Expert Systems With Applications 72 (2017) 327–334

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

Deep learning for decentralized parking lot occupancy detection

Giuseppe Amato, Fabio Carrara, Fabrizio Falchi, Claudio Gennaro, Carlo Meghini, Claudio Vairo*

Institute of Information Science and Technologies of the National Research Council of Italy (ISTI-CNR), via G. Moruzzi 1, 56124 Pisa, Italy

ARTICLE INFO

Article history:
Received 14 July 2016
Revised 21 September 2016
Accepted 27 October 2016
Available online 29 October 2016

Keywords:
Machine learning
Classification
Deep learning
Convolutional neural networks
Parking space dataset

ABSTRACT

A smart camera is a vision system capable of extracting application-specific information from the captured images. The paper proposes a decentralized and efficient solution for visual parking lot occupancy detection based on a deep Convolutional Neural Network (CNN) specifically designed for smart cameras. This solution is compared with state-of-the-art approaches using two visual datasets: PKLot, already existing in literature, and CNRPark-EXT. The former is an existing dataset, that allowed us to exhaustively compare with previous works. The latter dataset has been created in the context of this research, accumulating data across various seasons of the year, to test our approach in particularly challenging situations, exhibiting occlusions, and diverse and difficult viewpoints. This dataset is public available to the scientific community and is another contribution of our research. Our experiments show that our solution outperforms and generalizes the best performing approaches on both datasets. The performance of our proposed CNN architecture on the parking lot occupancy detection task, is comparable to the well-known AlexNet, which is three orders of magnitude larger.

1. Introduction

A smart camera is a vision system that has an image capture circuitry and enough computing power to process and extract application-specific information from the captured images. Smart cameras are also able to generate event descriptions or make decisions that are used in intelligent and automated systems (Belbachir, 2010).

Recently there has been a growing interest in developing smart camera solutions able to detect parking lot occupancy. The approach that we propose performs this task in real-time directly on smart cameras, without using a central server. It is a decentralized, effective, efficient, and scalable approach, based on deep learning techniques (Bengio, 2009). It relies on a deep Convolutional Neural Network (CNN) specifically designed to be executed on smart cameras.

The clear advantages of the decentralization are the reduction of the communication overhead and the elimination of computing bottleneck. As a consequence, the system scales better when the number of monitored parking spaces increases.

We believe that the proposed approach is also advantageous with respect to those using ground sensors (*e.g.* magnetic sensors) placed on every parking space. Indeed, a single smart camera can simultaneously monitor several parking lots at a cost that is significantly lower than the cost required to install and maintain sensors in every parking lot.

The usage of video to monitor occupancy of parking lots is not new, see for instance (de Almeida, Oliveira, Britto, Silva, & Koerich, 2015; Dan, 2002; del Postigo, Torres, & Menéndez, 2015; Wu, Huang, Wang, Chiu, & Chen, 2007). However, vacant parking space detection *using only visual information* is still an open problem. Many techniques using video cameras are tailored and fine-tuned to specific contexts and scenarios. However, these techniques cannnot be easily generalized, and even the adaptation of one solution to a different parking lot is not easy.

Thanks to the use of deep CNN, the proposed solution is robust to disturbances created by partial occlusions, by the presence of shadows and by the variation of light conditions. Moreover, it exhibits a good generalization property: in fact, the quality of the results is maintained when we consider parking lots and scenarios significantly different from the ones used during the CNN training phase. Furthermore, the classification phase needs fewer computational resources than the training phase, making it possible to run it on distributed, embedded, and low computing-power frameworks.

* Corresponding author.
E-mail addresses: giuseppe.amato@isti.cnr.it (G. Amato), fabio.carrara@isti.cnr.it (F. Carrara), fabrizio.falchi@isti.cnr.it (F. Falchi), claudio.gennaro@isti.cnr.it (C. Gennaro), carlo.meghini@isti.cnr.it (C. Meghini), claudio.vairo@isti.cnr.it (C. Vairo).

http://dx.doi.org/10.1016/j.eswa.2016.10.055
0957-4174/

To validate our approach, we built a dataset, called *CNRPark-EXT*, collecting images from the parking lots in the experimentation area, which is the campus of the National Research Council (CNR) in Pisa.

The images in the *CNRPark-EXT* dataset are taken by 9 smart cameras with different point of views and different perspectives, in different days with different weather and light conditions, and includes occlusion and shadow situations that make the occupancy detection task more challenging. The dataset has been exhaustively, manually annotated, and is available to the scientific community. More details about the *CNRPark-EXT* dataset will be given in Section 4.

In addition, we tested our method on PKLot, a dataset for parking lot occupancy detection, so as to be able to compare our method against the state-of-the-art methods discussed in de Almeida et al. (2015).

The usage of datasets coming from different parking lots and scenarios allowed us to test the generalization property of our approach. To this end, we trained the CNN on one scenario and tested it 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.

The paper is organized as follows. Section 2 introduces other works related to our proposal. Section 3 describes the convolutional neural network implied in the classification process. Section 4 presents the datasets used to evaluate and compare our approach. Section 5 discusses the experiments and the obtained results. Section 6 discusses how the framework was deployed in a real scenario and gives an overview of the overall system. Finally, Section 7 concludes the paper.

2. Related work

One of the earliest attempts of using machine learning to approach the problem of parking lot monitoring was due to Dan (2002) who used color vector features on a support vector machine (SVM) classifier to distinguish car regions from space regions inside the parking lot. Wu et al. (2007) tried to overcome the occlusions problem of this approach by classifying the state of three neighboring spaces as a unit and defining the color histogram across three spaces as the feature in their SVM classifier.

To deal with the problem of light changes, Tsai, Hsieh, and Fan (2007) trained a Bayesian classifier to verify the detection of vehicles based on corners, edges, and wavelet features. Huang, Tai, and Wang (2013) used a Bayesian hierarchical framework to build a vacant parking space detection system that operates day and night based on a 3D model for parking spaces. Similarly, the method presented in Delibaltov, Wu, Loce, Bernal et al. (2013) models every parking space as a volume in the 3D space, and thus is able to account for occlusions when estimating the probability of a vehicle being present in a parking space. Jermsurawong, Ahsan, Haidar, Haiwei, and Mavridis (2014) used specially trained customized neural networks to determine occupancy status and parking demand based on visual features extracted from parking spaces. They present robust results for night and day classifiers in a one-day long evaluation based on 126 parking spaces.

A recent work that approaches the problem by machine learning techniques is (de Almeida et al., 2015). The authors use a dataset of roughly 700.000 images of parking spaces coming from three different cameras to train SVM classifiers on multiple textural features, such as LBP, LPQ, and their variations. They also improve the detection performance using ensembles of SVMs, applying simple aggregation functions, such as maximum or average, to the confidence values given by the classifiers.

The work proposed in del Postigo et al. (2015) is based on a temporal analysis of the video frames to detect the occupancy variation of the parking areas. It combines background subtraction using a mixture of Gaussians to detect and track vehicles, and the creation of a transience map to detect the parking and leaving of vehicles. Masmoudi, Wali, Jamoussi, and Alimi (2014) tackles the problem of occlusions between parking spaces, where one or more space of a parking can be hidden by another parked vehicle. To this end, vehicle tracking is performed to detect the events of entering and leaving of a car in a parking space.

In addition to approaches using visual techniques and ground sensors, there are techniques that use sensors installed on cars or carried by car drivers. For instance, (Caicedo, Blazquez, & Miranda, 2012) proposes a framework for predicting parking occupancy by interacting with in-vehicle navigation systems. In Lan and Shih (2014), a crowdsourcing solution, leveraging on sensors in smartphones, was proposed to collect real-time parking availability information.

In the context of vehicle detection, the only work of which we are aware of using CNN is the one presented in Chen, Xiang, Liu, and Pan (2014), which uses a multi-scale CNN to detect vehicles in high-resolution satellite images.

To the best of our knowledge, ours is the first work that employs deep Convolutional Neural Networks in the context of parking lot monitoring.

2.1. Deep learning

Deep Learning (DL) (Bengio, 2009) 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. The traditional approaches to the classification problem use *ad-hoc* functions to extract from an image specific features that are considered to be indicative of certain objects. For example, hard corners and straight edges might be believed to indicate the presence of man-made objects in the scene. The outputs of these feature extraction functions are then given in input to a classification function, which determines whether or not a particular object has been detected in the image. However, this approach leads to weak and false-alarm prone detectors. In addition, it presents the following problems:

- it is hard to think of general, robust, reliable features, which map to specific object types;
- it is a huge task to find the right combination of features for every type of object to classify;
- it is difficult to design functions that are robust to translations, rotations and scaling of objects in the image.

All these problems make developing high accuracy object detectors and classifiers for a broad range of objects very hard .

The Deep Learning approach, on the other hand, exploits a large number of ground-truth labeled data to discover which features and combinations of features are most discriminative for each class of objects to be recognized, and builds a combined feature extraction and classification model. The model thus obtained can be employed not only to classify the specific objects it was trained on, but also to recognize previously unseen objects that are similar to them.

A Deep Learning approach particularly effective for vision tasks exploits *Convolutional Neural Networks* (CNN) (Girshick, Donahue, Darrell, & Malik, 2014; Krizhevsky, Sutskever, & Hinton, 2012; Simonyan & Zisserman, 2014). A CNN is composed of a possibly large number of hidden layers, each of which performs mathematical computations on the input provided by the previous layer and produces an output that is given in input to the following layer. A CNN

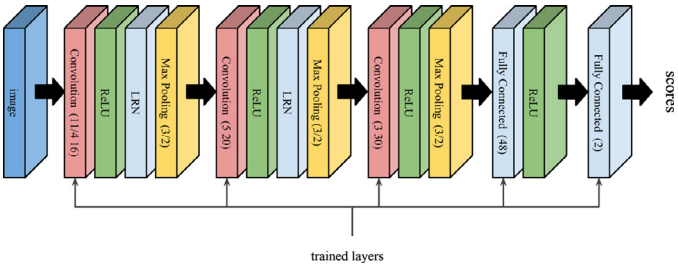


Fig. 1. CNN Architecture: for convolutional layers *conv1-3*, parameters are specified as "*size*/*stride num*". For max-pooling, parameters are specified as "*size*/*stride*". For fully connected layers we report their dimensionality. The last fully connected layer is followed by a 2-way soft-max classifier.

differs from classical neural networks for the presence of convolutional layers, which can better model and discern the spatial correlation of neighboring pixels than normal fully connected layers. For a classification problem, the final outputs of the CNN are the classes which the network has been trained on. The training phase is usually extremely expensive from a computational point of view, and may take a long time to complete. Once the network has been trained and the classifier has been initialized accordingly, the run time phase of prediction is quite fast and efficient.

3. Deep embedded convolutional neural networks for occupancy detection

One of the objectives of our proposal is to run the occupancy detection software entirely on smart cameras, *i.e.* cameras capable of processing the acquired images and of transmitting just the result to a remote server. As a reference, we considered Raspberry Pi 2 model B[1] equipped with the standard Raspberry Pi camera module[2] as smart camera.

A very popular deep convolutional neural network, used as reference in many works, is the so called *AlexNet* (Krizhevsky et al., 2012). The architecture of an AlexNet consists of 60 million parameters and 500,000 neurons. It is organized into five convolutional layers, some followed by max-pooling layers, and two fully connected layers with a 1000-way softmax (more details can be found in Krizhevsky et al. (2012)). Using such an architecture directly on a low-computing power device, poses a very difficult challenge, especially in light of the fact that a single camera might need to monitor several parking places simultaneously and that each parking place needs one independent occupancy detection task to be executed.

In order to make the detection software able to efficiently run directly on the smart cameras, we defined a smaller deep CNN architecture and we compared its performance with respect to the use of the AlexNet architecture. This simplification of the network is also justified by the fact that the original AlexNet architecture was designed for visual recognition tasks much more complex than our binary classification problem. Originally, AlexNet was trained on a one million images dataset to recognize 1000 different classes. In our case, we just have to distinguish two classes. In fact, the experiments reported in Section 5 show that our proposed architecture can cope easily and effectively with the car parking occupancy detection problem.

The deep CNN architecture that we defined is inspired to the AlexNet. We called the new network *mAlexNet*, as *mini* AlexNet. Details of the architecture are reported in Fig. 1. In the mAlexNet, we used three convolutional layers and two fully connected layers, including the output layer. The first and the second convolutional layers (*conv1-2*) are followed by max pooling, local response normalization (LRN), and linear rectification (ReLU). The third convolutional layer (*conv3*) does not use LRN. The number of filters of *conv1-3* and the number of neurons in the fully connected layer (*fc4*) are drastically reduced to fit the problem dimension, obtaining an architecture with roughly $\frac{1}{1340}$ parameters than AlexNet. In *fc4* and *fc5* (the output layer), no dropout regularization is used. The mAlexNet takes a 224 × 224 RGB image, corresponding to a crop representing one single parking space as input. The cropped image might need to be resized if its size is different than what needed.

The mAlexNet has a number of layers that makes the detection task executable in real-time on an embedded device. On average, occupancy detection of 50 parking spaces takes around 15 s on a Raspberry Pi model B.

Our CNN was trained to directly decide about the occupancy status of the individual parking spaces seen by the video cameras. During the training phase, we used random cropping and horizontal flipping techniques of the training images for data augmentation: images are squashed to a resolution of 256 × 256, then are horizontally flipped with a 0.5 probability, and finally a random 224 × 224 crop is taken as input of the neural network. Further details on the training phase are given in Section 5. At prediction time, images are resized to 224 × 224 resolution and no flip takes place.

(a) Overview of CNRPark CAM A (b) Overview of CNRPark CAM B

(c) Overview of CNRPark-EXT CAM 1 (d) Overview of CNRPark-EXT CAM 8

Fig. 2. Parking space patches are segmented and numbered as shown in the images. Top images belong to the small dataset *CNRPark*, while bottom images belong to the full *CNRPark-EXT* dataset.

4. Datasets

A contribution of this paper is also the publication of *CNRPark-EXT*, a dataset of roughly 150.000 labeled images of vacant and occupied parking spaces, built on a parking lot of 164 parking spaces.

CNRPark-EXT includes and significantly extends *CNRPark* (Amato, Carrara, Falchi, Gennaro, & Vairo, 2016), a smaller dataset of roughly 12.000 labeled images, which we also used to perform some of the experiments.

The smaller *CNRPark* dataset contains images of the parking lot collected in different days of July 2015, from 2 distinct cameras *A* and *B*, (see Fig. 2, top row), which were placed in order to have

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

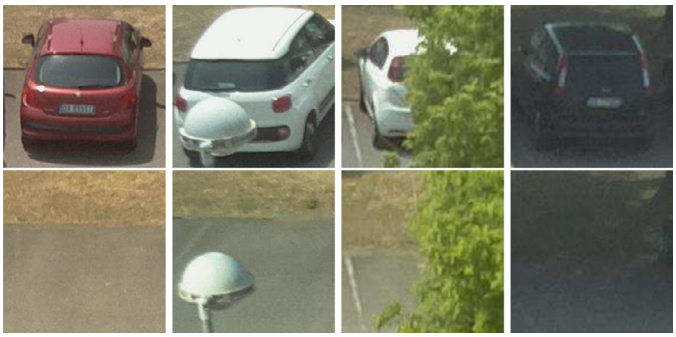


Fig. 3. Training set patches segmented from the camera view. Images show four parking spaces in both status: busy (first row) and free (second row). They also present some occlusion and shadow situations that we faced.

different perspectives and angles of view. The *CNRPark* dataset is also available for downloading.[3]

The full *CNRPark-EXT* extends *CNRPark* with images collected from November 2015 to February 2016 under various weather conditions by 9 cameras (see Fig. 2, bottom row) with different perspectives and angles of view. It captures different situations of light conditions, and it includes partial occlusion patterns due to obstacles (trees, lampposts, other cars) and partial or global shadowed cars (see Fig. 3). This allows training a classifier that is able to distinguish most of the difficult situations that can be found in a real scenario.

We built masks that allow cropping the full pictures taken by the camera in smaller pictures, each containing one single parking space. In the reminder of the paper we refer to such smaller pictures as *patches* (see Fig. 3 for some examples). A patch is a square of size proportional to the distance from the camera, the nearest patches are bigger than the farthest. Finally, we manually labeled all the patches according to the occupancy status of the corresponding parking space, 0 for vacant and 1 for occupied.

Patches of the *CNRPark-EXT* dataset are grouped together into subsets corresponding to different weather conditions (Sunny, Overcast, Rainy), days of capture, and camera IDs. Patches are also grouped into training, validation, and test subsets, to provide a common and objective ground for training and testing classification algorithms. *CNRPark-EXT* is composed of 4287 screen-shots acquired in 23 different days, resulting in a dataset of 144.965 labeled parking space patches.

The *CNRPark-EXT* dataset is made available for downloading.[4] Table 1 reports detailed information about the composition of *CNRPark, CNRPark-EXT*, and *PKLot*(de Almeida et al., 2015). *PKLot* is an additional dataset, existing in literature, which we also used to perform evaluation of the proposed techniques. More information on *PKLot* can be found in Section 5, and discussion on its grouping into subsets can be found in de Almeida et al. (2015).

5. Evaluation

In this section, we present the methodology and the results of the experimental evaluation. We performed different experiments to investigate different aspects of the proposed solution:

1. How does the proposed solution compare against state-of-the-art approaches?
2. How much the generalization performance degrades when using a reduced CNN instead of state-of-the-art ones?
3. How robust the proposed solution is to weather and viewpoint changes?

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

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

Table 1
Details of datasets used in the experiments, with the various proposed subsets. Values refer to the number of patches contained in every dataset or subset.

DATASETS	FREE SPACES	BUSY SPACES	TOTAL
CNRPark	4181	8403	12,584
CNRPark-EXT	65,658	79,307	144,965
PKLot	337,780	358,119	695,899
SUBSETS	FREE SPACES	BUSY SPACES	TOTAL
CNRParkOdd	2201	3970	6171
CNRParkEven	1980	4433	6413
CNRPark-EXT TRAIN	46,877	47,616	94,493
CNRPark-EXT VAL	5232	13,415	18,647
CNRPark-EXT TEST	13,549	18,276	31,825
CNRPark-EXT SUNNY	25,665	37,513	63,178
CNRPark-EXT OVCST	21,067	23,176	44,243
CNRPark-EXT RAINY	18,926	18,618	37,544
CNRPark-EXT *C1*	6407	9308	15,715
CNRPark-EXT *C2*	1454	2641	4095
CNRPark-EXT *C3*	4101	5370	9471
CNRPark-EXT *C4*	7219	9357	16,576
CNRPark-EXT *C5*	9582	11,256	20,838
CNRPark-EXT *C6*	9462	10,646	20,108
CNRPark-EXT *C7*	10,595	10,519	21,114
CNRPark-EXT *C8*	11,237	12,847	24,084
CNRPark-EXT *C9*	5601	7363	12,964
PKLot2Days	27,314	41,744	69,058
PKLotNot2Days	310,466	316,375	626,841
PKLot UFPR04 TRAIN	25,894	23,266	49,160
PKLot UFPR04 TEST	33,824	22,859	56,683
PKLot UFPR05 TRAIN	45,759	48,196	93,955
PKLot UFPR05 TEST	22,600	49,230	71,830
PKLot PUC TRAIN	114,424	106,334	220,758
PKLot PUC TEST	115,616	87,895	203,511
PKLot TRAIN	27,314	41,744	105,843
PKLot VAL	54,909	47,453	165,785
PKLot TEST	275,894	248,583	424,269

We used two datasets in our experiments: *CNRPark-EXT*, the dataset generated by us, and *PKLot* (de Almeida et al., 2015). The two datasets are significantly different. Besides the fact that they contain pictures taken from different parking lots, it is worth highlighting the following differences:

(a) in *CNRPark-EXT* parking spaces masks are non-rotated squares; often images do not cover precisely or entirely the parking space volume, whereas in *PKLot* images are extracted using rotated rectangular masks, which are subsequently straightened, resulting in a more precise coverage of the parking space;
(b) *CNRPark-EXT* is composed also of heavily occluded spaces (almost entirely covered by trees and lampposts) which are not included in the set of segmented spaces of *PKLot*; moreover in *CNRPark-EXT* 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-EXT*, which shows higher variability between images, and include more noisy factors.

The usage of two completely different datasets allowed us to extensively validate and compare the proposed approach. We performed our experiments using several subsets drawn from the datasets, performing training and test on subsets coming from the same dataset, and also training on subsets from one dataset while testing on the other one.

The performed experiments are described in the following subsections, and all the details of the subsets used are summarized in Table 1. All trained models produced are available for download.[5]

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

5.1. Comparisons with the state of the art

We have compared *mAlexNet* against the method proposed in de Almeida et al. (2015), a state-of-the-art approach for car parking occupancy detection that relies on RBF kernel SVMs trained on histograms of textural features. The authors specifically used LBP, LPQ features and their variations (Ojala, Pietikäinen, & Mäenpää, 2002; Ojansivu & Heikkilä, 2008; Rahtu, Heikkilä, Ojansivu, & Ahonen, 2012) as input of the SVM. In their experiment, they show that there is no absolute best among those textural features for this task. However, we noticed that in most of the cases, LPQu and LPQg (Local Phase Quantization with respectively uniform and Gaussian initialization), give better performance. They also tested *ensembles* of classifiers, fusing the confidence values coming from SVMs applied to different selections of the above features. Confidence values were fused using simple aggregation functions, such as *Max* and *Mean*. We noticed in their results that taking the mean of the confidence coming from different classifiers (to which we refer with *Mean Ensemble*) usually improves the performance.

Tests were performed by using both *CNRPark* and *PKLot* datasets. *CNRPark* was split into *even* and *odd* parking spaces (*CNRParkEven* and *CNRParkOdd*). Training was executed on one of the two and testing on the other.

For *PKLot*, we used the same configuration reported in de Almeida et al. (2015), splitting each of the three subsets of *PKLot* (corresponding to different cameras UPFR04, UFPR05, and PUC, see de Almeida et al. (2015) for details) in training and test sets with a 50–50 proportion. We made sure that images captured the same day do not appear simultaneously in the train and in the test sets. We trained our proposed model *mAlexNet* on each of the three training sets individually, and at the end of the training phase, we tested each trained model on all three testing sets.

All the models were trained with gradient descent for at most 18 epochs, with a learning rate of 0.01 halved every 6 epochs, a batch size of 64, a momentum of 0.9, and a weight decay of 0.0005.

We computed two evaluation metrics as in de Almeida et al. (2015). The first one is the accuracy on the test set when choosing the most confident class as output (which is using a threshold of 0.5 for a binary classification problem). The second one is the Area Under the Curve (AUC) of the 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. This measure gives us a threshold-independent evaluation of the classifier.

5.1.1. Results

Results are reported in Table 2. For simplicity, for each experiment we report only the variant of LBP or LPQ that yielded the best performance. We notice that our solution generally performs much better than the other compared methods in terms of both accuracy and AUC. In particular, *mAlexNet* outperforms the other techniques by 3% to 10%, when it is tested on images taken from a subset different from the training set. In fact, *mAlexNet* reaches accuracy values of 98.27% in the UFPR04/PUC training/test set configuration. This is roughly 10% more accurate than the best compared method, that is *Max Ensemble*, which reaches 88.40%.

Experiments on *CNRPark* revealed that classification on this dataset is more challenging than *PKLot*. This is probably due to the high variability of views and occlusion patterns. In this dataset, again, *mAlexNet* still outperforms the other compared classifiers in terms of accuracy. Our approach is roughly 3% better than the best of the others. In fact, it reaches 90.13% in the CNRParkOdd/CNRParkEven configuration and 90.71% in the CNRParkEven/CNRParkOdd one. The other compared methods reach respectively 87.65% and 87.21%.

Table 2
Comparison with state-of-the-art approaches.

Method	Test set	Accuracy	AUC
Train on UFPR04			
mAlexNet	UFPR04	99.54%	0.99
LPQu*	UFPR04	99.55%	0.99
Mean Ensemble*	UFPR04	99.64%	0.99
mAlexNet	UFPR05	93.29%	0.99
LPQg*	UFPR05	84.92%	0.94
Max Ensemble*	UFPR05	88.33%	0.95
mAlexNet	PUC	98.27%	0.99
LPQg*	PUC	84.25%	0.94
Max Ensemble*	PUC	88.40%	0.95
Train on UFPR05			
mAlexNet	UFPR04	93.69%	0.98
LPQgd*	UFPR04	85.76%	0.93
Mean Ensemble*	UFPR04	85.53%	0.95
mAlexNet	UFPR05	99.49%	0.99
LPQu*	UFPR05	98.90%	0.99
Mean Ensemble*	UFPR05	99.30%	0.99
mAlexNet	PUC	92.72%	0.98
LPQu*	PUC	87.74%	0.94
Mean Ensemble*	PUC	89.83%	0.97
Train on PUC			
mAlexNet	UFPR04	98.03%	0.99
LPQg*	UFPR04	87.15%	0.94
Mean Ensemble*	UFPR04	88.88%	0.95
mAlexNet	UFPR05	96.00%	0.99
LBPri*	UFPR05	82.78%	0.91
Mean Ensemble*	UFPR05	84.20%	0.91
mAlexNet	PUC	99.90%	0.99
LPQu*	PUC	99.58%	0.99
Mean Ensemble*	PUC	99.61%	0.99
Train on CNRParkOdd			
mAlexNet	CNRParkEven	90.13 %	0.94
LPQgd*	CNRParkEven	87.65 %	0.95
Train on CNRParkEven			
mAlexNet	CNRParkOdd	90.71 %	0.92
LBP*	CNRParkOdd	87.21 %	0.92

* de Almeida et al. (2015).

5.2. Evaluation of the generalization property

Here we compare the generalization performance of our network architecture *mAlexNet* against the approach proposed in de Almeida et al. (2015) and the full architecture of the *AlexNet* (Krizhevsky et al., 2012). To do so, we perform different experiments where we train with a dataset and we test with a different one. Details about the subsets used in these experiments are reported in Table 1, while the performed experiments are summarized in Table 3.

To compare generalization performance of *mAlexNet* and (de Almeida et al., 2015), we first trained on *PKLot* and tested on *CNRPark*, then viceversa. In order to reduce training times for the SVMs, training was performed on a subset of *PKLot*, called *PKLot2Days*. This subset is formed choosing from *PKLot* the images of the first two days, in chronological order, for each camera (UFPR04, UFPR05, and PUC) and for each weather condition (SUNNY, OVERCAST, RAINY).

We also compared the generalization performance of *mAlexNet* and *AlexNet*. In this case, we trained separately with *CNRPark*, *CNRPark* plus cameras *C1* and *C8* of *CNRPark-EXT*, the whole *CNRPark-EXT*, and *PKLot*. Validation was performed on the corresponding validation sets. Test with accuracy and AUC evaluation were performed on all available test sets.

Table 3
Experiments performed to test the generalization performance of *mAlexNet* and *AlexNet*. Accuracies on test sets are reported, for each combination of (model, training set, test set).

Method	Test set	Accuracy	AUC
Train on CNRPark			
mAlexNet	CNRPark-EXT TEST	93.52%	0.9838
AlexNet	CNRPark-EXT TEST	93.63%	0.9877
mAlexNet	PKLot TEST	95.28%	0.9916
AlexNet	PKLot TEST	95.60%	0.9910
Train on CNRPark+EXT TRAIN C1-C8			
mAlexNet	CNRPark-EXT TEST	95.88%	0.9937
AlexNet	CNRPark-EXT TEST	96.85%	0.9957
mAlexNet	PKLot TEST	90.48%	0.9738
AlexNet	PKLot TEST	96.51%	0.9937
Train on CNRPark+EXT TRAIN			
mAlexNet	CNRPark-EXT TEST	97.71%	0.9967
AlexNet	CNRPark-EXT TEST	98.00%	0.9974
mAlexNet	PKLot TEST	84.53%	0.9699
AlexNet	PKLot TEST	93.70%	0.9923
Train on PKLot TRAIN			
mAlexNet	PKLot TEST	98.07%	0.9967
AlexNet	PKLot TEST	98.81%	0.9984
mAlexNet	CNRPark-EXT TEST	83.83%	0.9139
AlexNet	CNRPark-EXT TEST	90.52%	0.9684
Train on PKLot2Days			
mAlexNet	CNRPark	82.88%	0.899
LBP*	CNRPark	65.31 %	0.580
Train on CNRPark			
mAlexNet	PKLot	90.38%	0.989
LBP*	PKLot	52.88 %	0.391

* de Almeida et al. (2015).

Experiments with *CNRPark* plus cameras *C1* and *C8* of *CNRPark-EXT* were performed in order evaluate with a training set containing a balanced set of different viewpoints. In fact, the majority of the images in *CNRPark-EXT TRAIN* are captured from a frontal viewpoint. *C1* and *C8* have very different viewpoints of the parking lot. As depicted in Fig. 2, *C1* has a side view of the parking lot, while *C8* has a pure front view.

All the models have been trained for at most 6 epochs, with a learning rate of 0.0008, which is multiplied by 0.75 every 2 epochs. Other hyper-parameters are the following: batch size 64, momentum 0.9, and weight decay 0.0005. The final models were chosen as the models with the best performance on validation sets.

5.2.1. Results

Table 3 reports the accuracy values obtained by the various methods for each experiment. We can state that our architecture has comparable performance with respect to *AlexNet* when trained and tested with data coming from the same dataset. In fact, the accuracy values of both models differ at most of 1% in all experiments where training and test subsets are taken from the same dataset.

When training and test is performed on different datasets, *AlexNet* reaches slightly higher accuracy values. Note that when using a viewpoint-balanced training set (i.e. when using *CNRPark*), we always obtain performance higher than 90%. In the best case, there is practically no difference between the two methods. In the worst case, we measured a difference of $\sim$ 9% with respect to *mAlexNet*.

Remember that *AlexNet* is three orders of magnitude more complex than *mAlexNet*. Obviously, a bigger model offers a greater generalization performance at the cost of more resources needed.

Finally, we report that *mAlexNet* generalizes always better than the approach proposed in de Almeida et al. (2015). Accuracy of our methods is respectively 15% and 35% better than the others.

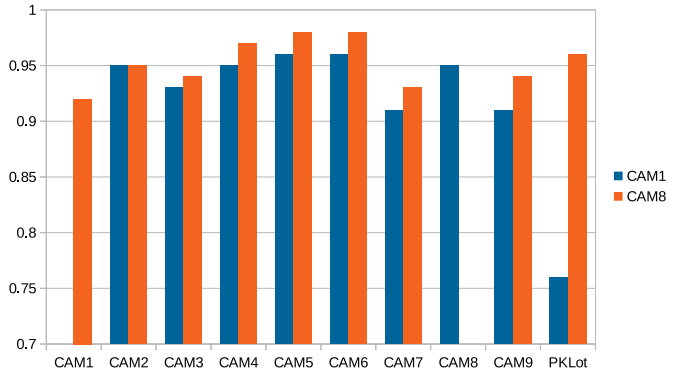


Fig. 4. Results of inter-camera experiments in terms of accuracy obtained when training on camera 1 (in blue), and on camera 8 (in red). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

5.3. Inter-camera and inter-weather evaluation

Errors in the occupancy detection of parking spaces are due to many reasons. For instance, the lighting condition changes during different periods of the year; moreover, occlusions and reflection patterns might introduce a fixed source of error. The weather condition might produce significant illumination changes as well. During a rainy weather, puddles and wet floor create textural patterns that may lead to a misclassification. Sunbeams can create reflections on the car's windscreen or on water, covering the majority of the images with saturated patterns. As we discussed in previous experiments, errors might also be due to low generalization properties of the classifier. When a classifier does not generalize well, it works well just in the conditions where it was trained. For instance, a bad classifier trained on a certain point of view of the parking lot, does not work well when tested with images coming from a camera seeing the parking lot from a different point of view.

To measure the robustness of our approach to these scenarios, we performed two types of experiments: *inter-camera* and *inter-weather* experiments.

In the former, we trained our neural network using images from one single camera of the *CNRPark-EXT* dataset. Then we measured the accuracy obtained with the trained network on pictures captured by another camera. To give maximum emphasis to robustness to viewpoint changes, we performed two different trainings with pictures coming respectively from *C1* and *C8*. *C1* is a side view of the parking lot, while *C8* has a front view of the parking lot. Examples of images taken from these two cameras are depicted in Fig. 2.

In the latter experiment, we trained on images taken during one particular weather condition, and we measured the accuracy obtained on images with different weather conditions. We performed three experiments, training respectively on *CNRPark-EXT SUNNY, OVERCAST*, and *RAINY* subsets.

We used the same training hyper-parameters used in the experiments described in Subsection 5.2.

5.3.1. Results

Results of *inter-camera* and *inter-weather* experiments are reported in Figs. 4 and 5, respectively. The histograms compare the accuracy of a classifier trained on a specific scenario (a specific camera or a specific whether condition) when tested on all other possible scenarios.

In *inter-camera* experiments, we noticed that the best accuracy is given by the model trained on *C8*. This is reasonable because it has a front view of the parking lot, which is common to most

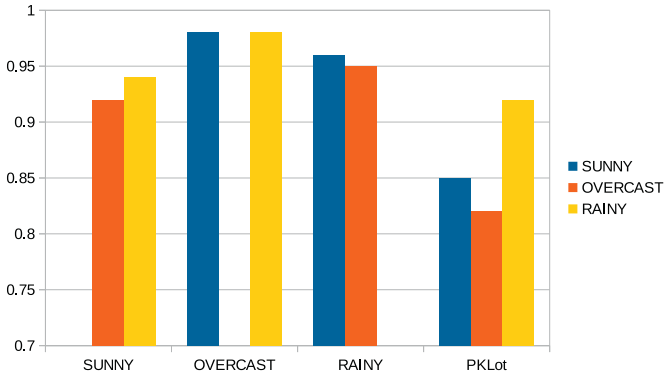
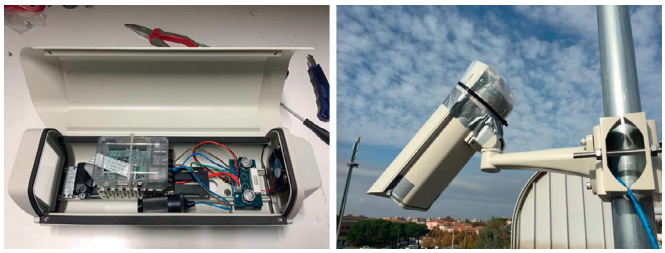


Fig. 5. Results of inter-weather experiments in terms of accuracy obtained when training on a sunny (in blue), overcast (in red), or rainy (in yellow) weather. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)



(A) Inside of a camera box (B) The complete camera box

Fig. 6. Each Raspberry Pi is mounted inside an outdoor camera box (Figure *A* on the left) and it is mounted on top of the roof of the building, attached to a steel pole (Figure *B* on the right).

of the cameras and to other parking lots scenarios (like *PKLot*). *C1* often captures only partial images of parking spaces, and it has a very skewed view of a portion of the parking lot. In fact, it reaches a lower accuracy on *PKLot*, which is mainly composed by images with no occlusions, with a central and vertical view of the parking lots. Nevertheless, it reaches accuracy values over 90% when tested with pictures coming from all other cameras of *CNRPark-EXT*.

A very good generalization is achieved even in *inter-weather* experiments. We noticed that the amount of error made by our model is related with the difference between the training and testing weather conditions. For example, our model trained on “sunny” images performs better on “overcast” images than “rainy” ones. The same goes for the model trained on “rainy” images, which is more accurate on “overcast’ images than “sunny” ones. However, performance differences are small. Rainy training is the winner when tested with the *PKLot* dataset. This is probably due to a bias in the *PKLot* dataset where most images seems to be similar to “rainy” images.

6. Deployment of the proposed solution in a real scenario

As we already said, smart cameras were built around Rasbperry Pi 2 model B, equipped with standard Raspberry Pi camera modules. Each smart camera has been mounted in an outdoor camera box and has been installed on the roof of a building in front of the parking lot (see Fig. 6).

The entire framework was deployed in the parking lot of the research campus of the CNR in Pisa as a Smart City application. The monitored parking lot consists of 164 parking spaces, organized in five rows, four of which are composed of about 35 parking spaces each, and one row is composed of 18 parking spaces. Although a single Raspberry Pi equipped with the standard camera module is able to monitor more than 50 parking spaces (i.e. with the given height and distance of the cameras from the parking lot), due to the conformation of the parking lot monitored, we had to deploy 9 smart cameras in order to monitor all the parking spaces. Some cameras monitor about 20 parking spaces, while some others more than 50.

Due to the angle of view and the perspective of the camera module used, and the position of the smart cameras, 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 used the redundancy of the overlapping parking spaces to reduce occlusion problems (for examples trees).

In particular, we assigned a weight value to each pair $\langle parking\_space, camera \rangle$, representing how good is the view of that parking space seen by that camera. A high value of this parameter means that the parking space is in the center of the image and that there is no obstacle between the camera and the parking space. The confidence returned by the CNN for a given parking space is weighted with this value and, in case of a parking space monitored by more cameras, the highest weighted confidence value is selected at server side. In this way, we are able to correct classification errors on parking spaces occluded for some cameras, by choosing the confidence value of another camera that has a clearer view of that parking space.

Using smart cameras, rather than ground sensors, has two relevant advantages: lower cost *per* parking space 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. These are very limited costs, and, as shown in Amato et al. (2016) and in Section 5, the accuracy of this approach is very good, and it is comparable to the accuracy of a ground sensor. Therefore, it is possible to monitor a large parking lot with a cost per parking space which is an order of magnitude lower than ground sensors, and still achieving comparable accuracy results.

Moreover, with smart cameras we are not limited solely to parking lots monitoring applications.

We could exploit the smart camera to perform additional activities like, for example, video surveillance activities, such as face/people recognition, or tracking and logging of people and moving vehicles.

6.1. Implementation

The software running on the smart cameras periodically captures the image of a portion of the parking lot and, for each parking space, determines the occupancy status by using a CNN trained offline.

Pictures captured by cameras are filtered by a mask that identifies the various parking spaces. The mask was built manually once and for all. Examples of masks built for different cameras are shown in Fig. 2. At run time, each frame is automatically segmented in patches (we recall that a *patch* is 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 trained CNN, to decide whether the corresponding parking space is empty or busy. On the Raspberry Pi 2, the classification of 50 parking spaces and the transmission of the results to a web server takes about 15 s. Fig. 7 shows an example of the classification of a portion of the parking lot monitored by a smart camera. As can be seen in figure, our approach deals very well with common challenging visual classification problems such as shadows, obstacles (trees or lamps) or

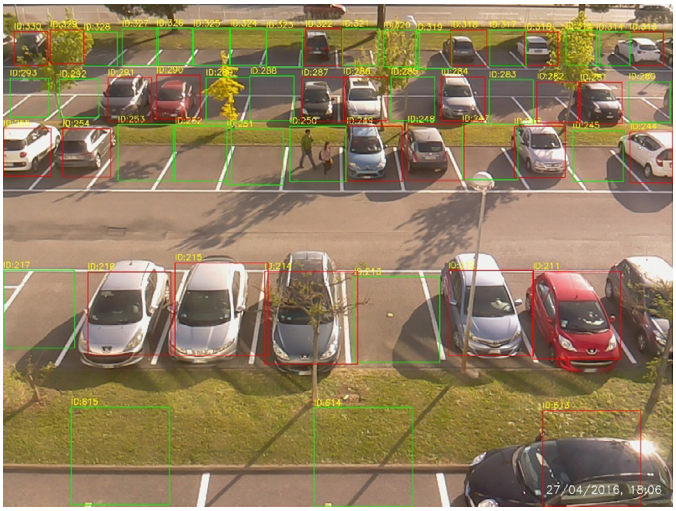


Fig. 7. Example of classification of a portion of the parking lot.

even people occupying the parking spaces. We prepared a website[6] to access the live view of all the cameras by authorized people.

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 3 MB each) are sent to the server which would thus become a bottleneck and a single point of failure.

Hardware and software details are briefly reported for completeness. The smart cameras are equipped with an ARM Cortex-A7 CPU, 1GB RAM DDR2, and a 32GB micro SD card for storage. The camera module is a 5 MP fixed-focus camera that supports 1080p30, 720p60 and VGA90 video modes, as well as still captures. The view angles of the camera are 53.50° horizontally and 41.41° vertically. We capture still pictures at a resolution of 2592 × 1944 pixels.

We used the OpenCV[7] library to elaborate the frames acquired by the cameras, and Caffe (Jia et al., 2014) to train and use neural networks.

7. Conclusions

A decentralized and efficient solution for visual detection of the parking status was presented, which exploits deep Convolutional Neural Networks (CNNs) to classify the parking space occupancy. The solution employees smart cameras built using Raspberry Pi platform equipped with a camera module. Each smart camera can simultaneously monitor up to fifty parking spaces.

A deep CNN architecture designed to run on embedded systems such as smart cameras, is used to classify images of parking spaces as occupied or vacant directly on board of the smart camera. In this way, the only information that is sent to a central server for visualization is the binary output of the classification.

As a further contribution, we collected and made publicly available *CNRPark-EXT*, a dataset containing images of a real parking lot taken by nine smart cameras, in different days, with different weather and light conditions. *CNRPark-EXT* contains images with high variability related to occlusions, point of views, illumination and weather conditions. This makes the dataset more compatible with a real scenario of an outdoor parking lot, and represents a good complement to other publicly available datasets, for more reliable assessments.

Using both *CNRPark-EXT* and *PKLot*, another publicly available dataset for parking occupancy detection, we performed experiments to compare the performance and generalization capabilities of our approach against other state-of-the-art techniques. These experiments show that our approach outperforms other ones based on shallow models, such as SVMs. Specifically, our CNN exhibits very high accuracy, even in presence of noise due to light conditions variation, shadows, and partial occlusions.

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

[7] http://opencv.org/.

Acknowledgments

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. We gratefully acknowledge the support of NVIDIA Corporation with the donation of the Tesla K40 GPU used for this research.

References

de Almeida, P. R., Oliveira, L. S., Britto, A. S., Silva, E. J., & Koerich, A. L. (2015). Pklot–a robust dataset for parking lot classification. *Expert Systems with Applications, 42*, 4937–4949.

Amato, G., Carrara, F., Falchi, F., Gennaro, C., & Vairo, C. (2016). Car parking occupancy detection using smart camera networks and deep learning. In *21th IEEE symposium on computers and communications (iscc)* (pp. 1212–1217). IEEE.

Belbachir, A. N. (2010). *Smart cameras*: Vol. 2. Springer.

Bengio, Y. (2009). Learning deep architectures for AI. *Foundations and Trends® in Machine Learning, 2*, 1–127.

Caicedo, F., Blazquez, C., & Miranda, P. (2012). Prediction of parking space availability in real time. *Expert Systems with Applications, 39*, 7281–7290.

Chen, X., Xiang, S., Liu, C.-L., & Pan, C.-H. (2014). Vehicle detection in satellite images by hybrid deep convolutional neural networks. *Geoscience and Remote Sensing Letters, IEEE, 11*, 1797–1801.

Dan, N. (2002). Parking management system and method. US Patent App. 10/066,215.

Delibaltov, D., Wu, W., Loce, R. P., Bernal, E., et al. (2013). Parking lot occupancy determination from lamp-post camera images. In *Intelligent transportation systems-(itsc), 2013 16th international ieee conference on* (pp. 2387–2392). IEEE.

Girshick, R., Donahue, J., Darrell, T., & Malik, J. (2014). Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the ieee conference on computer vision and pattern recognition* (pp. 580–587).

Huang, C.-C., Tai, Y.-S., & Wang, S.-J. (2013). Vacant parking space detection based on plane-based bayesian hierarchical framework. *Circuits and Systems for Video Technology, IEEE Transactions on, 23*, 1598–1610.

Jermsurawong, J., Ahsan, U., Haidar, A., Haiwei, D., & Mavridis, N. (2014). One-day long statistical analysis of parking demand by using single-camera vacancy detection. *Journal of Transportation Systems Engineering and Information Technology, 14*, 33–44.

Jia, Y., Shelhamer, E., Donahue, J., Karayev, S., Long, J., Girshick, R., … Darrell, T. (2014). Caffe: Convolutional architecture for fast feature embedding. In *Proceedings of the ACM international conference on multimedia* (pp. 675–678). ACM.

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems* (pp. 1097–1105).

Lan, K.-C., & Shih, W.-Y. (2014). An intelligent driver location system for smart parking. *Expert Systems with Applications, 41*, 2443–2456.

Masmoudi, I., Wali, A., Jamoussi, A., & Alimi, A. M. (2014). Parking spaces modelling for inter spaces occlusion handling. In *22nd international conference on computer graphics, visualization and computer vision* (pp. 119–124).

Ojala, T., Pietikäinen, M., & Mäenpää, T. (2002). Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *Pattern Analysis and Machine Intelligence, IEEE Transactions on, 24*, 971–987.

Ojansivu, V., & Heikkilä, J. (2008). Blur insensitive texture classification using local phase quantization. In *International conference on image and signal processing* (pp. 236–243). Springer.

del Postigo, C. G., Torres, J., & Menéndez, J. M. (2015). Vacant parking area estimation through background subtraction and transience map analysis. *IET Intelligent Transport Systems, 9*, 835–841.

Rahtu, E., Heikkilä, J., Ojansivu, V., & Ahonen, T. (2012). Local phase quantization for blur-insensitive image analysis. *Image and Vision Computing, 30*, 501–512.

Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scalearXiv:1409.1556, arXiv preprint.

Tsai, L.-W., Hsieh, J.-W., & Fan, K.-C. (2007). Vehicle detection using normalized color and edge map. *Image Processing, IEEE Transactions on, 16*, 850–864.

Wu, Q., Huang, C., Wang, S.-y., Chiu, W.-C., & Chen, T. (2007). Robust parking space detection considering inter-space correlation. In *Multimedia and expo, 2007 ieee international conference on* (pp. 659–662). IEEE.