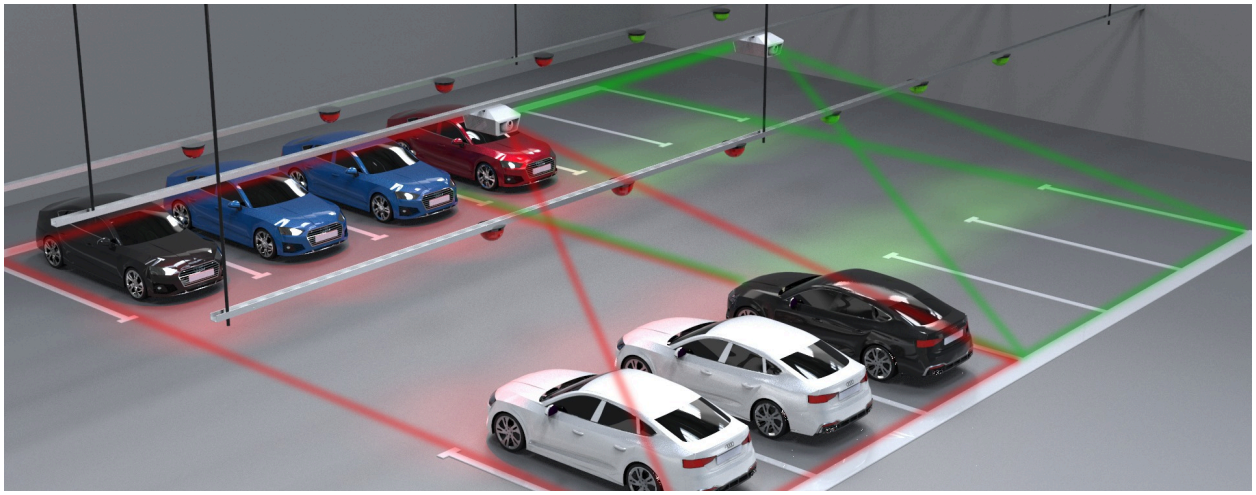
UNIVERSITY OF NORTH CAROLINA
CHARLOTTE

# Parking Lot Occupancy Detection using Deep Learning

**Detecting occupancy status on parking spaces using deep learning neural networks.**

Jeffery Sengsy | Devangi Samal | Jessica Ricks



# Introduction

## Problem, Motivation, Approach

Are there enough parking spots available? At glance, this question seems a banal topic. However in context of an average American spending 17 hours per year searching for a parking space makes the question more worthy of research. For the driver, the search for a spot is a waste of time, and unnecessarily contributes to their carbon footprint. A city planner may struggle against the search to tackle traffic congestion and improve urban safety. A property manager would seek to reduce search time to improve experience, revenue while minimizing costs.

Current technology  for lot detection includes manual counting by human staff. In-ground sensors and light sensors with magnetic and sonic technology are highly accurate yet difficult to maintain and costly.

Our interest lies with enhancing the existing infrastructure of CCTV cameras to incorporate computer vision. A promising thought to provide real time monitoring with high accuracy, cost effective and low maintenance. We seek to apply deep learning algorithms of CNN realm to classify occupancy or availability of a spot with ResNet, UNet and Faster RCNN on CNR Parking Images and Patches. The dataset was created and available open source on CNR+Ext Parking data. This was gathered by a team of researchers to ensure that different angles and conditions were taken to help teach their model. We apply resizing and color transformations to augment the data to mimic real world disturbances during image collection such as light glares, low light, and blocking of camera lens. Our UNet architecture achieves maximum accuracy in detection parking availability.

## Backgrounds

Urban areas face increasing pressure on parking infrastructure due to vehicle growth and limited space. Using traditional parking systems, such as ground sensors, RFID, or manual monitoring, are often expensive and inefficient. As a solution, computer vision-based methods using deep learning have gained attention for their scalability and cost-effective characteristics.

Traditional radar detection requires specific camera angles and distinct reference lines, while parking vision effectiveness is greatly affected by lighting conditions and somewhat clear camera images. Few years ago a team of researchers made an effort to tackle lighting problems with a two part solution. Prior to their work, other solutions included end-to-end CNN detection while simultaneously obtaining global and local information about the spot. Some CNN methods are trained to identify spots and locate vehicles with 2D color tags to protect against lighting change. Some use special fish-eye cameras with modified YoloV3 architecture to speed up detection. Detection with multiple camera inputs combined with enhancement algorithms suggested that the model could recognize lot boundaries without specific lane markers which was also effective in day and night scenes.

Most of these solutions work in a closed parking environment and do not fully address uneven lighting conditions. This team first utilizes fisheye cameras to collect data then applies Faster RCNN to detect spots quickly. The last step of their approach is an image processing sequence. This processing includes converting pixel matrix to grayscale by weighted average method. Next image enhancement with median filtering with a 3x3 pixel window to suppress noise while preserving important image details. Then binarization is applied to the enhanced image where pixels that exceed the threshold are reassigned to 255 and rest to 0. Lastly a connected regions method is used to extract parking lines. Put together, this solution is good at

detecting spots with lots of interference and uneven lighting, especially in lots with lane markers. Our experiment continues to work with Faster RCNN, but seeks to train a model with images taken without a specific type of lens and keep RGB colors. Rather use color augmentation on images to account for change in lighting.

## Methods

### Dataset

A dataset for visual occupancy detection consisted of 100K+ labelled images from nearly 160 parking spaces in a parking lot in Pisa, Italy. Images are collected between November 2015 and February 2016 in various conditions from up to 9 camera angles. Also includes partial shadows and obstacles like trees and lamp posts.

*Fig. 1 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.*

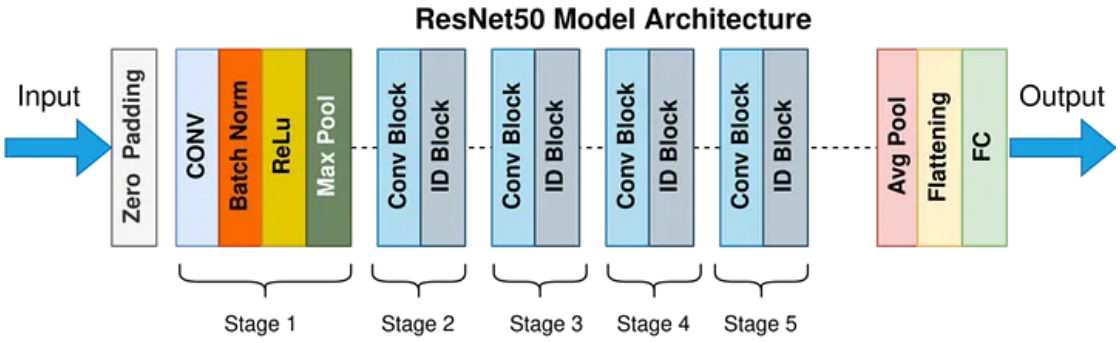
(A) Overview of CNRPark CAM A (B) Overview of CNRPark CAM B

(C) Overview of CNRPark-EXT CAM 1 (D) Overview of CNRPark-EXT CAM 8

*Fig2. 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.*

## Method: Residual Neural Network

This method was performed on the patch images (CNRPARK-Patches-150x150).




Here we use ResNet50 which is an image classification model that can be trained on large datasets and achieve results more efficiently. The key to this method is using residual connections, which allow the network to learn a set of residual functions that map the input to the desired output. The use of ResNet50 here is to train a deep learning model that classifies the 150x150 parking spot image patches (free = empty, busy = occupied). We first filter the metadata

CSV so that it only includes entries for which images are actually available. Now that we have created our new labels to work with the patch data, we can prepare our training and validation sets to train the model. We use ImageDataGenerator to automatically augment and split the data during training. Then the base model is initialized (ResNet50) and is pre-trained on ImageNet. Base model layers are frozen to avoid updating them during the training phase. The layers for the model goes as follow:

- Base_model (Resnet50)
- GlobalAveragePooling2D (converts feature maps to a 1D vector)
- Dense (256, relu) : Adds a fully connected layer
- Dropout(0.5) : prevents overfitting
- Dense (1, sigmoid): Outputs a probability for binary classification.

This is using transfer learning with ResNet50 to efficiently classify parking occupancy in the patch images. The model has an option to be extended later by unfreezing and fine-tuning ResNet layers.

## Method: U-Net

The dataset CNR-EXT-FULL_IMAGE was used for larger image sizes, 256x256, and full parking lot photos to create the U-Net mode l. The full image dataset presented better quality images since the images were not cropped. The full images were resized to 512x356 for image preprocessing and generating masks. The initial amount of masks generated for each folder, Rainy, Overcast, and Sunny, was 100 images, but after running through the U-Net model, the final number of masks generated was 1,000 per folder subgroup to get a better-performing model. Once the masks were generated using DeeplabV3, the 3,000 images were split into training and testing with an 80/20 split. U-Net models are used for precise segmentation of pixels

and historically have worked well with medical data. Pytorch is used to create a custom dataset to preprocess all the images in RGB and also in grayscale before implementing the U-Net model. To help with the larger number of generated masked images, data augmentation techniques were used to help resize the images down to 256x256. The U-Net model architecture consists of an encoder-decoder with skip connections. In the model, there are 4 convolutional blocks in the encoder, a bottleneck layer, an alternation between transposed convolutional layers and concatenation, and a final layer output. Following these layers is a convolution blocks with 2 Conv2D layers to increase complexity and the activation by ReLU. Upsampling is implemented to help increase the resolution of the images and provide flexibility for the spatial information.

## Method Faster R-CNN

The dataset includes bounding boxes in the form of (x,y) center coordinates and height and width. A custom class converts the bounding box to (x,y) coordinates of four corners. The class converts to tensors to hold tuples of image and target; dictionary of boxes and labels. Another custom class resizes the image to 128x128, applies horizontal flip, random rotation and color jitter to adjust brightness, contrast and saturation. The expectation here is that Faster RCNN can learn occupancy and lot space despite variable coloring and lighting conditions. This transformations class also handles scaling down of bounding boxes size to maintain integrity of the dataset. Once data was loaded with appropriate splits, a Faster RCNN with a feature pyramid network (FPN) backbone was trained. The usual foundation for a Faster RCNN is ResNet50, but ResNet50 can lose effectiveness when identifying small objects. Our dataset includes images of variable angles, far and near aerial views, a FPN is added on top of the residual network. A ResNet50 will output one feature map, then the FPN will extract features at multiple scales. FPN outputs a pyramid of feature maps that have combinations of low resolution with semantically

strong features and high resolution with semantically weak features which is fed into the next portion: regional proposal network (RPN).

This network will output a list of regions that may have an object: predict if there is a car. RPN consists of three convolution blocks with each subsequent block reducing in number of output channels refining the bounding box predictions. The next portion is region of interest (ROI) that converts the proposed bounding boxes to fixed feature maps that can be used by the following fully connected layers. This last layer will classify images by background or object and refine the position of the predicted bounding box. The training loop consisted of 10 epochs with an adam optimizer and learning rate scheduler updating learning rate every 3 epochs by gamma of 0.1 starting at learning rate of 0.001.

## Experiments

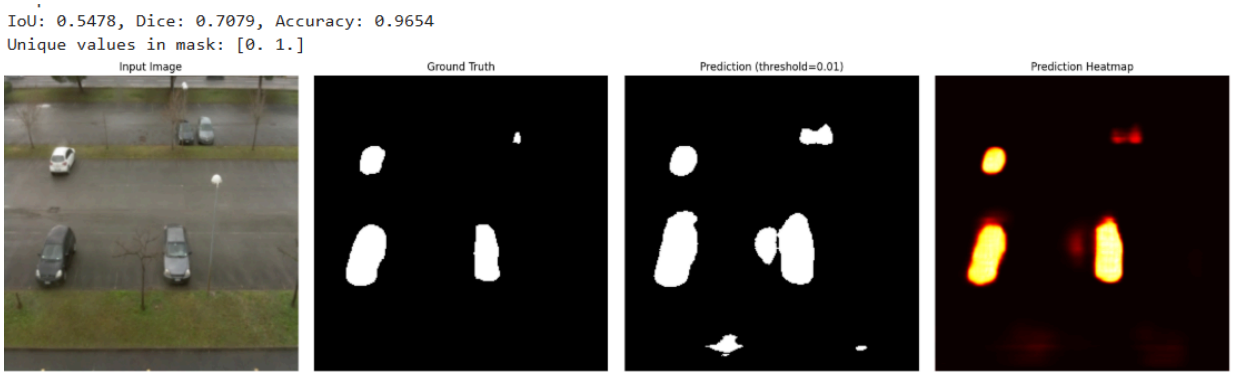
### Results: ResNet50

This model ran for 10 epochs on batch size 32. The given dataset was split into two folders being “A” and “B”. They contained the same patched images but different numbers of busy/free photos. On Patch A, the results had an accuracy score of 93.17% and validation loss of 26.09%. This tells us the model could possibly be overfitting due to the amount of biased number of busy/free photos. The same training was run on patch B and resulted in an accuracy score of 75.94% and validation loss of 36.38%. When this model was given unseen data, it had a harder time guessing due to the quality and conditions of the picture. The model did a better job of guessing empty parking spaces vs occupied.


The green represents that the space is free with a 75% probability. While the occupied space is blue, predicting a probability of 29%. However, due this being binary, this could be sufficient because we know for sure something is occupying the parking space.

## Results U-Net:

After running the model through 20 epochs, the results from the U-Net model show a validation accuracy of 87.35%, a continued downward trend for validation loss, and better segmentation results when the parking lot has fewer cars versus a full parking lot. An example of a high




accuracy image in the model shows an accuracy of 96.54%, Dice loss of 0.7079, and IoU of 54.78%(Fig. 3). The measurements indicate that a high number of pixels are classified correctly as occupied or free, moderately high dice loss indicates good precision boundary detections, and there is moderate accuracy between predicted images and actual images. For low accuracy

images, the results are IoU: 0.2121, Dice: 0.3500, Accuracy: 0.5792(Fig. 4). This example shows that a little over 50% of pixels are classified correctly, mismatch between predicted and actual regions, and poor segmentation. Lastly, when comparing the high accuracy or low accuracy images, the sunny images(Fig.4,6) have better accuracy than any of the rainy images.

*Fig 3. High-accuracy image for pixel segmentation. Rainy Day.*

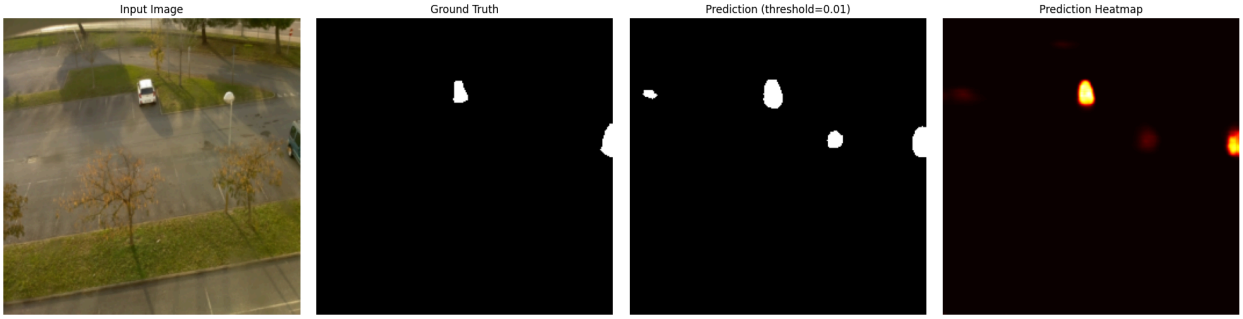



*Fig 4. High-accuracy image for pixel segmentation. Sunny Day.*

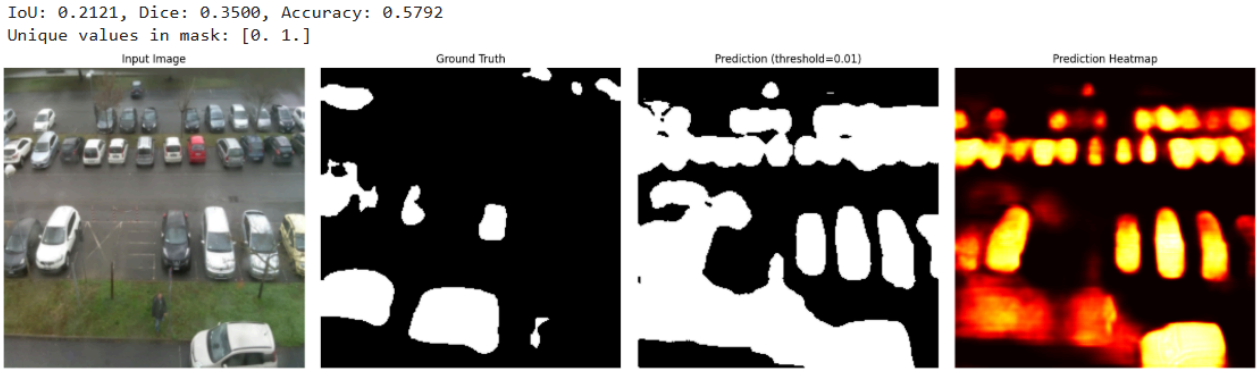


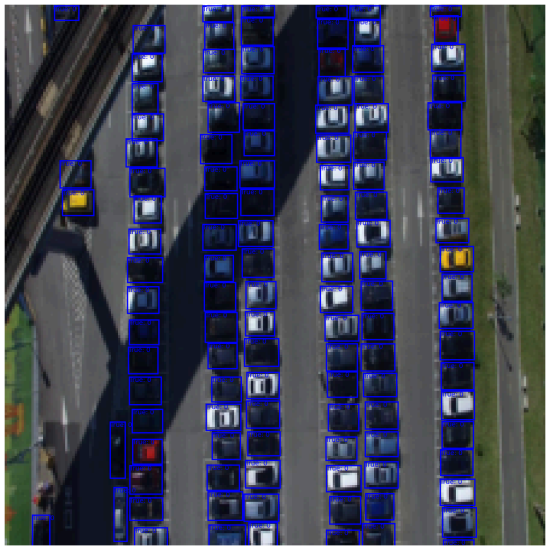

*Fig 5.. Low-accuracy image for pixel segmentation. Rainy Day.*


*Fig6. Low-accuracy image for pixel segmentation. Sunny Day*

When comparing the high and low accuracy images, the results indicate that there are some inconsistencies when classifying and segmenting different types of images. The biggest visual difference between these images, which is seen throughout all the images, is the number of cars in the image. The more cars in the image, the less precise the decision boundaries are, the lower the pixel classification accuracy, and the higher the mismatch is between the predicted and actual image. Another difference that can affect the evaluation metrics is the type of weather. Sunny images show way better accuracy than pixels classified in rainy images.

## Results Faster RCNN:

The training of Faster RCNN resulted in a choppy downward training loss curve. Accuracy on the test data came in at 65.23%. The unsmooth training is likely due to poor optimization. While Adam is a good option, the learning rate schedule may have been too large and needs more fine tuning. Also the batch size of 8 may have been too small for this kind of architecture.
Upon visualizing the model predictions, this following  image has many accurately predicted parking spots bounded in blue.



## Conclusions

Much of these ideas and portions of these methods are prevalent in autonomous driving research, but largely specialized detection with specific camera angles and requirement of lot boundaries. This project sought to apply residual networks and segmentation on data that has various resolutions, uneven lighting and represents many weather conditions.

The results of the U-Net model show that there is better classification of pixels when images have fewer cars and the weather is sunny. The biggest challenge in creating the U-Net model was the class imbalance, which classified pixels to be majority 0s for the prediction images. To combat this problem, BCEDiceLoss was implemented to evaluate the overlap between the masks and the prediction. For future work, running the model for more epochs, tweaking the loss function, and adaptive cropping to help the model classify pixels when cars are parked close to each other.

Challenges for Faster RCNN included long training periods and realizing to ensure that bounding boxes scaled down with resizing transformations. Faster RCNN could benefit from a larger batch size, and fine tuning of learning rate schedule. Also would like to apply data augmentation one at a time to see the effects of each transformation. The current experiment results in a heavy model, but modifications to the back bone and residual network could be done to adapt for a lightweight network, such as MobileNet, to be able to embed in small devices for real time detection.

## References

Institute of Information Science and Technologies of the National Research Council of Italy. (n.d.). *Deep learning for decentralized parking lot occupancy detection*. ISTI-CNR. https://www.isti.cnr.it/

Arifovic, A., & Sijercic, E. (2022). *Image-based parking occupancy detection using deep learning and Faster R-CNN*. In 2022 26th International Conference on Information Technology (IT) (pp. 1–5). IEEE. https://doi.org/10.1109/IT57578.2022.10010638

Huang, C., Yang, S., Luo, Y., Wang, Y., & Liu, Z. (2022). Visual detection and image processing of parking space based on deep learning. *Sensors*, *22*(17), 6672. https://doi.org/10.3390/s22176672

PyTorch. (n.d.). *DeepLabV3 ResNet50 — Torchvision main documentation*. PyTorch. Retrieved May 3, 2025, from https://pytorch.org/vision/stable/models/generated/torchvision.models.segmentation.deeplabv3_resnet50.html

Ronneberger, O., Fischer, P., & Brox, T. (2015). *U-Net: Convolutional networks for biomedical image segmentation*. In N. Navab, J. Hornegger, W. M. Wells, & A. F. Frangi (Eds.), *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015* (pp. 234–241). Springer. https://doi.org/10.1007/978-3-319-24574-4_28

Iglovikov, V. (2018). *Pytorch-UNet: Convolutional neural networks for biomedical image segmentation*. GitHub. https://github.com/milesial/Pytorch-UNet

Amato, G., Carrara, F., Falchi, F., Gennaro, C., & Vairo, C. (2016). *CNRPark+EXT: A dataset for visual occupancy detection of parking lots* [Data set]. ISTI-CNR. http://cnrpark.it/

Parking lot occupancy detection. Home. (n.d.). https://www.visive.ai/parking-lot-occupancy-detection

Huang, C., Yang, S., Luo, Y., Wang, Y., & Liu, Z. (2022, September 3). Visual detection and image processing of parking space based on Deep Learning. MDPI. https://www.mdpi.com/1424-8220/22/17/6672