Fine grained “automatic vehicle classification” system development for accurately measuring passenger-freight interactions

Final Report

METRANS Project 16-13

April 2018

Principal Investigator:

Mohammad Mozumdar, Ph.D.

Department of Electrical Engineering

California State University, Long Beach

DISCLAIMER

The contents of this report reflect the views of the authors, who are responsible for the accuracy of the data and information presented herein. This document is disseminated under the sponsorship of the Department of Transportation, University Transportation Centers Program, the California Department of Transportation and the METRANS Transportation Center in the interest of information exchange. The U.S. Government, the California Department of Transportation, and California State University, Long Beach assume no liability for the contents or use thereof. The contents do not necessarily reflect the official views or policies of the State of California, CSULB, or the Department of Transportation. This report does not constitute a standard, specification, or regulation

ABSTRACT

For the last decade, advances in machine-learning algorithms provided easy and efficient ways to analyze large sets of data in search of correlations that would otherwise be extremely time-consuming without the use of computers. The use of machine-learning algorithms for smart roads to track and analyze traffic attributes allows for highly accurate classifications while still being scalable and flexible enough to identify new types of vehicles that have yet to hit the market. Furthermore, extremely low power microprocessors have made it possible in the last few years to develop embedded systems that can run solely on battery power in multi-year applications without the necessity of recharging. Combined with low-power focused communications protocols and efficient vehicle classification algorithms, the lifetime of embedded systems can operate beyond a decade without any physical interaction after initial setup. This proposed research centers on the development of a distributed wireless sensing network that utilizes low power processors in conjunction to "in-sensor-node" machine learning algorithms for computation and a power-aware communications protocol for the development of a lightweight low-power multi-node MEMS sensing network. The collected data can be used for developing advanced models of urban traffic flow and for developing better policies to manage the impacts of transportation in metropolitan areas.

TABLE OF CONTENTS

ILLUSTRATIONS

DISCLOSURE

This research project report was funded by a grant from the California Department of Transportation (Caltrans) through the Metrans Transportation Center (METRANS) under a cooperative agreement between the University of Southern California (USC) and California State University, Long Beach (CSULB).

ACKNOWLEDGEMENTS

The authors acknowledge support given from California Department of Transportation (Caltrans) through the METRANS under a cooperative agreement between the University of Southern California (USC) and California State University, Long Beach (CSULB). Special thanks to Matt Hanson, Caltrans Freight Research for his valuable comments and suggestions during the timespan of the project. We would also like to acknowledge all graduate and undergraduate students who worked very hard day-and-night to get the best result possible.

I. SCOPE

This report is organized as follows: Section 2 talks about the issues with the current state of highway vehicular detection systems and technologies typically associated with such systems. Section 3 discusses in depth about the related classification work of previous papers with similar ideas and concepts for vehicular detection and classification. Section 4 presents our approach on how we decided to design a wireless sensor network, how we tackled the classification problem, and the background to our selected machine learning algorithm. Section 5 provides details of the developed hardware and, in section 6 we discuss about algorithms for vehicle classification with their performance. We conclude the report at section 7 highlighting the research result and directions to moving forward with this research.

II. INTRODUCTION

Ever increasing highway traffic and inadequate construction of new highways across the US is causing the congestion level on our nation's roadways to spiral out of control. According to the US Department of Transportation, surface road and highway traffic has increased by 2 percent and 33 percent, respectively, between 1987 and 1997 [2]. It is estimated that due to further demand for mobility, traffic will continue to increase by more than 40 percent by 2020. Therefore, demands for improving and extending the existing road infrastructure, especially around megacities, are becoming a major concern that could cost billions of dollars. By efficiently using existing transportation networks and advanced traffic control management techniques, cost effective and environmentally friendly solutions can be achieved. Thus, there is an essential need for an affordable and environmentally friendly solution in order to maximize the capacity and efficiency of existing transportation networks. ITS provides a solution that reuses the existing transportation network without the need to scrap the whole system. The ITS goals are to provide an efficient

solution for reduced travel time, easing delay and congestion, and improving safety. Advanced sensing, electronic surveillance, and traffic analysis and control are the main technologies employed by ITS to provide real time traffic information to users and policy makers of the transportation system. Moreover, different transportation service providers can use the ITS data to monitor, route, and control traffic flow.

The success of these intelligent transportation systems significantly depends on the proper design, installation, and maintenance of the sensor unit of the system. Currently available traffic sensor systems such as: inductive loop, video, sonar, radar, magnetic, capacitive, PVDF wire, and pneumatic treadle, are costly and use electrical power from the power distribution network. Current system can cost thousands of dollars for each sensor (video, sonar, and radar) installed on utility poles [2]. Moreover, costs for road-installed sensors (pavement) such as inductive, magnetic, PVDF wire, capacitive, and pneumatic treadle can span several thousand dollars each. Regardless, in-pavement sensors are still popular, due to their accuracy, ability to provide direct information with very little ambiguity, ability to monitor road conditions (i.e. presence of ice), all while not requiring a human operator. In the US, we have millions of miles of highways however power may not be available over all portions. Therefore, to collect traffic data in these areas, a system is needed that will be inexpensive to install and maintain, possess a small physical footprint, and can be deployed anywhere regardless of direct power availability. Motivated by these novel causes, in this report the PI suggests a sensing system based on low-power MEMS where installation is cost-effective and can remain operational without a direct power supply for years.

III. RELATED WORKS

For vehicle detection, transportation agencies mostly use Inductive Loop Detector (ILD) scheme which utilizes inductive current loop (6ft x 6ft, 6ft x 20ft, etc.) that needs to be implanted into the road pavement. Our proposed approach requires the installation of a "quarter" sized sensor platform that includes a wireless transceiver module, sensors, a low power microcontroller, and other minor electronics modules. Furthermore, the sensors will communicate with Electronics Communication Unit (ECU) wirelessly, therefore the need to cut into the pavement, as required in ILD approach, will no longer be necessary. The power savings in our proposed method would be achieved at two levels, at the node level and the network level. At node level, energy savings can be achieved through an optimized vehicle classification algorithm, improved hardware design, and also by employing a power-aware communication protocol. Moreover, the sensor node can be operated at different power modes such as TI's Low Power Mode (LPM) modes, also known as sleep mode, built into their microprocessor platform. In addition, node level power consumption can be optimized through reducing the active processing time. Such low power operating modes can be easily controlled by interrupt-driven program flow. To comply with a stringent power budget, more savings can be achieved by adjusting the sampling rate on-the-fly.

In [2-16], various authors described different configurations of wireless sensor packages to instrument roadways that count passing vehicles, measure roadway vehicle speed, and classify vehicles based on axle count and spacing. In [10], the authors proposed a wireless magnetic sensor node for vehicle detection with optical wake-up. In [11], the authors described a method for vehicle counting for ITS using a noninvasive magnetic-based WSN which would not be affected by weather conditions, allowing for very stable and robust data to operate with. The authors, in [15] investigated the task of classifying the types of moving vehicles in a distributed WSN in terms of

data collection procedure, feature extraction, pre-processing steps, and baseline classifier development. Furthermore, [17] describes methods for processing range imagery and performing vehicle detection and classification where classification rates of over 92% accuracy were obtained. In [18,19], the authors developed an algorithm that detects and tracks a moving target then subsequently alerts sensor nodes along the projected path of the target. The method proposed in [20] by researchers from UC Berkeley use hill-pattern as its primary selected feature for the vehicle detection and classification. The hill pattern is extracted from the vehicle's magnetic signature. While utilizing different aspects of the related works referenced above, we propose the employment of a low-power "*in-node"* machine learning based approach that uses appropriate sensors to increase traffic detection and classification to a near 90% accuracy (most of the work mentioned above use **out-of-node detection and classification**). Furthermore, our proposed approach utilizes low power operations and energy scavenging schemes in access points to help make the system extremely scalable for large-scale deployments. Considering all these capabilities, our proposed methodology is a significant milestone towards achieving better traffic data collection for the design of smart roads.

IV. METHODOLOGY

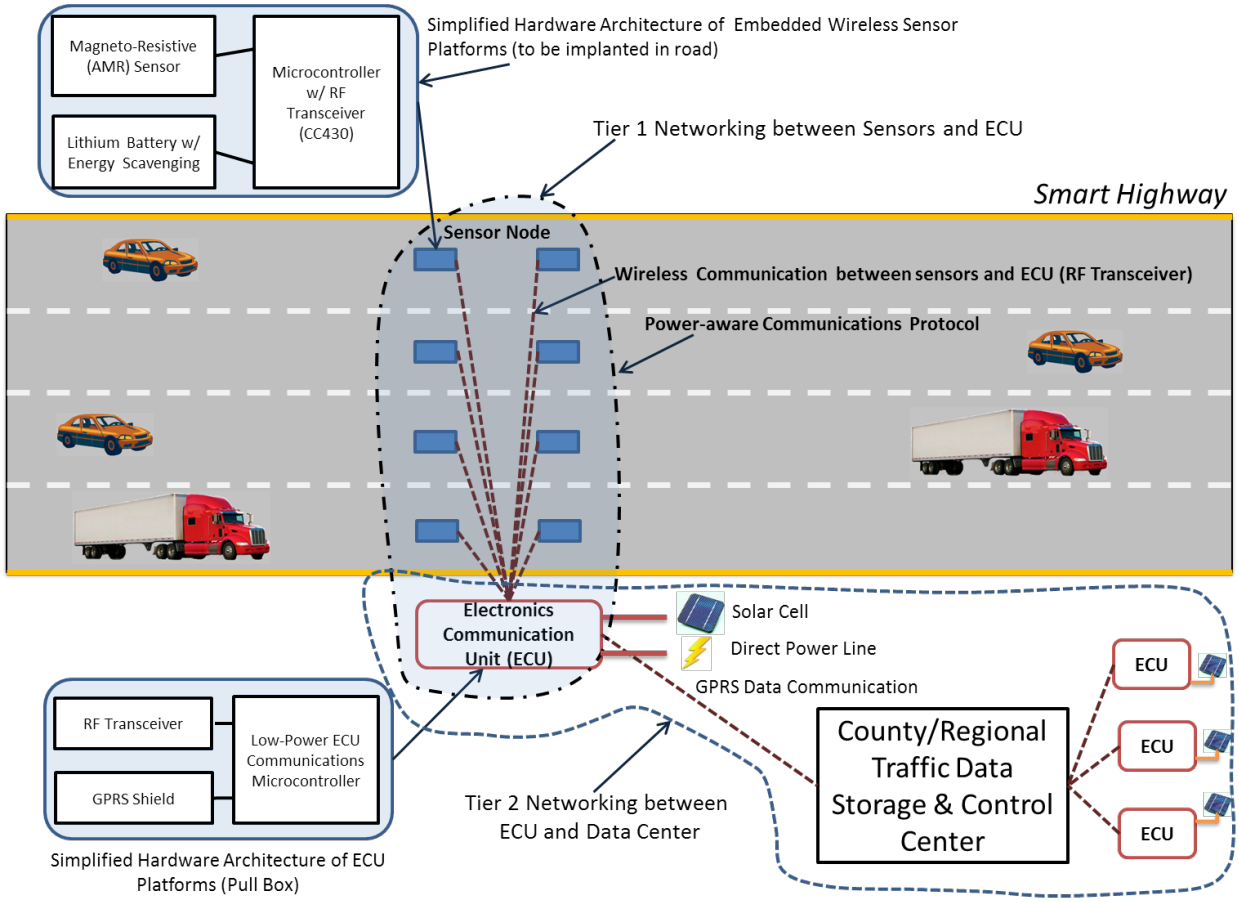

Figure 1: Proposed Architecture for Smart Road Sensing Networks

A high-level system architecture of our proposed method is shown in Fig. 1. The system is composed of sensor nodes to be implanted into the pavement of the road. In each lane, two sensor nodes will be implanted to detect vehicle presence, speed, and type. Each sensor node will be

wirelessly connected with an ECU in a STAR networking topology. The ECU will be an on-site wireless transceiver and controller that will perform traffic data aggregation from the deployed sensor nodes. Furthermore, the ECU will be powered mainly from photovoltaic cells with a compact auxiliary battery. If a direct power supply is available, the ECU can be connected directly to an AC power line. Whenever a sensor node detects any vehicle event, such as arrival or departure, the node will report the event to the ECU with classification information. Once the ECU receives the traffic event packets from the sensors, it will store the data locally and after a certain interval, the ECU will transmit aggregated traffic data to the regional/county-wide transportation data center.

Our in-node approach utilizes TI's CC430 microprocessor to process the data from a magnetic field sensor in addition to transmitting the classification data to the receiver or ECU. The sensor we use is a STMicroelectronics LIS3MDL magnetometer sensor utilizing AMR (Anisotropic Magneto-Resistive) technologies. Most AMR sensors work using a Wheatstone bridge configuration into an operational amplifier. The bridge elements are variable resistors that alter with changes of the magnetic field. Converting this output voltage from the Op-amp using an analog-to-digital converter (ADC) allows for the microprocessor to work with data of the relative magnetic field. The LIS3MDL supplies the magnetic field data on 3-axes in a 16-bit, signed integer format from its integrated ADC over a SPI bus controlled by the bus master. The sensors we are using are set to a sampling rate of 80Hz in continuous (uniform sampling) mode for accurate and consistent data, though the interrupt capability of this sensor allows for the sensor node to be placed into a very low-power state and be woken up only once a vehicle approaches the sensing field of the magnetometer.

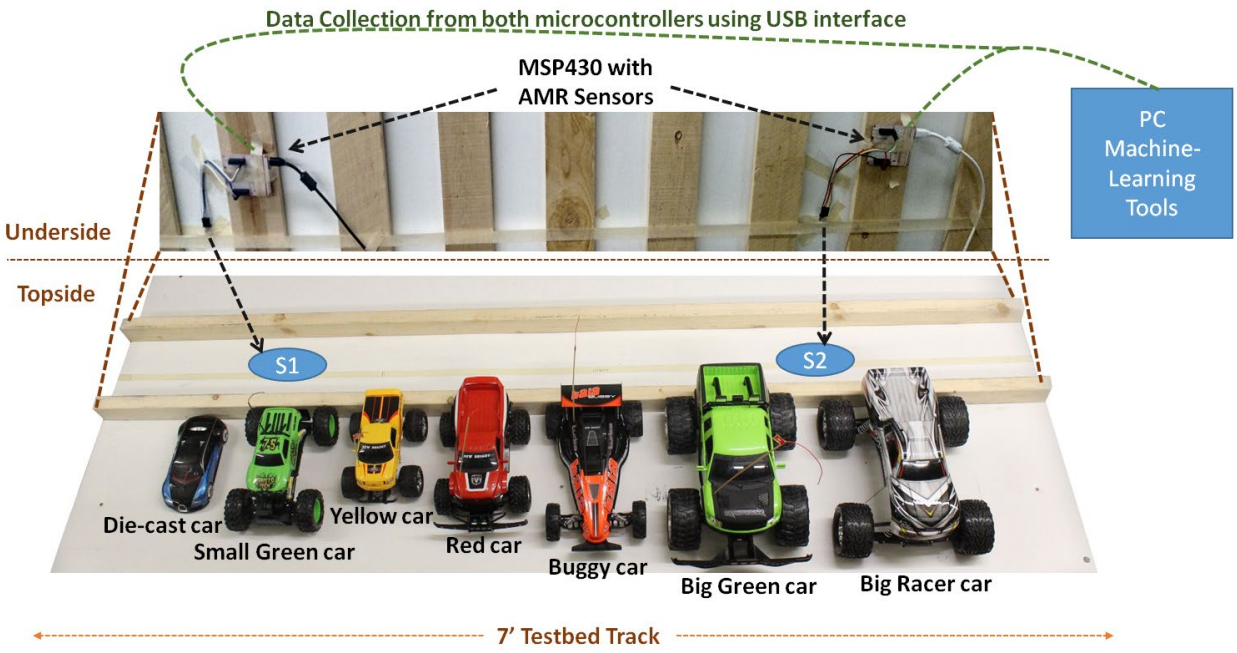

Figure 2: System Test Bed with Lineup of Radio Controlled Cars

Our small-scale test track along with radio controlled (RC) cars that we use as an analog to real world vehicles are shown in Fig. 2. The test bed is 7' long and is composed of a wooden fence section underneath for rigidity and a large plastic sheet above to provide a smooth surface for the RC cars to travel over the sensors. Two wooden guide rails are used to keep the RC cars along the paths of the AMR sensors and to act similarly to marked lanes in the real world. The RC cars tend to stray from driving straight at higher speeds, likely due to poor wheel alignment, causing the data gathered between the sensors to fluctuate wildly, thus requiring the railing. The points marked with the blue circles in Fig. 2 signify the locations where the two STMicroelectronics AMR sensors are located, each attached to a microprocessor used for in-node classification.

To determine when a vehicle passes over our AMR sensor, we first have to determine the background magnetic field for where our sensor is placed; this is the baseline. In order to use our sensor system in multiple environments, the ability to reset baseline values in each new environment

is crucial. The adaptive baseline gives us this ability. Once the baseline is determined, the values are subtracted from each raw axis data to effectively zero out the background magnetic noise floor.

Next, a threshold must be determined such that any ferromagnetic material that causes the magnitude to rise past will start the detection window for the vehicle. The threshold detection value should be adjusted such that only vehicles would be detected while lighter ferrous materials and other sources of noise will stay below the selected threshold. During the detection window, the processor initially extracts raw data used by the PC to calculate vehicle features such as the minimum, maximum, mean, range, standard deviation, and other features of each of the axes' data. The best features according to the algorithm are then implemented in-node with no further assistance from the PC.

V. HARDWARE DEVELOPMENT

In order to produce the desired performance and provide for a long endurance system, new hardware was designed to take account for the low power requirements of the project scope. While the new hardware uses the same processor and radio interface from the previous version, the CC430F5137, but switch out the old Honeywell magnetometer for a LIS3MDL magnetometer from STMicroelectronics. This new magnetometer has several benefits over the older sensor: uses SPI as opposed to I^2C for lower power communication, uses less power during sampling, has a wider set of options for full scale range detection, and can generate interrupts when the magnetic field magnitude exceeds a programmable threshold, thereby allowing for the system to only sample from the device when a vehicle is nearby. Readings are updated by the magnetometer's on-board controller in bulk and a data ready (#DRDY) signal is sent to the sensor node controller to indicate new data is available to be read. Magnetometer data is read over a 3-wire SPI bus.

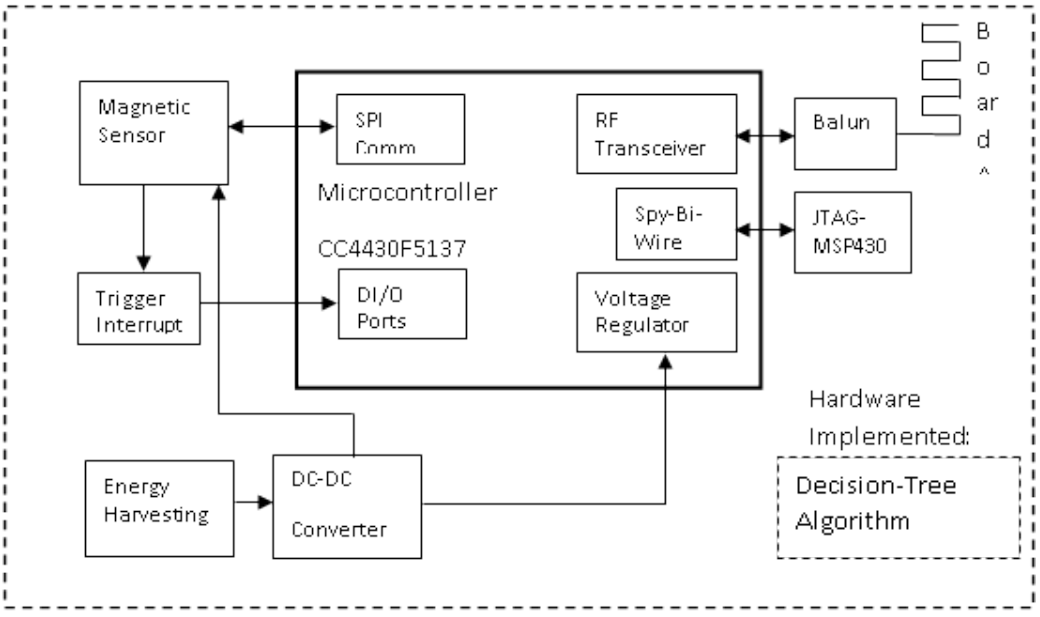

Figure 3: A simplified Wireless Sensor Node block diagram of the hardware architecture

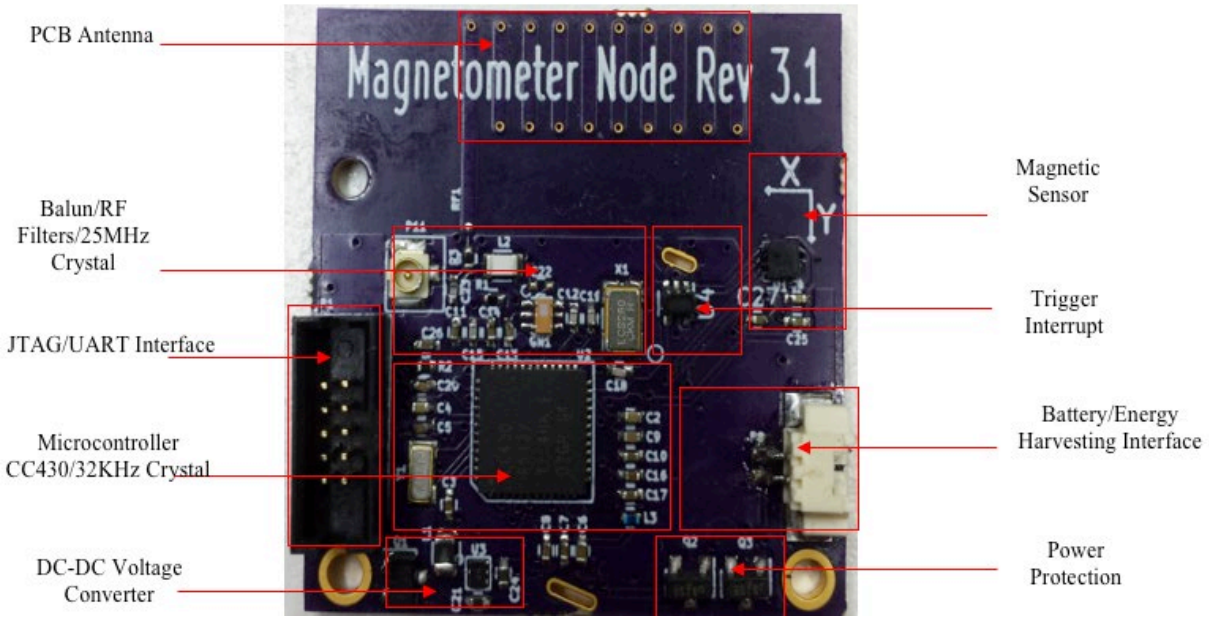

Figure 4: Wireless Sensor node desinged for vehicle detection

The new design also switched out the chip antenna for a meandering PCB antenna as a feasibility approach to implement future cost reduction options, should it become necessary for a production version. This design still utilizes the 915MHz ISM frequency band.

The sensor node uses a 10-pin, 1.27mm pitch header for programming and interfacing to a power analysis module, referred to as the Node Analyzer (NA), shown in Figure 5. The NA is a custom designed power logging tool which allows for sensor power profiling. The NA contains a microcontroller that will be able to communication through a UART to the sensor node and put the sensor node into the commanded state so the power utilization might be read. The NA contains power monitoring circuitry as well as a digitally controlled, variable voltage regulator. The power sourced from the NA is fed into the sensor node during operation while the power usage of the device is closely monitored via the NA.

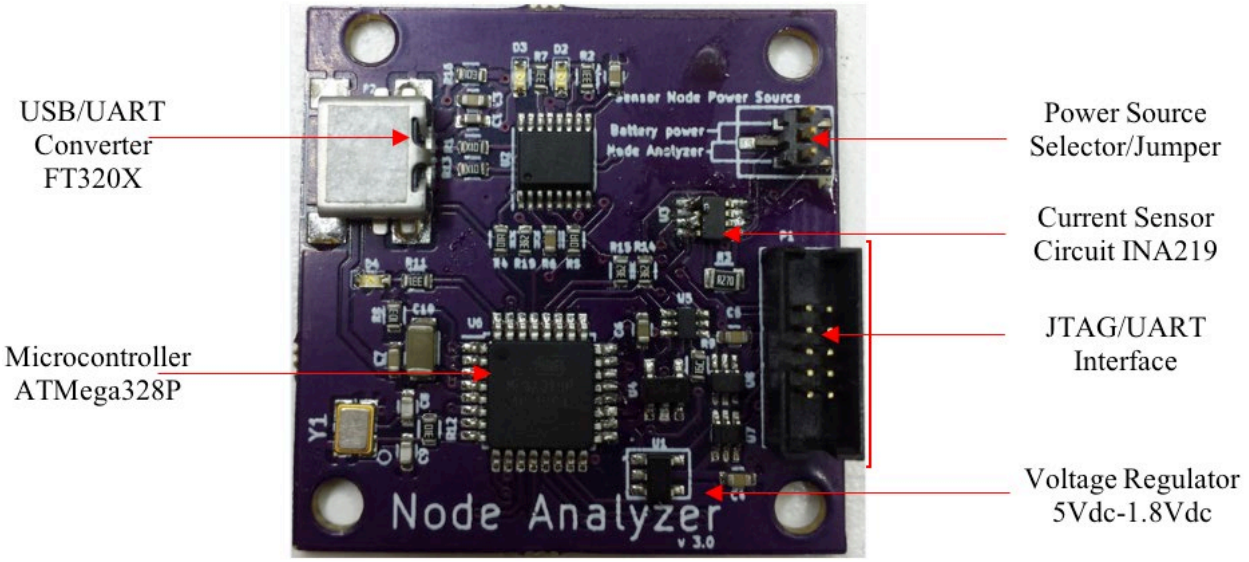

Figure 5: Node Analyzer (NA)

To program the Sensor Node, MSP-FET, or MSP-FET430UIF is required as well as a programming adapter board designed to convert the programmer's 14-pin, 2.54mm pitch header to the 10-pin, 1.27mm pitch header on the sensor node. Sensor nodes are programmed and debugged over the Spy-Bi-Wire interface. The sensor node boards utilize the 10-pin header to reduce the overall sensor node size. The programming setup requires the use of an adapter board to convert the 14-pin, 2.54mm programmer header to the 10-pin, 1.27mm header used on the sensor node (shown in Figure 6).

A. SENSOR NODE HARDWARE DESIGN

As described in the hardware architecture above, the block diagrams are shown here in a schematic format to provide detailed information of each components and circuitry used in the wireless sensor hardware. As shown in Figure 7, this schematic capture and PCB layout is done using the open source KiCad EDA software.

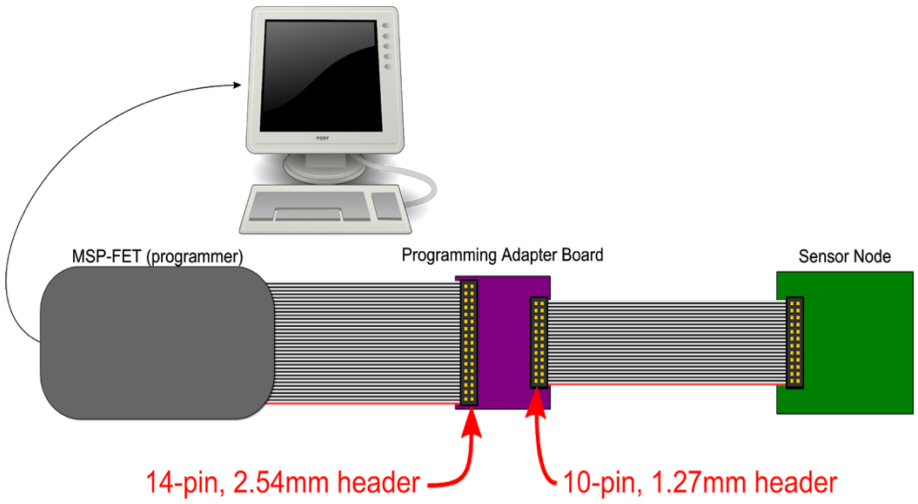

Figure 6: Sensor node programming setup

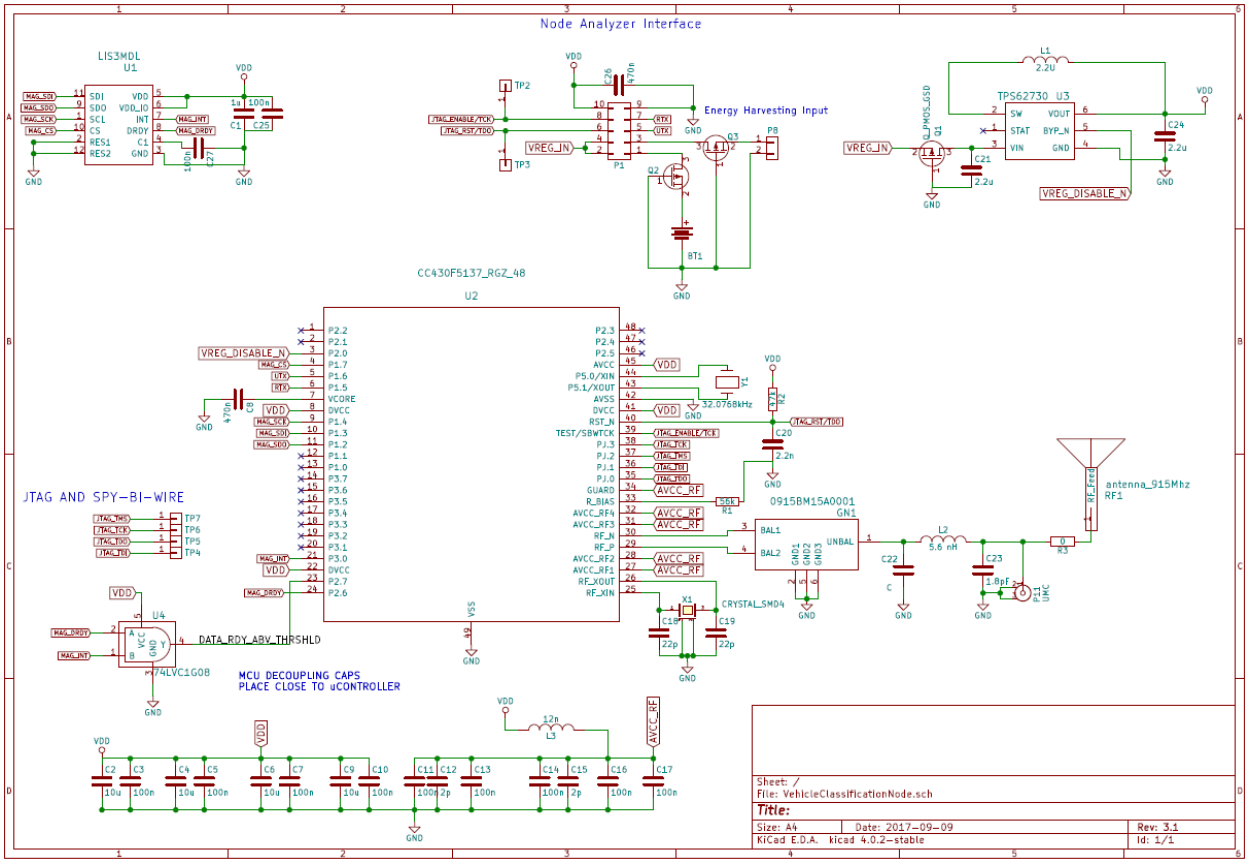

Figure 7: PCB layout of the Wireless Sensor Node

B. HARDWARE COMPONENTS

The Wireless Sensor Node that we designed and tested for the vehicle classification algorithms is shown in Figure 4 and 7. The following major components on the boards and its features are elaborated further in this section.

Microcontroller – The CC430F5137 is SoC from Texas Instruments designed for low-power wireless communication applications with its built-in RF transceiver. It consists of two Universal Serial Communication Interfaces (USCI), where USCI-B0 is used for the SPI to interface to the magnetometer. In the schematic the SPI lines where mark as MAG_SDI, MAG_SDO, MAG_SCK, and MAG_CS, which stands for the magnetic sensor to the microcontroller with the corresponding description: Slave Data Input (SDI), Slave Data Output (SDO), Slave Clock (SCK), and the Chip Select (CS), respectively. These lines serve the data and configuration commands between the CC430 microcontroller and the magnetometer.

Magnetometer – the LSI3MDL is connected through the SPI lines. The sensor is capable of providing interrupt capability for when the magnetic field intensity changes beyond some determined threshold. The magnetometer is able to select a full-scale range of ±4/±8/±12/±16 gauss.

Trigger Interrupt – 74LVC1G08 is an AND gate logic circuit which allows the processor to be configured to interrupt when both the magnetic field is above the interrupt threshold and there is data ready.

Energy Harvesting Interface – is an input port for an energy harvesting power source beside the Lithium-based, 3.3V 1Ah CR2477 battery coin-cell. In this project, two MOSFETs are used for reverse polarity protection for both input power sources.

DC-DC Converter – TPS62730 – is step-down DC-DC converter optimized for ultra-low power wireless applications. It has an input voltage range from 1.9VDC to 3.9VDC where in our case we convert the 3.6VDC to 2.1VDC. Also, the TPS62730 provides up to 100mA output current and allows the use of tiny and low cost chip inductors and capacitors to further reduce the size of the sensor node.

Balance-Unbalance (Balun) – 0915BM15A0001 is a 915MHz impedance matching circuitry for RF transceiver of the CC430F5137 chip. It is connected in between the microprocessor chip's RF front and the inductor-capacitor (LC) bandpass filter of the antenna.

PCB-Antenna – the radio antenna is a PCB design, etched to the tracing on the PCB. This approach provided for a medium size antenna that fits perfectly on the 30mm x 30mm PCB and is low cost. This antenna design was based on the Evaluation Module (EM) board recommendation for 868/915/955 MHz PCB spiral-type found in Application Note 058 from Texas Instruments. This approach allows for a reduce component count and serves as a feasibility study as an option for cost reduction.

Spy-Bi-Wire JTAG Protocol - is a serial communication protocol for programming and debugging firmware using Spy-Bi-Wire interface on the programmer. The pins from the microcontroller are then connected to the Molex 10-pin header for connection to the MSP-FET programmer.

C. HARDWARE FOR POWER PROFILING

Another board that we developed is the Node Analyzer (shown in Figure 5 and 8), as discussed in the above section was used in order to determine the power consumption of the Wireless Sensor Node board in the actual implementation. The NA is composed of a "current sensor" that was used for monitoring the power consumption of the attached sensor node. The main idea of implementing low power profiling to the SN board is to be able to make sure the

sensors can keep on running for years without replacing the battery. To realize this scenario, we developed this board - Node Analyzer board (NA), which is used to determine the power consumption of the SN board. The schematic of the Node Analyzer board is shown in Figure 8.

D. NODE ANALYZER BOARD

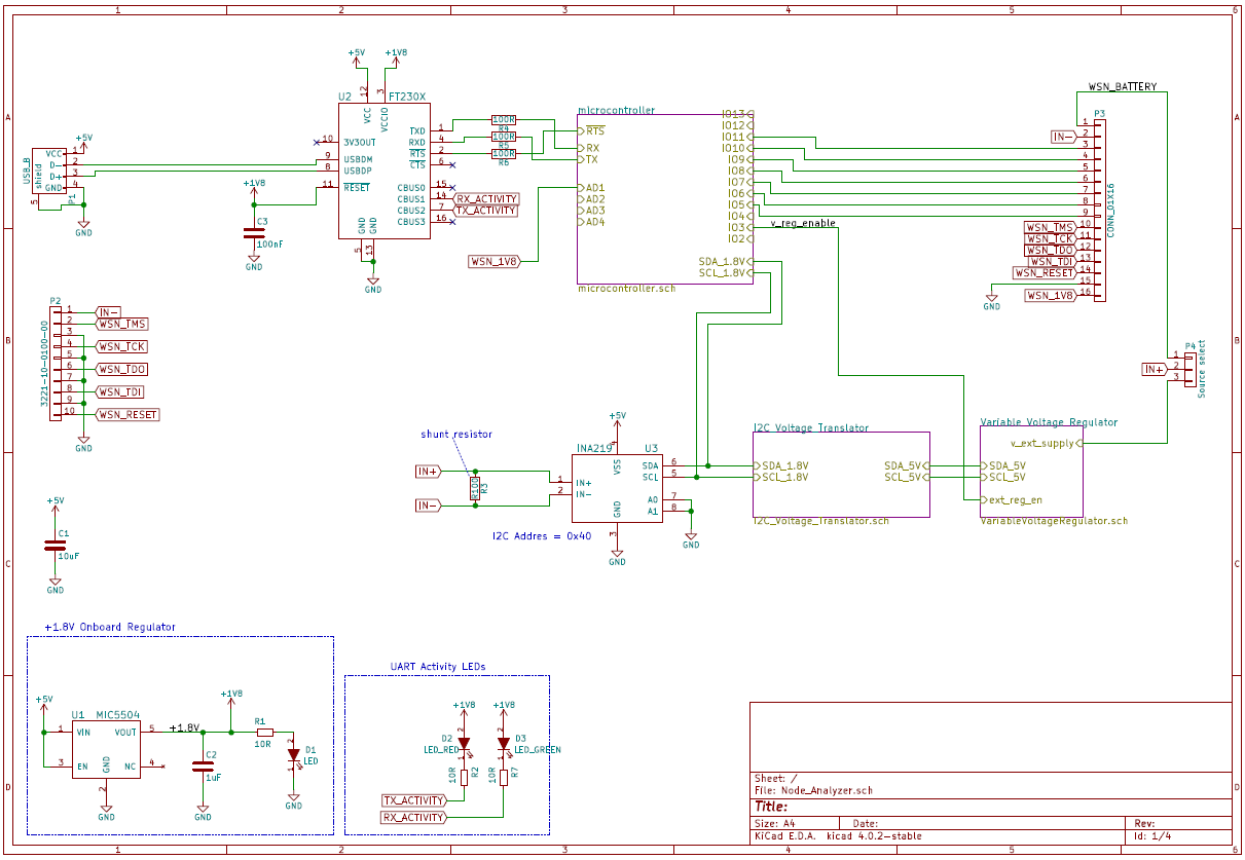

Figure 8: Schematic diagram of the Node Analyzer

The NA board is mainly composed of the following chips:

ATMega328P: Microcontroller that reads the current consumed in the SN board through other component on the NA board, which is the **Current Sensor INA219** chip. It is also composed of a voltage regulator that provides the power to the SN board while under test. Also, a **UART-to-USB chip FT320X** is used to interface to a PC that runs the **Node Analyzer Companion Software (NACS)** that provides the control of the power profiling sequences and reads the log data to view

the power consumption. The NA board is able to control and analyze the power profiling sequences of the SN board and to be able view the power consumption parameters while under test.

UART communication: The UART connection to the node analyzer will enable for a set of commands. Some specific to the analyzer and some intended to be sent to the sensor nodes. The profiles sent to the sensor nodes are static and arguments are not needed. Therefore, a single byte will be encoded to determine what operation to take. Commands for the analyzer are for power settings, or resets. The node analyzer will continuously send power information and a timestamp of the operation to the PC. There is a companion application that provides a GUI of this information. This data is the main advantage which will enable better refinement and iterations of device settings or algorithms.

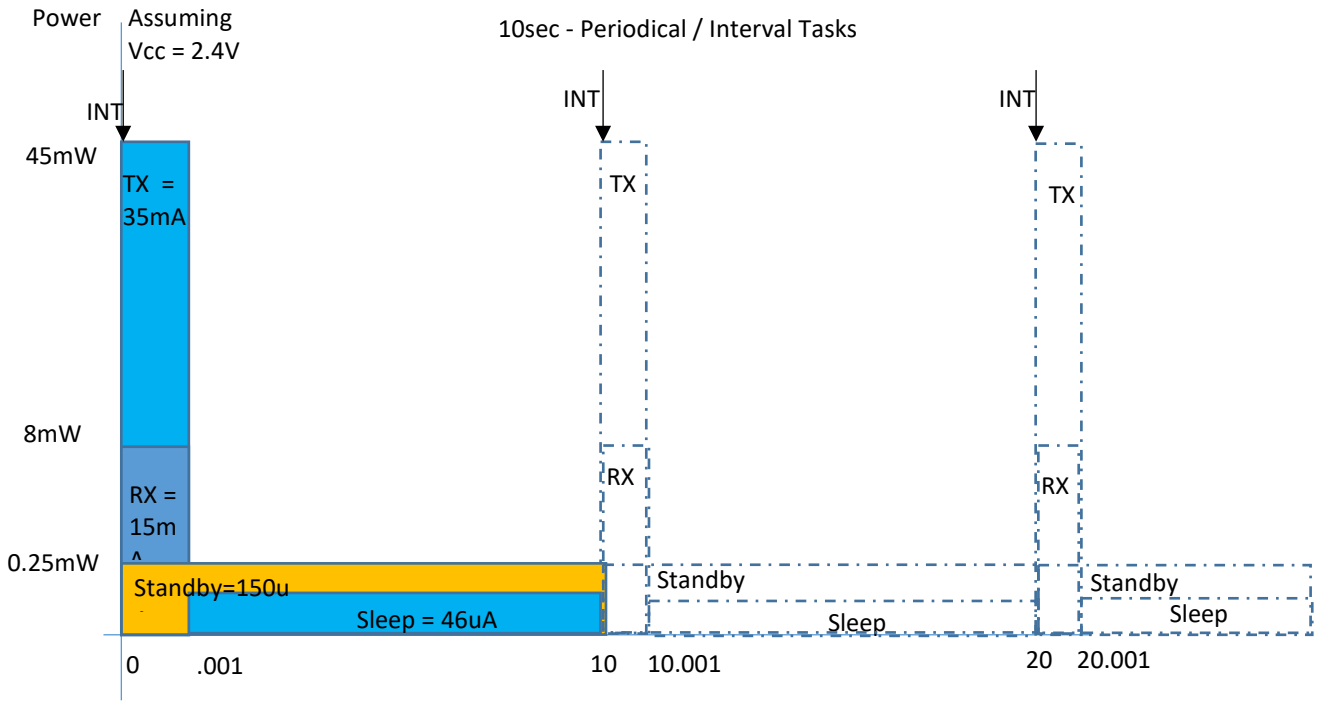

Figure 9: Sample power profiling capture, denoting power consumption levels at various operating modes.

E. POWER PROFILING TASK

We have included here an example of a power profiling graph that could be applied to the Wireless Sensor Board which can be controlled by the Node Analyzer board. This power profile graph shown in Figure 9 indicates that when an interrupt from a vehicle passing over the Sensor Node, the SN starts to process the data, extract the features, perform the Decision-Tree algorithm, and transmit the result for about 0.001ms with a current consumption of 35mA to the receiver. Once the processing the transmission is done the Wireless Sensor Board goes back to Standby Mode or Sleep Mode, consuming approximately 156uA and 46uA, respectively, until the next interrupt occurs. While no interrupt occurs, the SN board will remain in a low-power mode, thus, consuming less power that would allow the battery to last for years without any replacement.

VI. VEHICLE CLASSIFICATION AND RESULTS

The arduousness of vehicle identification partially stems from the difficulty of visually observing a single vehicle amongst a multitude of surrounding vehicles that are moving in parallel or opposing directions. Even so, there are a number of external and internal characteristics that can't be easily observed, which prevents these characteristics from successfully classifying these vehicles. Due to this, this task has the potential to be more feasible with the use of wireless sensors that are embedded within the pavement of these roads. In our research, we utilized a sensor that collects the distinctive features that will greatly assist us in the process of vehicle classification, and to further clarify the procedure we enacted, vehicle detection involved the use of a 3-axis anisotropic magnetoresistive (AMR) sensor that is wirelessly connected to a central controller while it obtains data detected from passing vehicles. These sensors measure the amount of variation in the magnetic field (MGauss) in x, y, and z directions based on the direction of the electric current and magnetization [21].

By using these embedded systems and machine learning algorithms, we are also simplifying the process of vehicle classification. Moreover, in implementing an approach that involves machine-learning algorithms, these systems are able to observe and determine various patterns of the collected data and associate them to a vehicles classification. There are many approaches to how machine learning could be used for the present task such as: Decision Tree Classification, Linear Regression, Support Vector Machines, Neural Networks, etc. However, the general methods that should be considered are machine-learning algorithms that can be used for automated classification. We have considered applying a select few of the many possibilities of machine learning classification algorithms, and generally, despite the fact that the aforementioned classification algorithms differ conceptually in terms of how it performs acts of classification, there are certain parallels including the goals each algorithm aims to achieve as well as the needed information

required for each machine learning algorithm. The machine learning classification algorithms used in our research follow the principle of supervised learning, which uses datasets composed of multiple data samples where each set of data represents a certain sample [22]. Attached to these samples are labels, which indicate the type of classification the data corresponds to, and the nexus between the label and the data sample is essential considering that it allows us and machine learning algorithms to distinguish the types of classifications that are associated to certain types of samples. Supervised machine learning algorithms require two different types of datasets: a training dataset that represents a sample of the data collected that will be used to effectively train the machine learning algorithm in order to predict the classification of a new set of data and a testing dataset, which, after the machine learning algorithm has been trained, will then be analyzed by the classification model. Broadly, the machine-learning algorithm observes each of the sample data while remaining unaware of the labels associated with it. The machine learning algorithm attempts to predict the classification of the sample data it observes by basing its analysis from the dataset that was used in order to train the machine learning algorithm. After the machine learning algorithm predicts all of the sample data from the testing dataset, we are able to measure its performance by juxtaposing the algorithms predicted classifications with real classifications from the labels in the provided testing dataset. After we determine the performance of the implemented machine learning classification algorithms, we can then execute our methodology to observe which classification algorithms achieve the best results.

In order to provide an overview of each classification algorithms performance, we applied Machine Learning based Vehicle Classification (MLVC) architecture; furthermore, a software implementation was designed to obtain, process, and display necessary experimental and resulting data. Python was used for the proposed architecture with the use of libraries to help process and

output data. These libraries include: scikit-learn [23], openpyxl [24], matplotlib [25], etc., and in utilizing these libraries, we will be able to implement the proposed architecture in order to solve the issue of classifying vehicles using machine learning algorithms. Figure 10, depicts the flow diagram of the MLVC architecture:

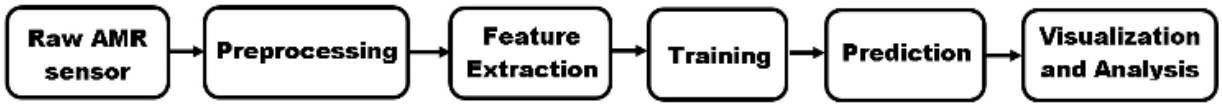

Figure 10: Flow diagram of MLVC Architecture

The first block in the architecture is where we retrieve raw data from the sensor, consisting of measurements from the AMR sensor that detect changes in the magnetic field with relation to the x, y, and z dimensional space (as shown in Figure 11 and 12). Each sample of data consists of a trial run where an experimented vehicle is driven over the sensor, thereby allowing the sensor to collect data. In addition, all trial runs would be stored onto a readable file that would be used in the next blocks of the architecture.

Figure 11: AMR Sensor data in three dimensional space

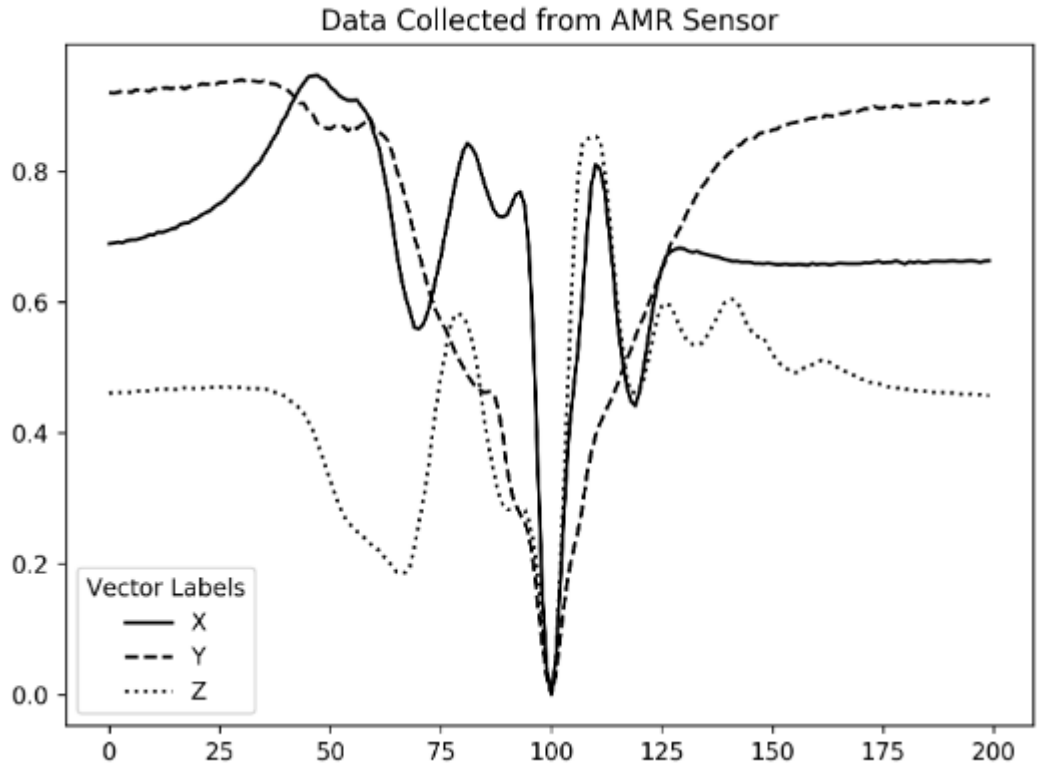

Figure 12: AMR Sensor data when car passes over sensor

Our trials were done with six different vehicles: three sedans (2003 Honda Accord, 2009 Honda Accord LX, 2007 Hyundai Elantra), two hatchbacks (2004 Ford Focus ZX3, 1999 Subaru Outback Impreza), and one truck (2001 Chevrolet Silverado LT), and it is after retrieving the raw sensor data that it would be preprocessed before any features could be calculated and extracted from the raw data. Here, we have created a training dataset and testing dataset, and we project that any implemented machine learning classification algorithm in our research would be able to analyze and distinguish the vehicles using the dataset. The implemented machine learning algorithms require that the datasets be in a specific structure; thus, it is imperative that the raw data from the sensor be preprocessed for it to meet the standard.

In the preprocessing of the raw data, we reconsidered the vectors that would be applied when generating which datasets for the machine learning algorithms to employ. Rather than utilizing the

raw x, y, and z axes from the data collected from the sensor, we considered vectors z, xy, and xyz. Where z consists of the raw data from the sensor according to the z axis, xy and xyz signify the magnitude of the specified vectors [26], which is calculated as follows:

$$z = Z$$

$$xy = \sqrt{X^2 + Y^2}$$

$$xyz = \sqrt{X^2 + Y^2 + Z^2}$$

In order to allow the machine learning algorithms to reduce any inaccuracies in the data, it was necessary to crop the data to remove background noise (depicted in Figure 13). This would allow a cars processed data to not only consist of when the car is detected by the sensor but also the moment it has left its scope and ensures that the data only comprises relevant vehicle data once it is processed.

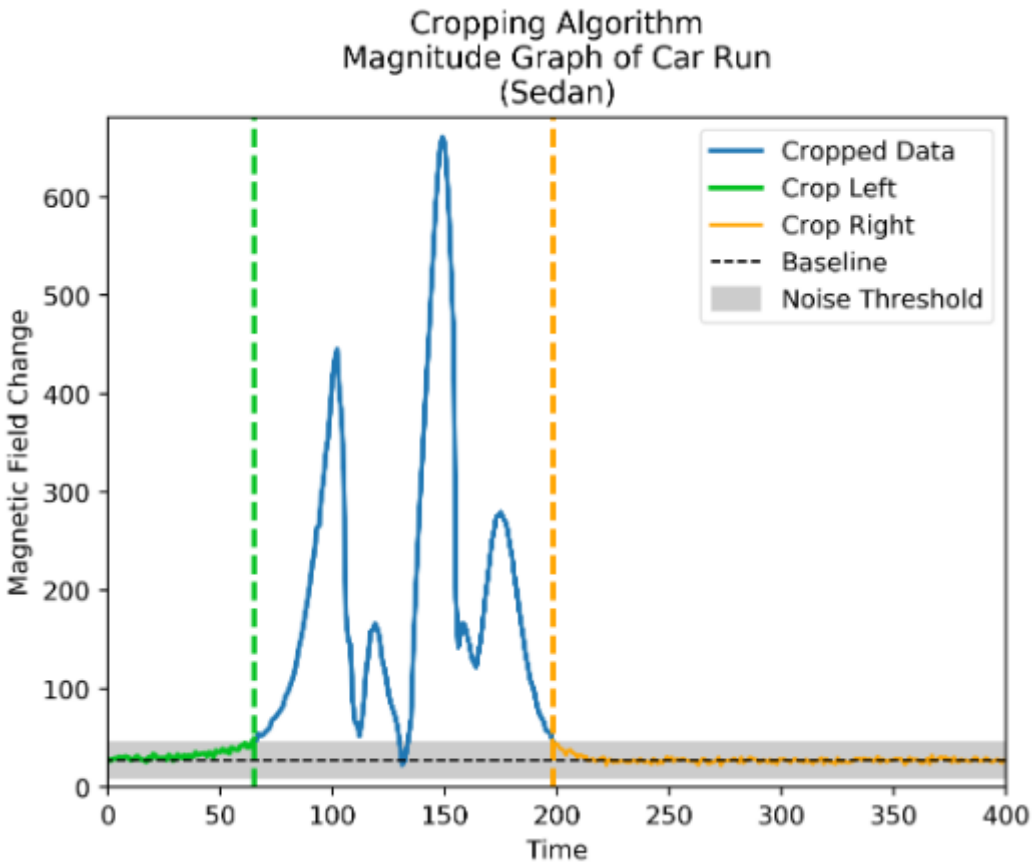

Figure 13: Cropping sensor data

There was also the consideration of applying different types of transformations to the raw data in order to compare the different results. For instance, we considered normalizing the raw data when moving through the Machine Learning Architecture. Applying this transformation allows us to compare the results of two instances when testing the classification algorithms: one, where only raw data is used, and two, where the raw data has been normalized.

In Machine Learning, features are unique characteristics or attributes that can be measured to produce a numeric value; furthermore, they are essential. Analyzing them will allow us the ability to distinguish the classifications of each vehicle in our sample. For vehicle classification using AMR sensors, such features signify the changes in magnetic fields in x, y, and z directions, and in using this preprocessed data, we can apply more diverse transformations (either statistic or

frequency-based) to the data in order to obtain clear results. The statistical features comprise: minimum, maximum, mean, median, peak-to-peak (p2p), root-mean-square (rms) [27], p2p-rms, standard deviation, variance, skewness, and kurtosis. The frequency-based features encompassed Fast Fourier Transform characteristics which comprised: fftmagmax, fftmagmax, and fftmean. Likewise, we must consider the vectors that are relevant for each sample, which are the aforementioned Z, XY, and XYZ vectors. For our experiments, we gathered the preprocessed data pertaining to the vehicles and focused on applying a combination of features to a particular vector. This resulted in the final step of preparation for the training and testing datasets so that they could be applied onto the Machine Learning Classification Algorithms.

Throughout this project it was observed that some features yielded more favorable results than others and that some combinations of features worked together more cohesively than others. We primarily used the Classification Tree and Random Forest algorithms in order to determine which features had the most influence in the classification process. However, statistical tools such as Scatterplot Matrices were also used in order to visually compare the feature data for each vehicle upon each considered vector. When a successful combination of features are used together in a classification algorithm, they typically achieve high scores in the predictions phase. One of the primary reasons we spent time searching for a wide array of features was to conduct a survey to test their performance and cohesion when used as combinations in various classification algorithms. In addition to applying brute force approach to test the prediction accuracy, we also used Scatterplot matrices to aid in our search for the optimal combination. One of the scatterplot matrices is shown in Figure 5 which demonstrates the relationship between peak2peak value of xyz vector, mean value of xy vector, median value of xyz vector, and mean value of xyz vector.

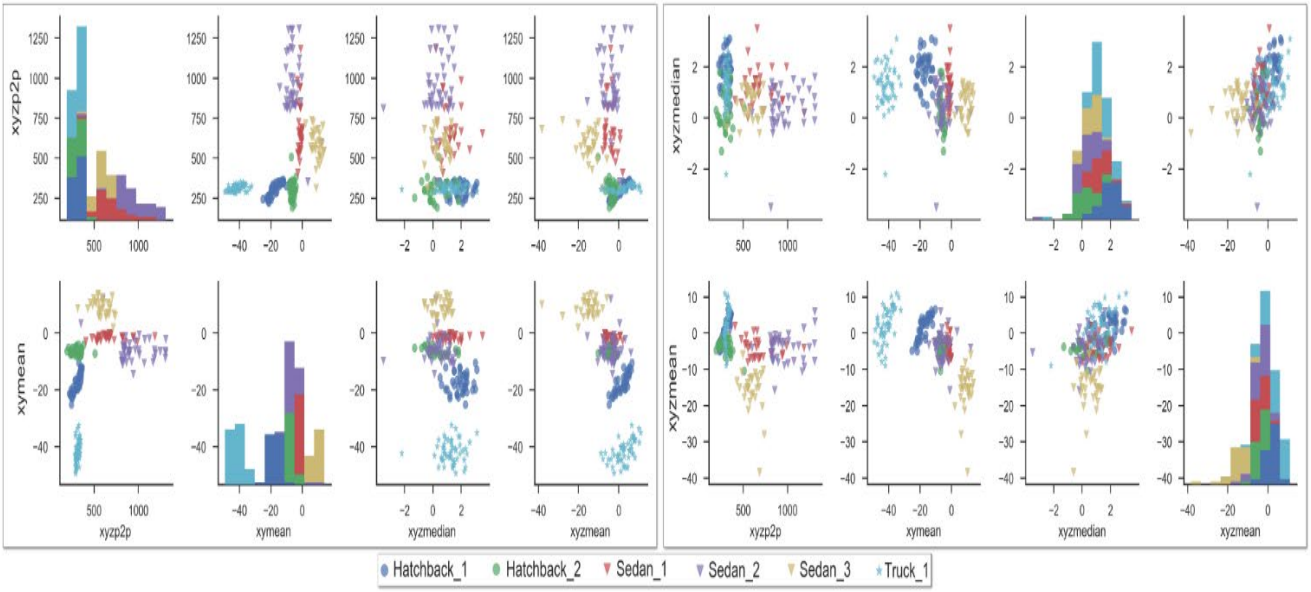

Figure 14: Scatterplot Matrix of xyz_p2p, xy_mean, xyz_median, xyz_mean

Throughout this project, it was observed that some features yielded more favorable results than others and the combinations of features worked together more cohesively. Primarily, we used the Classification Tree and Random Forest algorithms to determine which features had the strongest influence in the classification process; however, statistical tools such as Scatterplot Matrices were also used in order to visually compare each vehicles feature data upon each considered vector. When successful combinations of features are used together in a classification algorithm, they typically achieve high scores in the predictions phase, and one of the primary reasons we searched for an eclectic array of features was to conduct a survey to test their performance and cohesion when used as combinations in various classification algorithms. In addition to applying a brute force approach to test the prediction accuracy, we also used Scatterplot matrices to aid in our search for the optimal combination. One of the scatterplot matrices is shown in Figure 14, which demonstrates the relationship between peak2peak value of the xyz vector, mean value of the xy vector, median value of the xyz vector, and the mean value of xyz vector.

In consideration of our exploration for the ideal feature combinations, we must acknowledge the limitations of our testing data and how external factors may interfere with our features. Although some features might seem to produce favorable results regarding vehicle classification when trained on numerous iterations of the same car, it is naive to think that all cars of a single make and model will be identical on public roadways. Statistical features such as min and max, although scoring high classification rates, are negatively affected by certain factors (e.g. improper installation, calibration error, carriage weight, or tire pressure) that may cause slight height differences between the car and the sensor. An improperly installed sensor might be off center or at a different depth in the pavement, which could exacerbate errors in a rigid prediction model. Similarly, a lowered car will most likely read higher levels of magnetic field disturbance due to its closer proximity to the sensor. Although one could argue that a minimal difference in proximity in the magnitude will not affect the sensor readings on a macro scale, we can't make assumptions until more comprehensive test samples are gathered. Because of this, these factors are a concern that will likely need to be addressed in the future.

The machine learning classification algorithms that were implemented in our research were: **Decision Tree Classification Algorithm, Random Forest, and Multilayer Perceptron**. The implemented machine learning classification algorithms would go through the training process with the necessary dataset before analyzing the testing dataset and predicting the classifications of the vehicles within the sample data. Also, we will use Decision Tree and Random Forest algorithms regarding these classification algorithms in order to determine which features had the largest influence in the classification process. Then, we will use these features in Multilayer Perceptron.

A. DECISION TREE ALGORITHM

In our previous research [28], we made use of algorithms, such as C4.5a machine learning algorithm that outputs a decision tree model structured on the number of feature or attributes that are used to train the model of the decision tree [23]. This study proposes a similar method of classifying the sample data, but instead, it uses the Classification and Regression Tree (CART) Algorithm in order to generate a decision tree. Moreover, the CART implementation processes the training data sets that it receives and determines which features possess the most variance between each vehicle [26]. In other words, we fed an abundant amount of features and allowed the algorithm to determine which features were the most important. The algorithm then defines a certain threshold for each important feature in which a particular class of vehicles would either surpass or sustain. After defining the threshold for each comparable feature, the decision tree algorithm would generate an appropriate structure allowing it to classify vehicles based on their features. Moreover, the advantages of the Decision Tree Algorithm permit us to easily understand the decision making process structure for the algorithm. For instance, by observing a visualized decision tree of the algorithm, we are able to see the possible decision making paths that the algorithm traverses. Once this tree structure is generated, it is quite feasible to implement in low-powered electronic devices such as our vehicle classification node device; however, some notable flaws of the Decision Tree Algorithm are that the structure of the tree that was generated must be balanced in order for the algorithm to perform efficiently. Unfortunately, this is not always a guarantee when the tree is generated. By applying the testing dataset onto the CART algorithm, each data sample from the dataset is analyzed by the decision tree root's machine learning technique. Here, it is compared to the threshold defined at each node of the tree and traverses to either the left or right child of the node depending on how the vehicles data equates to the threshold defined in that node. A prediction is made when the decision is rested on a leaf of the tree [29].

B. RANDOM FOREST ALGORITHM

Another machine learning approach related to the methods used in the Decision Tree Classification Algorithm is the random forest method, which is deemed as an ensemble machine learning technique that utilizes a set of classifiers and averages their predictions before applying their decision onto an instance of data [30]. The random forest method uses the principle of a decision tree. Nevertheless, it expands the concept further by training multiple decision trees based on subsets of the dataset that are inputted into the machine learning algorithm in order to create a forest of decision trees [22] through the training process. The process consists of building each decision tree from the training datasets bootstrap sample where a random subset is selected from the dataset, and by creating multiple decision trees, the machine learning algorithm can combine the predictions of all the trees at their disposal. While this would potentially increase bias within the prediction process, the variance decreases upon using this technique. With these statistical differences, combining the predictions from the decision trees in the random forest algorithm potentially increases the accuracy of its predictions, which is significant compared to those made by relying on a single decision tree. Once this decision tree is made, it is easily implementable into low-powered electronic devices, including our vehicle classification node. Unfortunately, due to the amount of decision trees that would be utilized, both the training process and testing process can be costly in time and memory resources [22].

The prediction process abides similarly when applying the testing dataset onto a single decision tree; however, since the random forest is an ensemble machine learning technique, the process is compounded numerous times based on the amount of decision trees generated from the random forest method. The algorithm then averages all of the predictions made from each application to conclude a decision of a sample data link to a vehicles particular classification.

C. MULTILAYER PERCEPTRON (MLP) ALGORITHM

Another machine learning approach used in our project is the Multilayer Perceptron (MLP) classifier; it is a common subset of forward feeding neural networks that utilizes one or more hidden layers of perceptrons that can approximate non-linear functions of the input [22]. Training data is passed into the nodes of the input layer each representing a feature where the data gets fed through the neurons. As the data is passed through the neural network, the errors are calculated, and minor updates to the weights of the nodes are back propagated into the network. The nonlinear discriminant perceptrons in the hidden layers allow MLP to identify complex nonlinear relationships such as those found in vehicle classification. In fact, artificial neural networks such as MLP have been used extensively in studies involving vehicle transportation [31]. In our project, we used the Stochastic Gradient Descent (SGD) learning mode (an iterative optimization technique that is used to find the local minima of a cost function by making an initial estimate and updating it in small steps until the derivative reaches zero [22]) to train our MLP neural networks. We experimented with our alpha tuning parameter by using values between 0.0001 and 0.8 to find the most optimal settings for convergence in our training algorithm. Also, it is necessary to note that Stochastic Gradient Descent is an efficient training algorithm in large datasets since it does not require going through each sample to come to a conclusion. This makes SGD the de facto training algorithm in back propagation neural networks, especially considering that the networks often have very large training sets [32].

D. PERFORMANCE ANALYSIS

The classification algorithms implemented in our project include Decision Trees, Random Forest, and Multilayer Perceptron. Each algorithm yielded different results in performance and prediction accuracy. Principally, some algorithms performed more favorably depending on the combination

of features that were applied on the dataset it was trained on. On the contrary, there were also algorithms that performed less efficiently than average when using certain feature combinations. The use of Confusion Matrices allows us to visually determine a classification algorithms accuracy by comparing its predictions with the true value when analyzing a testing dataset.

The results of the CART Decision Tree Algorithm were generally favorable, with most feature combinations scoring an average prediction accuracy of 94% when using a raw dataset from statistical features or frequency based features. However, using a normalized dataset with statistical features yielded results approximating 90% (Figure 15). While the scores of a raw dataset are higher, it is generally more applicable to use a normalized dataset since the scalability would remain consistent when using different sensors; this is contrary to a raw dataset, which is not likely to possess such results. Using the frequency based features, the classification tree algorithm yielded mixed results with accuracies as high as 85% and as low as 77%, indicating the features ineffectiveness in the classification process. Using the decision tree algorithm, we noticed the significance of the statistical features including variance, kurtosis, Fig. 6: Normalized Confusion Matrix: Classification Tree z, xy, xyz - median, mean, rms, p2p, p2prms, stdev, variance, kurtosis, skewness and the skewness of a dataset. By using these features, the classification tree was able to perform better than other instances that did not use these features, which would indicate that such statistical features would have a strong influence in the classification process. Due to its inheritably low computation costs, the decision tree outperformed most of the other algorithms, and for a sensor node implementation, decision trees provide the easiest and most direct solution because it is fundamentally based on conditional logic [29]. Given these qualities, the CART decision tree

algorithm is a considerable method for vehicle classification as it allows us to visualize the decision making process (Figure 16).

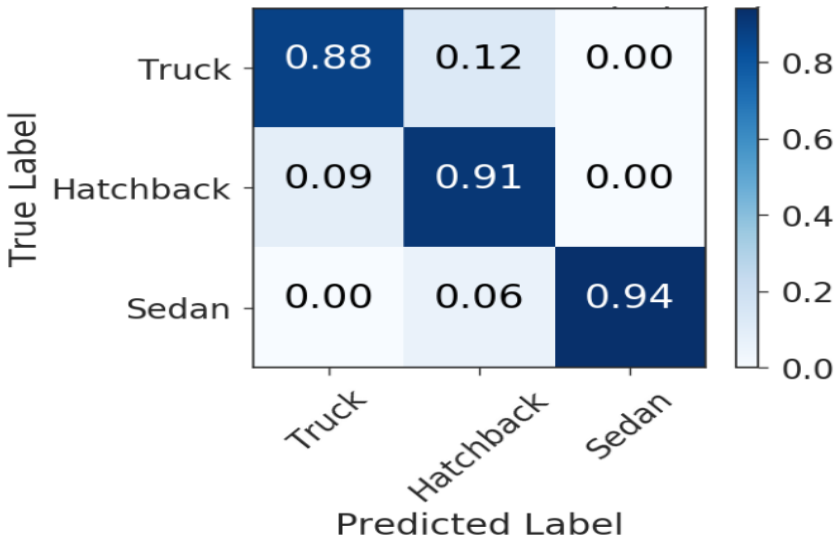

Figure 15: Confusion matrix for classification tree algorithm

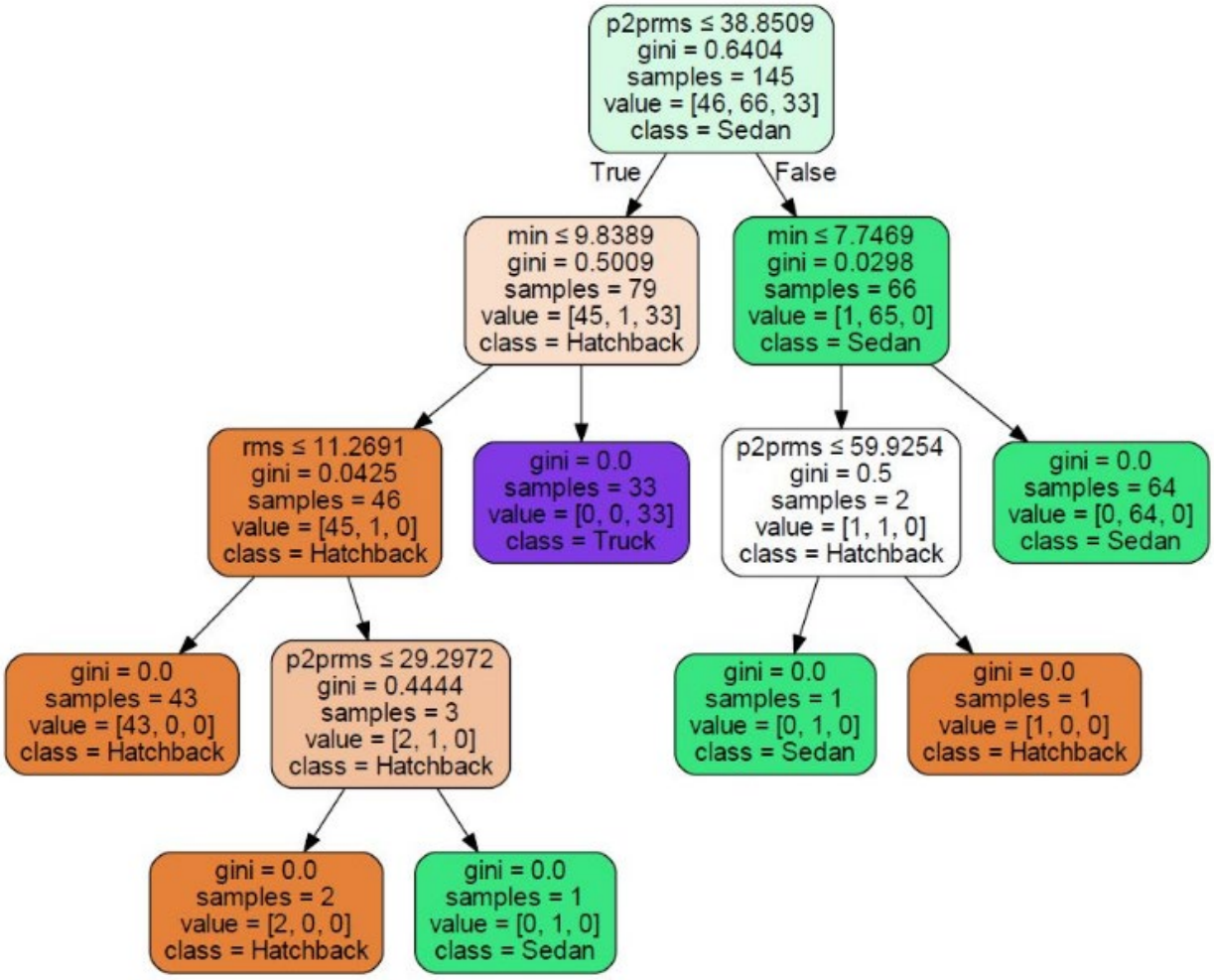

Figure 16: Decision Tree Classification Demonstration

The Random Forest algorithm performed slightly better than the Classification Tree algorithm with most results averaging a 93% prediction accuracy when using statistical features from a normalized dataset. This is contrary to the Random Forest algorithm, which averaged an 87% prediction accuracy when using frequency based features upon a normalized dataset. While the random forest algorithm may have produced slightly better results than the classification tree algorithm, it is essential to note that the computation level exceeds that of the classification tree algorithm. As mentioned in the analysis of the classification tree algorithm, having more features considered in the random forest algorithm would result in the same effect on the level of

computation required. In other words, the more features that the algorithm needs equates more computations. When applied onto a sensor node for vehicle classification, this would have detrimental effects, which are further compounded considering that the random forest algorithm uses multiple decision trees when formulating its decisions. Generating multiple decision trees would certainly have a mass effect on the computation needed to determine a vehicles classification; thus, it is important to optimize the use of the random forest algorithm so that no additional computation is needed when ideal results can be achieved via an adequate performance. One form of optimization is limiting the number of decision trees generated by the random forest algorithm. Figure 17 shows that the random forest algorithm is able to achieve an ideal level of prediction accuracy with less trees than instances (note: especially when compared to those used by the algorithm). If we applied the random forest algorithm onto a sensor node, we could tune the algorithm so that it only generates the necessary amount of decision trees.

An alternative to optimizing the random forest algorithm is restricting the maximum depth of the tree's structure, which would reduce the computation level at the potential cost of reducing its ability in attaining a high prediction accuracy. Figure 18 depicts the influence of the trees depth on the algorithm's total accuracy.

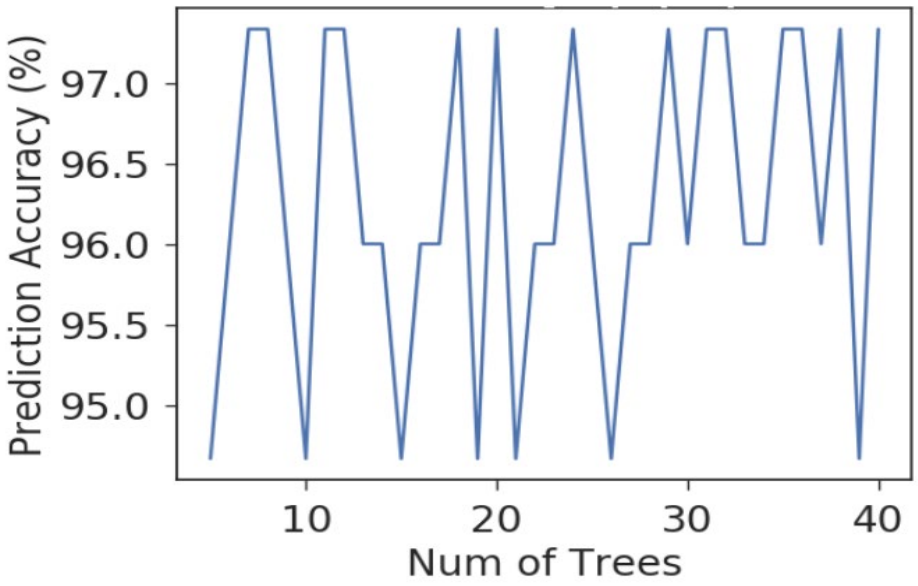

Figure 17: Number of Trees in Random Forest VS Accuracy

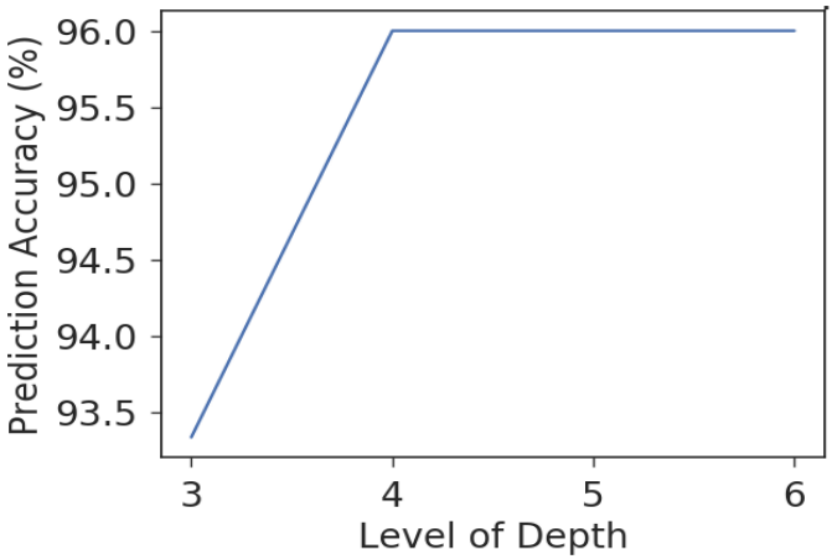

Figure 18: Depth of Trees in Random Forest VS Accuracy

Multilayer Perceptron was another promising classification algorithm that we focalized in our research, and it is essential to note that MLP and neural networks operate as a universal tool for solving an array of classification problems, while providing a variety of tuning parameters that optimize both its speed and complexity. Additionally, the fact that it can solve complex problems, such as car classification, proved its necessity for our project. After tinkering with tuning parameters

to find the most optimal combinations, they provided accuracies above 85 percent, with some even scoring in the high 90 percentiles. Contrarily to Decision Trees, MLP supplies us with a higher degree of flexibility and sustainability but at the cost of lower efficiency and difficult implementation. Although neural networks are one of the more computationally expensive classification techniques, it is imperative to recognize that with the acquisition of more sample data and vehicle categories, a properly implemented MLP algorithm will have a higher scalability than other more rigid models. Essentially, having a higher-level solution provides a safety net in the event that simpler methods (e.g. decision trees) become inadequate. In such cases, we could adjust our architecture by outsourcing road sensor data to a powered central processing node, which could process and classify with more powerful and sustainable algorithms.

VII. CONCLUSIONS

In this project, we have designed Wireless Sensor Node for automatic vehicle classification that could potentially replace traditional inductive loop systems. The new hardware was designed to take into account the low power requirements of the project scope. The designed sensor note uses CC430F5137 system-on-chip micro-controller, and LIS3MDL ARM magnetometer from STMicroelectronics. This new magnetometer has several benefits over the older sensor: uses SPI as opposed to I^2C for lower power communication, uses less power during sampling, has a wider set of options for full scale range detection, and can generate interrupts when the magnetic field magnitude exceeds a programmable threshold, thereby allowing for the system to only sample from the device when a vehicle is nearby.

In this project, we experimented different machine learning based vehicle classification algorithms. We were able to extract various statistical and frequency-based features from the raw sensor data vectors. Then, we prioritized different machine learning algorithms in order to remedy vehicle classification problems. Broadly, algorithms such as decision tree and random forest are potentially easier to be implemented on sensor nodes with computation constraints; nevertheless, they are more rigid in learning complex patterns in the dataset and less scalable for practical industrial solutions. This proves contrary to multilayer perceptron, which is not only more scalable but also capable of providing a higher accuracy while still requiring more computation resources. For future developments in our research, we are interested in gathering more data from different types of cars; in addition, there are more machine learning algorithms that we can evaluate in the machine learning based vehicle classification algorithm, which would hopefully result in a higher

accuracy and yield more generalized models. Based on our research results, we have submitted an article [1] in IEEE Sensors.

VIII. REFERENCES

[1] A. Ameri, H. Agnote, S. Cagle and M. Mozumdar, "Machine Learning Based Fine Grained Vehicle Classification For Next Generation Smart Roads", *IEEE Sensors Journal*, 2018. (submitted, under review)

[2] Gordon, R. L., Reiss, R. A., Haenel, H., Case, E. R., French, R. L., Mohaddes, A., & Wolcott, R. (1996). TRAFFIC CONTROL SYSTEMS HANDBOOK-REVISED EDITION 1996 (No. FHWA-SA-95-032).

[3] Knaian, A. N. (2000). A wireless sensor network for smart roadbeds and intelligent transportation systems (Doctoral dissertation, MIT Media Lab).

[4] Bajwa, R., Rajagopal, R., Varaiya, P., & Kavaler, R. (2011, April). In-pavement wireless sensor network for vehicle classification. In Information processing in sensor networks (ipsn), 2011 10th international conference on (pp. 85-96). IEEE.

[5] El-Tawab, S., Olariu, S., & Almalag, M Friend: A cyber-physical system for traffic flow related information aggregation and dissemination. InWorld of Wireless, Mobile and Multimedia Networks (WoWMoM), 2012 IEEE International Symposium on a (pp. 1-6)..

[6] Bathula, M., Ramezanali, M., Pradhan, I., Patel, N., Gotschall, J., & Sridhar, N. (2009). A sensor network system for measuring traffic in short-term construction work zones. In Distributed Computing in Sensor Systems (pp. 216-230).

[7]. Cheung, S. Y., Ergen, S. C., & Varaiya, P. (2005, November). Traffic surveillance with wireless magnetic sensors. In Proceedings of the 12th ITS world congress (Vol. 1917, p. 173181).

[8]. Yoo, S. E. (2013). A wireless sensor network-Based portable vehicle detector evaluation system. Sensors, 13(1), 1160-1182.

[9]. Papp, Z., Sijs, J., & Lagioia, M. (2009, December). Sensor network for real-time vehicle tracking onroad networks. In Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP), 2009 5th International Conference on (pp. 85-90). IEEE.

[10]. Sifuentes, E., Casas, O., & Pallas-Areny, R. (2011). Wireless magnetic sensor node for vehicle detection with optical wake-up. Sensors Journal, IEEE, 11(8), 1669-1676.

[11]. Liepins, M., & Severdaks, A. (2013, November). Vehicle detection using non-invasive magneticwireless sensor network. In Telecommunications Forum (TELFOR), 2013 21st (pp. 601-604). IEEE.

[12] Xing, J., & Zeng, Q. (2007). A model based vehicle detection and classification using magnetic sensor data (Master Thesis).

[13] Padmavathi, G., Shanmugapriya, D., & Kalaivani, M. (2010). A study on vehicle detection and tracking using wireless sensor networks. Wireless Sensor Network, 2(02), 173.

[14] Caruso, M. J., & Withanawasam, L. S. (1999, May). Vehicle detection and compass applications using AMR magnetic sensors. In Sensors Expo Proceedings (Vol. 477).16

[15] Duarte, M. F., & Hu, Y. H. (2004). Vehicle classification in distributed sensor networks. Journal of Parallel and Distributed Computing, 64(7), 826-838.

[16] Haoui, A., Kavaler, R., & Varaiya, P. (2008). Wireless magnetic sensors for traffic surveillance. Transportation Research Part C: Emerging Technologies,16(3), 294-306.

[17] Harlow, C., & Peng, S. (2001). Automatic vehicle classification system with range sensors. Transportation Research Part C: Emerging Technologies, 9(4), 231-247.

[18] Gupta, R., & Das, S. R. (2003, October). Tracking moving targets in a smart sensor network. In Vehicular Technology Conference, 2003. VTC 2003-Fall. 2003 IEEE 58th (Vol. 5, pp. 3035-3039). IEEE.

[19] Li, D., Wong, K. D., Hu, Y. H., & Sayeed, A. M. (2002). Detection, classification, and tracking of targets. Signal Processing Magazine, IEEE,19(2), 17-29.

[20] Cheung, Sing Yiu, Sinem Coleri Ergen, and Pravin Varaiya. "Traffic surveillance with wireless magnetic sensors." Proceedings of the 12th ITS world congress. Vol. 1917.2005.

[21] L. Jogschies, D. Klaas, R. Kruppe, J. Rittinger, P. Taptimthong, A. Wienecke, L. Rissing and M. Wurz, ”Recent Developments of Magnetoresistive Sensors for Industrial Applications,” Sensors, vol. 15, no. 11, pp. 28665-28689, 2015.

[22] E. Alpaydin, Introduction to Machine Learning, 3rd ed. Cambridge: MIT Press, 2014.

[23] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel and B. Thirion, ”Scikit-learn: Machine Learning in Python,” Journal of Machine Learning Research, vol. 12, pp.2825-2830, 2017.

[24] C. Clark, E. Gazoni, (2017, Aug 29). ”openpyxl - A Python library to read/write Excel xlsx/xlsm files,” [Online] Available: https:openpyxl.readthedocs.io/en/default/#

[25] J. D. Hunter, ”Matplotlib: A 2D Graphics Environment,” in Computing in Science & Engineering, vol. 9, no. 3, pp. 90-95, May-June 2007.

[26] Breiman, Leo; Friedman, J. H.; Olshen, R. A.; Stone, C. J. (1984). Classification and regression trees. Monterey, CA: Wadsworth & Brooks/Cole Advanced Books & Software. ISBN 978-0-412-04841-8.

[27] D. Harnett, Statistical methods, 3rd ed. Reading, Mass.: Addison-Wesley, 1982.
[28] K. Ying, A. Ameri, A. Trivedi, D. Ravindra, D. Patel, and M. Mozumdar. ”Decision tree-based machine learning algorithm for in-node vehicle classification,” In Green Energy and Systems Conference (IGESC), 2015. IEEE, pp. 71-76. IEEE, 2015.

[29] T. Cormen, C. Leiserson, R. Rivest and C. Stein, Introduction to algorithms. Cambridge, Massachusetts: The MIT Press, 2014.

[30] D. Opitz and R. Maclin, ”Popular Ensemble Methods: An Empirical Study,” Journal of Artificial Intelligence Research (JAIR), vol. 11, pp. 169-198, 1999.

[31] I. Nitz, U. Schulthess and H. Asche, ”COMPARISON OF MACHINE LEARNING ALGORITHMS RANDOM FOREST, ARTIFICIAL NEURAL NETWORK AND SUPPORT VECTOR MACHINE TO MAXIMUM LIKELIHOOD FOR SUPERVISED CROP TYPE CLASSIFICATION,” Proc. of the 4th GEOBIA., 2012.

[32] G. Zhang, Y. Wang, and H. Wei, ”Artificial neural network method for length-based vehicle classification using single-loop outputs,” Transp. Res. Rec., vol. 1945, pp. 100108, 2006.