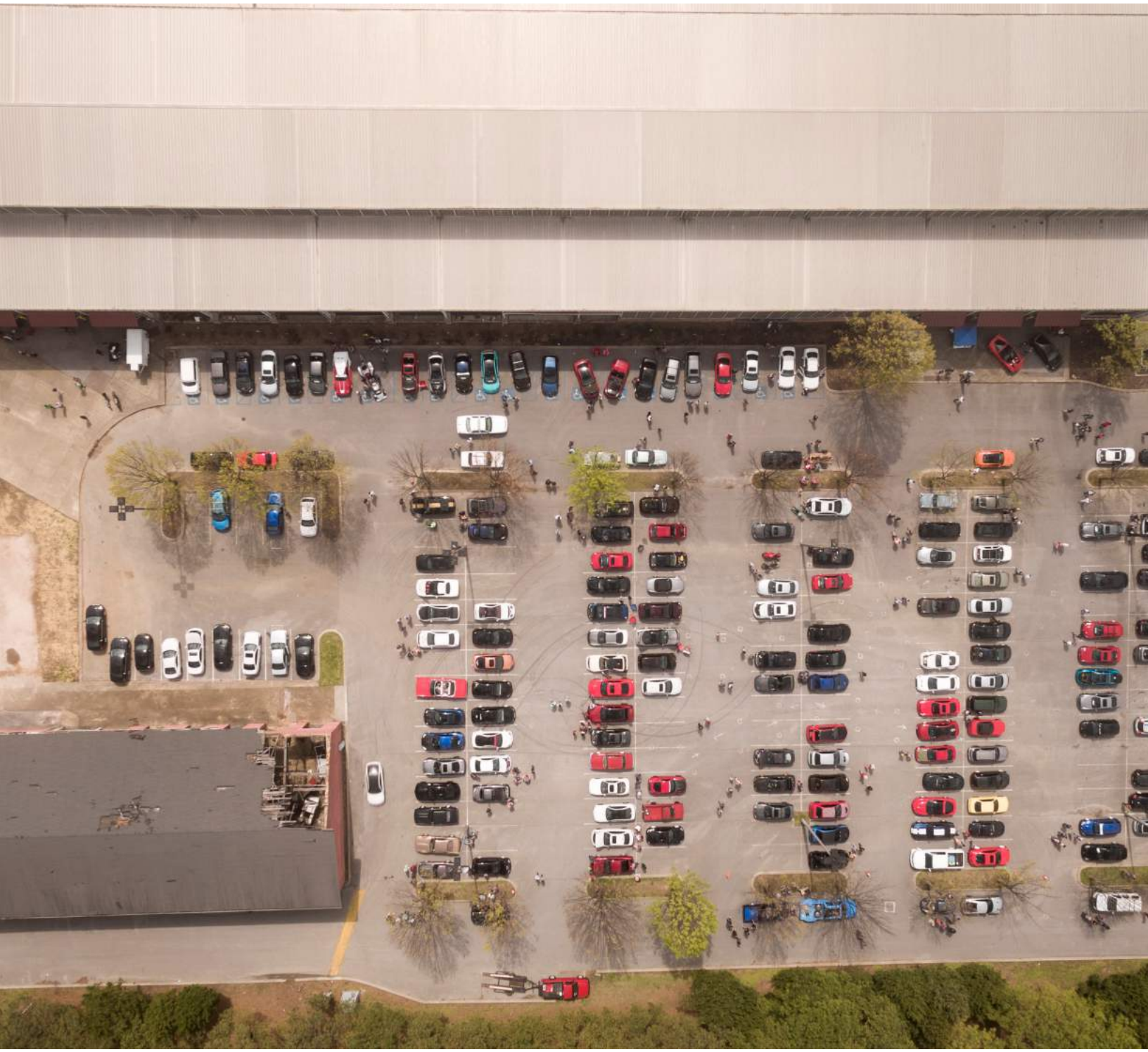
# SPOT OR NOT?
## PARKING SPOT VACANCY WITH MACHINE LEARNING

Jeffrey Jex - Final Capstone



# PROBLEM

It's hard to find a parking spot in busy places (like Jackson Hole's town square). It would be great to know how many spots were available on demand, with potential to keep some record.

Not only would this help travelers, it could be used by city planners to identify under used parking or areas that need more parking. Everyone wants to get a selfie with the elk antler arches, why make them wonder if town square parking is available?

This is 2021 - there's data for that.

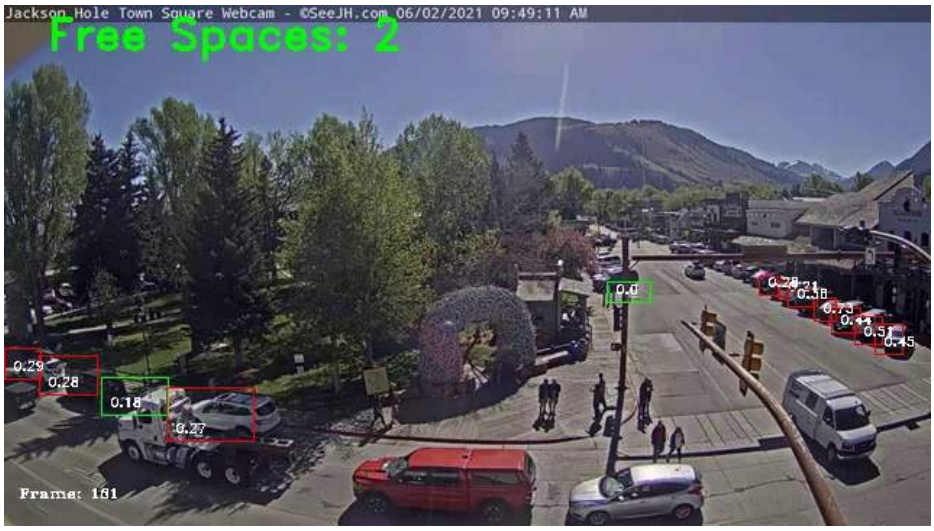
# OBJECTIVE

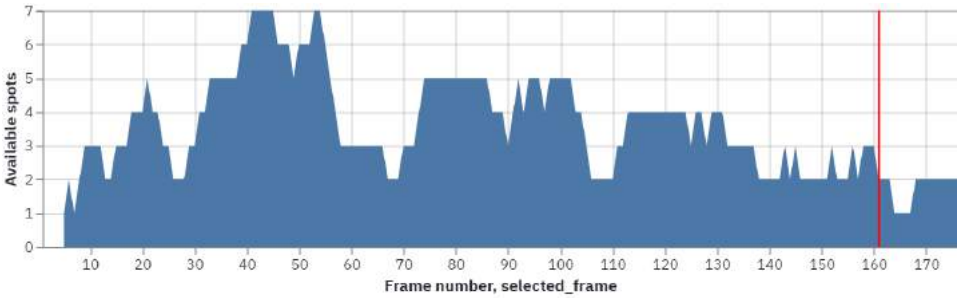
Identify vacant spots in live webcam data using machine learning. Specifically, the Jackson Hole Town Square camera.  This can be viewed as an unsupervised learning problem, with the potential for user feedback on incorrect labeling of parking spots to feed back into the mode to improve precision.

# OUTCOMES

To measure accuracy, I took every 10th frame of the 175 frame demo video clip (which covers both night and day) and counted parking spaces. Considering myself an expert at finding available parking spaces, I used the human count as "Truth" to compute the following confusion matrix and derived metrics.

Mean Avg Precision 0.57
Sensitivity (recall): 0.15
Specificity: 0.87
Misclassification: 0.11




**Sample of processed data, Frame 161, showing red occupied spots and green vacant spots. The number indicates how much of the spot is occupied by a car (0 = not occupied, 1= completely filled)**

# MODEL METRICS

This model gives zero false negatives. This is seen in the high specificity score of 0.87. This means that whenever a parking spot was marked as 'taken' it was always correct! The number of false positives indicates that the model often predicts spots as 'available' when they are not (see the not-so-great mean average precision score of 0.57.

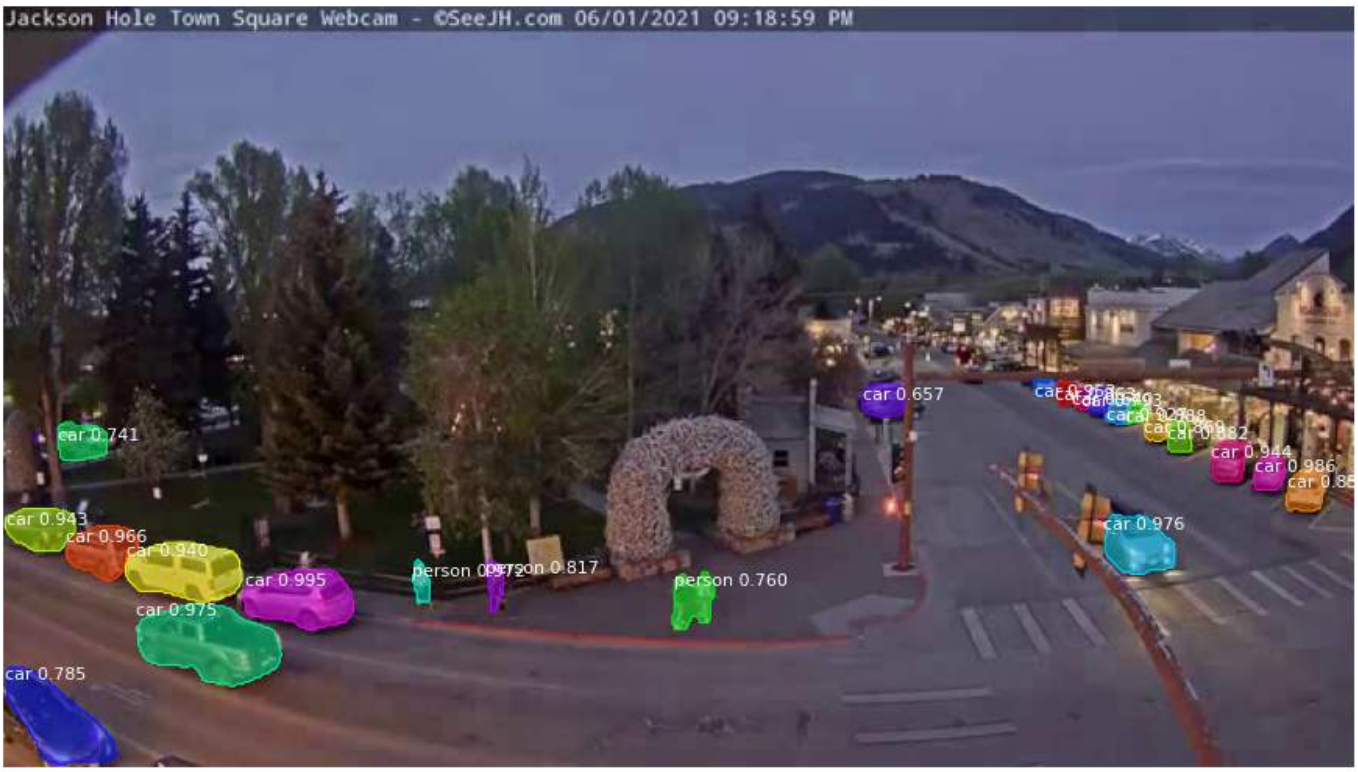
Positive: Vacant Spot
Negative: Filled Spot

| | | Actual Values | |
|---|---|---|---|
| | | Positive | Negative |
| **Predicted Values** | Positive | (TP) 36 | (FP) 31 |
| | Negative | (FN) 0 | (TN) 205 |

# MISCLASSIFICATION CAUSES AND SOLUTIONS

Common false positive causes and potential solutions:

- Parking spots that are very few pixels (in the back) were frequently misclassified. Use higher resolution video stream or ignore those spots. **This could bring the mean average precision up to 0.84, a significant gain.**
- Black or white cars were often misclassified. Higher resolution, or more specific model training to identify those types of cars would probably help.
- Temporary occlusion of spaces (such as a garbage truck or bus driving in front of them) caused the model to think spaces were 'available' because the space was blocked by something it didn't recognize as a car. This could be fixed by waiting a few extra frames before declaring a spot as 'vacant' to make sure its really vacant. This code is already implemented in the 'live demo' version, but wasn't used on this set to make it clear exactly what the algorithm is doing.




**Jackson Hole town square with instance segmentation masks from Mask R-CNN. labeled on each detected instance are the classification (person, car) and confidence of classification (between zero and one)**

# ADDITIONAL USE CASES

Data on how full a parking lot is can also be used for:

- Vehicle routing - send vehicles to open spots via navigation app or digital signage.
- Proxy for number of customers visiting a business over time.
- Proxy for how busy an area is - for example: shopping malls, airports, conference centers. Could be used to inform visitors before attending.
- Identify under utilized portions of parking lots to flag for repurposing.
- Identify commonly used 'parking spots' that aren't allowed (parking illegally).

# METHOD

I used Mask R-CNN (Region-proposal Convolutional Neural Network), originally produced by facebook research (He et al. 2017). This algorithm is used to process each video frame to identify cars and trucks. If a car or truck's bounding box overlaps with a designated parking place, that spot is considered occupied. If not, the spot is vacant. I used an updated version that works with TensorFlow 2 (see repository here). To save time training, I used the well documented Matterport model weights, trained on the COCOs dataset.

## Step 1: Identify parking spots

- Using a video clip during 'rush hour' when all spots were filled, and employed Mask R-CNN to detect vehicles. This identifies all cars, and assumes they were parked. Using a second frame from the clip a few seconds or minutes later (still with all spots full), I detect with Mask R-CNN again. Any vehicle bounding boxes from the first frame that were still full in the later frame were considered to be parking spaces. Any boxes that weren't full were considered as noise (i.e. a car driving on the road). An initial implementation of this is shown by Geitgey (2019).
- Methods used by other workers have included:
  - training a specialized neural network to identify empty (or full) spaces (Amato, 2021).
  - Looking at painted lines (Jackson Hole's town square's prime spots are parallel parking that don't have any lines). (Dwivedi, 2018).
  - Have somebody draw boxes manually, or map it with a drone

## Step 2: Determine if spots are full

- I compare the parking spot bounding box with all the cars bounding boxes. If there's a significant overlap, the spot is considered full.
- This is done using IoU - intercept on union. IoU compares how much of the car's box overlaps with the parking spot box. If it's above a threshold, I count the spot as being occupied. Side note: IoU is used internally by R-CNN networks to give one bounding box per instance and avoid double-counting objects. This means there's already great functions to compare if two boxes overlap significantly or not.

# CHALLENGES

- Most parking lot cameras are not situated at a very good angle for detecting available spots. They are only ~10 feet off the ground and the first line of cars obscures all the other parking lots. The higher up and the more 'in-line', the easier it is for the algorithm to tell if spots are there. The Jackson Hole camera offers several different types of parking, so it's a decent test.

- It took 10% of my time getting the image detection model up and running on my laptop, 40% getting code to reliably and smoothly pull youTube clips and 50% getting it working in streamlit and deployed streamlit sharing. On the bright side, streamlit's servers are much faster than my little laptop

# FUTURE WORK

**Algorithm and Statistics**

- Try using a higher resolution stream to capture cars farther from the camera, and add in blurring faces of pedestrians (for privacy).
- Implement YOLO algorithm, which is faster but less accurate than R-CNN. Since I'm not currently using the mask, there's no good reason to spend extra processing time obtaining it.
- More samples on model accuracy vs. human counting open/filled spots.
- Try cnrpark's training images for open/vacant parking spots and see if it performs better than car detection with IOU (Amato, 2021).
- Try on other parking lots (such as Jackson Hole's airport).
- Currently struggles with black cars or trucks, try re-training the model, or using higher resolution stream to resolve this.

**App Features**

- Ability for user feedback on if spots were correctly identified as vacant or not
- Add options to run on any video:
  - Enter your own video URL or upload one
  - Download parking place bounding boxes after processing
  - Upload saved parking place bounding boxes
  - Download processed video
- Store 'parking lot vacancy' data to glean insights about location.

# REFERENCES


- Acharya, Yan. Real-time image-based parking occupancy detection using deep learning. http://ceur-ws.org/Vol-2087/paper5.pdf
- Cazamias, Marek, 2016. Parking Space Classification using Convolutional Neural Networks. See paper. They use a single image to count parking spots. Stanford paper
- Amato, Giuseppe; Carrara, Fabio; Falchi, Fabrizio; Gennaro, Claudio; Vairo, Claudio. Accessed June 2021, CNR park: http://cnrpark.it/
- COCO dataset, used to train Matterport model: http://cocodatas
- Dwivedi, Priya, 2018, Find where to park in real time using OpenCV and Tensorflow
- Geitgey, Adam, 2019, Snagging parking spaces with mask R CNN and Python.
- et.org/
- He, Kaiming; Gkioxari, Georgia; Dollár, Piotr; Girshick, Ross. 2017, Original Mask R-CNN paper by facebook research. https://arxiv.org/abs/1703.06870