Autonomous Decision Making for a Driver-less Car

Nicolas Gallardo, *Student Member, IEEE*, Nicholas Gamez, *Student Member, IEEE*, Paul Rad, and Mo Jamshidi *Fellow, IEEE*
Department of Electrical and Computer Engineering
The University of Texas at San Antonio
One UTSA Circle, San Antonio, TX 78249, USA
{nicolas.gallardo, nicholas.gamez}@my.utsa.edu, paul.rad@rackspace.com, moj@wacong.org

***Abstract*—Autonomous driving has been a hot topic with companies like Google, Uber, and Tesla because of the complexity of the problem, seemingly endless applications, and capital gain. The technology's brain child is DARPA's autonomous urban challenge from over a decade ago. Few companies have had some success in applying algorithms to commercial cars. These algorithms range from classical control approaches to Deep Learning. In this paper, we will use Deep Learning techniques and the Tensorflow framework with the goal of navigating a driverless car through an urban environment. The novelty in this system is the use of Deep Learning vs. traditional methods of real-time autonomous operation as well as the application of the Tensorflow framework itself. This paper provides an implementation of AlexNet's Deep Learning model for identifying driving indicators, how to implement them in a real system, and any unforeseen drawbacks to these techniques and how these are minimized and overcome.**

***Index Terms*—Autonomous Driving, Deep Learning, Machine Learning, Neural-Network, Vision, Internet of Things**

I. Introduction

AUTONOMOUS Driving is not a new problem, the idea has been around since the 1980's with notable success coming from Carnegie Mellon's ALVINN autonomous driving platform [1].

In general, Deep Learning approaches for autonomous driving has been split into two models: i.) Mediated perception uses the driving markers in the image to create a world representation surrounding the car. Items such as traffic signs and obstacles in the road are classified to determine a driving action ii.) Behavior Reflex directly maps the input data to a driving action. The entire image is converted to steering and velocity commands.

Both models have issues such as requiring a multitude of sensors such as GPS, radar, and accurate maps for mediated perception, while behavior reflex methods can be confused by different human decisions made for similar events [2]. In the DeepDriving project by Princeton [2], they propose a new approach that maps only specific inputs to steering angle and speed. It is called the direct perception approach which can be considered a mix between the first two autonomous driving approaches. The data set is obtained from the Princeton DeepDriving project [2]; it is comprised of twelve hours of simulation data from The Open Racing Car Simulator (TORCS) environment emulating real life driving situations for training and testing. Learning from this data allows the system to determine the best course of action (steering angle) to safely navigate to a desired endpoint. The Deep Neural Network (DNN) architecture used is based on AlexNet and also is used in this papers implementation of DeepDriving [3].

Autonomous driving has a wide range of potential social, economic and environmental impacts. Each of which revolves around the inherent efficiency gained from implementing an optimized computer algorithm to do a task normally performed by a human. A "blue-paper" written by Morgan Stanley Research [4] breaks down the types of savings that can be expected by the US if we integrated a completely autonomous driving infrastructure. It is estimated by [4] that $1.3 trillion should be expected in savings. Even better, some of these savings directly relate to the reduction of the U.S.'s carbon footprint. Figure 1 outlines the savings categories.

Savings Category	$ Saved
Total savings from accident avoidance	$488 billion
Fuel savings	$158 billion
Fuel savings from congestion avoidance	$11 billion
Productivity gain from congestion avoidance	$138 billion
Productivity gain from autonomous cars	$507 billion
Total	$1.3 trillion net

Fig. 1: Savings using autonomous driving systems [4]

Other impacts include contributions to the science of Machine Learning and Deep Learning as yet another example of how these techniques may be used to solve the worlds extremely complex problem.

> "A breakthrough in machine learning would be worth ten Microsofts" -Bill Gates

A. Deep Learning for Image Classification

Deep Learning is the an extension of artificial neural networks, a subset of artificial intelligence. "Deep" Learning is in reference to the number of artificial layers needed to map from an input value to its output value in a network. Deep Learning with its increased complexity can solve more

* This work was supported in part by Grant number FA8750-15-2-0116 from Air Force Research Laboratory and OSD through a contract with North Carolina Agricultural and Technical State University.

problems than a conventional neural network such as license plate recognition [5]. Deep Learning is also being used to augment and enhance conventional controllers [6]. A neural network as shown in Figure 2 [7] combines two different types of layers, convolutional and fully connected, to determine the probability of an image being 1 of the 10 possible outcomes. The first layer is the input, an image in this case. The middle layers are described as hidden layers and typically only connect to the adjacent layers. The last layer gives the outputs of the network and can typically sent to a function to determine the distribution probability between each output. Specifically for image classification, we receive the probability of the input image being one of the predetermined labels.

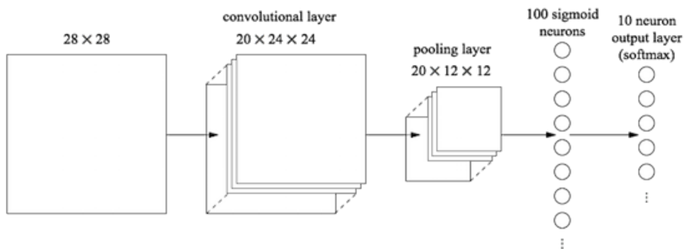

Fig. 2: Neural Network Example

Image classification with deep neural networks took off with the creation of AlexNet [8] which was used as the base model for the proposed architecture of this paper. Since 2012 and the inception of AlexNet for image recognition, multiple models have been developed to increase the accuracy and develop a better understanding of the inherent success for convolutional deep neural networks for images.

Convolution neural networks are advantageous to use for image processing since the convolution can express a large amount of pixels in an image more compactly. Through multiple convolutional layers, an image of size $(227, 227, 3)$, over 180,000 pixels, can be described with 4,096 data points. With the reduced number of outputs, a model can utilize fully connected layers to reduce the remaining data points into the required number of outputs. A network like this is described in [9]. The Inception model, debuted at the 2014 ImageNet Large-Scale Visional Recognition Challenge, built off the original success from convolution neural networks to create a newer network decreasing the top-5 error rate of image recognition lower than 7%.

B. Driverless Cars

Multiple approaches can be used to solve a problem like autonomous navigation. A classical control approach uses techniques such as simultaneous localization and mapping (SLAM) [10], path planning with A^* and Dijkstra theorems, extended Kalman filtering (EKF), IMU data, and other raw sensor data. Using these techniques have been proven to be useful in the systems described in [11] Many of these examples require a powerful middleware infrastructure in order to efficiently share sensor data along with other information necessary for the system to perform properly. One solution that is increasing in popularity is Robot Operating System (ROS). ROS handles the exchange of information automatically, and distributes it through the system into *nodes*. Each *node* performs individual functions within the system and as a whole, work as a team to perform complex functions of a robot. [12]
Urmson et. al. [13] designed an autonomous vehicle to complete the following tasks as part of the 2008 DARPA Urban Challenge.

1) Merge into and and turn across dense moving traffic
2) Vehicles were challenged to find their way through a neighborhoods road network while avoiding parked cars, construction areas, and other road obstacles but did not encounter moving traffic
3) Demonstrate correct behavior with traffic at four-way intersections and to demonstrate rerouting around an unexpectedly blocked road.
4) Navigate heavy traffic

The vehicles designed for the tasks were a joint effort between Carnegie Mellon university, General Motors Research Lab, Caterpillar, and Intel. The combined effort of this team successfully created a system using a 2007 Chevy Tahoe equipped with a drive by radio system that would allow for remote steering, throttle and brake management. Along with this, processing resources were mounted inside the SUV to provide the computation capability required for this system to work. In all, the autonomous system was tested over 3000 km and placed first at the 2008 DARPA Grand Challenge. This system uses an extremely complicated control system that calculates its desired trajectory based heavily on local cost maps calculated using input from an extremely expensive sensor array. The following sensors were used on the Tahoe.

1) Applanix POS-LV 220/420 GPS/IMU (APLX) x1
2) SICK LMS 291-S05/S14 LIDAR (LMS) x6
3) Velodyne HDL-64 LIDAR (HDL) x1
4) Continental ISF 172 LIDAR (ISF) x2
5) IBEO Alasca XT LIDAR (XT) x2
6) Continental ARS 300 Radar (ARS) x5
7) Point Grey Firefly (PGF) x2

The above items resulted in a system that costs well over $100k and would not be cost effective until the hardware needed became cheap and readily available for millions of cars, trucks, and SUV's. Overall the system was extremely capable and came out in first place, dominating the DARPA challenge with a 19 minute faster finish time compared to the second place contestants.

This challenge helped to boost interest in autonomous driving research. Since the early 1990's, Caterpillar had several systems that had been demonstrated live to customers that showed their capability to automate many heavy machinery tasks. The issue was that before the first 2004 DARPA Grand Challenge, no one wanted to automate their driving systems.

After the 2004 Challenge, interest peaked and research was fully underway.

Caterpillar hosts a large research program with over $2 billion in funding and 10,000 engineers with 350 at the PhD level. A large effort goes into semi-autonomous and fully autonomous machinery. Currently, Caterpillar's systems can provide solutions for hauling, dozing, underground, longwall, and drilling. These functions collectively work together to successfully mine an area for precious material. Several advantages are gained using these systems [14]:

1) Safety - A large piece of inspiration for robotics systems is for situations that are dangerous and have the potential to cause the user harm. In mining processes, dangerous situations such as mine collapse, explosions, drilling, etc. are very common and the number of mining related injuries is reduced by removing the human.
2) Productivity - An autonomous system may be monitored and logged in order to maximize the process over all. Things such as reduction in crew numbers, time to process material, drilling accuracy are all improvements that have been observed when using these systems.

All of the previously presented research relies on a combination of image data fused with data from supplemental sensors. Many techniques have been developed in order to create a local map around the robot so that it has enough knowledge to plan a trajectory and follow it without crashing or enacting fatal behaviors. A Deep Learning algorithm in [15] is used to obtain range estimations based on a pair of stereo images. The Deep Learning algorithm extracted features from the images and trained a classifier to separate the features into 5 different classes. Each class allowed the system to predict range data of the features detected in each pair of images.

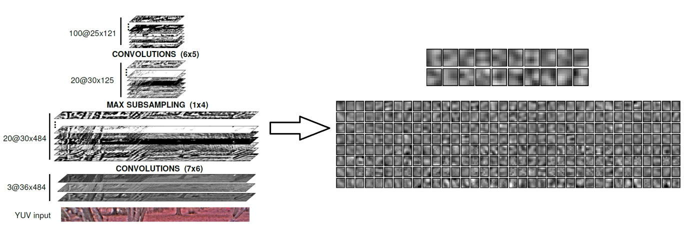

Fig. 3: DNN Used by [15]

II. Problem Statement

Autonomous driving is one of the next frontiers for Deep Learning and machine learning. The purpose of this research was to produce a deep neural network using Tensorflow [16] python framework to correctly identify the distance to leading cars and the lane markings in two and three lane configurations. A successful model should be able to navigate traffic given that no cars behind the target vehicle are driving faster than the target vehicle.

The rest of the paper is structured as follows: First the graph model is presented, the model used was AlexNet. Next, training using TensorFlow Deep Learning tools is explained in Section III.B. along with the different experiments conducted using these tools in Section III.C. AlexNet is further explained in Section III.D. in order to show how this model applies to the autonomous driving problem. Finally, results and analysis along with challenges are highlighted in Sections III.E. and III.F. respectively.

III. Proposed Method, Design, and Approach

A. *Proposed Solution Approach*

AlexNet is the base model used for this implementation. AlexNet is one of the original image classification models containing 5 convolution layers and three fully connected layers originally [8]. The model's convolution layers filter the images down to 4096 outputs where the fully connected layers feed their output to a softmax function producing a distribution probability of the 1000 possible labels.

AlexNet was the one of the first highly popular computer vision convolution neural networks because of its low top-5 error rate ($\sim$15%). AlexNet won the 2012 ImageNet Large-Scale Visual Recognition Challenge beginning the race to lower error rates even further using convolution neural networks.

The Deep Learning model is the initial step in the autonomous system. The results from the model are sent into a controller which calculates the steering and pedal commands. The vehicle itself is a combination of systems all working towards the final goal of autonomous driving.

B. *Training Methodology*

TensorFlow plays a large part in the training process, it allows the creation of large and complex neural networks. Furthermore, TensorFlow is an open source platform that allows computation to be carried out on GPU's or multiple CPU's [16]. Our approach investigated the advantages/disadvantages of first duplicating what the Princeton team did in the Deep-Driving project using TensorFlow instead of Caffe. This acts as a benchmark to investigate as to why and where in the code optimization or degradation was occurring. Once the initial experiment setup was completed, modification of parameters pertaining to the neural network were investigated to see what behavior is obtained from the network when layers are added/subtracted, filter sizes are changed, padding is used, or optimization schemes are altered. Our investigation into these parameters provided a deeper understanding of the effects of each tuning parameter. Furthermore, by taking a different approach we hope to uncover a better approach to the autonomous driving problem. The final autonomous driving system is capable of providing steering and speed commands that allows the car to drive around the simulated track in TORCS without colliding with other cars or driving off the track.

Figure 4 shows the input and output from the designed network. An first person image is given from the driver's perspective, and the 14 indicators itemized in Table I are determined by the neural network. This network is called a multiple target regression problem.

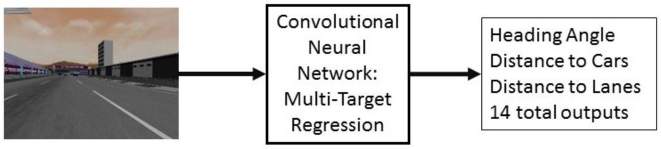

Fig. 4: Input images are mapped through the neural network into 14 different affordance labels

Label	Low	High
Angle of the Car	-0.5	0.5
Range to left lane marker	-7	-2.5
Range to middle lane marker	-2	3.5
Range to light lane marker	2.5	7
Distance to car in left lane	0	75
Distance to car in right lane	0	75
Range to far left lane marker	-9.5	-4
Range to near left lane marker	-5.5	-0.5
Range to near right lane marker	0.5	5.5
Range to far right lane marker	4	9.5
Distance to the left lane	0	75
Distance to the middle lane	0	75
Distance to the right lane	0	75
Fast	0	1

TABLE I: Table of affordance indicators. Output of neural network

C. *AlexNet Based Model*

Reducing the learning curve of machine learning and Deep Learning heavily depends on developing well known models for specific applications. Utilizing a predefined neural network allows the developer to start with a foundation to implement immediately then improve upon. AlexNet is made up of five convolution layers followed by four fully connected layers. The AlexNet model implementation can be seen in Section III.D.

D. *Experiments, Results & Analysis*

Another python package called TFlearn is a higher level abstraction of Tensorflow. TFlearn was used to create the DNN consisting of five convolution layers followed by four fully connected layers. Figures 8 and 6 depict the actual network created taken from the Tensorboard visualization of the training DNN.

The model features all the same components from the original work but in Tensorflow. The Caffe model did not come with accuracies but with an absolute mean error value for each output. Therefore, the two models do not have a common point for comparison yet which is expanded upon in the next section. For the final training, sets of 50,000 images from the DeepDriving data-set were used. The most successful training models settled between 55 - 60 % accuracy. For the results shown, the DNN was as closely modeled to the original Caffe model as Princeton's DeepDriving project with the following variables.

- Learning Rate = 0.01
- Optimizer = Stochastic Gradient Descent
- Loss Function = Mean Square
- Batch Size = 64

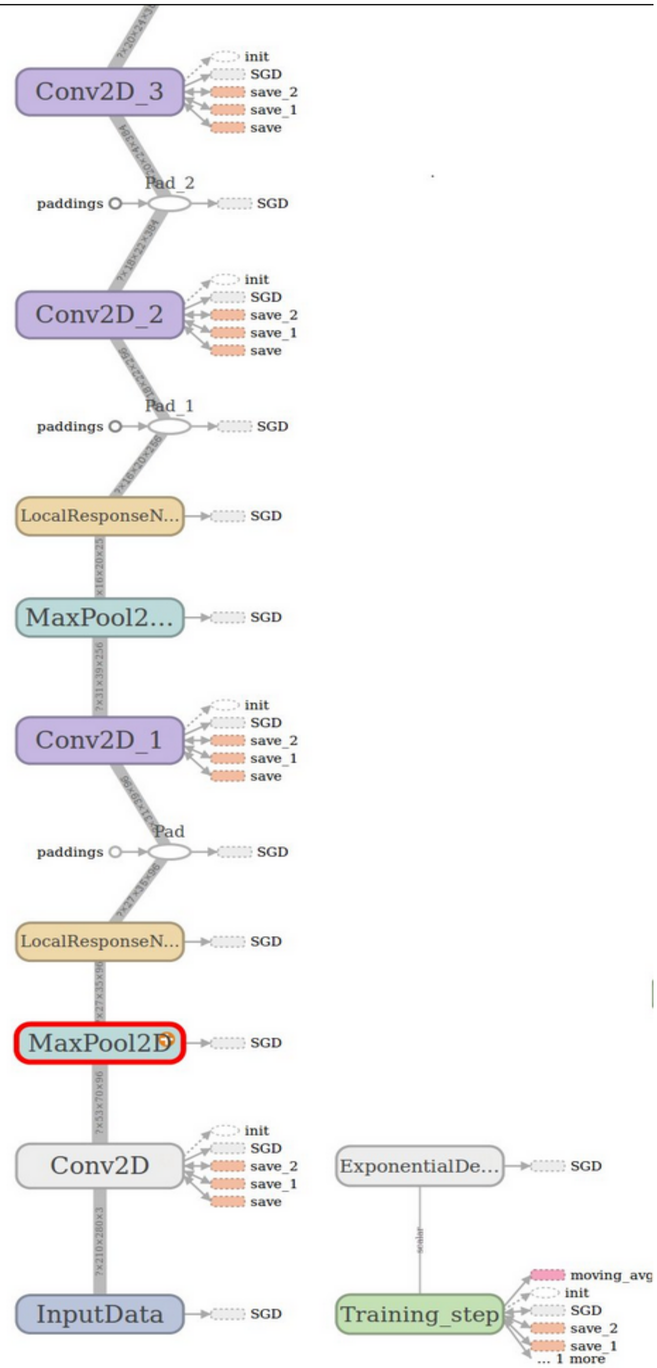

Fig. 5: Implementation of the convolution network from Tensorboard for the driving affordance indicators Part 1

The data was originally saved as a Caffe protobuf data-type requiring some pre-processing to extract separately the image and labels. The images are size (280, 210, 3) differing from the normal AlexNet architecture of a square image. Originally we attempted the DNN while re-sizing the image, but the results were less accurate than not resizing them. The fourteen labels are shown in Table I.

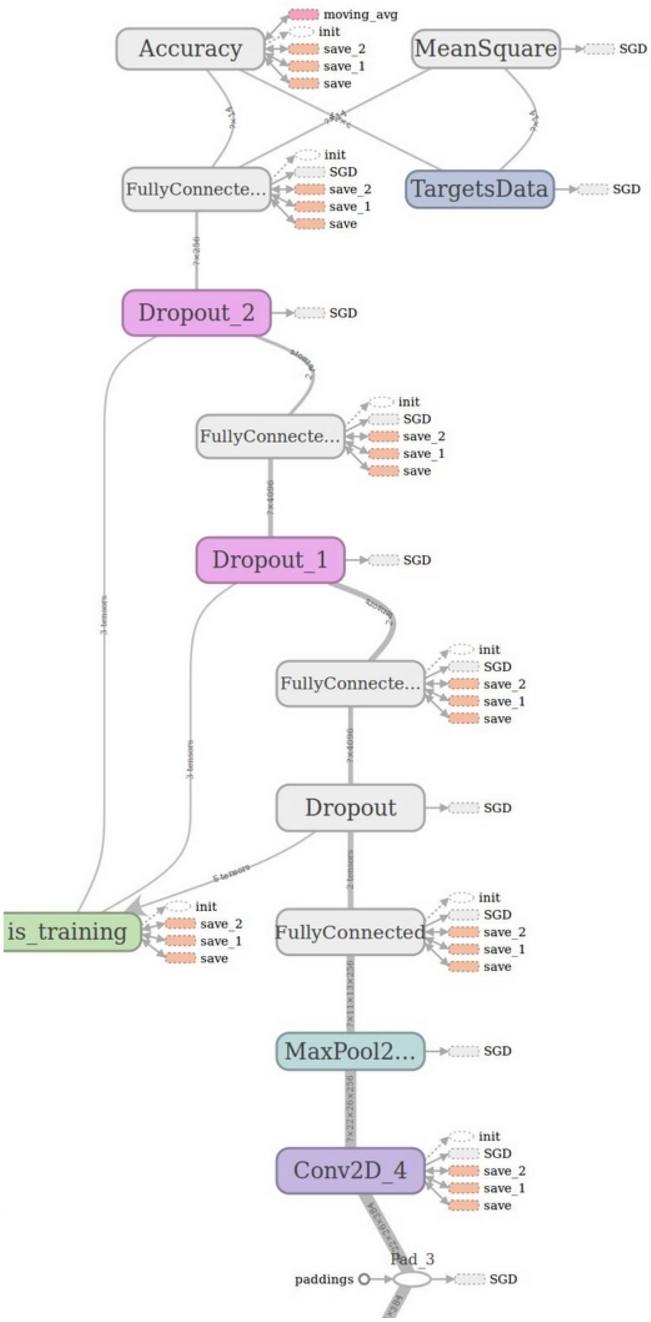

Fig. 6: Implementation of the convolution network from Tensorboard for the driving affordance indicators Part 2

Each indicator was normalized to the range 0.1 - 0.9 in the Caffe model originally, but during training, the non-normalized data showed better accuracy than the normalized ones. Consequently, in the Tensorflow implementation, the original values for the labels are used. The model is tuned by an accuracy metric that categorizes how close the values of all the indicators are to the labels given for each image. This higher the accuracy, the closer all the 13 indicators are to the correct value. Figures 7 - 10 depict the accuracy and loss of the training and validation models steps. The validation test set is 5% of the data set taken randomly and removed from the training set. The model reaches 66% accuracy before settling around 55%, but the validation is continually at 78% accurate.

The loss, calculated by mean square, is high due to some of the indicators having a large range. The model is not as accurate as one would want. To try to increase the accuracy, different optimizers and loss functions were attempted to no avail. The optimizers Adam and Momentum were used over SGD but did show any improvement, and binary crossentropy and softmax also did not improve the model. Because of this, the authors continued with the original variables.

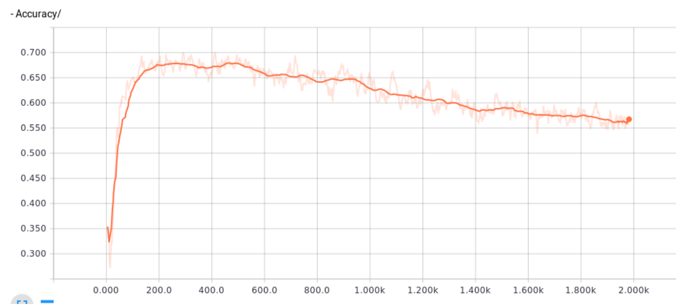

Fig. 7: Accuracy of trained model per step

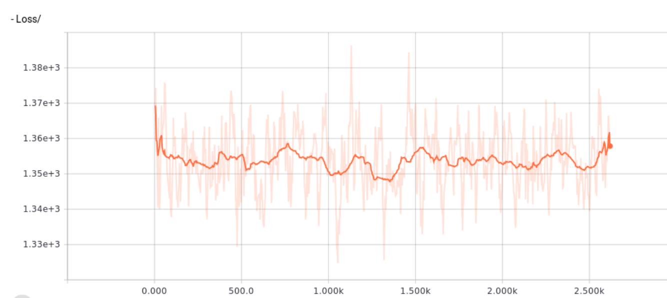

Fig. 8: Loss of trained model per step

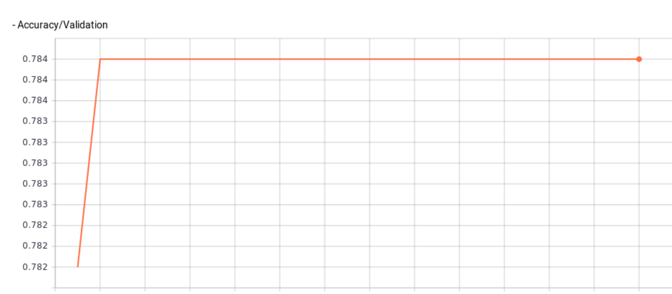

Fig. 9: Accuracy for validation set. Validation occurred every 100 steps

Fig. 10: Loss for validation set. Validation occurred every 100 steps

A large burden pertaining to deep learning is the data set that is used. In this case, the Princeton DeepDriving data set was used. First, this is a large data set so any manipulation done to it must be propagated across a large amount of data that can make manipulating the set quite cumbersome for whatever hardware is being used. Second, similar to the last challenge, training on a large data set can take a very long time.

When comparing the Caffe and Tensorflow models, the idea was to use the metric from Princeton's Caffe model. The mean absolute error,

$$e_{mean} = \frac{1}{N} \sum_{i=1}^{N} |f_i - y_i|$$

where f_i is the predicted output, N is the number of samples, and y_i is the known output, could not be calculated for the Tensorflow model because of a consistent error when attempt to predict values of a saved model. A C-type error "bad_alloc" would terminate the program and no new predictions would show. This is typically a memory error, but it occurred even when attempting small test sets.

IV. Conclusion and Future Work

Through the research conducted, several goals were achieved:

1) Obtain an understanding of the Autonomous Automotive Industry, the state of the art autonomous driving technology, and broader impacts of a world where a majority of the vehicles on the road are autonomous.
2) Research different approaches that can be used to implement a autonomous driving system with machine learning and Deep Learning techniques at the core
3) Implement what Princeton did in their DeepDriving project in the TensorFlow machine learning framework
4) Draw conclusions about future improvements and real-world implementation.

In this paper we discussed how exactly each goal was met. Through literature review of previous work, a deep understanding of existing solutions along with their drawbacks were analyzed in order to validate the alternative machine learning approach to this problem.

Using deep learning techniques such as CNN, AlexNet, and supervised learning, the research shows that autonomous driving can successfully be simulated in a game environment using the deep learning architecture presented in this paper. The results of this experiment show that these principles are ready to be applied to a real world platform in order to experimentally verify its ability to become an everyday part of our lives. As autonomous vehicles are embedded with Internet of Things (IoT) data gathering, a decision making and interactive driving experience will emerge. The next steps are to study the interaction and decision making process among Smart IoT devices, much like the implementation of IoT for agriculture in [17], and vehicles.

Acknowledgment

The authors would like to thank their families for their continued support and encouragement.

References

[1] D. A. Pomerleau, "Alvinn: An autonomous land vehicle in a neural network," DTIC Document, Tech. Rep., 1989.
[2] C. Chen, A. Seff, A. Kornhauser, and J. Xiao, "Deepdriving: Learning affordance for direct perception in autonomous driving," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 2722–2730.
[3] C. Chen, "Extracting cognition out of images for the purpose of autonomous driving," Ph.D. dissertation, PRINCETON UNIVERSITY, 2016.
[4] R. Shanker, A. Jonas, S. Devitt, K. Huberty, S. Flannery, W. Greene, B. Swinburne, G. Locraft, A. Wood, K. Weiss *et al.*, "Autonomous cars: Self-driving the new auto industry paradigm," *Morgan Stanley Blue Paper, Morgan Stanley & Co. LLC*, 2013.
[5] R. Polishetty, M. Roopaei, and P. Rad, "A next-generation secure cloud-based deep learning license plate recognition for smart cities," in *Machine Learning and Applications (ICMLA), 2016 15th IEEE International Conference on*. IEEE, 2016, pp. 286–293.
[6] M. Roopaei, P. Rad, and M. Jamshidi, "Deep learning control for complex and large scale cloud systems," *Intelligent Automation and Soft Computing (AUTOSOFT)*.
[7] M. A. Nielsen, *Neural Networks and Deep Learning*. Deetermination Press, 2015.
[8] A. Krizhevsky, I. Sutskever, and G. Hinton, "Imagenet classification with deep convolutional neural networks," in *NIPS*, 2012.
[9] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015, pp. 1–9.
[10] P. Benavidez, M. Muppidi, P. Rad, J. J. Prevost, M. Jamshidi, and L. Brown, "Cloud-based realtime robotic visual slam," in *Systems Conference (SysCon), 2015 9th Annual IEEE International*. IEEE, 2015, pp. 773–777.
[11] Y. Feng, P. Iravani, and C. Brace, "Position estimation and autonomous control of a quad vehicle," SAE Technical Paper, Tech. Rep., 2016.
[12] "Ros introduction," http://wiki.ros.org/ROS/Introduction, accessed: 2016-12-11.
[13] C. Urmson, J. Anhalt, D. Bagnell, C. Baker, R. Bittner, M. Clark, J. Dolan, D. Duggins, T. Galatali, C. Geyer *et al.*, "Autonomous driving in urban environments: Boss and the urban challenge," *Journal of Field Robotics*, vol. 25, no. 8, pp. 425–466, 2008.
[14] M. Glover, "Caterpillars autonomous journey-the argument for autonomy," SAE Technical Paper, Tech. Rep., 2016.
[15] R. Hadsell, A. Erkan, P. Sermanet, M. Scoffier, U. Muller, and Y. LeCun, "Deep belief net learning in a long-range vision system for autonomous off-road driving," in *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2008, pp. 628–633.
[16] "Tensorflow," https://www.tensorflow.org/, accessed: 2016-12-11.
[17] M. Roopaei, P. Rad, and K.-K. R. Choo, "Cloud of things in smart agriculture: Intelligent irrigation monitoring by thermal imaging," *IEEE Cloud Computing*, vol. 4, no. 1, pp. 10–15, 2017.