Electric Vehicle Waiting Time Prediction with OpenCV and Support Vector Regression

Rifqi Muhammad Fikri1), Mintae Hwang2)

Abstract

This paper proposes a system that aimed to solve the lack of a charging station infrastructure by providing waiting time information to electric vehicle (EV) drivers for optimal charging options. This system determines the number of vehicle movements, such as entering or leaving the charging station in real-time. Furthermore, the Changwon National University area’s charging station is simulated with a PC server and camera as the real-time video input; the OpenCV and MobileNet SSD object detection network is used to detect and track vehicle movements in the charging station, and the Firebase real-time database is used as the server. For the mobile application, maps featured with Kakao Maps API in the user interface are installed. In addition to the information regarding the current status of the charging station, a prediction system with Support Vector Regression (SVR) was added so that EV drivers can schedule charging ahead of time. Therefore, the proposed system aimed to assist EV drivers in navigation and reduce waiting times for charging. The simulation was conducted to evaluate the proposed system performance and got an average time-response result of 7.77 seconds from the moment EVs were detected until it was displayed in the mobile application interface.

Keywords: Electric Vehicles, Waiting Time Prediction, OpenCV, Support Vector Regression, Mobile Application

1. Introduction

In recent times, electric vehicles (EVs) have been recognized as a promising approach to reduce traffic emissions; they promise better energy conversion efficiency and greenhouse gas (GHG) emission[1][2]. However, EVs have limitations and high costs that prevent them from competing with conventional petroleum-based vehicles. Charging requires significant time and expenses, while recharging is slow. It is imperative to manage EV limitations for attracting private charging stations (CSs) relative to their capacity[3][4]. However, one of the critical challenges in
fully adopting EVs is the lack of charging station infrastructure, although it is predicted that a sufficient number of CSs will eventually be constructed in the future[5]. Therefore, determining the optimal option for CSs located in a town is important. Providing information regarding the nearest and most suitable CS would assist EV drivers. Most of the time, EV drivers do not know the availability condition of the CS beforehand. There are various map applications on the internet, which provide easy access and interaction with the users. However, most of the information is merely related to the location and navigation to the CS. Therefore, using the navigation feature in addition to providing information regarding CS availability in real-time scenarios would be beneficial for the EV drivers.

In this study, the researchers developed a system that provides information regarding CS availability in a mobile application. The system can also determine EV movement, such as entering or leaving the CS. Moreover, a prediction system with machine learning technology is added to predict the future occupancy of the CS. This study is limited to the CS in the Changwon National University area. A web camera was used as the video streaming input, and a PC was used as the simulation platform. The vehicles in the video frame are classified by using the MobileNet SSD object detection method. On the user side, the android application provides them with a maps function interface with Kakao Maps application programming interface (API). The information is shared with the Firebase real-time database to store all of the data and provide the users with information regarding real-time scenarios. Thus, EV drivers can choose an optimal location to charge their vehicles based on the application's information.

The rest of this paper is organized as follows. Section 2 introduces the related works. Section 3 illustrates the model and the architecture of the system. Section 4 presents the performance evaluation of the proposed system. Section 5 concludes this paper and discusses future research possibilities.

2. Related Works

A waiting time estimation (WTE) system has been developed for a hospital in previous studies to minimize patient waiting times and avoid overcrowding[6]. The system is implemented with a message queue telemetry transport (MQTT) protocol for a smartphone-based queuing theory. The results indicated that the proposed algorithm used in the system can present new patient waiting times that are close to the actual waiting times.

Another study developed a mobile ticket dispenser system (MTDS) with a waiting time prediction that enhances user experience in the ordering process[7]. This system allowed
customers to remotely draw a ticket for ordering meals anywhere through a smartphone before they arrive at a restaurant, thus, reducing their waiting time.

Another study aimed to develop a procedure to evaluate the configuration of electric fast-charging station services in charging infrastructure scenarios in Rome’s selected urban area[8]. The system estimated the number of potential battery electrical vehicle (BEV) users in a selected urban area using a big data analysis method by developing a fuzzy model to designate BEV users to convenience charging stations criteria. Then, a simulation procedure, in which all the charge events were obtained, was utilized to obtain a customer crowding time profile at the stations.

In this study, the newly proposed system is based on a previously proposed system[9]. A method that can count the movement of vehicles by using YOLO-LITE real-time object detection is utilized to count the number of hydrogen fuel-based vehicles that are charging or queuing in the CS for the driver’s needs. The proposed system upgrades the system with changes and introduces a prediction system feature. The prediction system will offer insight and help EV drivers forecast their plans for charging their EVs in CSs.

3. Proposed Waiting Time Prediction System

3.1 Proposed System Configuration

Considering the underlying architecture, the proposed system aimed to assist EV drivers by providing information regarding the charging station’s availability in the form of a mobile application by using object detection and a prediction system. Because this system is based on a previously proposed system[9], several main structures and features are similar. Four main points structure the system. A charging station with an installed IP camera records vehicle position and movement (in or out), and a PC server counts the number of vehicles in the charging station in real time using machine learning and object detection. A Firebase database server responds to the request and saves all the data from the object detection and prediction system through a mobile application. The mobile application with a map feature on the user-end checks the CS waiting time. The prediction system creates future prediction forecasts relative to the time of day using environmental data from the CS. [Fig. 1] illustrates the underlying architecture of the system.
Electric Vehicle Waiting Time Prediction with OpenCV and Support Vector Regression

3.2 PC Server

In this system, a webcam attached to a laptop was used for the simulation. The Open source computer vision library (OpenCV)[10], and MobileNet SSD object detection network[11] was installed for vehicle tracking and counting. For locating vehicle position, a centroid tracker was used to track the vehicle object’s centroid. Imaginary boundaries were also created to determine whether the movement of the vehicle is inwards or outwards. For every instance of change for the detected vehicles, the PC server sends the information to the Firebase Database. The PC Server is also attached to Arduino Uno with sensors to collect the environmental data. Whenever there is an event of movement in the CS (EV entering or leaving), the PC server will immediately send new information to the database. Therefore, the database will always have the latest updated information and CS status. [Fig. 2] illustrates the centroid tracker and the PC server simulation set-up.

[Fig. 1] Proposed System Configuration

[Fig. 2] Centroid Tracker and PC Server in the Form of Laptop Attached to a Web-camera and Arduino
3.3 Firebase Database Server

In this study, Firebase was used as the database platform because it supports a direct connection between the PC server (vehicle object detection) and the mobile application user. The Firebase real-time database is a cloud-hosted NoSQL database that enables users to store and synchronize data between multiple users in real-time. Charging station information such as the charging station addresses, the number of charging ports, and phone numbers were manually stored. For each event that occurred in the CS, the number of vehicles going in and out were stored on the database. After storing the new information, the Firebase DB sends the newest information to the mobile application to provide the users with the latest information regarding the number of CS vehicles.

3.4 Prediction System

The goal of the prediction system is to provide insights and allow users to forecast future events. To create a prediction system, a descriptive analysis of the average number of EVs charging daily is needed. The generated EV number and environmental data is combined to create a daily prediction. Support vector regression (SVR) is used for predictive analytics to generate a prediction outcome. The typical hyperplane formula is as follows:

\[ y = wx + b \]  

(1)

Where \( y \) is the prediction outcome hyperplane. Then the equation of decision boundary is as follows:

\[ \begin{align*}
wx + b &= +a \\
wx + b &= -a
\end{align*} \]  

(2)

[Fig. 3] presents an illustration of the SVR example.

![Fig. 3] Simple SVR Example
As the hyperplane shows the prediction line, the two boundary lines will show the minimum and maximum number of daily EV predictions using SVR. After the prediction is generated, the result will be sent to the Firebase DB; then, the prediction information will be shown in the mobile application. In this study, the number of vehicles is generated by counting the number of vehicles and recording the environmental data with Arduino and sensors, which amounts to datasets of 30 days. Then, based on the daily dataset, an SVR solution is created for a prediction for the next 30 days (month).

3.5 Android Application

The mobile application for object detection is developed using the Android studio. There are two main features in the mobile application: a charging station navigation (CSN) page and charging station information (CSI) page. The primary function of the CSN page is to navigate and search the location of the charging station. The CSI page will show the condition of the CS in real time. Information such as CS name, address, number of ports, number of vehicles in the charging station in real time, and the forecast of the prediction system is displayed on this page. [Fig. 5] illustrates the storyboard of the android application.
4. Performance Evaluation

As seen in [Fig. 6], a demo field test is conducted by assuming a university parking lot as a CS. Then, each vehicle that comes in and goes out is assumed to be an EV that will be charging and leaving the CS. The demo field test was performed to measure the time response when an event occurred and was recorded in the PC server. The demo field test was conducted at a busy time (coming to and leaving work). [Fig. 6] presents the CS simulation recorded by the laptop.
For the five days of recording, the time-responses were counted by saving time-stamps in the PC server, Firebase database, and the mobile application, then subtracting each of the time-stamps; time-response A, from the PC server to the Firebase database, time-response B, from the Firebase database to the mobile application. The time-stamps for PC server vehicle detection were saved by the python source code into a CSV file, while in the Firebase DB and mobile application, it is automatically recorded with their feature. The demo field test results are shown in [Fig. 7] and [Fig. 8], and the daily average time-response test results are shown in [Table 1].

![Fig. 7] PC Server to Firebase DB Time-responses Test Result

![Fig. 8] Firebase DB to Mobile Application Time-responses Test Result
Based on the table above, the test has 7.77 seconds average response time for storing and updating data from the point when an event occurred in the PC server (time response A) until it is displayed on the users mobile application (time-response B). Time-response A is faster than time response B as it is one-way connectivity, the PC server sends new information, and Firebase DB only retrieves it. While mobile application needs to send a request to update the CS information and then Firebase DB will respond with the updated information. Another reason for higher response time for mobile applications is the limited feature of the mobile application. Users have to manually refresh (slide down the page) and wait for the page to be updated with the latest CS information.

5. Conclusion

In this study, a system that can determine the movement of vehicles entering and leaving a CS in real-time was proposed with an android application as the user interface. The proposed system is based on previous study which aimed to assist EV drivers in reducing navigation and wait times for charging their EVs.

Considering the performance evaluation results, the proposed system worked with a time-response of 7.77 seconds from the event recorded and detected by the web-camera in the PC Server, to the display in the mobile application. In the future, increasing the dataset for prediction and creating more CS locations will be considered. Also, an optimal location feature in the mobile application will be created so that the users can search the most suitable location to charge their EVs.

Acknowledgement

This work was supported by Gyeongnam SW Convergence Cluster 2.0 under the contract S0312-20-1002.
References


