Car Parking Occupancy Detection Using Smart Camera Networks and Deep Learning

Giuseppe Amato*, Fabio Carrara*, Fabrizio Falchi*, Claudio Gennaro*, Carlo Meghini* and Claudio Vairo*
*ISTI-CNR, via G. Moruzzi 1, Pisa, Italy
Email: {giuseppe.amato, fabio.carrara, fabrizio.falchi, claudio.gennaro, carlo.meghini, claudio.vairo}@isti.cnr.it

***Abstract*—This paper presents an approach for real-time car parking occupancy detection that uses a Convolutional Neural Network (CNN) classifier running on-board of a smart camera with limited resources. Experiments show that our technique is very effective and robust to light condition changes, presence of shadows, and partial occlusions. The detection is reliable, even when tests are performed using images captured from a viewpoint different than the viewpoint used for training. In addition, it also demonstrates its robustness when training and tests are executed on different parking lots. We have tested and compared our solution against state of the art techniques, using a reference benchmark for parking occupancy detection. We have also produced and made publicly available an additional dataset that contains images of the parking lot taken from different viewpoints and in different days with different light conditions. The dataset captures occlusion and shadows that might disturb the classification of the parking spaces status.**

***Index Terms*—Machine Learning, Classification, Deep Learning, Convolutional Neural Networks**

I. Introduction

Techniques for car parking occupancy detection are of great importance for an effective management of car parking lots. Knowing in real-time the availability of free parking spaces and communicating to the users can be of great help in reducing the queues, improve scalability, and the time required to find an empty space in a parking lot.

In many parking lots, ground sensors are used to determine the status of the various spaces. This requires installation and maintenance of sensors in every parking space, which might be expensive, especially in parking lots with high number of spaces available.

Recently, various techniques relying on the use of video cameras have been proposed to monitor the occupancy of parking lots [1], [2], [3], [4]. However, despite these fine efforts, vacant parking space detection using only visual information is still an open problem. Most of these approaches rely on specialized visual techniques tailored to the specific scenario, and lack of generalization when applied to different parking lots. In this paper, we provide a distributed, effective, efficient and scalable solution for real-time parking occupancy detection, based on deep Convolutional Neural Networks (CNN).

The proposed solution for parking detection follows the trend of applying Deep Learning techniques [5], specifically CNNs, to problems that require a high-level of abstraction to be solved, such as vision.

Thanks to deep CNNs, the proposed solution is robust to disturbances due to partial occlusions, presence of shadows, variation of light conditions, and exhibits a good generalization property. In fact, the quality of the results are maintained when we considered parking lots and scenarios significantly different from the ones used during the training phase. Moreover, the classification phase needs low computational resources, making it suitable for embedded environments such as Raspberry Pi. In the context of vehicle detection, the only work of which we are aware of using CNN is the one presented in [6], which uses a multi-scale CNN to detect vehicles in high-resolution satellite images.

To validate our approach we performed tests using *PKLot* [4], a dataset for parking lot occupancy detection. Besides, we built an additional dataset, called *CNRPark*, using images coming from smart cameras placed in two different places, with different point of views and different perspectives of the parking lot of the research area of the National Research Council (CNR) in Pisa.

The *CNRPark* dataset contains images taken in different days, with different light conditions, and includes occlusion and shadow situations that make the occupancy detection task more difficult. The proposed dataset has been manually and exhaustively annotated, and made available to scientific community for defining new algorithms for car park occupancy detection. More details about the *CNRPark* dataset will be given in Section III-A.

The usage of these two datasets, coming from two different parking lots and scenarios, allowed us to test the generalization property of our approach, by training on one scenario and testing in a completely different one. To the best of our knowledge, there are no other experiments where this type of generalization property has been tested. We compared our approach against the state of the art methods discussed in [4].

The paper is organized as follows. Section II discusses the proposed approach and gives and overview of the overall system. Section III discusses the experiments and the obtained results. Finally Section IV concludes the paper.

II. Deep Learning for occupancy detection

In this paper, we propose an approach for the detection of car parking occupancy based on *Deep Learning*. Deep Learning (DL) [5] is a branch of Artificial Intelligence that aims at developing techniques that allow computers to learn complex perception tasks, such as seeing and hearing, at human level of accuracy. It provides near-human level accuracy in image classification, object detection, speech recognition, natural language processing, vehicle and pedestrian detection, and more.

A Deep Learning approach particularly effective for vision tasks exploits *Convolutional Neural Networks* (CNN) [7], [8], [9]. A CNN is composed of a possibly large number of hidden layers, each of which performs mathematical computations on the input and produces an output that is given in input to the following layer. A CNN differs from classical neural networks for the presence of convolutional layers, that are able to model and discern the spatial correlation of neighboring pixels better than normal fully connected layers. For a classification problem, the final outputs of the CNN are the classes for which the network has been trained on. On one hand, the training phase is usually computationally expensive and it may take a long time. On the other hand, once the network has been trained, the prediction phase is quite fast and efficient.

We trained two different CNNs and we used the trained neural networks to decide about the occupancy status of the individual parking spaces seen by the video cameras.

In order to validate our approach, we performed experiments on both *CNRPark* and the *PKLot* datasets, which are discussed in more details in Section III-A.

Our solution is intended to run directly on smart cameras, i.e. cameras provided with computational capabilities. In our experiments we used Raspberry Pi 2 model B[1], equipped with the standard Raspberry Pi camera module[2] and mounted in an outdoor camera box installed on the roof of the building in front of the parking lot. We call these devices *Raspberry smart cameras*.

Due to the angle of view and the perspective of the camera module used, most of the parking spaces closest to the building are monitored by just one camera, while the parking spaces farthest from the building are monitored by more cameras. We use this redundancy to solve some obstacle problems (for examples trees) by selecting the confidence value of the camera that has the clearest view of that parking space.

Using smart cameras, rather than ground sensors, has two relevant advantages: lower cost and versatility. The cost of a Raspberry Pi equipped with a camera module is about 80€, and the outdoor camera box with pole support has about the same price. One single camera placed on the roof of a 3-floors building, as in our case, can monitor, on average, about 50 parking spaces located in front of the building, thus reducing the cost per single space to a few Euros. The cost of using one ground sensor, per parking space, is one order of magnitude higher. As will be shown in Section III, the accuracy of the proposed approach is high, and it is comparable to the accuracy of the ground sensors. Therefore with a limited expense, compared to installing a single ground sensor for each parking space, it is possible to monitor a large parking lot with comparable accuracy results.

In addition, by using cameras, we are not limited to parking monitoring applications. We could also exploit the cameras to perform video surveillance activities, like face/people recognition or video tracking of people and moving vehicles. Moreover, if a large database to store all the processed images is available, it is possible to reconstruct past events at a given time and in a given area of the parking lot, by submitting a query to the database. A key aspect of the proposed solution is its decentralized strategy and the delegation of the parking decision to the smart cameras themselves. This solution has the clear advantage of being scalable, as it requires no additional elaboration on the server side. In a centralized solution, images of the parking lot acquired at high resolution (of about 3MB each) should be sent to the server which would thus become a bottleneck and a single point of failure.

The size of our software is 3.1MB. On the Raspberry Pi 2, the load is around 0.25, using one of the four cores of the CPU, and about 22% of the RAM memory. It takes about 15 seconds to execute an iteration of classification of 50 parking spaces, and to send the classification result to the web server for visualization. We prepared a website[3] to access the live view of all the cameras. By clicking on each image thumbnail, a larger view of that camera is shown with the real time parking spaces occupancy status highlighted. We also prepared a time-lapse of the previuos day for each camera, to provide a quick overview of how our approach works.

In the following, we describe the algorithm for determining the parking space status, which rely on the use of CNN and we give a brief description of the CNNs that we used in our solution.

A. Parking Space Status Classification

The system that we built periodically captures the image of a portion of the parking lot and, for each parking space, determines the occupancy status by using the trained CNN. Pictures captured by cameras are filtered by a mask that identifies the various parking spaces. The mask was built once for all manually. Examples of masks built for two different cameras are shown in Figure 1. At run time, each frame is automatically segmented in the patches (the portion of the original image containing a single parking space) corresponding to the parking spaces monitored by that camera, by using the generated masks. Every patch is then classified using the

[1]https://www.raspberrypi.org/products/raspberry-pi-2-model-b
[2]https://www.raspberrypi.org/documentation/hardware/camera.md

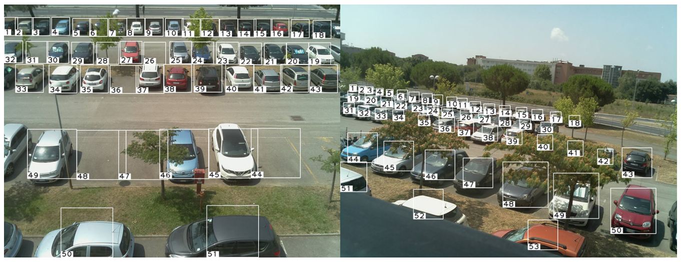


Fig. 1. *Parking space patches are segmented and numbered as shown in the two images, respectively forming subset A (on the left) and B (on the right).*

[3]http://claudiotest.isti.cnr.it/telecamere.html

net	*conv1*	*conv2*	*conv3*	*fc4*	*fc5*
mLeNet	30x11x11+4 pool 5x5+5 -	20x5x5+1 pool 2x2+2 -	-	100 ReLU	2 soft-max
mAlexNet	16x11x11+4 pool 3x3+2 LRN, ReLU	20x5x5+1 pool 3x3+2 LRN, ReLU	30x3x3+1 pool 3x3+2 ReLU	48 ReLU	2 soft-max

TABLE I

CNN Architectures: for conv1-3 *layers, each cell specifies: the number and the size of filters as "$num \times width \times height + stride$", the max-pooling operation applied as "$width \times height + stride$", and whether Local Response Normalization (LNR) and/or Rectified Linear Unit (ReLU) activation are applied. For fully connected layers, we report their dimensionality. A 2-way softmax follows the last fully connected layer.*

trained CNN, to decide if the corresponding parking space is empty or busy.

B. Convolutional Neural Networks

We tested two CNN architectures, named *mAlexNet* and *mLeNet* ('m' stands for 'mini'), respectively based on [7] and [10], which are known to be successful in visual recognition tasks. These networks have less than five trainable layers in order to make the computations feasible in real-time on the embedded device. This limitation is reasonable since the original architectures are designed for large scale visual recognition tasks, which are way more complex than our binary classification problem. In fact, we observed throught experiments that the reduced architectures can cope easly with the car parking occupancy detection problem.

Both architectures take a 224x224 RGB image as input, therefore each patch is normalized accordingly before the training.

mLeNet is based on LeNet-5, proposed by [10], having two convolutional layers followed by max pooling and two fully connected layers. The first layer (*conv1*) is similar to the one proposed by [7] to be more suitable for a 224x224x3 input image. Layers *conv2* and *fc4* have a reduced number of filters and neurons with respect to [10] to better suit a binary classification task without overfitting. For *fc5*, the last Gaussian RBF (radial basis function) layer is replaced with a classical inner product layer acting like a 2-way soft max classifier.

mAlexNet is based on AlexNet, proposed in [7], where we dropped the third and fourth convolutional layers and a fully connected layer. All convolutional layers *conv1-3* are followed by max pooling, local response normalization (LRN), and linear rectification (ReLU), except for *conv3* where no LRN takes place. The number of filters of *conv1-3* and the number of neurons in *fc4* are drastically reduced to fit the problem dimension. For *fc4-5*, no dropout regulation is adopted.

Unlike [7] and [10], both architectures have a dense connection between convolutional layers. The details of both architectures are reported in Table I.

III. Evaluation

In this section, we present the methodology and the results of the experimental evaluation. We first assessed the two proposed CNN architectures to select the one that offers the best performance. Afterwards, we compared our approach against other state-of-the-art methods.

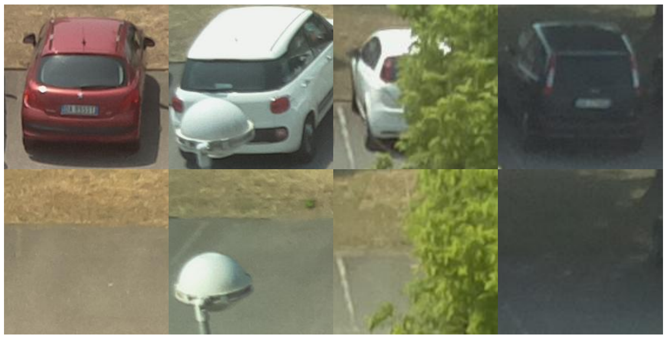


Fig. 2. *Training set patches segmented from the camera view. Images show four parking spaces in both status: busy (first row) and free (second row). They also present some occlusion and shadow situations that we faced.*

A. Datasets used for validation

We used the *PKLot* and *CNRPark* datasets for validation.

PKLot [4], consists of ~ 700.000 images of parking spaces coming from multiple parking lots organized by weather condition (Sunny, Cloudy, Rainy) and by day of capture.

CNRPark is a benchmark we built to test our approach. It was obtained by about 250 images of the parking lot collected in different days, from 2 distinct cameras, which were placed to have different perspectives and angles of view, various light conditions and several occlusion patterns. We built a mask for each parking space in order to segment the original screenshots in several patches, one for each parking space (see Figure 2 for some examples). Each of these patches is a square of size proportional to the distance from the camera, the nearest are bigger then the farthest. We then labelled all the patches according to the occupancy status of the corresponding parking space.

CNRPark is composed of 12,584 labelled patches and it is available for download[4]. It captures different situations of light conditions and it includes partial occlusions due to obstacles (lampposts, trees or other cars) and partial or global shadowed cars (see again Figure 2). This allows to train a classifier that is able to distinguish most of the situations that can be found in a real scenario.

The main differences between *CNRPark* and *PKLot* are the following:

a) *PKLot* is larger than *CNRPark* (695.899 vs 12.584 total images) and contains images spanning across months with different weather conditions;

b) in *CNRPark* parking spaces masks are non-rotated squares and often images do not cover precisely or entirely the parking space volume, whereas in *PKLot* images are extracted using rotated rectangular masks and subsequently straighten, resulting in a more precise coverage of the parking space;

[4]http://claudiotest.isti.cnr.it/park-datasets/CNRPark/

c) *CNRPark* is composed also by heavily occluded spaces (almost entirely covered by trees and lampposts) which are not included in the set of segmented spaces of *PKLot*; moreover images are taken from lower point of views with respect to *PKLot*, resulting in more occlusions due to adjacent vehicles.

These aspects makes the classification of *PKLot* an easier challenge with respect to *CNRPark*, which shows higher variability between images.

The usage of two completely different datasets allowed us to validate and compare the proposed approach both performing training and test on the same dataset, and training on one dataset while testing on the other one.

B. Assessment of the Proposed CNNs

In order to assess the performance of the two CNN architectures *mAlexNet* and *mLeNet*, presented in Section II-B, we performed experiments using the *CNRPark* dataset, considering two possible application scenarios: a *single camera scenario*, in which train and test data come from the same viewpoint, and a *multiple camera scenario*, in which training data and test data come from different viewpoints.

Experiments are performed offline using the manually labelled data generated as described in Subsection III-A. Images are divided in two subsets, *CNRPark A* and *CNRPark B*, containing images respectively taken from a camera with central view and a camera with a side view of the parking lots (Figure 1). This produces roughly a 50%-50% split. Details are reported in Table II. All patches are shuffled and resized to a fixed size of 256x256 pixels. No information about the previous classification of a space is used to classify the same or other spaces, hence each image is classified independently from each other. For both the CNNs (mLeNet and mAlexNet), we perform a learning phase and a classification phase.

For the single camera scenario, in order to limit problems of overfitting, we further divided the two subsets (*CNRPark A* and *CNRPark B*) considering independently odd numbered and even numbered spaces, which maintains a 50%-50% split proportion. Specifically, when we train on odd numbered we test on even numbered, and viceversa, for a total of eight experiments. This scenario allowed us to test the robustness of the proposed solution to possible changes that may occur during outdoor monitoring with fixed cameras, such as illumination changes, shadows and partial occlusions.

For the multi camera scenario, we train both networks on an entire subset and then we test them on the other subset, for a total of four experiments. We tested the robustness of the proposed solution to viewpoint variations, allowing us to measure the ability of the solution to transfer the learned knowledge to a new unseen scenario.

Training images are randomly cropped to 224x224 and randomly flipped horizontally, while for test images the central 224x224 crop is taken. We train our CNN models using the Caffe framework [11] with gradient descend with momentum. The following hyper-parameters are used: momentum 0.9; weight decay $5 \cdot 10^{-4}$. The initial learning rates are chosen independently for each experiment and they are reported in Table III (*base lr* column). The learning rate is decreased two times by a factor 10 when the loss stabilizes, or after at most 10 epochs, resulting in at most 30 epochs of training. The training phase took at most 15 minutes on a NVIDIA GTX 980 for each single experiment. Pretrained models are available for download.[5]

Table III reports the accuracy on the test sets for each configuration obtained by the models at the end of the learning phase.

Both CNNs, when tested in the single camera scenario, perform better on the subset *CNRPark A*, which contains less occlusions and in which there are less variations between parking spaces. In fact, *mAlexNet* reaches an accuracy of 0.996 in *CNRPark A*. However, very good results are also obtained on subset *CNRPark B*, which poses additional challenges due to the very skewed viewpoint, the higher number of obstacles in the field of view, and the imbalance of the set.

subset	free	busy	total
CNRPark *A*	2549	3622	6171
CNRPark *B*	1632	4781	6413
CNRPark	4181	8403	12584
CNRParkOdd	2201	3970	6171
CNRParkEven	1980	4433	6413
CNRPark	4181	8403	12584
PKLot2Days	27314	41744	69058
PKLotNot2Days	310466	316375	626841
PKLot	337780	358119	695899

TABLE II
Details of datasets and image subsets used in the experiments.

SINGLE CAMERA EXPERIMENTS

train	*test*	*net*	*base lr*	*accuracy*
A (even)	A (odd)	mLeNet	0.001	0.993
		mAlexNet	0.01	0.996
A (odd)	A (even)	mLeNet	0.001	0.982
		mAlexNet	0.005	0.993
B (even)	B (odd)	mLeNet	0.001	0.861
		mAlexNet	0.01	0.911
B (odd)	B (even)	mLeNet	0.001	0.893
		mAlexNet	0.005	0.898

MULTI CAMERA EXPERIMENTS

train	*test*	*net*	*base lr*	*accuracy*
A	B	mLeNet	0.0001	0.843
		mAlexNet	0.001	0.863
B	A	mLeNet	0.001	0.842
		mAlexNet	0.0005	0.907

TABLE III
Settings and results of experiments performed on subsets A and B of CNRPark. *The even/odd indication tells whether training or testing is performed only on images of even or odd numbered spaces of that particular subset.*

[5]http://claudiotest.isti.cnr.it/park-datasets/CNRPark/models/

method		*input dims*	*input description*
mAlexNet		224x224x3	a 224x224 crop of the original image
SVM	LBP	256	classical LBP [12]
	LBPu	59	uniform LBP [12]
	LBPri	36	rotational invariant LBP [12]
	LBPuri	10	uniform and rotational invariant LBP [12]
	LPQu	256	LPQ (uniform init.) [13]
	LPQg	256	LPQ (gaussian init.) [14]
	LPQgd	256	LPQ (gaussian derivative init.) [14]

TABLE IV
Summary of the compared methods.

	INTRA-DATASET			**INTER-DATASET**	
train	PKLot-2D	CNRP.-Odd	CNRP.-Even	PKLot-2D	CNRP.
test	PKLot-N2D	CNRP.-Even	CNRP.-Odd	CNRP.	PKLot
model					
mAlex	0.981	0.901	0.907	0.829	0.904
LPQu	0.966	0.869	0.815	0.646	0.398
LPQgd	0.970	0.877	0.813	0.617	0.410
LPQg	0.957	0.869	0.816	0.638	0.438
LBPuri	0.874	0.850	0.763	0.653	0.497
LBPu	0.951	0.868	0.800	0.643	0.467
LBPri	0.879	0.865	0.820	0.642	0.487
LBP	0.945	0.874	0.872	0.631	0.529

TABLE V
Settings and results (accuracy) of intra- *and* inter-dataset *experiments.*

mAlexNet reaches an accuracy of 0.911. Factors like the heigth of the camera, its angle of view, and its distance from the parking spaces also marginally affect the accuracy. However, our approach maintains good results also in these conditions.

For the multi camera scenario, higher values of accuracy are obtained when training on the more complex subset *CNRPark B*, from which the model can extract richer information and better generalize. In this case, *mAlexNet* reaches an accuracy of 0.907.

In all configurations *mAlexNet* offers the best performance, thanks to the use of a larger model that always boosts accuracy, although more effort is required to train it.

C. Comparisons with State of the Art

We have compared our most promising CNN architecture, which is *mAlexNet*, against the method proposed in [4]. The classification techniques proposed in [4] rely on RBF kernel SVMs trained on histograms of textural features. The authors specifically use LBP, LPQ and their variations ([12], [13], [14]). We performed two types of comparative experiments using both *CNRPark* and *PKLot* datasets: *intra-dataset* experiments and *inter-dataset* experiments.

With *intra-dataset* experiments, we evaluated both techniques using, individually, one of the two dataset for both training and testing purposes. In this way, we estimated the performance of both techniques when the statistics of both train and test sets are similar.

With *inter-dataset* experiments, we used one of the two datasets to train and fine-tune each method, and we used the other dataset to test them. Hence, we estimated the ability of both techniques to generalize from the particular statistics of the training dataset.

A summary of the compared methods is reported in Table IV. Details of the parameter settings for the methods in [4] are as follows. All LBP-type features are extracted with radius 1 and 8 neighbors. LPQ features are extracted using window size of 3x3. The SVMs are trained following the methodology reported in [4]. In details, the C and γ parameters of the SVM are chosen using a grid search and a 5-fold cross-validation on the training set. We choose the parameters setting (C, γ) that gives the best 5-fold accuracy, that is the best accuracy obtained classifying each fold of the dataset with the model trained on the other folds. The final SVM is then trained on the entire training set using the chosen parameters, and tested on the testing set. To obtain a probabilistic score from the output of the SVM, we used the same strategy used in [4], which evaluates the posterior probability using a sigmoid function with two parameters [15].

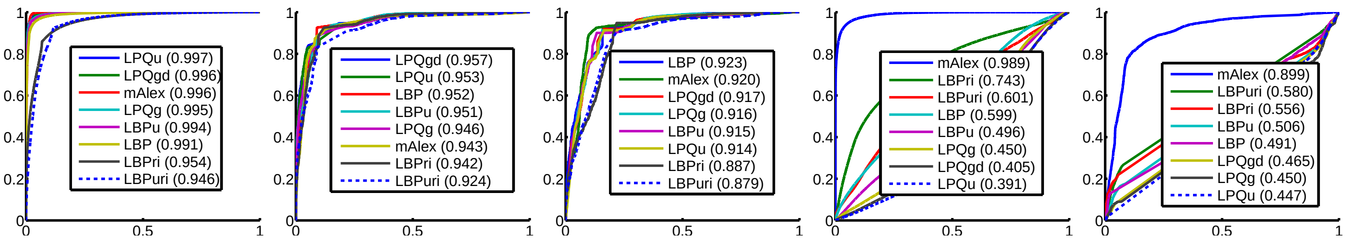
For *intra-dataset* experiments, both datasets have been divided in two partitions. *CNRPark* has been divided in *CNRParkEven*, containing images of even-numbered spaces, and *CNRParkOdd*, containing images of odd-numbered spaces. Both partitions have been used as train set and test set. *PKLot* dataset has been divided in *PKLot2Days* and *PKLotNot2Days*. The former is formed choosing for each parking lot and for each weather condition the images of the first two days in chronological order, and the latter contains the remaining images. This partition strategy has been adopted to reduce training time, since only the smaller partition (*PKLot2Days*) is used as train set.

For *inter-dataset* experiments, we trained on *PKLot2Days* instead of using the whole *PKLot* to reduce training times. Tests are however performed on the whole dataset.

All the details of the datasets are reported in Table II, and results of performed experiments are summarized in Table V. For the comparisons, we evaluated the performance of the trained classifiers on the test sets measuring the accuracy and the Area Under the Curve (AUC) of Receiver Operating Characteristic (ROC) curves. ROC curves show how True Positive Rate (TPR), on y-axis, and False Positive Rate (FPR), on x-axis, vary as its score threshold is varied. AUC measures how much a curve leans near the perfect classification point, that is the point (0,1) on the ROC plot. AUC values range from 0 (perfect misclassification) to 1 (perfect classification), where 0.5 indicates a classifier that performs like the random guessing classifier. In Table V, we report the achieved accuracies, and in Figure 3 the ROC curves are show together with the achieved AUC values for each single experiment.

In the *intra-dataset* experiments, our method is comparable with the ones proposed in [4] in terms of AUC. In fact, our *mAlexNet* method reaches AUCs of 0.943 on *CNRParkEven*, 0.920 on *CNRParkOdd*, and 0.996 on *PKLotNot2Days*, which are very close to respectively 0.957, 0.923, and 0.997 of the best performing compared methods, as can be seen in Figure 3.

In terms of accuracy measured when using a threshold of 0.5, the *mAlexNet* achieves slightly higher accuracy values. In fact, we reach and accuracy of 0.901 on *CNRParkEven*, 0.907



(a) TRAIN: PKLot2Days TEST: PKLotNot2Days
(b) TRAIN: CNRParkOdd TEST: CNRParkEven
(c) TRAIN: CNRParkEven TEST: CNRParkOdd
(d) TRAIN: CNRPark TEST: PKLot
(e) TRAIN: PKLot TEST: CNRPark

Fig. 3. *ROC curves for intra-dataset (a-c) and inter-dataset (d-e) experiments, where the positive class is 'busy'. Methods in the legend are sorted by descending values of AUC.*

on *CNRParkOdd*, and 0.981 on *PKLotNot2Days*, which are higher than respectively 0.877, 0.872, and 0.970 of the best performing compared methods, as can be seen in Table V.

In the *inter-dataset* experiments, our method demonstrates a higher level of generalization, outperforming the other tested methods. In fact, the *mAlexNet* method reaches AUCs of 0.899 on *CNRPark*, and 0.989 on *PKLot*, which are definitively better than respectively 0.580, and 0.743, of the best performing compared methods, as can be seen still in Figure 3.

In terms of accuracy, the *mAlexNet* achieves significantly higher accuracy values. In fact, as can be seen in Table V, we reach and accuracy of 0.829 on *CNRPark*, and 0.904 on *PKLot*, versus respectively 0.646, and 0.529, of the best performing compared methods.

IV. Conclusions

In this paper, we proposed and evaluated an approach for parking status detection, which uses Convolutional Neural Networks to classify the parking space occupancy directly on board of a Raspberry Pi camera. Each camera is able to cover a large portion of a parking lot by monitoring simultaneously about 50 parking spaces.

In order to validate our approach, we have created a dataset, *CNRPark*, containing images of a real parking lot taken by smart cameras. The dataset has been made available to the scientific community to develop and compare new techniques of car parking occupancy detection. Moreover, we compared the presented approach with other state-of-the-art approaches based on non-deep (shallow) machine learning, and tested on the *PKLot* dataset, a very large publicly available dataset of parking lot images.

Our experiments show that convolutional neural networks are effective to be applied to the addressed problem. Specifically, they exhibit very high accuracy, even in presence of noise due to light conditions variation, shadows, and partial occlusions. We have also tested our approach using a test set produced with a camera placed in a different place (viewpoint and perspective), with respect to the training set. Results also show that CNNs have good generalization capabilities in predicting parking status when tested on a dataset completely different from the training dataset. The proposed method is also robust to non-standard parking behaviours, such as cars occupying multiple parking lots. In that case, our solution tends to classify both slots as busy. However, an in-depth analysis of the performance in non-ideal situations, such as night, foggy, or snowy conditions, is left to future work.

Acknowledgment

This work has been partially funded by the DIITET Department of CNR, in the framework of the "Renewable Energy and ICT for Sustainability Energy" project.

References

[1] N. Dan, "Parking management system and method," Jan. 2002, uS Patent App. 10/066,215.
[2] Q. Wu, C. Huang, S.-y. Wang, W.-c. Chiu, and T. Chen, "Robust parking space detection considering inter-space correlation," in *Multimedia and Expo, IEEE International Conference on*. IEEE, 2007, pp. 659–662.
[3] C. G. del Postigo, J. Torres, and J. M. Menéndez, "Vacant parking area estimation through background subtraction and transience map analysis," *IET Intelligent Transport Systems*, 2015.
[4] P. R. de Almeida, L. S. Oliveira, A. S. Britto, E. J. Silva, and A. L. Koerich, "Pklot–a robust dataset for parking lot classification," *Expert Systems with Applications*, vol. 42, no. 11, pp. 4937–4949, 2015.
[5] Y. Bengio, "Learning deep architectures for AI," *Foundations and trends® in Machine Learning*, vol. 2, no. 1, pp. 1–127, 2009.
[6] X. Chen, S. Xiang, C.-L. Liu, and C.-H. Pan, "Vehicle detection in satellite images by hybrid deep convolutional neural networks," *Geoscience and Remote Sensing Letters, IEEE*, vol. 11, no. 10, pp. 1797–1801, 2014.
[7] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.
[8] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.
[9] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Computer Vision and Pattern Recognition*, 2014.
[10] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.
[11] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, "Caffe: Convolutional architecture for fast feature embedding," *arXiv preprint arXiv:1408.5093*, 2014.
[12] T. Ojala, M. Pietikäinen, and T. Mäenpää, "Multiresolution gray-scale and rotation invariant texture classification with local binary patterns," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 24, no. 7, pp. 971–987, 2002.
[13] V. Ojansivu and J. Heikkilä, "Blur insensitive texture classification using local phase quantization," in *Image and signal processing*. Springer, 2008, pp. 236–243.
[14] E. Rahtu, J. Heikkilä, V. Ojansivu, and T. Ahonen, "Local phase quantization for blur-insensitive image analysis," *Image and Vision Computing*, vol. 30, no. 8, pp. 501–512, 2012.
[15] J. Platt *et al.*, "Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods," *Advances in large margin classifiers*, vol. 10, no. 3, pp. 61–74, 1999.