Applied Machine Learning Intensive (AMLI), Summer 2021
Capstone Report

UARK CAR DETECTION AND CAR TYPE IDENTIFICATION

July 30, 2021

Trinity Colbert, Industrial Engineering, tkcolber@uark.edu
Giancarlos Cabrera, Electrical and Computer Engineering, gcabrera@uark.edu
Marvin Violantes, Computer Science, mgviolan@uark.edu

1 Abstract

With a new parking verification system being implemented at the University of Arkansas, Team 4 got the idea to use machine learning and deep learning to create a system that would help the parking department. This system would be able to detect objects as cars, the car type, and the car color with the use of an API. Later on a license plate recognition system would be integrated into this system. The way data was collected was by going to multiple different streets around the University of Arkansas and recording cars driving by. This video was then sent to the model to output a new video showing bounding boxes and information about the vehicle. After running multiple test we found out that certain situations work better than others, allowing for better or worse results. This system is on the right track, but still has more to build upon.

2 Introduction

Team 4, "Universal Encoders", has chosen to do the UARK car detection and car type identification project. The University of Arkansas is transitioning their parking verification methods from having a sticker in the front windshield to plate detection. This parking permit has to be verified because if a person is parked in the wrong location, they will get a warning or a ticket. To verify a permit, they will now have to match the car description to the licence plate to the correct parking lot. If the University of Arkansas decides to verify each car without the help of a computer, this would be a lengthy and inaccurate process due to the inefficiency of human intervention. Going through with this method would result in a plethora of errors and wasted manpower since someone would have to be delegate to operate this task. The employee tasked with making sure every plate matches the correct description could easily misread the plate or description, which would cause an unfortunate

chain of events. This is where our car detection project comes into play. With the ability to have a program that runs a camera through parking lots and garages, gathering the type and color of the vehicle; parking enforcement will now be able to quickly match the plate to car description. This would greatly reduce errors, but not completely eliminate errors because there could be some flukes. Our goal of this project is to develop a working program that can seamlessly identify an object is a car, along with the make and color of that car. In the future our goals would be to completely complete the verification system. We would start by making sure the program can work with vehicles not moving and the integrate the license plate match making system.

2.1 Team Roles:

- Project Manager: Trinity Colbert
- Programming Manager: Giancarlos Cabrera
- Progress Analyst: Marvin Violantes

3 Data Analysis

The data input for this project is video recordings of cars around the UARK campus. The output will be that same video, along with results, if a vehicle is detected. If so, it will demonstrate the type of vehicle it is, and its color. For example, it would be able to identify the difference between a truck and a SUV and then tell the user what color each vehicle is. For our first method, we gathered footage of cars traveling down MLK Blvd, using an iPhone 11 and the duration was fluctuated between about 30 seconds and 1 minute. This method ensures we capture a random variety of vehicles with multiple colors to properly test our program. Another method we could use if we were to integrate this program into

the parking garage is to ask the University of Arkansas if we could borrow footage from previously installed parking garage cameras around campus. This would reflect a more true depiction on how our program is meant to be used. Due to time restrictions, the first method was the only one that was adapted.

After we collected our data, we directly inserted it into our program to test the versatility and compatibility of our code. We made sure every video was recorded on the same camera and resolution to avoid any inconsistency. The video was recorded handheld by Giancarlos throughout the day time at different locations. The initial time included was during lunch time since this would be when the roads are the most crowded. Other times included 1 hour after lunch and then 1 more hour afterwards. The locations included M.L.K Jr. Blvd, and two different sides of Garland Ave.

4 Project Implementation

The idea of the project is to create an efficient program that can analyze an inputted video and correctly identify each vehicle's type, their general design color, and the amount of cars that have traveled within each frame. Utilizing these tools allows us to successfully resolve the problem we are addressing with our program's implementation. When contextualizing our project with the University of Arkansas' new parking transition, the key factors that will come into play are the detection of the vehicle's classification and its model's color scheme. This will create the foundation for a reliable system that can be used to ensure the accurate matchmaking between plate identification and vehicle detection for a more efficient method of permit verification. Our model covers the latter aspect of the project's purpose in the realm of observation. This is due to the fact that it's dealing with information gathering from the chosen method of transportation's appearance. Since this is only a baseline, there are a multitude of traits that can be implemented to allow the program to be integrated on

a broader spectrum. In our situation, plate identification would the trait that would need to be incorporated, though with the time-frame we were given, this was not possible.

When running the program with a video input, the following process will occur and the described output will be shown. As any vehicle that has been represented within the training dataset is within frame of the video, it will immediately be wrapped in a bounding box with its color and vehicle type predicted at the ceiling section of the box. There is a probability percentage given to represent the program's assurance of its prediction. The way the program accomplishes this is by utilizing an API titled "Vehicle Detection, Tracking, and Counting". Through the use of this program, videos can be analyzed to provide the necessary information we need to approach our goals. On a conceptual standpoint, the way the program works is by taking in a video and breaking it down it frame by frame for further investigation. This has been done through the use of an open source machine learning software library called OpenCV.

For our project, we inputted 1080p videos recorded at 30 FPS consistently throughout each data collection. The reason we chose 1080p is because going through more pixels in a frame takes more time to compile so we didn't want the highest quality video choice available to us. However, we still wanted a high enough quality option so that the program has plenty to look through and allow it to be more accurate in its predictions. Since our recording devices offered between 720p, 1080p, and 4k, it was decided that the middle option was the moderate choice. As for the use of 30 FPS, since we want our model to be efficient, the analysis of too many frames can result in an exponentially higher runtime as the length of the video increases. While using a video at 60 FPS could be beneficial for our accuracy, we felt like it could be sacrificed for a higher pixel count per frame.

Once a frame from the video has been obtained, it will then be implemented into the TensorFlow-operated deep-learning model that is associated with the aforementioned API.

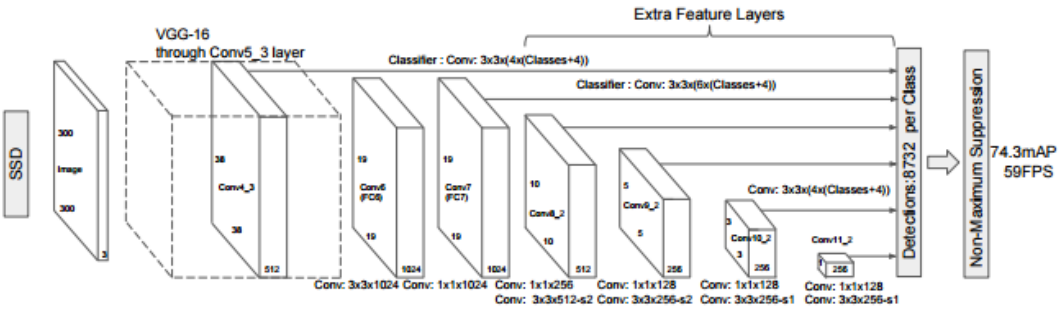

Figure 1: How The Model Layers Its Detection From SSD

The model takes the frame and starts to gather detections from it using the pretrained dataset that was provided from the API. This creates the bounding boxes around the cars but does not give any vital information about the object within the box. Once the detections are made, the frame will then go through a classification phase where the objects' information within the bounding boxes are predicted and displayed above. The two categories that the classification phase accounts for is the color and vehicle type. The API is designed to recognize several shades of black, blue, green, orange, red, violet, white, and yellow. As for the vehicle type, it can detect cars, motorcycles, buses, trucks, and even boats. There are other predictions that the program can make that are not related to our project, but they include anything from airplanes to parking meters to animals to people. Figure 1 provides a visualization as to how an image is derived from the SSD and is constantly put through several layers to pin point detections to output our desired results. When it comes to counting the amount of vehicles, there is a vertical ROI Line that is drawn at the right-end of the screen and acts as a detection-line. When a detection crosses the line, it will take the coordinates of its location and cross reference it will the x-position of the ROI Line. If the x-coordinates are greater than or equal and lays anywhere on the y-axis to the ROI Line, it will consider the object to have passed the line and will add to

the counter positioned to the top-right of the screen. The final image is then inserted into a new video which becomes compromised of each frame from the input video after going through the following process. Figure 2 demonstrates a general depiction as to how the video was processed along with the methods in which the end result's characteristics were found through a horizontal flow chart.

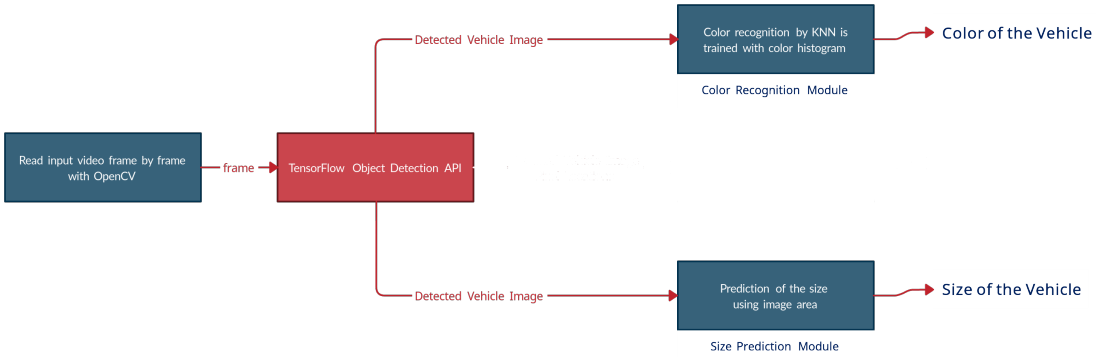

Figure 2: Conceptual Flow Chart of Vehicle Detection

While going through the process of submitting the data through the program, there were several roadblocks that were encountered. To begin, our understanding of pretrained models was not refined enough to fully grasp their mutability. Since this is one of, if not, the first time the members of our group have been exposed to utilizing pretrained models, our knowledge on how we can utilize its features was very limited. This resulted in all of our members to be under the impression that a pretrained model could not be trained to improve its accuracy. This was found to not be the case. Upon communicating with several people, it was discovered that we should have annotated the data. Essentially, annotating data would have allowed the model to receive pre-classified inputs so that it can learn from those patterns. While we were able to overcome a misunderstanding on our behalf, it costed us several hours being spent trying to reach this understanding then actually implementing

the annotations to our raw data. This leads to our second issue that we ran into, time to implement the annotations. Due to our lack in training for our model using annotations, our model's accuracy was significantly impacted in a negative way. We were capped at the accuracy that was available with the provided API. While we were not able to overcome this issue, we tried our hardest to modify the API as much as we could to reach the most accurate results possible. Apart from this, a difficult turnstile that we had to hop over was the fact that large aspects of the code were hardcoded. This means that the code was designed around a specific input rather than be able to handle a variety of inputs. The issue was not too hard, but it was a very tedious and time-consuming process to locate every aspect of the code we needed to modify so that it correlated correctly to our input.

5 Experimental Results

While the model does a good job detecting the type of vehicle that drives by and its color, there are also situations where the model doesn't perform the best. For example, Figure 3 depicts an instance where a singular car is being detected as both a green car and black car, both with relatively high certainty percentages. Since one car can't be two different cars with different colors, this error shows one of the many insecurities of our program. Both predictions were wrong because it is a gray car that is being shown in the picture, With more time, these imperfections could be alleviated through further training and testing.

5.1 Qualitative Results

After we tested our videos we learned a few new things. First our model did not work well with M. L. K. video. It had too many cars for the model to handle. It would miss a few cars or completely identify it incorrectly(Image shown below missing a vehicle). The model performed better on the Garland video. Since there were less vehicles it was able to get

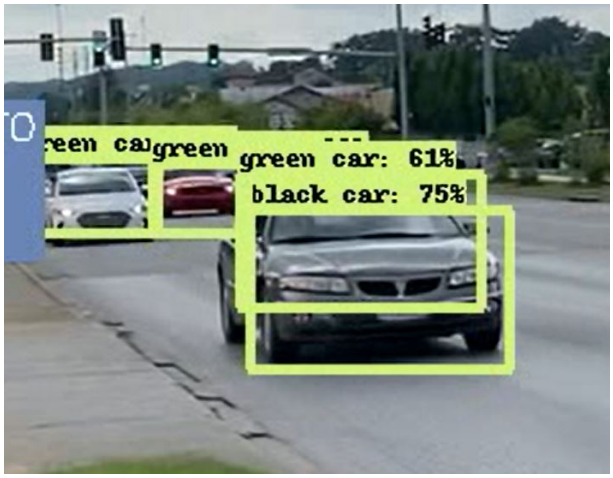

Figure 3: Counting Two Vehicles in One Car

Figure 4: Another Case of Counting Two Vehicles in One Car

Figure 5: Showing that the model missed a vehicle

all the vehicles that were shown in the video. This was the same result in the other side of Garland. After running the model, a CSV file is given that shows the car type and car color and it is visible that some of vehicles are not being recorded.

Vehicle Type/Size	Vehicle Color
car	black
car	white
truck	red
car	blue
truck	white
car	red
car	green
car	white
truck	blue
car	green
car	red
car	black

Figure 6: CSV file that model gives

5.2 Quantitative Results

In addition to our visual results, we also calculated some statistical results of our model. The detection accuracy was calculated by comparing the actual number of cars that should have been detected and the experimental number of cars that were detected by our program. We used the same logic for our classification accuracy except we are evaluating the validity of the description of the car. We then calculated a percent error deviation for both detection and classification of our final model using the actual and experimental statistics collected from our output.

	Detection	Classification
Accuracy	0.67	0.83
% Error	0.33	0.083

Figure 7: Accuracy Metrics

6 Conclusion

The purpose of the project was to provide an efficient API to a discrepancy that can be found in the University of Arkansas new paid parking system. With the transition of windshield stickers to tracking license plates, there opens the potential to mismatching the license plate with the wrong vehicle. It is important because this could prevent students/faculty from getting incorrectly charged with a ticket or further legal action. A vehicle detection model was utilized to gather the key characteristics to accurately identify a vehicle in real time. Once our program is given an input video, it breaks it down frame by frame, processes a frame through the deep-learning model, wraps each identifiable object in the frame with a bounding box, classifies and counts said objects, then adds the modified frame into an output video which is exported once every frame is complete. In terms of improvements, we

could have incorporated the method of annotating our raw data to gain a higher accuracy overall. As for future works, we could have explored other API's with a more general coding dynamic, rather than it mainly possess hardcode.

7 Acknowledgment

I want to thank my advisor, Dr Thi Hoang Ngan Le and our TA viet-Khoa Vo Ho for the guidance and patience throughout our project. I appreciate the financial and educational support from both google and NACME as this project has advanced us one step further to our future Engineering careers.

References

Main Advisor—— Dr. Thi Hoang Ngan Le: University or Arkansas

TA—— Viet-Khoa Vo Ho

Michele Lezama: NACME

Bryan Hill: University of Arkansas

Dr. Baker: University of Kentucky

Dr. Allen â Blanchette: University of Kentucky

Dr. Grant: University of Kentucky

Dr. Rainwater: University of Arkansas

Dr. Zhang: University of Arkansas

Dr. Lou: University of Arkansas