chi Chen et al.* [14]	*Samuel Ordonia* [15]
			Software and hardware		
Computer vision algorithm	Background subtraction (MOG) with classifier	Classification	Classification	Object detection	Object detection
Detection framework	Custom trained CNN for classification	Custom trained CNN for classification	Custom trained CNN for classification	SSD (YOLOv3)	SSD (YOLOv3)
Computation platform	Desktop	Edge device (Raspb. Pi)	Desktop	Edge device (Nvidia Jetson TX2)	Desktop
Specific hardware solution	No – proposed webcam	Yes	No	Yes	No
			Data		
Research environment	Simulated model	Real world application	Real world application	Simulated model + real photos	Real world application
Process mode	Real-time	Real-time	On test dataset	Real-time	Real-time
Training dataset	Custom	Public (PKLot); custom (CNRPark-EXT)	Custom (transfer learning on PKLot and ImageNet dataset)	CNRPark + EXT	Public (CNR, CPARK)
			Application		
End user interface	No	No	No	No	No
Practical application	No – just simulated in a model	No – no end user inter-face and no data workflow	No	No – just simulated in a model and no focus on status inter-pretation	No – missing Hardware and end user interface

4 Design requirements

Compared to design principles, which are more high-level and generic, design requirements in information systems (IS) research "are defined as documented physical and functional needs that a particular product or service must fulfil" [17]. The requirements here are of qualitative nature and the proposed solution needs to be evaluated against them later on. To address the challenges of parking space management, namely capacity tracking and forecasting, our proposed solution is designed to be user-centric, i.e. that it provides a software solution and hardware recommendations. The solution must primarily make a statement on a level that an end user understands, i.e. online information of "parking available or occupied" (DR1.1). An end user in our context can either be a parking lot operator (e.g., admin interface) or a driver (e.g., parking guidance). Since the result must be retrievable and easily interpretable for the user, it has to be processed online and it has to have some kind of easy-to-read visualization (DR1.2). The real-time data are supposed to be viewed and filtered via a web interface. The solution sought is primarily intended to provide a digital real-time overview of the parking situation, therefore, it is important that both the camera evaluation and the server updates are performed live (DR1.3). In addition, the system should be able to be operated permanently and long-term (DR3.2). As we aspire a low-cost solution, we aim to use off-the-shelf hardware (DR2.1). Further, no changes should be made to the parking spaces themselves, such as the installation of sensors or other infrastructure, except camera installation, if needed. The preferred use-case works with pre-installed camera systems. Hence, we are able to implement our solution without any interventions (DR2.2). This requirement is directly related to the scalability of the solution. It should be possible to apply the developed solution on new parking spaces without great effort, which could be important for operators of multiple or complex parking facilities (DR3.1). Since the artifact shall be available to public and feature latest frameworks, the solution is supposed to be built upon open-source software (DR3.3). In order to ensure usefulness of the solution, high accuracy is to be achieved, i.e. F_1 score of at least 95% (DR3.4). We summarized the design requirements in Table 2.

We are aware that privacy and data security play an extremely important role in video surveillance of parking lots. A public or commercial use of our solution should therefore consider these aspects. Our proposed solution shall be able to recognize individual vehicles, but no information about driver, passengers or license plate shall be tracked here. As we focus on the technical solution, we will not convert this into a separate design requirement. However, we want to emphasize that, in our approach, we do not need to store video stream or image data permanently.

Table 2. Summary of the design requirements

Number		*Description*
DR1		**Provide status information of current parking situation**
	DR1.1	Provide online interface
	DR1.2	Display visual representation
	DR1.3	Offer real-time status
DR2		**Ensure low-cost implementation**
	DR2.1	Use off-the-shelf hardware
	DR2.2	Minimize modification/intervention of main infrastructure, only camera with internet connection, if necessary
DR3		**Provide system package (hardware, software)**
	DR3.1	Ensure scalability
	DR3.2	Permanent operation
	DR3.3	Use open-source software modules
	DR3.4	Ensure reliability: F_1 score > 95%

5 Proposed Solution

In order to provide a useful artifact to parking space management on public, open-air parking lots and ultimately to address the introduced problem of "parking pain", we propose a solution, that can display real-time occupancy of monitored parking spaces via a web application. For this purpose, we use camera images that are automatically evaluated with a deep learning model. Utilizing cameras is a suitable approach, since many parking lots have already installed cameras or can mount them with as little interference as possible and at low cost.

Our proposed solution includes two parts. First, we describe our technical approach, which highlights the deep learning algorithm for the automated recording and evaluation of parking spaces. Second, the implemented approach is presented, which includes the hardware system and an end-user interface in the form of a web app for real-time display of free and occupied parking spaces. As we strive for an applicable artifact, the proposed solution must be consistent from input to output. The system we present includes three layers, namely the preprocessing of a camera stream, image interpretation with deep learning and the visual representation in a web app. The layers and the schematic data flow are presented in Figure 1.

5.1 Deep learning approach

In this section we present an object detection based solution for automated assessment of occupied and free parking spaces. The solution covers the object detection algorithm for detecting cars, the algorithm for determining the occupancy of parking spots, and object tracking for determining parking durations of each car.

Ordonia [15] compares the first three versions of YOLO, achieving an accuracy level over 90% with YOLOv3, and confirms YOLO's reliable and relatively resource-friendly approach as a single-shot-detector. The release of YOLOv4 promises even better performance.

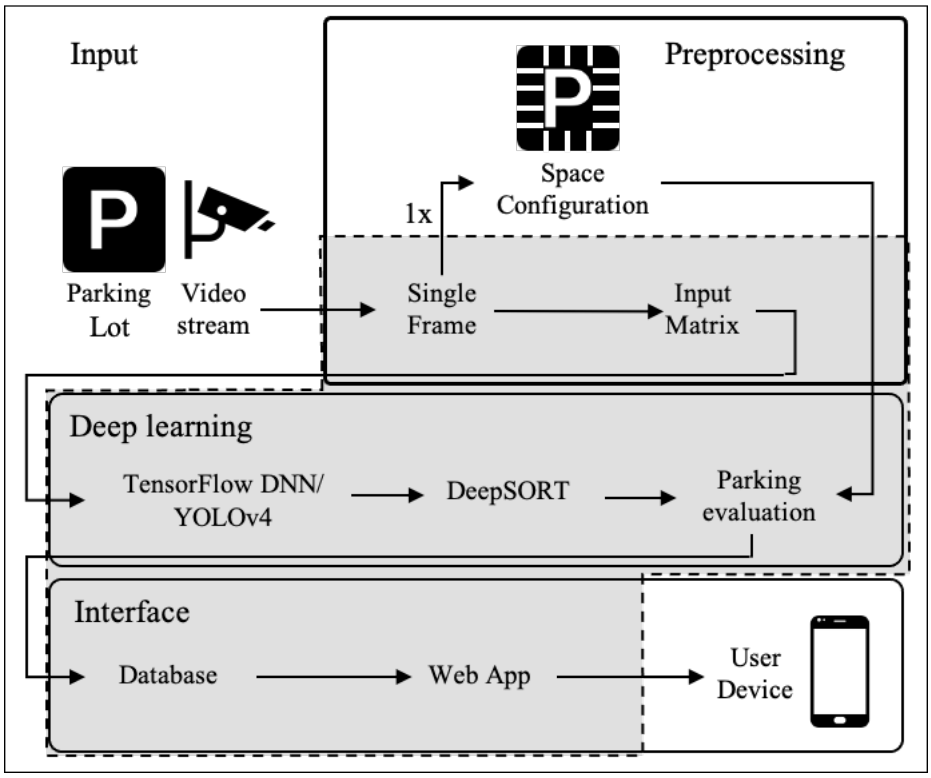

Figure 1. Schematic presentation of proposed solution highlighted in grey color.

[18] indicates that YOLOv4 is the most accurate open-source, real-time detector, tested on the Microsoft's Common Objects in Context (COCO) dataset.

Therefore, our solution takes up the YOLOv4 algorithm. We use YOLO's TensorFlow implementation, since TensorFlow is an essential anchor point in the artificial intelligence industry. Our model is pretrained on an ImageNet dataset [19]. This dataset includes images of people, animals, but also cars, trucks and other vehicles. Thus, this pretrained model serves as a perfect basis, since the deeper layers are already trained to detect vehicles. Fine tuning of the model is performed on a dataset of images taken over a 24-hour period at three-minute intervals in the test parking lot. 90% of the annotated data is used for training (395 images) and testing (113 images) and 10% (57 images) for later evaluation of the results. Therefore, we apply transfer learning, which is a technique to transfer and leverage the learning results of a trained neural network to a related task. This method requires comparatively small datasets, which is useful in our case, i.e. training a neural network to recognize parked cars [20].

Instead of just detecting an object on a single image, we can also track this object over several frames. This technique is called object tracking and enables us to identify and track cars over time by assigning a unique ID to each car. This allows our system to collect parking data, such as durations, spots, or daytime. Hence, our solution can automatically produce a variety of statistics and can ultimately be used to predict parking data. Object tracking algorithms are used in combination with object detection algorithms. A recent approach to object tracking is DeepSORT (Simple Online and Realtime Tracking) [21] which uses the Kalman Filter and deep learning. Our approach combines DeepSORT with our previously trained YOLOv4 model in order to track detected cars.

Figure 2. Frame with YOLOv4 (objects), DeepSORT (IDs) and occupation (green/red color)

Further, after detecting and tracking unique cars, we map their position on the regarding parking lot to evaluate the occupancy (Figure 2). For every pre-defined parking box, our algorithm compares each detected car object with this spot and considers a parking space to be occupied if a recognized car covers this space to at least 60%. We use the python package Shapely to calculate this intersection. The threshold value needs to be chosen based on the camera angle and the individual parking style of the drivers and might differ from parking lot to parking lot. If the percentage value is too low, the bounding box of a car may also cover parts of a parking spot behind it. Hence, this parking spot would be falsely marked as occupied. Furthermore, it is possible that a car was not parked exactly inside the parking spot and it is falsely interpreted as free.

5.2 Edge Device

Our hardware approach is based on the edge device Nvidia Jetson Xavier NX. We choose the Jetson Xavier NX because of its high performance, resource-efficient operation and small size. With 8 GB RAM, 4 CPU cores and a 32 GB micro SD card, the Xavier NX fulfills all requirements to perform the entire parking lot analysis. At the same time, the system runs at only 10 watts. To access video data, we embed the video stream of the corresponding parking lot. We don't specify the connected camera. Since we use the OpenCV library to convert the video streams into individual frames, our solution can handle a multitude of common video formats, that are supported by OpenCV. We load the latest image into our model. The results are then saved locally and sent to the central server running the web app. Experience shows that the process of entering and exiting the parking lot takes about half a minute, an actual real-time analysis is not necessary. Therefore, we decide to analyze and evaluate the most recent frame of the live video just every 20 seconds. Despite this huge reduction in data size and transfer, we can still provide almost real-time information within the web app.

5.3 Implemented Solution

The goal of our solution is to provide end users with an intuitive and clear overview of the current parking situation. In our opinion a web application offers the fastest and most flexible approach to this. The software is therefore platform independent and doesn't require installation. Users can access the URL[2] and view the parking lot with an internet-capable device. This is useful, since one-time parking visitors can quickly view the parking situation without any prerequisites. Figure 3 represents the bird's-eye view of the parking lot in a graphical representation and indicates them as free (green) or occupied (red). This is not only very useful for potential parking visitors, but also for parking lot operators. We further provide an admin interface. The admin panel offers a number of additional functionalities, such as a heat map of daily utilization of individual parking boxes, total and average parking times. An occupancy rate graph allows, on the one hand, to compare the daily utilization and on the other hand, it serves as a forecast for the current day. Another graph visualizes the number of unique visitors ().

[2] Link to the web app: https://sparkle-network.com/app/

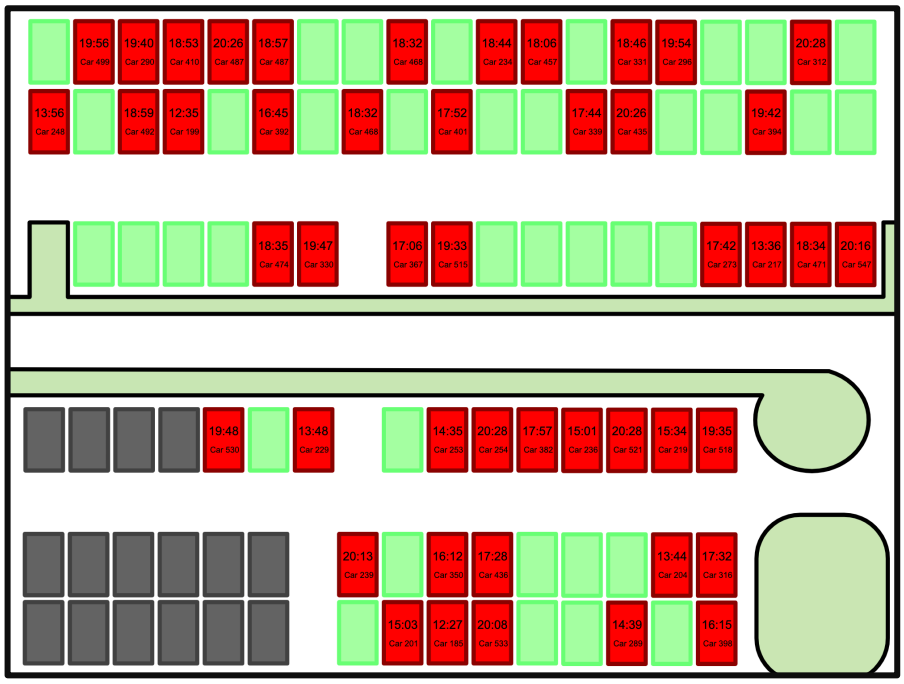

Figure 3. Live overview of the parking situation in the web interface

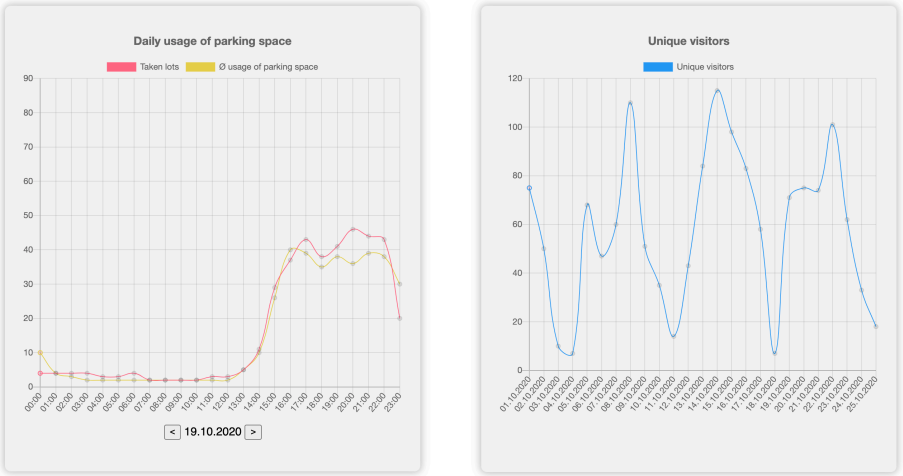

Figure 4. Visualization of parking data

6 Evaluation and Results

6.1 Evaluation of the deep learning application

In order to provide useful information to visitors and operators, accuracy and reliability of the algorithm are essential, i.e. the accuracy of the object recognition and the correct classification of free and occupied parking spaces.

We follow the approach of splitting our dataset into 70% training, 20% test and 10% evaluation data. Therefore, we evaluate our algorithm on 57 images of the parking lot containing both daytime and nighttime images. In total we evaluate our model on 4845 parking spaces – 3855 free and 990 occupied spaces (Table 3) – on accuracy, precision and recall [9]. Our model achieves an accuracy of 99.48% on our test data, i.e. nearly every classification (free or occupied) is correct. However, since the number of free and occupied spaces is imbalanced the accuracy could be misleading here. Therefore, we also take the precision and recall into account. We calculate a recall of 98.57% and a precision of 99.89%. This leads to an F_1 score of 99.22%, verifying the high reliability of our predictions.

Table 3. Confusion matrix

	Actual occupied spot	**Actual free spot**
Predicted occupied spot	966	1
Predicted free spot	24	3854

6.2 Evaluation of the requirements

We conduct a test of our solution for several weeks and qualitatively compare our result with the design requirements from chapter 4. The main goal of our solution is to provide status information of the current parking situation. We provide a web app, that displays visual information. Therefore, DR1.1 and DR1.2. are fulfilled.

As discussed in 5.2, the server updates occur in real-time, however, for a better system performance, it has proven to be best to analyze the parking lot only every 20 seconds, which in a narrow sense does not fulfill DR1.3 (real-time). Nevertheless, due to the fact that drivers need some time to park their car, this did not have any substantial effect on the quality of the results in the test system. Our system still meets the qualitative requirement of a current overview of the parking situation. The proposed solution has been developed with existing standard hardware. The Jetson Xavier NX has been optimized by Nvidia for the operation of neural networks. Thus, DR2.1 regarding off-the-shelf hardware is achieved. Furthermore, the one-time hardware costs for the Jetson computer, a suitable case and a SD card only add up to 530€. The annual operating costs of the Jetson at 10 watts and an electricity price of 0.28€/kWh are estimated to be around 25€. Especially with regard to comparable solutions such as built-in sensors in individual parking boxes, both the initial investment costs and the long-term operating costs are remarkably lower, satisfying DR2.1. Since our system

offers the possibility to use already existing cameras, the modification of the parking lot in this scenario is minimal. DR2.2 however, requires no or only little modification of the parking lot. This is the only way to realize a low-cost approach. We consider this objective as achieved with respect to alternative solutions. We highlight that the cameras need internet and power supply to send the analysis data to the server. DR3 refers to a fully functional system. Since our test phase shows that the Jetson is capable of performing the entire computation, the requirements DR3.1, a scalable system, and DR3.2, permanent operation, are also satisfied. Instead of having a bottleneck due to a central server approach, the entire computation is done on the local device. Only the result data are transferred to the server. This reduces the data traffic and allows a resource-saving implementation of further systems on other parking spaces. Additionally, the system is based on an open-source approach. The neural network YOLOv4 for image recognition is publicly available and was trained by us through transfer learning on a custom dataset. Thus, the procedure meets the requirements of DR3.3. In addition to the operation itself, the reliability and accuracy of the system are essential. An evaluation based on test data has resulted in an accuracy of over 99% for the used parking space and thus satisfies the requirement DR3.4.

Summarizing, the previously defined requirements have been taken into account and fulfilled in our system. Nevertheless, the developed solution is limited to some extent, e.g. the camera must be able to cover the whole parking lot. Potentially more than one camera has to be in use. Also, the solution still needs to be evaluated with end users.

7 Conclusion

In this paper we provide a solution for camera-based parking space management with a focus on low-cost public and open-air scenarios. Based on a design science approach, we apply existing deep learning approaches in the field and select hardware to construct a useful and feasible artifact, which is the very core of design science research. Our deep learning approach features TensorFlow, YOLOv4 and DeepSORT and is able to automatically identify free parking spaces at a 99% accuracy level. We use transfer learning to train the model on self-labeled data from a selected public parking lot. Our hardware approach, using an edge device, enables a resource efficient and scalable concept and ensures low costs. Additionally, we design a web application, that can potentially assist drivers as well as parking lot managers to view live information about the parking space.

We evaluate our solution qualitatively against the design requirements, that we derive from prior work and challenges in the field. We can successfully meet the majority of the requirements. Within our deep learning algorithm, we integrate the SSD Detector YOLOv4, which proves to be very reliable regarding object recognition of a large number of small objects. The achieved accuracy refers both to the car detection and to the parking lot evaluation. A major challenge in this evaluation arises with the camera angle, i.e. in an acute angle objects cannot be classified correctly. Another challenge is given by the computation power of the edge devices. We solve this issue by utilizing Nvidia Jetson Xavier NX, which offers sufficient memory (RAM).

A key challenge in the deep learning approach is certainly the generalization of the dataset. In order to avoid retraining the system for each new parking space, the neural network must reliably recognize cars under a wide variety of conditions and camera angles. Future research should therefore try to collect significantly larger and more diverse training datasets, in order to gain better generalization. Since we evaluate the solution only on a single parking environment, we engage other researches to use our approach to test a variety of other parking spaces.[3] The project was primarily driven by the focus on a useful artifact and viable prototype. Our solution was tested in real-world conditions and can be the basis for further development of camera-based parking space management and ultimately help to optimize urban land use and traffic routing.

Future research can also integrate advanced functionalities in the system, such as automatically recognizing parking lot boundary lines, or enhancing the system with a predictive algorithm for future occupancy scenarios. Another feature could be the integration of GPS functions in the web app to enable on-parking navigation. The goal would be to develop a complete program that combines all these individual functionalities and runs as a standalone product. A long-term goal could be the cross-platform integration of this solution with various digital parking management solutions such as sensors in parking garages or roadside scanning from car manufacturers. By pooling different solutions in one platform, individual solutions for specific parking scenarios can be developed and integrated, thus contributing even better to a holistic urban traffic and city planning.

[3] Our code is available on GitHub: https://github.com/derm1ch1/spARkle

References

1. Cookson, G., Pishue, B.: Stress durch Parkplatzsuche in Deutschland. INRIX, Inc. (2017)
2. Cookson, G., Pishue, B.: Impact of parking pain in the U.S. INRIX, Inc. (2017)
3. Cookson, G., https://inrix.com/blog/2017/07/parkingsurvey/ (Accessed: 04.10.2020)
4. Revathi, G., Dhulipala, V. R.: Smart Parking Systems and Sensors: A Survey. International Conference on Computing, Communication and Applications. (2012)
5. Pala, Z., Inanc, N.: Utilizing RFID for smart parking applications, pp. 101-118. (2009)
6. Hevner, A. R., March, S. T., Park, J.: Design Science in Information Systems Research, pp. 75-105. (2004)
7. Peffers, K., Tuunanen, T., Rothenberger, M. A., Chatterjee, S.: A Design Science Research Methodology for Information Systems Research, pp. 45-78. (2007)
8. Österle, H., Becker, J., Frank, U. H. T., Karagiannis, D., Krcmar, H., Sinz, E. J.: Memorandum on design-oriented information systems research. (2011)
9. Hossin, M., M.N, S.: A Review on Evaluation Metrics for Data Classification Evaluations. (2015).
10. Dargan, S., Kumar, M., Ayyagari, M. R., Kumar, G.: A Survey of Deep Learning and Its Applications: A New Paradigm to Machine Learning. (2019)
11. Soo, S.: Object detection using Haar-cascade Classifier. U. o. T. Institute of Computer Science. (2014)
12. Amato, G., Carraraa, F., Falchia, F., Gennaroa, C., Meghinia, C., Vairoa, C.: Deep Learning for Decentralized Parking Lot Occupancy Detection. ISTI-CNR. (2017)
13. Acharya, D., Yan, W., Khoshelham, K.: Real-time image-based parking occupancy detection using deep learning. Infrastructure Engineering, The University of Melbourne. (2018)
14. Chen, L.-C., Sheu, R.-K., Peng, W.-Y., Wu, J.-H., Tseng, C.-H.: Video-Based Parking Occupancy Detection for Smart Control System. MDPI. (2020)
15. Ordonia, S.: Detecting Cars in a Parking Lot using Deep Learning. San Jose State University. (2019)
16. Redmon, J., Divvala, S., Girshick, R., Farhadi, A.: You Only Look Once: Unified, Real-Time Object Detection. (2016)
17. Koppenhagen, N., Gaß, O., Müller, B.: Design Science Research in Action – Anatomy of Success Critical Activities for Rigor and Relevance. IISM. (2012)
18. Bochkovskiy, A., Wang, C.-Y., Liao, H.-Y. M.: YOLOv4: Optimal Speed and Accuracy of Object Detection. (2020)
19. ImageNet, http://image-net.org/ (Accessed: 04.10.2020)
20. Pratt, L. U., Mostow, J., Kamm, C. A.: Direct Transfer of Learned Information Among Neural Networks. AAAI. (1991)
21. Wojke, N., Bewley, A., Paulus, D.: Simple online and realtime tracking with a deep association metric. (2017)