PKSpace: An Open-Source Solution for Parking Space Occupancy Detection

Roman Števaňák*†
Adrián Matejov*‡
Ondrej Jariabka§
Marek Šuppa¶
*Supervised by: Marek Nagy‖ and Igor Farkaš***

Faculty of Mathematics, Physics and Informatics, Comenius University in Bratislava

Abstract

Parking space management is an issue that affects any building above certain size. While there are solutions which rely on networks of connected sensors for each parking space, these are usually difficult to maintain and expensive to implement. In this work we present PKSpace – an open source solution for detecting vacant and occupied parking spaces, using only an inexpensive camera and single-board computer. PKSpace aims to be the whole package: by being able to obtain the images, provide admin interfaces for specifying parking spaces, marking them as vacant or occupied, and offering tools for training machine learning models, which are capable of predicting the number of vacant or occupied parking spaces in new images, it allows the user to deploy the system in a short amount of time while minimizing the amount of resources necessary for its maintenance. This work also includes a comparison of methods used for this task and their empirical evaluation on the FMPH parking dataset, which is released as part of this work.

Keywords: parking lot monitoring, machine learning, computer vision, classification, object recognition, open source

1 Introduction

The problem of finding a free space to park one's car is one that inevitably concerns residents of large metropolitan areas. Since the issue is fairly pressing in many areas, the industry provides multiple solutions. De Almeida et al. [5] categorize them into three buckets: counter-based, sensor-based and image-based.

The counter-based systems usually utilize gate-arm counters located at entrances and exits of parking lots. They simply count the incoming and leaving vehicles and provide their difference as the number of occupied parking spaces. Due to their relatively low cost of deployment they can be found in a great number of parking lots around the globe. While a system like this does help one to get a sense of the occupancy of a given parking lot, it does not provide any additional information, which could for instance help navigate an incoming vehicle to a free parking spot.

The sensor-based systems use various types of detection sensors, such as for instance ultrasonic [12] or magnetic field [20] sensors, which need to be installed at every parking space and send gathered data to a centralized processing unit. This type of system provides more information that are of interest when dealing with parking space management, but does not scale well for bigger parking lots, since its cost increases with every added parking space.

Finally, the image-based systems usually take a video or image stream as their input (see Figure 1). This has multiple advantages, the biggest one being the fact that often times no additional infrastructure is required for their deployment, as many parking lots already have surveillance cameras, which overwatch the whole space. Moreover, with the recent improvements in the area of smart cameras [3], the installation costs for such systems are usually minimal. Since these systems use images as their input, the availability of public datasets they could be tested on is of great importance. Unfortunately, in the past the authors have been forced to use extreme measures, such as finding images via Google Image searches [6]. This has improved in the past few years thanks to the introduction of large scale (hundreds of thousands of parking spaces) datasets, such as those introduced in [5] and [2].

In this work we present an opensource image-based solution to the problem of parking space occupancy detection called PKSpace. It allows the end user to use single web interface to perform all tasks necessary for its deployment: viewing the live feed from the camera while adjusting its position, marking the positions of respective parking spaces in a given parking lots, creating a train-

*Both authors contributed equally to this work
†rstevanak@gmail.com
‡ado.matejov@gmail.com
§o.jariabka@gmail.com
¶marek@suppa.sk
‖mnagy@ii.fmph.uniba.sk
**farkas@fmph.uniba.sk

(a) (b)

Figure 1: Example of parking lot with visualized mask from PKLot dataset. Red polygons represent occupied spaces and green ones free spaces. Note that some spaces are not labeled due to heavy occlusion

ing dataset from the data this camera gathered, using the prepared training dataset to better align the provided pre-trained model with the given parking lot and finally view predictions for newly gathered image data. It was designed to be inexpensive, easy to use, and at the same time provide Application Programming Interfaces (APIs), which allow outside systems to consume data it gathers, and use it for further processing. Furthermore, we also introduce a new dataset which better approximates the situations the models may run into in a real deployment.

This paper is structured as follows. In Section 2 the work relevant to the proposed solution is discussed. Section 3 describes datasets which are later used to select the best performing methods and train models, which are available in the PKSpace package. Section 4 describes the architecture of the proposed solution, while Section 5 introduces the tested models and Section 6 summarizes the results of these tests. Finally, Section 7 presents our conclusions and discussion on possible future work.

2 Related Work

As Huang et al. [11] noted, most of the previous work in the area of vision-based parking spaces occupancy detection can be roughly classified into two categories: car-driven and space-driven. When using former approach, the task is usually formulated as "Given an image find objects closely resembling cars". Parking spaces which do not include such objects are then deemed to be empty. In order to find these objects a number of object detection algorithms can be used, such as [17] or [19]. However, this approach falls short when the effect of perspective distortion can be seen in the input images – the area of the car progressively decreases with its distance from the camera, and it is harder to match by an object detection algorithm.

The space-driven approaches usually make use of a simple assumption: there should be minimal variation in images of vacant parking spaces, given a short time window. In other words, it is assumed that the background is statistically static for a short amount of time [11]. In such context, a viable strategy is to use background subtraction [9]. Unfortunately, while the assumption from above may hold in controlled environments (for instance an indoor parking lot), the outdoor scenes are usually vastly different, even in short time spans. Consider for instance sudden weather changes, random scene shadows or just a moving cloud – all of these rendered background removal techniques close to useless, as the variations in background are no longer minimal. One possible way of dealing with this is to model various possible types of lighting conditions. This requires a lot of memory, so Funck et al. [8] have proposed a memory and computationally efficient model which uses eigen-space representation to model vast amounts of backgrounds. A more robust approach to this problem was proposed in [16] where features based on Gabor filters are used as input to a classifier which classifies empty spaces in different lighting conditions.

Despite the already mentioned rough classification into two categories, a new kind of approach appeared in the past few years. It tries to deal with the problem of light changes by modeling the volume of a parking space in 3D while using 2D data as input. To this end, Huang et al. [10] propose a Bayesian hierarchical framework, which in their experiments managed to identify shadowed regions it was asked to model as part of its training regime. In a similar fashion, the approach discussed in [6] estimates the probability of a vehicle being present in a specific parking space, which helps to account for occlusions in the input image.

Some of the very first work in using Machine Learning methods for classification of parking spaces as vacant or occupied can be found in [4], where Dan trained an SVM classifier for each considered parking space, using color vector features as its input. One of the biggest problems of this approach is that it does not take occlusions into account. To overcome this problem, Wu et al. [21] use histograms extracted from three neighboring parking spaces as features for their SVM classifier, which classifies all three parking spaces at the same time (into one of eight classes). This allows for better modeling of the inter-space correlation and the Markov Field Framework then helps to solve potential conflicts in predictions. While this approach has been shown to effectively deal with inter-space occlusions, it is still greatly affected by variations introduced by the environment.

(a) Sample image from UFPR04 (b) Sample image from UFPR05 (c) Sample image from PUCPR

Figure 2: Examples of each parking lot from PKLot dataset

While all of the aforementioned work provide important contributions towards the goal of efficient vision-based parking space classification, it still remains an open problem. Part of the reason is the lack of a well curated and established dataset. This situation was greatly improved by the introduction of the PKLot dataset in [5], which subsequently generated a lot of interest for this particular task. The baseline models proposed by Almeida et al. have already been improved using modern classification techniques used in Computer Vision, which are based on Convolutional Neural Networks (CNN) [2, 18]. These models usually have a lot of free parameters which are optimized during training, which gives them great representation power, but at the same time requires bigger amount of resources necessary for their training as well as actual real-world use. Ahrnbom et al. [1] try to propose a compromise by using an SVM-based classification with features which can be extracted very quickly, yielding a fast classifier with close to state-of-the-art performance. In [2] Amato et al. also acknowledge this problem and try to address it by limiting their proposed CNN in size. They further show that despite its smaller size, it achieved state of the art results on the PKLot dataset. Moreover, they also validate its generalization property by evaluating it on a new dataset called CNRPark-EXT, which was made available to the scientific community.

3 Parking Lot Datasets

Most recent works in this area utilize publicly available datasets like PKLot or a more recent one called CNRPark-Ext. Both of these use high-definition cameras, and feature a top-down view, so overlays of cars parked side by side are not that significant. Moreover, the images are taken on parking lots with lines separating spots clearly marked on the ground, so exact masks can be made for each parking spot independently. The parked cars usually adhere to the lines, and so cars parking over multiple parking spaces can be considered an anomaly (Figure 1a).

One issue with these datasets is that they were obtained in circumstances which cannot always be assumed in real world usage of parking space detection systems. To this end we also introduce the FMPH dataset, that represents a more real-world setup. All of these datasets are described in more detail in the following sections.

3.1 PKLot

This dataset is composed of pictures of two parking lots - one next to the administrative building of Pontifícia Universidade Católica do Paraná (PUCPR) (Figure 2c) and the other in front of the Federal University of Paraná (UFPR) (Figure 2a and Figure 2b), with the latter being captured by two cameras from fourth and fifth floor in 5 minute intervals for 30 days. It consits of 12,417 pictures with resolution of 1280x720 pixels. In total these pictures depict 695,899 parking spaces out of which 337,780 (48,5 %) are occupied and 358,119 (51,5%) are empty.

These photos are taken under various weather conditions, such as, overcast (Figure 2c), rainy weather (Figure 2a) or sunny weather (Figure 2b) and multiple lighting conditions throughout the day. All pixels corresponding to parking spaces are manually marked as such, and labeled as occupied or free by a human expert.

For each picture there exists an XML file that describes it. For each parking space it contains whether it is occupied or not, its center, rotation, height and width, so that it can be cut out from picture and possibly rotated.

The suggested ratio of dividing the data of this dataset into training and testing sets is 1:1. All pictures from one day should only be included in one of them. The recommended method for evaluation is overall error rate, which is defined as follows:

$$OER = \frac{FP + FN}{TP + TN + FP + FN}$$

where TP, TN, FP and FN stand for the amount of true positives, true negatives, false positives and false negatives.

PKlot is used as the standard dataset in a large body of academic literature on this topic (see for instance [7, 13, 1, 18]). It has parking spaces with small to no overlaps, clear lines for parking spaces and cars usually parked within these lines (Figure 2).

3.2 CNRPark-Ext

The CNRPark-EXT dataset is composed of roughly 4,287 pictures with resolution of 2592x1944 pixels.They are

(a) Snow and sunny (b) Snow (c) Fog

Figure 3: Images of various weather conditions during different seasons from FMPH dataset. Note, significant reflections from glass pane

taken by nine high-definition cameras over the course of 23 days. Individual parking spaces are extracted by cutting out so called patches. These are non-rotated rectangular shapes, that do not necessarily cover the space entirely or precisely (as can be seen in Figure 4). These are then resized into 150x150 pixels in size. The dataset is composed of 144,965 patches out of which 65,658 (45,2%) are free and 79,307 (54,8%) are occupied.

Photos are categorized by several parameters, for example, weather conditions observed at the time of their acquisition, such as, overcast (Figure 4a), rainy (Figure 4b) or sunny (Figure 4c). Another way is by the day and time they were taken, or by the ID of the camera. The occupation labels for each respective patch are loaded from one of the included text files. This dataset contains heavy occlusion, but it can be compensated by the fact that it also includes multiple views of the same parking spot. Suggested division into training and testing data can be found in the included text files.

(a) (b) (c)

Figure 4: Examples of patches from CNRPark-Ext

3.3 FMPH

FMPH dataset (Figure 3) is named after Faculty of Mathematics, Physics and Informatics in Bratislava, where it was created. It consists of 1,093 pictures that depict 25,139 parking spaces out of which 14,311 (57%) are occupied and 10,828 (43%) are not. They are shot in 15 minute intervals from the 31st of December 2016 to the 30th of January 2017. Pictures are taken in various weather conditions like fog (Figure 3c), sunny weather with snow (Figure 3a) or pure show (Figure 3b). Combined with different lighting conditions throughout the day they were captured at, they provide new setup which is lacking in the existing datasets and closely resembles a real-world scenario.

There is a JSON file created for each picture. It holds information about each parking space – a set of points defining its area, the angle and binary value whether it is occupied or vacant. The position of center is calculated by averaging defining points. The image is firstly rotated around this center by the specified angle. Then, minimal bounding rectangle of space defining points is cut out.

This dataset presents a more real-world setup a parking space vacancy detector may encounter. The camera is placed behind a window pane, so reflection artifacts can sometimes be seen on included images. It is also positioned not too high above the ground, so parking spaces have big overlays from the camera's point of view. It also means that at a certain time there is sun shining directly into the lens, so nothing can be seen. These pictures were removed from the dataset, as it is expected that such images will be ignored by the pre-processing steps.

This dataset was shot by low cost camera, meaning that the included pictures have lower resolution of 640x480. In this dataset there can be different masks for same parking space in different pictures, in order to remedy the lack of parking lines on the ground and subsequently unpredictable parking of cars.

4 PKSpace Architecture

The architecture of PKSpace is based on a standard approach utilized in the Internet of Things paradigm: a set of cameras monitor parking lots, each one with its own processing unit which is connected to a server in a local network (Figure 5).

All of the data gathering, processing, analysis and prediction procedures take place independently on each processing unit. Results prepared by separate processing units can be later aggregated by the central server. The only requirement the proposed architecture has on this server is that it needs to run an instance of an SSH server. This allows for creation of secure SSH tunnels via this central server, thanks to which processing units can be accessed from outside of the internal network, as well as simple

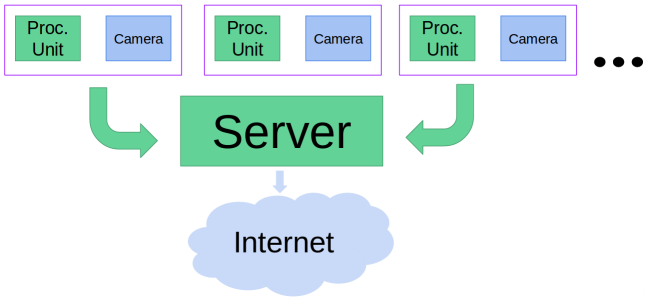

Figure 5: Separate processing units with their own cameras. Server is gathering results which are accessible from the internet

maintenance and administrative access.

The processing unit is structured into three separate modules – Capturer, Predictor and the Admin Interface (Figure 6). These are described in more detail in the following sections.

4.1 Capturer

The purpose of this module is to gather images from any suitable source, most probably a camera connected to the processing unit. It is also expected that the images produced by this module are suitable for further processing (i.e. there are parking spaces to be recognized).

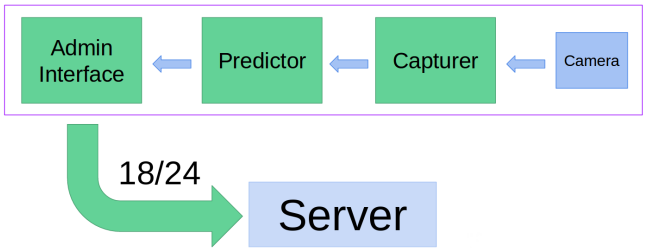

Figure 6: Deeper look at modules our processing unit consists of. The figure shows the flow of data and results (18 out of 24 spaces occupied) which are sent to the server.

To this end the presented version of the PKSpace package utilizes a simple script which decides whether picture is worth saving or not by comparing the amount of black pixels to a predefined threshold. Furthermore, the script is able to perform simple operations with the obtained image, such as translations, which may be required, given the specific requirements of a particular deployment. This script is intended to be executed in specific time intervals using standard time-based job schedulers, such as Cron in Unix-like operating systems.

4.2 Predictor

The predictor uses a trained model to predict whether a given parking space is vacant or occupied. In order to do that, it uses the mask of the parking space to extract respective parking spaces from the big image, according to the specification of the chosen model.

The user can choose either one of our pre-trained models, or retrain the model on new data, which can be created in the Admin Interface discussed in the following section.

As a result of the prediction, each image of a parking space is labeled with either 0 (vacant) or 1 (occupied), and results are sent to Admin Interface where they are summarized and a visualization (Figure 1b) is also created.

4.3 Admin Interface

In the Admin Interface, the user is capable of managing all settings regarding Capturer and Predictor, see a visualization of a predicted image and more. This is the only module the end user is intended to interact with.

In case the user wants to train a model on their own dataset, the Admin Interface provides tools for creating masks and labeling the gathered data. These are described in subsections below.

4.3.1 Mask Creator

Mask Creator is a tool for marking boundaries of parking spaces on the image. Due to various camera positions, the user is able to create multiple masks. The user is provided with a simple interface, which lets him choose an image from previously specified folder and mark regions of all of the parking spaces which will be predicted. Moreover, it is possible to specify the angle of a given parking space with regards to the whole image, which in turn makes sure that all images are rotated in the same way when they are used for training, testing or prediction.

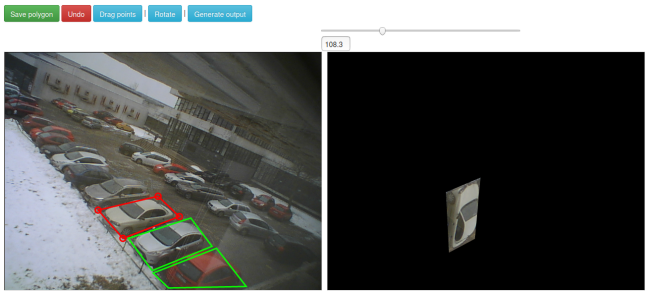

Figure 7: Red polygon represents currently marked parking space. Its mask is shown in canvas on the right side and can be rotated using slider above. When the area and rotation of particular parking space is set user can save it and the color changes to green

We chose well-known JSON format as the output. It contains information about boundaries of each parking space along with its rotation. When the user is done with creating the mask, the Admin Interface will store it in a JSON file which can later be used in Labeler.

Figure 8: Left box shows available masks. After choosing one, parking spaces are displayed on canvas and user can mark those which are being occupied with red color

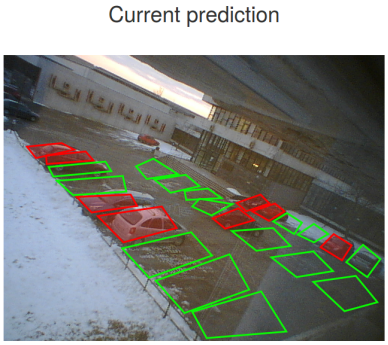

Figure 9: Live prediction

4.3.2 Labeler

For training the model one needs to have a good dataset. It is quite time-consuming to create one, so we decided to build a tool called Labeler to speed up the process.

Labeler loads all the images provided by Capturer. The user is then able to choose one of the masks that were created in Mask Creator, choose the image for labeling and click on those parking spaces which are occupied in the image. Labeler then generates a JSON file for this particular image, with information regarding which mask was used and which parking spaces are occupied. This way user creates a dataset which can be used for training and testing of their own model.

4.3.3 Live feed

The Admin Interface also provides a live feed of the last predicted image with a visualization of vacant and occupied parking spaces. The user is able to see for which parking spaces the mask has to be changed or if it seems that the model needs to be re-trained.

4.4 Sample Implementation

While the architecture of PKSpace is designed for general use on any suitable embedded architecture, in the experiments discussed in this work a specific instance is used: Raspberry PI 2 – a well known single-board computer used mainly in hobby and educational settings, but also for industrial applications – and a PlayStation Eye camera, known for its good performance to cost ratio, when used for Computer Vision-related applications, such as [14]. The hardware components used in this implementation are by far not the best in terms of quality or performance, but represent some of the most inexpensive choices in their respective categories. Since the whole package could be at the time of writing of this document under $50, it presents an affordable option for most use cases.

The Admin Interface is developed using Flask, which is a lightweight Python web framework based on Werkzeug and Jinja 2. Thanks to its minimality user doesn't have to install bunch of packages necessary to run a Flask application. The fron-end functionality is managed by JavaScript with jQuery library and design is done using widely known Bootstrap. Therefore, it is required to have JavaScript enabled in user's browser.

5 Models

Encouraged by recent work in this area, described in more detail in Section 2, especially by the use of Integral Channel Features in combination with an SVM and Logistic Regression classifier [1], which the Ahrnbom et al. suggest are well suited for running on low performance platforms, we sought to create an opensource implementation of this model, while replicating results are provided in the aforementioned paper on the PKLot dataset.

		Our results		Ahrnbom et. al.	
Training	Evaluation	LR	SVM	LR	SVM
UFPR04	UFPR04	0.9925	0.9917	0.9994	0.9996
UFPR04	UFPR05	0.8824	0.9109	0.9928	0.9772
UFPR05	UFPR04	0.8098	0.8410	0.9963	0.9943
UFPR05	UFPR05	0.9658	0.9720	0.9987	0.9988

Table 1: Comparison of results of our implementation of Integral Channel Features and results reported in Ahrnbom et al. [1]

Since we introduce a new FMPH dataset as part of this work, a set of baseline methods along with their performance is also reported to give researchers and practitioners a well defined starting point. As such we chose a kNN, Logistic Regression (LR) and Multi Layer Perceptron (MLP) classifiers, since they are well studied models and often times serve as baselines for new Computer Vision datasets (such as for instance [5]). All of the models received the RGB values of the parking space patch resized to 80x80 as their input. Given our memory constrain and mostly embedded setting, kNN is not of practical use, since it needs to load and keep all training data in memory to make a prediction. For MLP was used the L-BFGS solver, since we do not have large training datasets.

model	accuracy	F1 score	AUC
kNN k=1	0.82253	0.80996	0.76133
kNN k=3	0.82346	0.80962	0.75921
Log. Re.	0.74761	0.73643	0.69163
MLP (15, 15)	**0.88219**	0.88155	0.86550
ICF LR	0.86100	0.86300	0.86950
ICF SVM	0.87680	**0.88690**	**0.87680**

Table 2: Results of baseline models trained and evaluated on the FMPH dataset

kNN parameter for how many neighbours are considered in evaluation were selected by comparison of only F1 scores. For Logistic Regression we set $\lambda_1 = 1$, tolerance for stopping to 0.0001 and used the `liblinear` solver, for we have small datasets. These are sane defaults suggested by a well regarded scikit-learn library [15]. Results can be seen in Table 2.

6 Results

Let us first discuss the comparison between the results obtained by our implementation of Integral Channel Features and the results reported in [1]. As we can see in Table 1, when training and testing on the same dataset subset of the data the results are comparable. They differ to much greater extent when the training is done on a different subset of the data than testing. We suspect that this may be due to different distribution of days that are used in the training and testing sets, which is unfortunately not reported in [1]. Another explanation may be that there is an issue with our implementation, which seems less likely, given quite close results in the first considered case.

In Table 3 we also report results of the cross-validation of the Integral Channel Features method described above on the FMPH dataset and the previously mentioned UFPR subset of the PKLot dataset described in Section 3.1. As we can see, models trained on FMPH dataset did well on UFPR04 and UFPR05 dataset, while models which were trained on the UFPR subsets were not able to generalize well enough to the FMPH dataset. This can be mostly contributed to the fact that the FMPH dataset is more complex and therefore arguably harder to classify correctly, given all the issues mentioned in Section 3.3.

As we can see in Table 2, despite their success on other datasets, the ICF-based models were outperformed by an MLP with a configuration of two hidden layers with 15 neurons on each of them. This parameter was chosen from a pool of architectures with one hidden layer and $l \in \{5, 10, 15, 20, 25\}$ neurons and two hidden layers, where every combination of numbers $l1, l2 \in \{5, 10, 15, 20, 25\}$. The aforementioned combination was chosen because it gave the best performance on the validation set.

Since the ICF-based and MLP models perform well on separate types of datasets, they are both implemented in PKSpace and the user can choose any of them depending on their particular use case.

Training	Evaluation	LR	SVM
FMPH	FMPH	0.8695	0.8768
FMPH	UFPR04	**0.8721**	**0.8793**
FMPH	UFPR05	0.8236	0.7828
UFPR04	FMPH	0.7449	0.7355
UFPR05	FMPH	0.7652	0.7775

Table 3: Results of cross-validation of the Integral Channel Features method on the FMPH dataset and the UFPR subsets of the PKLot dataset. The numbers in the table represent the AUC score.

To summarize, results on conventional datasets (UFPR) were on par with the ones reported in paper from Ahrnbom et al. [1]. On FMPH dataset, while results are worse by considerable amount due to numerous factors discussed in Section 3.3, they are still good enough for our cause.

7 Conclusions and Future Work

In this work we present an opensource solution for the problem of parking space occupancy detection called PKSpace. It utilizes a vision-based approach using a machine learning model to classify images of parking spaces as either occupied or vacant. It further provides its user with the ability to choose either a pre-trained model which is part of the provided solution, or to make their own dataset for a specific parking lot and train the model on it. Furthermore, the user can choose one of many models which were benchmarked as part of this paper.

Since most of the published work in this area base their results on datasets that are of high quality and can be considered "too sterile" to resemble real-world use cases, we introduce a new parking lot dataset called FMPH, which was created using a low cost camera and better approximates the environment an opensource system like this may find itself in. In order to give a good starting point for future research using this dataset, we also provide baseline performance that can be further improved in the future.

In the future we would like to propose models that can achieve better classification results on the dataset we introduced. Moreover, we believe that, given the abundance of data in publicly available datasets and numerous good results when formulated as a classification problem, it may be time to deal with the whole problem more holistically and predicting the number of parking spaces directly from the input image as opposed to from a set of pre-defined parking space patches.

We would like to also extend the Admin Interface by a Model Trainer tool, for simplifying and automating the process of training (or re-training) a model on new data.

An implementation of the PKSpace package is available under the terms of the GNU GPL 3 license from `https:`

//github.com/NaiveNeuron/PKSpace.

References

[1] Martin Ahrnbom, Kalle Astrom, and Mikael Nilsson. Fast classification of empty and occupied parking spaces using integral channel features. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 9–15, 2016.

[2] Giuseppe Amato, Fabio Carrara, Fabrizio Falchi, Claudio Gennaro, Carlo Meghini, and Claudio Vairo. Deep learning for decentralized parking lot occupancy detection. *Expert Systems with Applications*, 72:327–334, 2017.

[3] Ahmed Nabil Belbachir. *Smart cameras*, volume 2. Springer, 2010.

[4] Noah Dan. Parking management system and method, January 31 2002. US Patent App. 10/066,215.

[5] Paulo RL De Almeida, Luiz S Oliveira, Alceu S Britto, Eunelson J Silva, and Alessandro L Koerich. Pklot–a robust dataset for parking lot classification. *Expert Systems with Applications*, 42(11):4937–4949, 2015.

[6] Diana Delibaltov, Wencheng Wu, Robert P Loce, and Edgar A Bernal. Parking lot occupancy determination from lamp-post camera images. In *Intelligent Transportation Systems-(ITSC), 2013 16th International IEEE Conference on*, pages 2387–2392. IEEE, 2013.

[7] Daniele Di Mauro, Sebastiano Battiato, Giuseppe Patanè, Marco Leotta, Daniele Maio, and Giovanni M Farinella. Learning approaches for parking lots classification. In *International Conference on Advanced Concepts for Intelligent Vision Systems*, pages 410–418. Springer, 2016.

[8] Stefan Funck, Nikolaus Mohler, and Wolfgang Oertel. Determining car-park occupancy from single images. In *Intelligent Vehicles Symposium, 2004 IEEE*, pages 325–328. IEEE, 2004.

[9] Thanarat Horprasert, David Harwood, and Larry S Davis. A statistical approach for real-time robust background subtraction and shadow detection. In *IEEE ICCV*, volume 99, pages 1–19, 1999.

[10] Ching-Chun Huang, Yu-Shu Tai, and Sheng-Jyh Wang. Vacant parking space detection based on plane-based bayesian hierarchical framework. *IEEE Trans. Circuits Syst. Video Technol.*, 23(9):1598–1610, 2013.

[11] Ching-Chun Huang and Sheng-Jyh Wang. A hierarchical bayesian generation framework for vacant parking space detection. *IEEE Trans. Circuits Syst. Video Technol.*, 20(12):1770–1785, 2010.

[12] Amin Kianpisheh, Norlia Mustaffa, Pakapan Limtrairut, and Pantea Keikhosrokiani. Smart parking system (sps) architecture using ultrasonic detector. *International Journal of Software Engineering and Its Applications*, 6(3):55–58, 2012.

[13] Elena Màrmol and Xavier Sevillano. Quickspot: a video analytics solution for on-street vacant parking spot detection. *Multimedia Tools and Applications*, 75(24):17711–17743, 2016.

[14] Charles Martin, Benjamin Forster, Hanna Cormick, et al. Cross-artform performance using networked interfaces: Last man to die's vital lmtd. In *NIME*, pages 204–207, 2010.

[15] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[16] RJ López Sastre, P Gil Jimenez, Francisco J Acevedo, and S Maldonado Bascon. Computer algebra algorithms applied to computer vision in a parking management system. In *Industrial Electronics, 2007. ISIE 2007. IEEE International Symposium on*, pages 1675–1680. IEEE, 2007.

[17] Henry Schneiderman and Takeo Kanade. Object detection using the statistics of parts. *International Journal of Computer Vision*, 56(3):151–177, 2004.

[18] Sepehr Valipour, Mennatullah Siam, Eleni Stroulia, and Martin Jagersand. Parking stall vacancy indicator system based on deep convolutional neural networks. *arXiv preprint arXiv:1606.09367*, 2016.

[19] Paul Viola and Michael J Jones. Robust real-time face detection. *International journal of computer vision*, 57(2):137–154, 2004.

[20] Joerg Wolff, Thomas Heuer, Haibin Gao, Michael Weinmann, Stefan Voit, and Uwe Hartmann. Parking monitor system based on magnetic field senso. In *Intelligent Transportation Systems Conference, 2006. ITSC'06. IEEE*, pages 1275–1279. IEEE, 2006.

[21] Qi Wu, Chingchun Huang, Shih-yu Wang, Wei-chen Chiu, and Tsuhan Chen. Robust parking space detection considering inter-space correlation. In *Multimedia and Expo, 2007 IEEE International Conference on*, pages 659–662. IEEE, 2007.