Proceedings of the 2020 Annual General Donald R. Keith Memorial Capstone Conference
West Point, New York, USA
April 30, 2020
A Regional Conference of the Society for Industrial and Systems Engineering

West Point Autonomous Vehicle Research and Design: Convolutional Neural Networks and Image Classification

Edward Kang[1], Sean Min[1], Will Anderson[2], Reed Burton[2], Jarred Fassett[2], Samuel Pool[2], Mitchell Stiffler[2], Preston Draelos[3], Chris Little[3], Austin Morock[3], Courtney Razon[1] Daniel Gonzalez[2], Kathryn Pegues[1], and Joseph Cymerman[3]

[1]Department of Systems Engineering
[2]Department of Electrical Engineering and Computer Science
[3]Department of Civil and Mechanical Engineering
United States Military Academy, West Point, NY

Corresponding Author: Edward.k.kang.mil@mail.mil

Author Note: The authors of this article participated in a year-long senior research project in the Departments of Systems Engineering (SE), Civil and Mechanical Engineering (CME), and Electrical Engineering and Computer Sciences (EECS) at the United States Military Academy (USMA). The research underpinning this paper was conducted under the supervision of Dr. Daniel Gonzalez, Major Courtney Razon, Lieutenant Colonel Kathryn Pegues, Major Joseph Cymerman, Major Mark Lesak, Dr. Peter Hanlon, Mr. Dominic Larkin, and Mr. Nicholas Livingston. Upon graduating in May 2020, the authors commissioned into the United States Army as Second Lieutenants, serving in the Aviation, Cyber, Engineer, Field Artillery, Infantry, and Military Intelligence branches.

Abstract: Technological development within the domain of autonomous driving systems remains an important area of focus for government, industry, and academia. Large corporations are continuing research and development in unmanned delivery platforms; this technology holds the potential to substantially decrease costs and increase efficiency in supply chain management. For the military, autonomous systems show the potential to increase capability and improve Soldier safety. The West Point Autonomous Vehicle Research and Design (AVRAD) team developed an autonomous system, capable of competing in the 2020 Intelligent Ground Vehicle Competition (IGVC)--Self Drive Competition. IGVC leadership decided to cancel the 202 IGVC due to the COVID-19 pandemic. However, for the past 27 years, this international competition challenged teams to think creatively about evolving technologies of vehicle sensors, robotics, and system integration. During the event, teams race their autonomous systems through a course which tests the entry's lane following ability and ability to avoid obstacles. While driving, the system builds a map of the immediate area by using data gathered from three front facing and two rear facing Mako G-319C cameras. Additionally, a Velodyne HDL-64E LiDAR senses and identifies each type of obstacle or critical image such as a stop sign. The images gathered are relayed to a Convolutional Neural Networks (CNN). The CNN algorithm classifies each image as a specific item, whether it is a traffic sign, person, or an entirely different obstacle. Finally, the vehicle management system uses the CNN's classification output to map the immediate surroundings. With an established map, the system then determines the most appropriate action or pathway. This paper will explain, in detail, how the AVRAD team selected, tested, and improved a CNN capable of quickly and accurately detecting and recognizing common street side objects along its path.

1. Background

The Department of Defense (DoD) remains interested in evolving robotic technology as it has the potential to increase soldier safety in volatile, uncertain, complex, and ambiguous environments. The U.S Army's Ground Vehicle Systems Center (GVSC), the leading development agency in robotic development in the Army, develops, integrates, and sustains technological solutions for manned and unmanned ground vehicle systems. To incentivize research in this domain, the GVSC sponsors the Intelligent Ground Vehicle Competition (IGVC—Self Drive) annually in Rochester, Michigan.

The focus of IGVC—Self Drive is to encourage teams from various undergraduate and graduate programs to gain experience in developing autonomous systems capable of navigating urban environments. The West Point Autonomous Vehicle Research and Design (AVRAD) team's physical system is a Polaris GEM E2 vehicle outfitted with off-the-shelf components capable of achieving autonomous driving. With a functioning vehicle, the team focused achieving a critical component of autonomous driving- the ability to detect, classify, and react to obstacles.

To achieve this, the team developed a software architecture that enables the system to recognize environmental obstacles. This paper will explain, in detail, how the AVRAD team selected, tested, and improved a Convolutional Neural Network (CNN) capable of quickly and accurately detecting and recognizing common street side objects.

2. Design Overview

The system's ability to visually detect images, classify images, and respond accordingly is critical in ensuring the system runs at a smooth rate in which the vehicle can appropriately respond to its kinetic environment. To achieve this, the system requires the use of a CNN. The CNN must classify the objects quickly and accurately to allow the vehicle management system to respond to a changing external environment. The system uses a scanning window box technique to view the entirety of an image. This efficient technique allows the CNN to further identify all items viewed in the box, rather than focus on specific regions of an image. The image is then run through the CNN to identify individual objects such as people, road signs, road obstacles, street lanes, and any other objects associated with driving. Next, this information is sent to the vehicle management system to determine the appropriate action.

2.1 Mechanical Design

The AVRAD system is composed of one Polaris GEM E2 vehicle with an autonomous sensing package including one Velodyne HDL-64E Light Detection and Ranging (LiDAR), five Mako G-319C cameras, one AStuff Spectra computer, one XSENS MTi-710 GPS/IMU, and one Multiplexed Vehicle Electrical system (MVEC). The Mako G-319C model camera serve as the "eyes" of the system with a 37.5 frames per second and a resolution of 2048x1544 (Gige Vision). The Velodyne LiDAR which uses a pulsed laser to determine distances works in conjunction with the five cameras to give the system a whole picture view of its immediate surroundings.

Figure 1. Polaris GEM E2 with LiDAR and Mako Cameras
Left to Right: Polaris GEM, Mako Camera, LiDAR

3.1 Software Design

The AVRAD team developed a program which interconnects the hardware systems on an operating system known as Robot Operating System (ROS). ROS is the primary platform in which information from various sensors and programs communicate. This information is compiled into "nodes." ROS then interconnects these nodes so that information can be passed along to different programs that enable the vehicle to function as a system.

In the case of image classification, AVRAD uses machine learning within ROS to accomplish this task. Without specific programming, machine learning uses algorithms and statistics to accomplish specific tasks, such as classifying a given image. In the context of image classification, machine learning is advantageous because it allows the vehicle to make educated—and mostly correct—"guesses" regarding what a camera detects without software programmers having to individually code each object onto the vehicle's computer. However, the CNN must be trained with previous, known data, allowing the network to "learn" how to differentiate between objects. After the neural network is properly trained, the program will output the correct identification of the object the camera detected.

3. Image Classification

A CNN is a specific type of neural network classifier that works well with images. Training a CNN takes a significant amount of time due to the requirement of inputting images into the computer and continuing until the margin of error is reduced to an acceptable confidence level. The amount of overhead work required to personally train the data would not have outweighed the benefit of using a pre-trained CNN. Thus, the AVRAD team made the design decision to use a pre-trained CNN on the AVRAD system.

3.1 Convolutional Neural Network Explained

The CNN enables the system to identify obstacles in the system's immediate surroundings. The means of classifying images into known objects assists the bigger mission of the system being able to react to different objects it encounters. This section of the report will explain how the CNN classifies and identifies a given image.

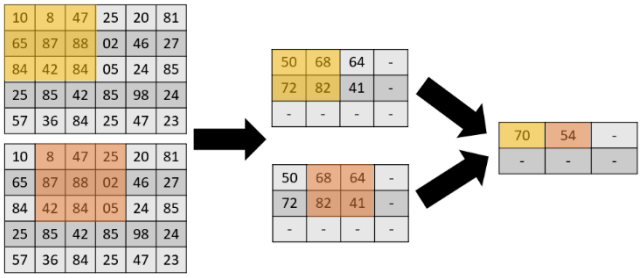

Figure 2. 2-Dimensional Bounding Boxes

The first step of classifying images occurs as the Mako cameras send images to the CNN. The neural network then scans the image by utilizing bounding boxes. Bounding boxes isolate the whole image by using predetermined equations to create a zoomed-in, cropped image. The cropped image is assigned a numerical value that represents the larger image, as shown in Figure 2. For instance, the highlighted far left 3x3 yellow entities are reduced to a 2x2 matrix. Furthermore, the 2x2 matrix is then further reduced to a single entity shown on the far right of Figure 2. The same concept applies to the highlighted red entities in Figure 2. Then, the CNN uses logistic regression and scans through the bounded boxes to predict an "objectness" score (Redmon & Farhadi, 2018). If a set of pixels and values that result from the bounding boxes resemble a known object, the CNN will assign an "objectness" score close to 1. When the neural network completes scanning and reducing the image from the cameras via bounding boxes, the CNN assigns an "objectness" score to the image. Those images with an "objectness" score greater than 0.5, which is the CNN's threshold of believing the analyzed image is indeed the known object, are classified and identified as such. For example, if the CNN recognizes a set of red pixels and upon assigning an "objectness" score of 0.75, the CNN will label the image from the camera as a potential street sign.

One important characteristic for the CNN is that the program utilizes multi-label classification. This is to account for overlaps in categories when a bounding box detects an image. For example, when the CNN identifies a female person in front of the vehicle, the CNN will label the image within the bounding box as "Human" and "Woman." (Redmon & Farhadi, 2018).

After the image is isolated into bounding boxes and the image within is given a label, the CNN uses convolutional analysis in determining whether the image contains an object that corresponds to its assigned label. Figure 3 shows how a CNN uses convolution to identify and classify an image. The top three images on the figure above show the overlap of two equations x(t) and h(t), which are a straight horizontal line and a parabolic curve, respectively. The bottom image in the same figure shows the convolution over time. As the overlapped area, denoted by the green highlighted section, between the two equations increase, the bottom image shows that the convolution-value increases as well. Lastly, the highest peak of the convolution-value is reported, which in the case of Figure 3 is represented by t_4. If the convolution-value passes a threshold of 0.7, the suspected object within the bounding box is finally classified as what the CNN believes the image to be (Redmon et al., 2018).

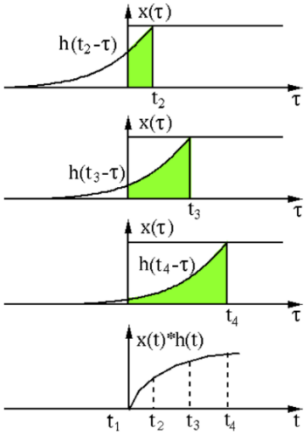

Figure 3. Convolution Over Time (Fourier)

3.2 Comparison of Candidate Convolutional Neural Networks

The AVRAD team analyzed pre-trained CNNs to determine which would provide the most value based on speed and accuracy; this resulted in three candidate solutions: Faster R-CNN, RetinaNet, and YOLOv3. Average Precision (AP) is the metric used when comparing a CNN's accuracy and effectiveness. Redmon et al., 2018 compared CNNs based on AP. His findings, shown in Figure 4, portrays the AP of different CNNs.

Faster R-CNN builds upon the already existing Fast R-CNN, becoming faster solely through the improvement of Selective Search. What makes Faster R-CNN operate more rapidly is that it forgoes the use of Selective Search; instead it uses a smaller CNN, known as Region Proposal Network (RPN) which utilizes gradient descent. The gradient of the final layer of weights is calculated first and the first layer of weights is calculated at the end. This allows for partial calculations of weights from one gradient to be reused and influence a previous layer resulting in efficient gradient computation. This is important for decreasing processing time and saving computational power in the CNN instead of having to approach each layer as a separate and new task. Faster R-CNN also incorporates anchor boxes, which are predefined bounding boxes. They have a certain height and width and are purposed to help scale and capture the aspect ratio of specific object classes that you would detect. This allows for Faster R-CNN to analyze 9 separate specific regions being almost 10x faster while holding a very similar level of accuracy to its predecessor, Fast R-CNN.

RetinaNet is also a pre-trained CNN, much like that of Faster R-CNN and YOLOv3. RetinaNet further distinguished itself by performing highest in overall AP as seen in Figure 4. However, speed is an equally important factor in selecting a CNN. RetinaNet's operating speed is significantly slower than the other alternatives. The team determined that a more reactive/responsive CNN is preferable over a slightly more accurate CNN such as RetinaNet.

YOLOv3 looks at the entire image when running. The CNN makes decisions based on the "global" scale, seeing the whole picture instead of segments. YOLOv3's ability to make predictions based solely on a single network evaluation allows the network to run 100x faster for a single image than that of Faster R-CNN. YOLOv3 runs on Darknet, an open source neural network frame which allows the user to interact with the CNN with ease.

AVRAD chose to utilize YOLOv3 over other CNNs based on (AP). The measurements of each CNN's AP when calculating over 50% region of any given image (AP_{50}), AP when calculating over 75% region of any given image (AP_{75}), and the AP when identifying objects of small, medium, or large size (AP_S, AP_M, AP_L). Looking at Figure 4 in depth, based off the AP metric average mean, the AP metric of YOLOv3, AP_{50}, shows YOLOv3 being very competitive with Faster R-CNN, performing far faster than other CNNs. In accordance with the detection metric value of AP_{50}, YOLOv3 excels in performance of identifying objects and producing the corresponding boxes around the object.

YOLOv3's ability to detect small-scale objects is competitive with Faster R-CNN and only outperformed by RetinaNet in accuracy of identifying smaller objects. And although RetinaNet appears to beat YOLOv3 on this category, comparing YOLOv3's metric values of AP_{50} with other operating CNNs, YOLOv3 holds the best attributes for this vehicle, namely operating faster and with higher efficiency than any other CNN (Tsang). RetinaNet is more accurate than YOLOv3, but its processing speed is significantly slower. In in the context of using a CNN for image classification in a self-driving vehicle, the AVRAD team determined that for a system reacting to external obstacles the team prioritized speed of processing over accuracy.

	backbone	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
Two-stage methods							
Faster R-CNN+++ [5]	ResNet-101-C4	34.9	55.7	37.4	15.6	38.7	50.9
Faster R-CNN w FPN [8]	ResNet-101-FPN	36.2	59.1	39.0	18.2	39.0	48.2
Faster R-CNN by G-RMI [6]	Inception-ResNet-v2 [21]	34.7	55.5	36.7	13.5	38.1	52.0
Faster R-CNN w TDM [20]	Inception-ResNet-v2-TDM	36.8	57.7	39.2	16.2	39.8	**52.1**
One-stage methods							
RetinaNet [9]	ResNet-101-FPN	39.1	59.1	42.3	21.8	42.7	50.2
YOLOv3 608 × 608	Darknet-53	33.0	57.9	34.4	18.3	35.4	41.9

Figure 4. Convolutional Neural Network Weighted Value Chart (Redmon et al., 2018)

4. YOLOv3 Testing and Implementation

After selecting YOLOv3, the team first tested its capability for classifying known objects using Google images. The leftmost picture in Figure 5 shows a sample of this test image. YOLOv3 successfully identified a stop sign and was able to differentiate between a car and a truck. The team then built a ROS package for the CNN in order to use the AVRAD system's machine learning algorithm.

After installing YOLOv3 in the system's software platform, the AVRAD team conducted further testing to ensure the CNN could correctly identify object from ".bag" files. These files provide quasi-real time imagery and allows the AVRAD team to ensure YOLOv3 can identify captured imagery from environments more closely aligned with the system's operational environment. The AVRAD team conducted these tests while the vehicle was stationary; the middle picture in Figure 5 shows an example image the system saw during this test.

Finally, the AVRAD team used Compute Unified Device Architecture (CUDA) parallel processing programs to enable the system to run YOLOv3 and identify multiple objects in real time as the system is moving forward. After modifying the CNN on the system to ensure it can function while in motion, the AVRAD team tested the functionality of YOLOv3 by having one team member stand in front of the camera with a computer displaying an image of a stop sign on the screen, this is the rightmost picture in Figure 5. The demonstrates that YOLOv3's ability to accurately detect a person, laptop, and a stop sign.

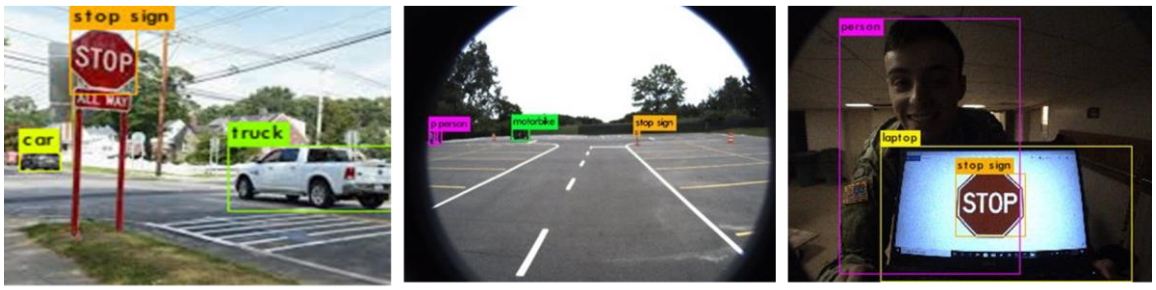

Figure 5. Progression of Testing Data
Left to Right: Image Captured from Newsday.com, Bag File, Real-time Imagery

5. Further Testing of YOLOv3

By utilizing the Velodyne HDL-64E LiDAR and five Mako G-319C cameras in conjunction with adapting a pre-trained YOLOv3 CNN, USMA AVRAD has made two significant findings about the current state of image classification in relation to autonomous driving systems. First, the team determined that YOLOv3 requires additional training to become fully autonomous. After confirming the functionality of known objects such as a stop sign, the team tested the ability to detect and follow street lanes. The lane following testing phase raised the question whether the CNN truly recognized the road lines. Because the training data used yellow lines instead of white, the AVRAD system failed to stay within the boundaries of the

white-painted lanes. It appears as if the CNN was correlating color of street lane lines as a higher correlation than viewing painted road lines. The discrepancy between the training data and real-life imagery would explain why the vehicle failed to stay within the boundaries of the pre-made lanes. The team may need to retrain the CNN with additional images of lane lines in different colors, or risk the car deviating from its intended path. The second finding occurred during the AVRAD's outdoor test. The system failed to stop after approaching a stop sign. Moving forward, the AVRAD team plans to conduct additional analysis to determine if this failure occurred due to the CNN not recognizing the stop sign, or if the vehicle management system failed to connect the CNN's output with the mechanical break command.

6. References

Convolution. (2020). Retrieved February 19, 2020, from Wikipedia: https://en.wikipedia.org/wiki/Convolution

Digital Convolution—E186 Handout. Retrieved March 3, 2020, from Fourier: http://fourier.eng.hmc.edu/e161/lectures/convolution/index.html

Gige Vision: Mako G-319C. Retrieved March 05, 2020, from RMAElectronics: https://www.rmaelectronics.com/avt-mako-g-319c/

O'Brien, R. (2019). *Residents' Outcry, Traffic Study Lead to New All-Way Stop Signs at Babylon Village Intersection*. Retrieved March 5, 2020, from Newsday: https://www.newsday.com/long-island/suffolk/intersection-all-way-stop-signs-crashes-traffic-study-1.35347517

Redmon, J. & Farhadi, A. (2018). *YOLOv3: An Incremental Improvement*. Retrieved February 19, 2020, from PJReddie: https://pjreddie.com/media/files/papers/YOLOv3.pdf

Sachan, A. (2017). *Zero to Hero: Guide to Object Detection using Deep Learning: Faster R-CNN, YOLO, SSD*. Retrieved February 8, 2020, from CV-Tricks: https://cv-tricks.com/object-detection/faster-r-cnn-yolo-ssd/

Tsang, S. (2019). *Review: YOLO$_v$3—You Only Look Once (Object Detection).* Retrieved March 05, 2020, from TowardDataScience: https://towardsdatascience.com/review-yolov3-you-only-look-once-object-detection-eab75d7a1b

ISBN: 97819384961-8-9