Parking Lot Detection

Automated detection of empty parking lots

Group 17

Alexandros, George, Laurens, Manolis, Sofia

Introduction

Finding a parking space has been a problem for everyone who uses a car. Even in organized parking lots sometimes it is difficult to find a space, let alone knowing if one's car fits in that space. A common approach to that issue is to use sensors in order to identify if a spot is occupied or not, however, this creates problems such as maintenance of the hardware and initial investment costs. Another approach is to use a camera and a deep learning architecture to keep track of the availability of the parking spaces. We found this research field extremely useful because not only we will get familiar with the deep learning algorithms, but also we will implement something that can be used in real life.

In this paper we propose a **convolutional neural network** (CNN), which classifies a parking spot image as available or occupied, based on the existence or absence of a car in this place. Our baseline is a pre-trained **VGG-16 neural network**, where the three fully connected layers at the end are optimized with the use of SGD and the use of our training data. For the training and evaluation of our model, we used the PKLot dataset.

Main Contributions

- Usage of VGG net for the parking spot classification issue.
- Investigate how different weather conditions affect the performance of the network.
- Investigate how the system would generalize when tested in different parking lots than the one trained with.
- Optimize the network to reduce the classification error and overfitting/underfitting.
- Investigate how to make the system fully automatic.

Dataset

- The PKLOT dataset [1] consists of 12,417 images of parking lots and 695,899 samples of parking spaces located in three different areas.
- Parking spaces are manually segmented and labeled by authors.
- Images are grouped by the weather condition of the day they were captured. Rainy,cloudy and sunny.

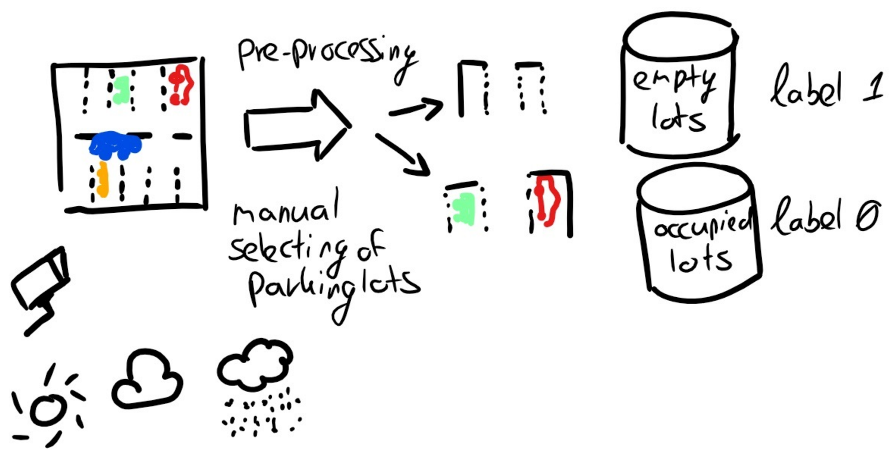


Figure 1: Process of dataset creation.

Technical approach

In this work we propose a system which employs deep CNN for detection of vacant and occupied parking slots.A pipeline of our algorithm can be depicted in the figure 2.The data needed for training a network like this should consist of a plethora of images which are captured in different angles and lighting conditions. The main reason behind this strategy is that a diversity in data facilitates the capability of a network to be generalized in a sound manner in new unseen data.

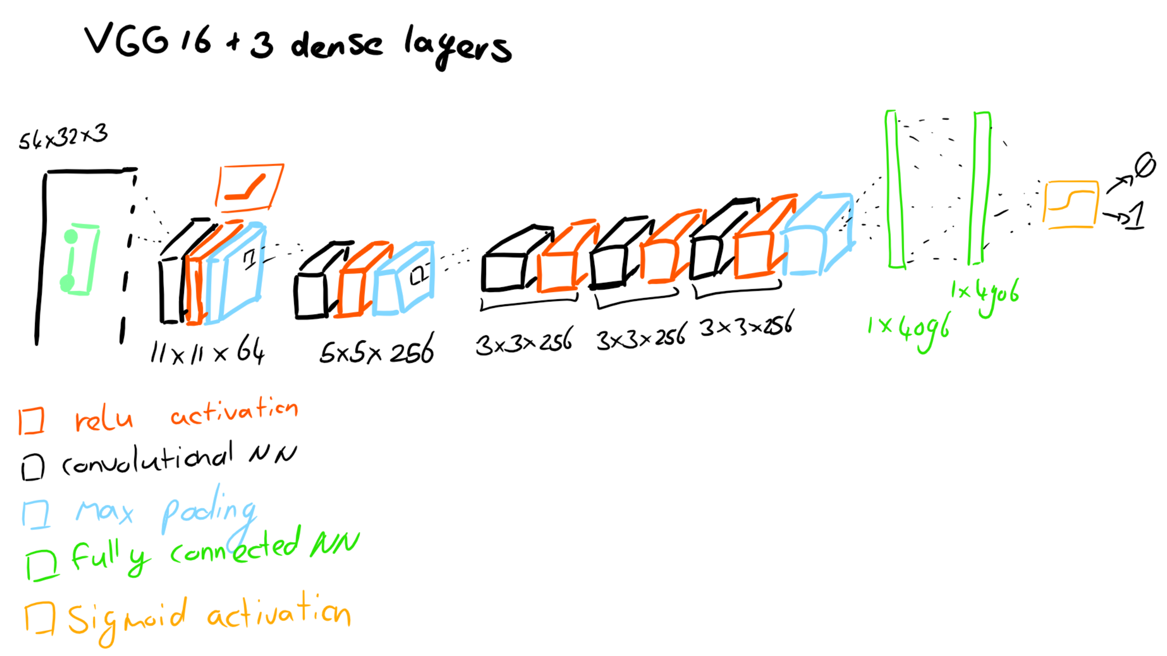


Figure 2: Architecture of the network.

CNN Architecture

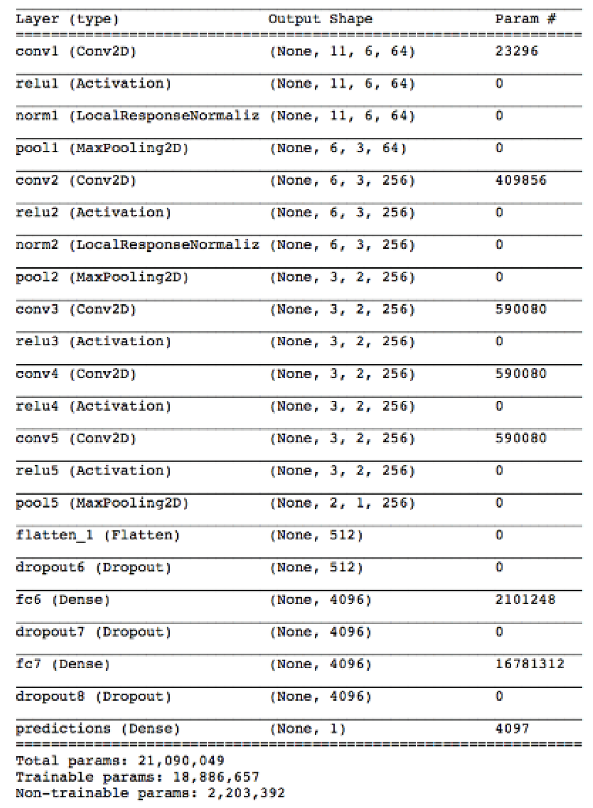
The architecture we chose for that project uses the pre-trained convolutional layers from VGGNet-F [3]. These convolutional layers were pre-trained in ImageNet. Fine-tuning the VGG net from scratch, would be impossible due to resources limitations. Beside this, since our data are associated with cars, which is a category included in ImageNet, we expect higher-level features in the CNN to be relevant to our dataset as well.

Mainly because of time limitations we decided to follow the Transfer Learning technique, thus we froze all the layers of our CNN and only fine-tuned the last three fully connected layers. After splitting our training set into two different sets (i.e. training-validation set) we were able to choose our hyperparameters through performing cross-validation. Namely, the hyperparameters we ended up can be depicted in Table 1 and we performed this procedure for 10 epochs.

Hyperparameters	Value
Learning Rate	10^{-3}
Batch Size	128
Weight Decay	5×10^{-3}
Optimizer	SGD with Nesterov
Momentum of Nesterov	0.99
Dropout of the 3 FC Layers	0.2

Table 1: Values of Hyperparameter

An overview of our architecture in combination with the shape of the output and the number of parameters of each layer can be depicted in figure 3. The final output of our CNN model is going to be a probability vector that includes the probabilities for which our input image belongs to the class 0 (i.e. vacant parking spot) or class 1 (i.e. occupied parking spot).



Layer (type)	Output Shape	Param #
conv1 (Conv2D)	(None, 11, 6, 64)	23296
relu1 (Activation)	(None, 11, 6, 64)	0
norm1 (LocalResponseNormaliz	(None, 11, 6, 64)	0
pool1 (MaxPooling2D)	(None, 6, 3, 64)	0
conv2 (Conv2D)	(None, 6, 3, 256)	409856
relu2 (Activation)	(None, 6, 3, 256)	0
norm2 (LocalResponseNormaliz	(None, 6, 3, 256)	0
pool2 (MaxPooling2D)	(None, 3, 2, 256)	0
conv3 (Conv2D)	(None, 3, 2, 256)	590080
relu3 (Activation)	(None, 3, 2, 256)	0
conv4 (Conv2D)	(None, 3, 2, 256)	590080
relu4 (Activation)	(None, 3, 2, 256)	0
conv5 (Conv2D)	(None, 3, 2, 256)	590080
relu5 (Activation)	(None, 3, 2, 256)	0
pool5 (MaxPooling2D)	(None, 2, 1, 256)	0
flatten_1 (Flatten)	(None, 512)	0
dropout6 (Dropout)	(None, 512)	0
fc6 (Dense)	(None, 4096)	2101248
dropout7 (Dropout)	(None, 4096)	0
fc7 (Dense)	(None, 4096)	16781312
dropout8 (Dropout)	(None, 4096)	0
predictions (Dense)	(None, 1)	4097

Total params: 21,090,049
Trainable params: 18,886,657
Non-trainable params: 2,203,392

Figure 3: Pipeline of our algorithm.

During the implementation of our algorithm, a split of the dataset in **50% training and 50% testing was employed**. Also we split the training set further in **70% training and 30% validation** set in order to tune our hyperparameters.

Results

The following figures, demonstrate the 'features' in which our network focuses, when trying to classify a lot as free or occupied. In figure 4 we the most important features of the network are clearly visible.

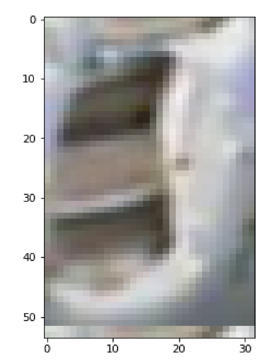


Figure 4: pre-saliency

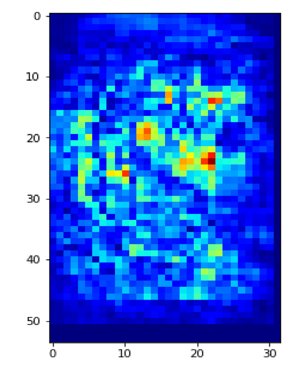


Figure 5: post-saliency

The analysis presented below, gives a good insight on how accurate our deep network is in identifying classes of both occupied and free parking places. As we can notice, both FP and FN values are quite low compared to TP and TN. Also, regarding our Quantitative Analysis, we will demonstrate four basic metrics named Accuracy, Precision, Recall and F1 score.

Predict \ True	Empty	Occupied
Empty	126024	666
Occupied	3758	86201

Metric	Value
Accuracy	97%
Precision	97%
Recall	99%
F-Measure	98%

One of the interesting research questions we would like to answer, is how would our network perform and generalize when tested with data from different weather conditions and different parking lots than the ones it was trained with.

In the following graph we can see a visualization of the train and test error throughout 100 epochs, for training and testing the network with different datasets.

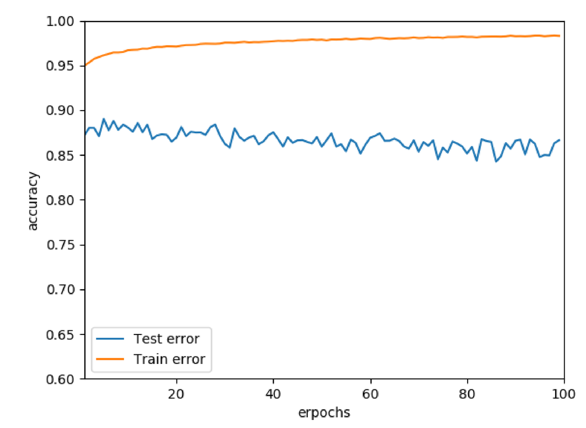
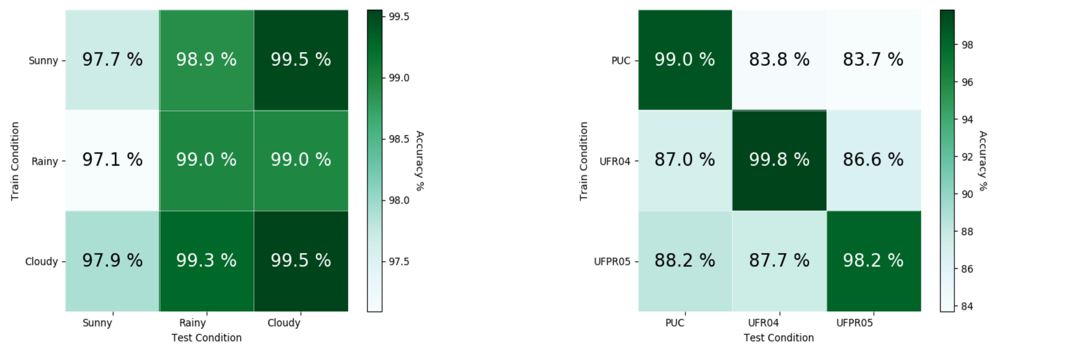


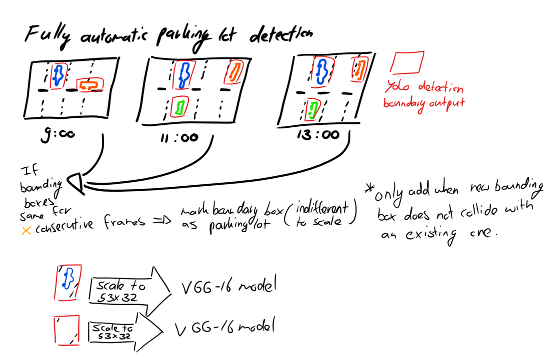
Figure 6: Train set: UFPR04 Test set: UFPR05

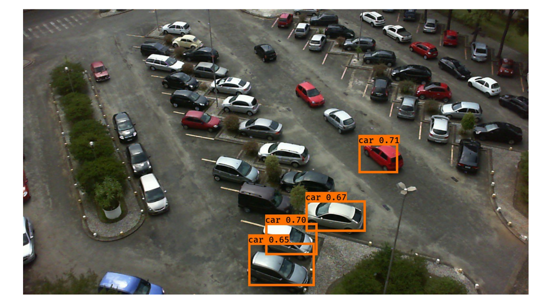
The first heatmap presents how the network performed for all the combinations of different weather condition for train and test. The second heatmap presents how the networks performs when trained and tested with data from different parking lots.



Forthcoming Research

As an extension, we wanted to fully automate the system of parking lot detection, without the manual segmentation of the parking lots. In the following images we present the necessary steps in order to detect the parking spaces automatically and the results of these steps using YOLO.





We tried different models for object detection, one of them was the well known YOLO [2], however, the model was only able to detect cars which were close enough to the camera. The extension is currently under development in order to make the network detect also the far away blocks.

Indicative References

[1] Paulo R.L. 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.

[2] Joseph Redmon, Santosh Kumar Divvala, Ross B. Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. *CoRR*, abs/1506.02640, 2015.

[3] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014.