Final Report

APS360: Group 21

09 December 2020

Olivia Tracey (1004862623)

Michael Boyadjian (1005109142)

Scott Oxholm (1005106788)

Matthew Ing (1005057260)

*Not including references, figures, or headings, the word count is **2468.***

1. Introduction

Parking is a time-consuming task because it requires drivers to explore all possible spots before determining whether the lot is full. This wasted time can lead to a decrease in customer satisfaction, or overall loss of customers who cannot find a spot. Solutions exist that use sensors to monitor individual spots in a parking lot (See Fig. 1). However, these sensors can cost anywhere from $200-400 per spot [1]. The goal of our project is to use our understanding of machine learning (ML) to develop a solution that can produce the number of available spots in a lot, without requiring these expensive sensors.

In class, we have explored the effective use of ML for a variety of classification tasks. As a result, we felt that we could utilize ML to classify whether a parking spot is full or empty in a parking lot. Data pre-processing will be used for segmenting a full parking lot into separated parking spots, which can then be classified as occupied or empty (See Fig. 3). This will then allow there to be a clear display of the number of available parking spots in a lot to save customers time and frustration when parking their car!

Figures 1 & 2: Images of Current Sensor Systems [1][2]

2. Illustration / Figure

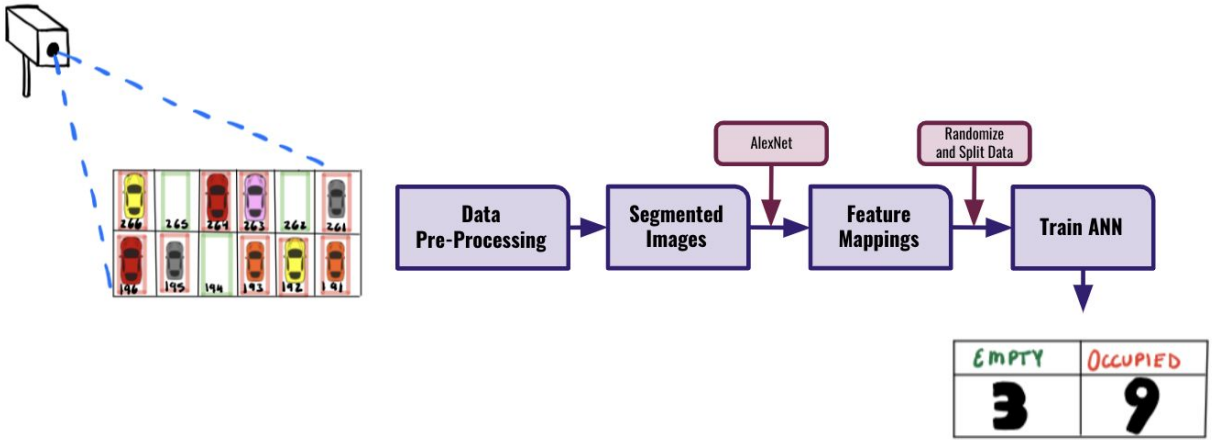


Figure 3: High-Level Overview of Model Architecture

3. Background & Related Work

3.1. PKLot – A Robust Dataset for Parking Lot Classification [3]

This paper classifies spots using surveillance systems, rather than expensive cameras (See [4] for an example of such a project). This reduces cost and avoids the need to send large data files wirelessly coming from these many sensors.

Their core method of classification utilizes two textural based classifiers - Local Binary Patterns (LBP) and Local Phase Quantization (LPQ). The paper explores the different accuracy results from training and testing on the same camera feeds, or training on one and testing on the other two. In general, LBP and LPQ achieved strong accuracy results wherever training and testing lots overlapped (90%+). Whereas when there were different lots used for testing and training, no accuracy above 90% was achieved.

3.2. Transfer Learning for Classification of Parking Spots using Residual Networks [5]

This paper classifies parking spots as occupied or unoccupied. In particular, they used the ResNet50 architecture as reference which uses an identity block and a convolution block as part of their network. Their project uses the PKLot dataset and data they collected at the University of Zilina.

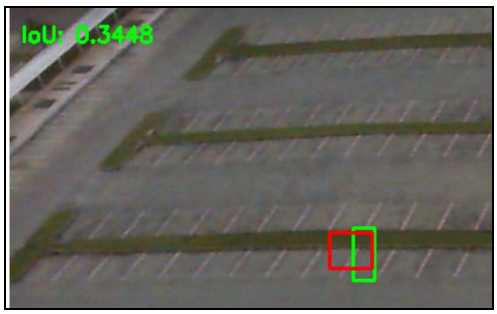
The datasets are composed of images of parking lots that have been segmented into cut-outs which show individual parking lot spaces which are labelled for whether they have a car present or absent. Their program tests the transfer of their PKLot trained classifier on their University of Zilina dataset. The team achieved an accuracy of roughly 99.92% on the PKLot dataset and 98.42% after the transfer learning on the dataset.

4. Data Processing

Our dataset consists of two parking lots, one with a single camera and another with two separate cameras. This data is sorted into three weather conditions of sunny, rainy, and cloudy. It is available as full lot images or cropped single spot images. To allow for flexibility, we wanted to design a system that could segment lots into individual spots. Additionally, because of the size of the dataset, we focused on a single camera to pull images from all three weather conditions. Each image had an associated XML file, which included the coordinates of every parking spot in the lot. We could have pre-cropped images using the coordinates, but this would reduce universality across other parking lots. We therefore then looked to find a way to segment the image ourselves.

We first created an intersection over union function that could calculate the accuracy of a predicted spot (See Figures 4 & 5). We wanted to segment lots using a model, but could not

assume one-to-one mapping for weights and coordinate labels (ex. First model output ≠ top left coordinate of first parking spot).




```
[ ]  def get_iou(bb1, bb2):
        assert bb1['x1'] < bb1['x2']
        assert bb1['y1'] < bb1['y2']
        assert bb2['x1'] < bb2['x2']
        assert bb2['y1'] < bb2['y2']
        x_left = max(bb1['x1'], bb2['x1'])
        y_top = max(bb1['y1'], bb2['y1'])
        x_right = min(bb1['x2'], bb2['x2'])
        y_bottom = min(bb1['y2'], bb2['y2'])
        if x_right < x_left or y_bottom < y_top:
            return 0.0
        intersection_area = (x_right - x_left) * (y_bottom - y_top)
        bb1_area = (bb1['x2'] - bb1['x1']) * (bb1['y2'] - bb1['y1'])
        bb2_area = (bb2['x2'] - bb2['x1']) * (bb2['y2'] - bb2['y1'])
        iou = intersection_area / float(bb1_area + bb2_area - intersection_area)
        assert iou >= 0.0
        assert iou <= 1.0
        return iou
```

Figure 4 (L): Graphical representation of parking estimation. Green represents the actual spot and red the predicted spot.

Figure 5 (R): Code that calculates 0.3448 prediction accuracy in Figure 4

We then explored sliding window object detection, which treats the classifier as a window that can be applied to any part of the image to locate the presence of a car. We defined a set crop size which was applied to the entire lot image as a grid [4].

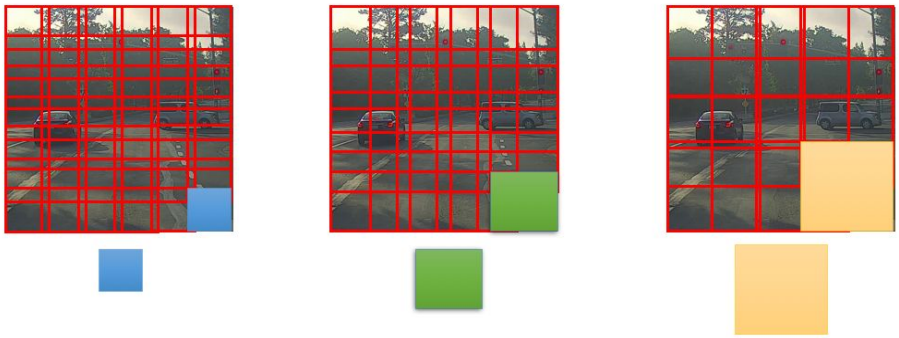


Figure 7: Example of sliding window sizes used to try and find cars located in this image [4]

This would count the number of cars present, making it easy to find the total spots available (available=total spots–#cars). The issues here arose due to image asymmetry, causing spots to often be sliced into random pieces.

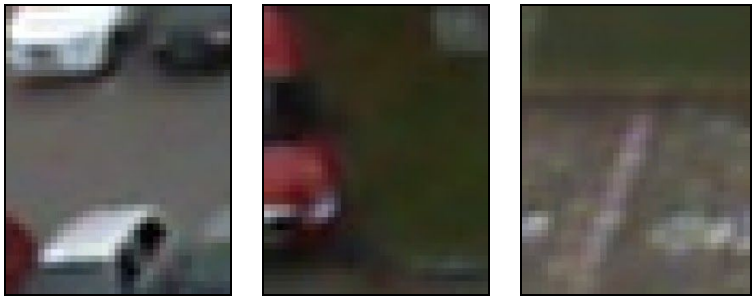


Figure 8: Segmentation results using the sliding window approach

Lastly, we looked at using computer vision to extract the parking lines on the images and to draw bounding boxes around each parking space. Using the OpenCV python library, we performed several operations, starting from thresholding the image to be binary, then denoising, and then identifying contour lines. These contoured lines could then be written on any image for testing.

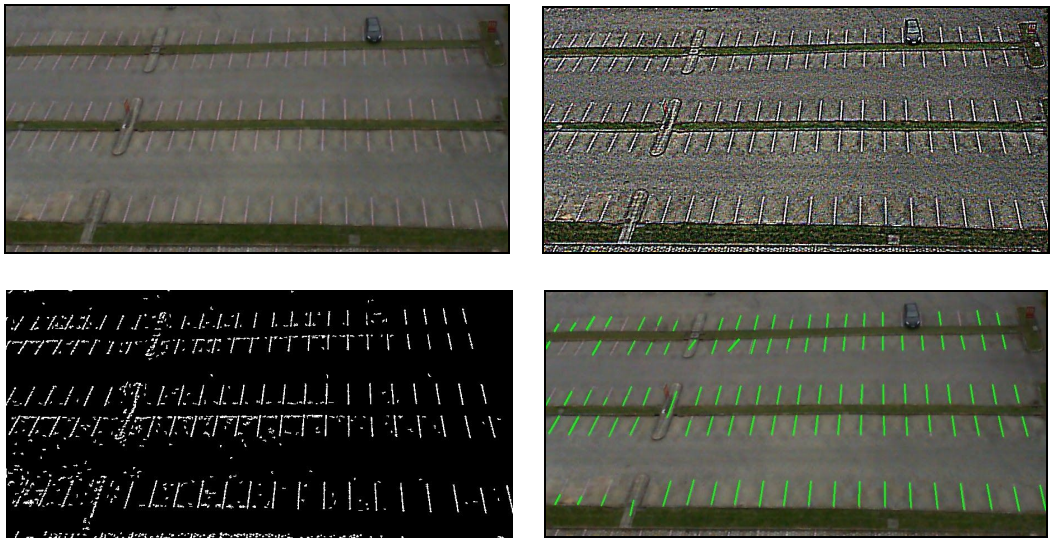


Figure 9: Processing steps to obtain contour lines: Empty Lot (Top-Left), Sharpened Image (Top-Right), Thresholding and Denoising (Bottom-Left), and Contour Drawing on Original Image (Bottom-Right)

After filtering by checking the line length and slope, the only remaining contour lines are those of parking spots. When each image is cropped, we also add a pixel value of 50 to a blank black image to account for regions cropped. By checking the amount of overlap between the region of interest and already cropped images on this mapping, we ensure no duplicate spots. While extremely effective, some parking lines were difficult to recognize with the contouring, so we account for these missed spots in our results.

The segmentation algorithm can be applied to any parking lot, which adds an important element of universality to our final product. Finally, we were able to gather a training set using these cropped images, classifying them into three categories - empty, occupied, and trash (ex. a tree).

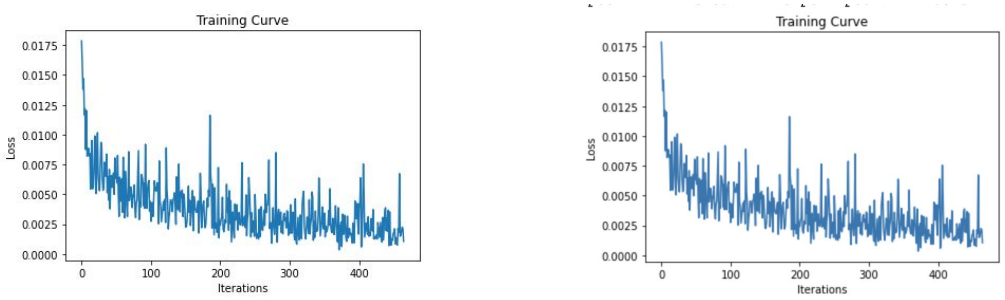
5. Architecture

At a high-level, our model takes as input our cropped images and classifies them as empty or occupied using a Convolutional Neural Network (CNN). To remedy some of the computational requirements of training a high accuracy model, we decided to use transfer learning. We used AlexNet to build feature mappings of our labelled dataset. Next, we randomized images, split the data, and trained an ANN model we built to classify images into the three classes we discussed. For our training set, we used 350 empty, 350 occupied, and 100 trash images. Sample images were then plotted to ensure they were all transformed to be the same size.

Figure 10: Plotting Some Images to Check Sizing

The ANN model takes an input with dimensions 256x6x6 and has two fully connected layers. Since AlexNet architecture already incorporates pooling and convolutional layers, we only chose to add two fully connected layers. While the architecture has no pooling layers, it has 300 hidden units to account for the non-linear separation of the data. The ReLU activation function was chosen because it yielded the highest training and validation accuracy. Finally, by implementing the Adam optimizer, we were able to account for momentum. This led to an improvement in accuracy of 6%. The hyperparameter choices and training curves for the final model can be seen below:

```
train(model, train_ft_loader, val_ft_loader, batch_size=64, num_epochs=15,lr=1e-4)
```



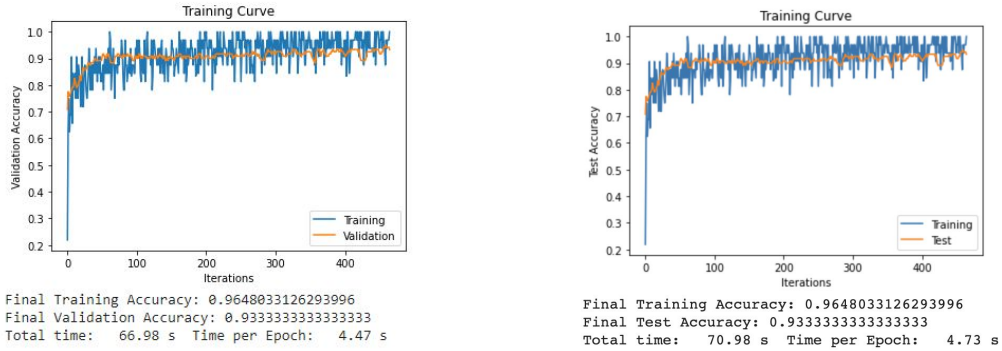


Figure 11: Hyperparameter choices and respective training and validation curves

6. Baseline Model

Since the task was image classification, the baseline model chosen was an SVM model. The model was trained on segmented images of spaces in the parking lot (segmented using the coordinates in a processed XML file) and flattened before being inputted into the SVM model. The model's regularization parameter (C-value) was then tuned to achieve the best testing result.

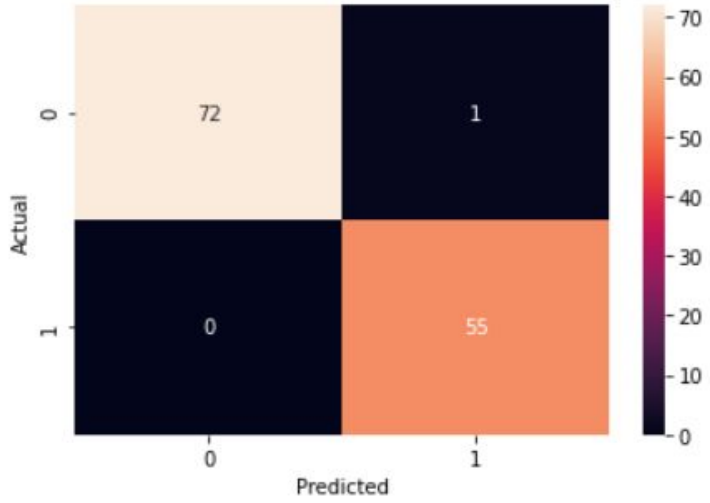


Figure 12: Confusion matrix for the baseline model's XML segmented image test results.

The baseline model was first tested on images segmented using given XML files. The model classified images as having a car present or absent. After splitting images into training and testing data, the model was tested and consistently scored > 99%.

While this first result was strong, after segmenting the images using our sliding window algorithm and manually organizing them, the accuracy of the SVM model was reduced. The accuracy was reduced to ~84%, which is less than the accuracy of our final model's accuracy of 98%.

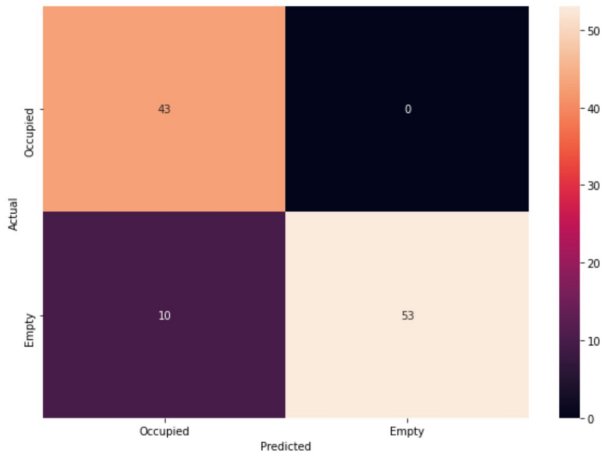


Figure 13: Confusion matrix for the baseline model's sliding algorithm image test results on the new data.

7. Model Evaluation of New Data

In order to properly evaluate the model's performance on new data, we obtained a sample that had not been used at any point for the development in the model. We show an image of the lot that was selected at random for testing. We apply our segmentation algorithm as discussed in the Data Processing section and have superimposed the resulting parking contours on our test image.

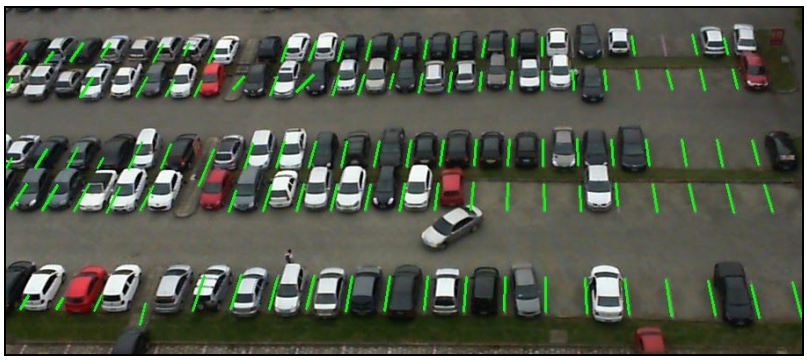


Figure 14: Parking contours written onto test image

After applying the segmentation algorithm on the image, it resulted in 106 individual images to be fed into our model.

Figure 15: Sample of segmented images from parking lot test image

Additionally, we obtain the overlap mapping to display all of the regions in our test image that have been cropped.

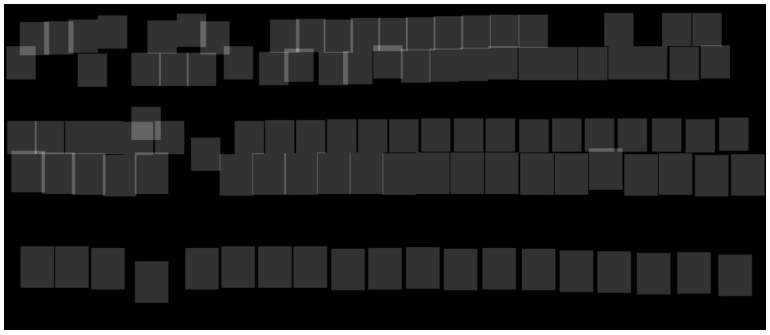


Figure 16: Overlap mapping of cropped regions

Finally, with the test image fully segmented, it is ready to be inputted into our model. The code outputs the number of cars identified and an image of each.

```
ALNC = alexnet.features
num_cars=0
for images, trash in iter(my_loader):
      imgs=ALNC(images)
      out = model(imgs)
      pred = out.max(1, keepdim=True)[1]

      if(pred==1):
        num_cars+=1
        image = images.numpy()
        # Plot the images in the batch - from sample code
        fig = plt.figure(figsize=(25, 4))
        ax = fig.add_subplot(2, 20/2, 1, xticks=[], yticks=[])
        plt.imshow(np.transpose(image[0], (1, 2, 0)))

      # print(pred)
      # image = images.numpy()
      # # Plot the images in the batch - from sample code
      # fig = plt.figure(figsize=(25, 4))
      # ax = fig.add_subplot(2, 20/2, 1, xticks=[], yticks=[])
      # plt.imshow(np.transpose(image[0], (1, 2, 0)))

print("total number of cars", num_cars)
```

```
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:12: RuntimeWarning: More than 20
  if sys.path[0] == '':
total number of cars 43
```

Figure 17: Code to classify segmented image and output number of identified cars

As shown in *Figure 17*, 43 cars were identified in the image and of the images shown, there were no empty spots misidentified as cars. These results are further analysed below in Quantitative and Qualitative Results.

8. Quantitative Results

After comparing the output with the actual image, we see that our model correctly predicted 43 out of 45 cars. That said, the model actually predicted all of the inputted images correctly. Because the segmentation algorithm missed two images of spots, two occupied images were never inputted into the model to be classified. These misses were labelled as false negatives.

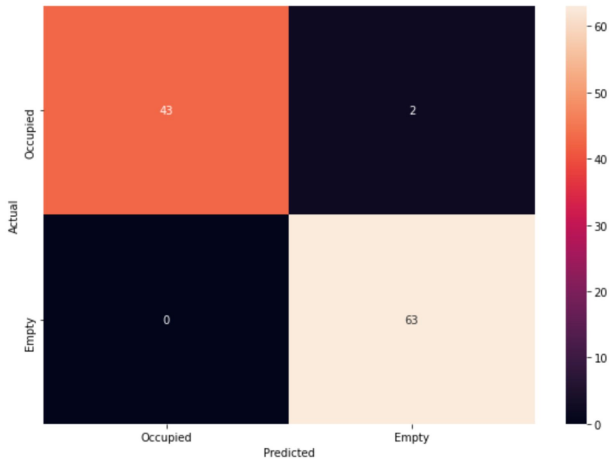


Figure 18: Confusion Matrix for Test on Unseen Data

Figure 18 shows that the model correctly predicted 43 cars (TPs) and 63 empty spots (TNs), It also shows that the model missed 2 cars (FNs) and not incorrectly labelling any spots as cars (FPs). Using these values, we calculate the following statistics:

$$Sensitivity = \frac{TP}{TP+FN} = \frac{43}{45} = 0.9555 = 95.6\,\%$$

$$Fall-Out = \frac{FP}{TP+FN} = \frac{2}{45} = 0.0445 = 4.5\,\%$$

$$Precision = \frac{TP}{TP+FP} = \frac{43}{43} = 1.0000 = 100\,\%$$

$$Specificity = \frac{TN}{TN+FP} = \frac{63}{63} = 1.0000 = 100\,\%$$

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} = \frac{106}{108} = 0.9814 = 98.1\,\%$$

It can be seen that the model performs extremely well on this unseen data, with an accuracy of 98.1%. Given that the model had a validation accuracy of 93.3%, this result is impressive. Moving forward, this test makes us optimistic about the flexibility of our model.

9. Qualitative Results

We can attribute the success of our model to the chosen architecture, including design choices like the optimizer and activation function. Furthermore, the pre-processing of the data played a huge role in the high success rate of our model. Proper parking lot segmentation is crucial for successful categorization of the parking spots and finding a strong method to perform this task was a major turning point in the accuracy of our model.

In addition to outputting the number of occupied spots in a lot, the model also outputs an image of each identified car:

Figure 19 : Pictures of identified cars from model output

While the model performed well, certain inputs were more challenging than others. Challenges included if the picture of the parking lot was not at an optimal angle, had slanted spots, or was not high quality. The lower quality of the image, the more data-processing and manual work required.

Furthermore, our model occasionally has difficulty differentiating between dark colored cars and empty spots. Depending on the angle and exposure of the input image, the model will occasionally mistake the dark colored car for pavement and not record it as an occupied spot. This leads to incorrect counting of occupied spots and can be corrected by increasing the exposure of images when they are input to the model.

10. Discussion

Our model's accuracy of ~98% is significantly higher than the SVM baseline's accuracy of ~84%. We credit this to the ability of the AlexNet CNN to process image "noise". The CNN's ability to interpret spatial features is a major strength and large reason for why it bests the SVM model. The SVM model and ANN model both process the 2D images into 1D vectors before being inputted. Had we used just an ANN, it's results would likely be similar to the SVM model.

Sensor System (Vendor)	Time Accuracy
Radar/Magnetometer (Fybr)	78%
Radar (Sensys)	98%
Infrared (CPT)	92%
Image Recognition (Cysen)	77%
Magnetometer (StreetSmart)	81%

Figure 20: Accuracy of Sensor System Methods Used to Classify Parking Spots as Empty/Occupied [6]

Our ability to produce results with the accuracy we did, given the timeframe we had, is very promising. Other methods of counting parking lot spots with comparable accuracies, like radar or infrared, are more expensive (see Figure 20). Furthermore, other models of equal cost and which also did image recognition, like Cysen, had lower accuracy.

Comparing our model to examples in literature also has promising results. The model from the paper in section 3.1 used the same dataset (PKLot), utilized textural based classifiers and was only able to achieve an accuracy of 89% when trained and tested on different lots [3]. Moreover, while the model used in section 3.2's paper achieved an equal accuracy to our model on new test data (~98%), their model was tested on images that were rigorously pre-processed, unlike our input images [5]. Since the quality of images was the greatest hindrance to our model, it can be interpreted that our result is superior (since it was tested on worse quality images, as a consequence of performing our own image segmentation). Segmenting the images was a real challenge but we found the best solution to be through the use of contouring as can be seen in Section 4.

The strong results on new data show the model's flexibility and demonstrates that it has immense potential with further improvements. We learned that the classification of an image is much easier than the segmentation of the image. Given good quality data, even an SVM model can be very successful. However, it is impossible to perform classification for a parking spot that "does not exist" according to the segmented images. Next steps for the project will focus on the improvement of image segmentation and data processing. This could entail looking at regions in the lot images that were missed to see if parking spots exist or even further tuning parameters in the segmentation algorithm.

Excluding data processing, two possible areas of future exploration include accommodating droplets that accumulate on the camera lense and processing images with snow. To deal with droplets from weather, the camera lense can be physically protected (either by where it is located or by protective equipment). For processing images with snow, it may be better that the model be solely trained on parking lots with snow and have it's weights switched from "snow" to "dry", based on a day's weather.

11. Ethical Considerations

To account for any ethical issues that may arise throughout this project, we discussed the possible impacts of using the model and how we will be collecting data. From the data collection perspective, we are ethically sound because the photos are only of cars, and not people. One possible ethical issue here would be with having personal data of license plates, but in the photos in the dataset, the license plate information cannot be seen due to the angle of the camera. As for the possible impacts of the model itself, we did not identify any possible ethical issues because there are no significant implied biases or severe effects for what we are designing it to do.

The metrics that were mentioned in class for measuring fairness; Demographic Parity, Equalized Odds, and Individual Fairness, do not apply for our model, but we have accounted for other causes of bias to ensure that our model is ethically sound. By checking for causes of biases that were discussed in class; Skewed Sample, Tainted Examples, Limited Features, Sample Size Disparity, and Proxies, we concluded that due to the lack of human involvement in our project, there are few ethical considerations needing to be considered.

12. References

[1] V. Ross, “Looking for Parking? Check Your Phone,” *Popular Mechanics*, 14-Nov-2017. [Online]. Available: https://www.popularmechanics.com/technology/gadgets/a6528/smart-parking-systems-steer-drivers-to-open-spaces/#:~:text=Smart parking systems don't,to Urbiotica spokeswoman Irene Compte. [Accessed: 15-Nov-2020].

[2] “Libelium integrates radar technology into its Smart Parking sensor,” *Traffic Technology Today*, 28-May-2019. [Online]. Available: https://www.traffictechnologytoday.com/news/smart-parking/libelium-integrates-radar-technology-into-its-smart-parking-sensor.html. [Accessed: 15-Nov-2020].

[3] Paulo R.L.de Almeidaa, Luiz S.Oliveiraa, Alceu S.Britto Jr., Eunelson J.Silva Jr., Alessandro L.Koerichbc. “PKLot – A Robust Dataset for Parking Lot Classification,*” Science Direct,* 01-July-2015. [Online]. Available: https://www.sciencedirect.com/science/article/abs/pii/ S0957417415001086. [Accessed: 15-Oct-2020].

[4] S. J. Han and J. Choi, “Parking Space Recognition for Autonomous Valet Parking Using Height and Salient‐Line Probability Maps,” *Wiley Online Library*, 01-Dec-2015. [Online]. Available: https://onlinelibrary.wiley.com/doi/full/10.4218/etrij.15.0114.0112. [Accessed: 15-Oct-2020].

[5] M. Gregor, R. Pirník, and D. Nemec, “Transfer Learning for Classification of Parking Spots using Residual Networks,” *Transportation Research Procedia*, 30-Jul-2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2352146519303527. [Accessed: 15-Oct-2020].

[6] B. Y. Cai, R. Alvarez, M. Sit, F. Duarte, and C. Ratti, “Deep Learning Based Video System for Accurate and Real-Time Parking Measurement,” *arXiv.org*, 20-Feb-2019. [Online]. Available: https://arxiv.org/abs/1902.07401. [Accessed: 09-Dec-2020].