DEEP LEARNING ON CLOUD FOR SELF-DRIVING CARS
Contents

1. Project Background.................................................................................................................. 2
   1.1 Problem Statement ............................................................................................................. 2
   1.2 Project Motivation ............................................................................................................ 2
   1.3 Current solutions and their shortcomings........................................................................ 3
2. Project Objective.................................................................................................................... 4
3. Project Methodology ............................................................................................................. 5
   3.1 Main software to be employed ......................................................................................... 5
   3.2 Step by step methodology of proposed project ............................................................... 6
   3.3 Brief testing strategy ......................................................................................................... 6
4. Project Schedule and Milestones.......................................................................................... 7
References .................................................................................................................................. 8
1. Project Background

1.1 Problem Statement

Injuries sustained from road traffic accidents are the leading cause of death for people aged 15 to 29 and more than 1.25 million people lose their lives due to them every year. If sustained action is not taken, road accidents are predicted to become the 7th largest cause of death by 2030 [1]. Technological research and inventions, as in other matters, can be a great tool for the world to drive down this risk.

Self-driving cars have the potential to reduce the number of road accidents and the associated death toll [2]. Nearly 90% of road accidents happen due to human error [3] and by solely being focused on driving and being ever so ready to react to urgent situations, self-driving cars avoid the risk of accidents caused by distracted drivers behind the wheel [4]. They can also be set to obey the law, respect road signs, and never lose track of their speed and the official speed limit. Moreover, they cut out emotions and irrational factors behind reckless driving such as being late for appointments, personal frustrations or prolonged stop-and-go traffic.

1.2 Project Motivation

To be able to navigate their way around roads, self-driving cars use a variety of techniques to detect their surroundings such as radar, lidar (laser lights), GPS, the odometer and computerized vision. Truly autonomous cars do not require human intervention other than setting the destination and starting the system. Hence, the automatic system needs to be able to drive, navigate and make its own decisions using its various tools and technologies.

For total automation to be possible, autonomous cars need precise locating and positioning which can only be achieved by real-time, or near-real-time, updating of the mapping environment. Simply using GPS-based maps is not enough for next generation cars that won’t have drivers at all. Moreover, roads are rarely predictable and situations that require instant decision making may occur anytime. For example, self-driving cars may have to navigate through heavy rain or snowstorms that could hide or distort the painted lines on roads and highways. This can make autonomous navigation systems, if not useless, at least, erratic.

Additionally, self-driving cars need to detect, recognize and track objects like other cars, pedestrians and cyclists around them. Since the appearance and movement of these entities is highly unpredictable, the object detection and tracking system needs to be accurate and have an instant feedback time. The smallest of delays in obtaining object tracking data can lead to a delayed reaction by the driving system, potentially leading to an accident.
1.3 Current solutions and their shortcomings

Currently, deep learning-based object detection approaches especially those involving convolutional neural networks have made great breakthroughs in using camera images in object detection. However, the error rates are still too high for a safety critical application and the performance of the object detection and tracking systems is mainly computed based on prerecorded datasets. [5] Edge cases for systems supported by these deep learning solutions are still erratic and not production ready for fully autonomous self-driving cars.

Moreover, neural networks used in such systems need to be trained on representative datasets that include examples of all possible driving, weather and situational conditions. This translates to petabytes of training data which requires humongous computational power and a very long period of time to process. Training these networks on single GPUs takes years and there is a dire need to quicken and optimize the training process. Besides the training of the networks, processing of camera image data and feeding it to trained networks in real-time are time and resource intensive processes as well and require serious optimization if self-driving cars are ever to achieve real-time response rates and decision making amidst changing road conditions.
2. Project Objective

Since self-driving car systems need to be safety critical applications, it is of utmost importance to detect surrounding objects accurately and track them in real time in order to make a driving decision. For real time detection to be possible, data computation needs to be faster and the project aims to harness the power of cloud computing to parallelize processes and reduce latency.

The project proposes a cloud-based self-driving system with two key functions:

1. Cloud based multi-object tracking system
2. Deep learning engine for real-time process on road-like conditions

In order to detect, recognize and track multiple objects given a series of videos and image data, the project aims to use YOLO (You Only Look Once): a deep neural network-based object detection system. Trained and plugged with DeepLearning4Java (Java machine learning platform), the aim of the object detection system will be to achieve an increase in accuracy of object detection, recognition and tracking for the input data.

One of the obstacles for real time object detection and tracking is the huge amount of sensor data that needs to be processed instantly and the computing power required for this process. In an attempt to optimize this process, the project aims to make use of Apache Flink: a distributed streaming dataflow engine. By deploying the deep learning engine(YOLO) on Flink, data computation will happen over a cloud cluster with effective parallelization and reduced latency.

To ensure that the object detection and tracking system works just as well with live data streams as it does with pre-recorded video and image data, the system will be implemented on a mobile phone to test the deep learning platform with a live data stream supplied from the phone camera and processed through Flink.
3. Project Methodology

3.1 Main software to be employed

1. For object recognition and tracking:

   YOLO (You Only Look Once)

YOLO is a deep neural network based system that achieves fast object detection and tracking by dividing image frames into sub-regions, predicting the bounding boxes and probabilities for these regions.

YOLO is a suitable choice for the project’s main object detection feature as it is one of the fastest open-source systems and allows us to analyze our system easily by enabling a tradeoff between speed and accuracy of object detection without the need to retrain the network.

The project aims to use either YOLO version 2 or 3 based on further research and updated project scope.

2. For data computation

   Apache Flink

Flink is an open source distributed streaming dataflow engine. It is capable of supporting large amounts input data to various different deep learning models and creating a real-time stream to transmit the input data to the models in order to achieve near real time computation and results [6].

Moreover, Flink supports stateful computation and can therefore maintain data which has been processed over a long period of time.

These features make Flink a convenient choice for our application that requires inputting and handling a large amount of image and video data for real time processing by the deep learning engine.

3. Deep Learning

   Deeplearning4Java (DL4J)

DL4J is an open-source, distributed deep-learning library written for Java and Scala. It takes advantage of the latest distributed computing frameworks including Apache Spark to accelerate training of deep learning models [7].

DL4J will be an ideal choice to build a YOLO model on because of its widespread support for and compatibility with all Java Virtual Machine (JVM) languages.
3.2 Step by step methodology of proposed project

- The essential first step for the project will be to train a deep learning neural network offline on a local machine to detect objects and track them.
- Once the network is sufficiently trained, it will be deployed on DL4J on a Java Virtual Machine (JVM) and integrated with Apache Flink. This will ensure that parallel computation takes place on a cloud cluster to save time and reduce latency.
- The project aims to make use of drive-time videos as an input for the system through a front-end. The input data will be streamed through Apache Flink and after cleaning and verification, it will be fed to the deep learning neural network model set up on DL4J.
- After processing the input data, the neural network will be expected to recognize the various objects in the video stream and continuously track their movement.
- The next potential step will be to integrate the system with a mobile phone camera and test the network using a real-time camera input stream.

3.3 Brief testing strategy

The performance of the object detection model and the deep learning component deployed on Flink will be tested and analyzed based on the following criteria:

- Error rate (or accuracy) achieved
- Throughput (e.g., training time, time to process real-time data stream)
- Speedup of overall object detection and tracking results
- Performance testing with different data sets and sizes (Video data as well as real-time camera stream)
# 4. Project Schedule and Milestones

<table>
<thead>
<tr>
<th>Timeframe</th>
<th>Milestone</th>
</tr>
</thead>
<tbody>
<tr>
<td>September-October 2018</td>
<td>Background research</td>
</tr>
<tr>
<td></td>
<td>Self-learning and setup of opensource software</td>
</tr>
<tr>
<td></td>
<td>Environment Setup</td>
</tr>
<tr>
<td>November 2018</td>
<td>Setup and training of YOLO on DL4J</td>
</tr>
<tr>
<td></td>
<td>Setup of Apache Flink for data streaming</td>
</tr>
<tr>
<td>December 2018</td>
<td>Integrate YOLO on DLJ4 with Apache Flink</td>
</tr>
<tr>
<td>January 2019</td>
<td>Setup frontend to handle data input</td>
</tr>
<tr>
<td></td>
<td>Testing with video data</td>
</tr>
<tr>
<td>February 2019</td>
<td>Framework setup on mobile phone</td>
</tr>
<tr>
<td></td>
<td>Testing and bug fixing</td>
</tr>
<tr>
<td></td>
<td>Analysis of performance and optimization</td>
</tr>
<tr>
<td>March 2019</td>
<td>Collection and final documentation of test results</td>
</tr>
</tbody>
</table>
References

http://www.who.int/news-room/fact-sheets/detail/road-traffic-injuries

Andrew Del-Colle, The 12 Most Important Questions About Self-Driving Cars, POPULAR MECHANICS (Oct. 8, 2013)

Bryant Walker Smith, Human Error as a Cause of Vehicle Crashes, CTR. FOR INTERNET & SOC’Y (Dec. 18, 2013, 3:15 PM)
http://cyberlaw.stanford.edu/blog/2013/12/human-error-cause-vehicle-crashes


[6] Apache Flink-Stateful Computations over Data Streams
https://flink.apache.org/

[7] Deep Learning for Java- Open-source, distributed, scientific computing for the JVM
https://deeplearning4j.org/