Hindawi
Journal of Advanced Transportation
Volume 2019, Article ID 6372597, 15 pages
https://doi.org/10.1155/2019/6372597

Research Article

Public Transport Driver Identification System Using Histogram of Acceleration Data

Nuttun Virojboonkiate, Adsadawut Chanakitkarnchok, Peerapon Vateekul, and Kultida Rojviboonchai

Chulalongkorn University Big Data Analytics and IoT Center (CUBIC), Department of Computer Engineering, Faculty of Engineering, Chulalongkorn University, Bangkok, Thailand

Correspondence should be addressed to Kultida Rojviboonchai; kultida.r@chula.ac.th

Received 1 October 2018; Revised 19 December 2018; Accepted 15 January 2019; Published 5 February 2019

Guest Editor: Mohammad H. Y. Moghaddam

This paper introduces a driver identification system architecture for public transport which utilizes only acceleration sensor data. The system architecture consists of three main modules which are the data collection, data preprocessing, and driver identification module. Data were collected from real operation of campus shuttle buses. In the data preprocessing module, a filtering module is proposed to remove the inactive period of the public transport data. To extract the unique behavior of the driver, a histogram of acceleration sensor data is proposed as a main feature of driver identification. The performance of our system is evaluated in many important aspects, considering axis of acceleration, sliding window size, number of drivers, classifier algorithms, and driving period. Additionally, the case study of impostor detection is implemented by modifying the driver identification module to identify a car thief or carjacking. Our driver identification system can achieve up to 99% accuracy and the impostor detection system can achieve the F1 score of 0.87. As a result, our system architecture can be used as a guideline for implementing the real driver identification system and further driver identification researches.

1. Introduction

Twenty billion units is the number of the Internet of Things (IoT) that is predicted to be installed in 2020 [1–3]. Technology of sensors and communication devices dramatically increases this number of connected things. One of the key connected things is the connected vehicle. It is reported that there will be a quarter billion connected vehicles on the road in 2020 [4]. Nowadays, consumer vehicles are equipped with a lot of sensors in order to provide driver-assistant services, such as adaptive cruise control, automatic parking, and automatic emergency breaking. These sensors in vehicles lead to an advancement in transportation research. One of the popular research topics is the study of the driving style to detect the driver's behavior. The survey of these researches has already been discussed thoroughly in [5]. Apart from detecting driving style, if a driver has his own driving style, the driver himself could also be identified. This leads to the driver identification research that can automatically identify an individual driver using sensors in the vehicle.

There are a lot of situations where a vehicle is shared by more than one person. These situations can occur when the vehicle is a city bus, a company van, or even a consumer car. The driver identification system has four significant points of benefit. First, it helps companies or governments know the actual driver of their vehicles. Until now, to recognize the driver, one could use some simple approaches such as recorded notes or an identification card. One could use even more sophisticated approaches such as a fingerprint scanner or face detection. However, by using sensors in a vehicle, the identification system can achieve the goal easier because it can automatically identify the drivers while they are driving the vehicle. Moreover, an identification card or a fingerprint system can be cheated easily by replacing the identifying item. Real-time face detection using a video camera can solve the cheating problem. However, the drivers might not feel happy and comfortable to be observed by the camera all the time. By using the driver identification detected by sensors, the driver will not feel being watched by the camera. More importantly, the drivers cannot allow other persons

to drive on their behalf. This is because it is difficult to copy the driving behavior. The second benefit of the driver identification system is that it can be applied to a car thief or carjacking detection. According to an Interpol statistic [6], more than seven million vehicles have been stolen almost every year since 2012. With the hidden sensors and the driver identification system, a thief will not even know that he is being detected. The system can immediately send a carjacking notification to the owner of the vehicle or the police. In this situation, a GPS module can also be used to detect the vehicle location. The third benefit of the driver identification system is that it can help in adjusting the environment based on the driver's preferences. For instance, once the system detects the specific driver, it can control the air temperature, songs, and seat position based on the preferences of this driver. The fourth benefit is that it can be used in the insurance business. As drivers have different behaviors causing different risk levels, insurance companies can adjust charges of insurance according to the drivers and their driving styles.

There have recently been several proposed researches for the driver identification system, which can be classified into two categories. The former is the research using multiple sensors [7–17]. The latter is the research using a single sensor [18–20].

For the first category, in [7–9], multiple external sensors were installed inside the vehicle. These sensors included GPS, gas and brake pedal force, velocity, steering angle, and engine speed. The vehicle data was analyzed by a spectral and Gaussian mixture model [7] and a neural network [8, 9]. These works achieved 75-95% of accuracy. As a Controller Area Network (CAN) bus became a signaling standard, many researchers attempted to exploit the sensor data from the CAN bus to perform driver identification [10–15]. In [10], the data of a single car driven by 15 volunteer participants was used. The variables such as time of day and radio station were controlled. An overlapped sliding window of the data was considered and optimized. There were many features evaluated including statistical features, descriptive features, and frequency features. The authors showed the best algorithm, the top sensor, and the top feature that can differentiate the drivers in the experiments. In [11], the driver identification method using only data obtained at a single turning event has been proposed. The authors classified turning locations into 12 types and conducted experiment for each of them. The model used 12 signals of the CAN bus in an Audi car driven by multiple drivers on real roads. The model could achieve 76.9% for two-driver classification and 50.1% for five-driver classification. In [12], driver identification with impostor detection has been proposed. The authors used an Extreme Learning Machine (ELM) to detect the impostor. The ELM had 8 input variables including data obtained from the CAN bus and other sensors. For driver identification, the accuracy was greater than 80% in all cases and still above 90% for 2-3-driver classification. For impostor detection, the accuracy was above 80% when a single genuine driver was considered. However, the accuracy decreased to about 50% when the number of authorized drivers increased. In [13], driver identification using vehicle acceleration and deceleration events has been proposed. The data including duration, speed, acceleration, jerk, and path of defined acceleration and deceleration events were selected to extract statistical features. Linear Discriminant Analysis (LDA) was used as an algorithm to identify the drivers. Accuracy of up to 80% can be achieved. In [15], driver identification was conducted using vehicle data without a feature extraction process and accuracy of up to 99% was achieved. In [14], three open data were evaluated in the proposed model. The minimum training and testing time required to achieve a desired accuracy were also investigated. Apart from using data from the CAN bus, a different method has been proposed in [16]. Instead of using real-time data like other previous work, the trip-based data obtained from an In-Vehicle Data Recorder (IVDR) was exploited to identify the driver. With the proposed machine learning pipeline, the average accuracy of 88% can be achieved.

Although using data obtained from multiple sensors via the CAN bus can help achieve high accuracy, there are still a large number of vehicles that are not compatible with the CAN bus standard. Moreover, using multiple external sensors suffers from installation, data processing, implementation, and computation. Therefore, a number of recent approaches have been proposed to use only data obtained from a single sensor [18–20]. In [20], GPS data from a smartphone is used to identify the driver by deriving other data from the location such as acceleration, angular speed, and jerk. A Random Forest was used to identify the driver from statistical features extracting from those derived data. The experiment was done in nine groups of drivers separately. Each group consisted of 4-5 drivers, who drove in a similar location and time. The average accuracy for all groups of 82.3% was obtained. Many aspects were mentioned such as number of drivers, classifier algorithms, and sliding window. Another approach using only a single sensor was proposed in [18]. In this work, only acceleration data was used by collecting from smartphones located in five campus buses for three months. The PCA (Principle Component Analysis) was done on statistical variables derived from the acceleration data, which is collected only for six days. The PCA showed some unique patterns for each driver. However, some drivers still had the same pattern of variables. Moreover, the analysis was done on the whole-day data, which is not practical for the real-time driver identification system. Nevertheless, this work reveals that the acceleration data has potential to identify the driver. Furthermore, the acceleration sensor is low cost and low energy consumption and can directly represent the vehicle's accelerating behavior of the driver. Thus, in our previous work [19], we proposed driver identification using a histogram of acceleration. We selected the histogram as our main feature. This is because the histogram can tolerate noise of data much more than other statistical features. The data was collected from 13 campus buses for 10 months. A neural network was used to identify the driver from the histogram. The identification was done separately for four periods of time of a day to reduce traffic variations. Using this method, 77-88% of accuracy was achieved. Although all of the abovementioned works use only data obtained from a single sensor and provide promising results, some important issues still need to be addressed for public transport driver

identification. First, the data preprocessing method can be improved because of the characteristic of the public vehicle, which frequently stops along the trip. Second, practical issues need to be addressed such as the traffic environment, number of drivers, detection time, and training period. Third, a case study such as impostor detection should be implemented and evaluated to show the possibility and the importance of driver identification.

Therefore, in this paper, we extend our previous work in many aspects. First, we introduce the system architecture for driver identification. This system is simple to implement and apply to any type of vehicles. Second, the data preprocessing method is improved. Specifically, we observe that there exists a large amount of data recorded when the vehicles are not moving. Eliminating such data before the processing can significantly increase the accuracy. This is because public vehicles normally stop at every service station and such data does not represent the drivers' behaviors. With our new data preprocessing method, the accuracy of our system can greatly improve from 77-88% to 94-99%. Third, the experiment has been conducted by using a random time of the day to identify the driver regardless of the traffic environment. Fourth, we extensively investigate variables that could affect the accuracy. These variables include axis of data, sliding window, overlapping technique, classifiers, number of drivers, and driving period. Fifth, we show that our system can be applied to detect an impostor. Thus, it can be beneficial for detecting a car thief or carjacking.

The rest of this paper is organized as follows: the details of related work are provided in Section 2. In Section 3, we propose our system architecture, which consists of the data collection module, data preprocessing module, and driver identification module. In Section 4, the case study of impostor detection is explained by modifying the identification model. In Section 5, we evaluate the performance of our system in various aspects including the inactive-period filtering module, axis of acceleration, sliding window and overlapping technique, classifiers, number of drivers, and impostor detection. The conclusion of the work is provided in Section 6.

2. Related Work

In this section, we thoroughly discuss the researches that utilize data obtained from a single sensor to perform driver identification. Chowdhury et al. proposed a driver identification method using only GPS data [20]. Other data such as speed, acceleration, jerk, and jerk energy were computed from the GPS data. Then, statistical features of those data were extracted such as mean, median, skewness, kurtosis, standard deviation, min, max, and some percentile values. After analysis of variance (ANOVA), 137 features were then selected to be input of the Random Forest. The experiment involved 38 drivers, but they were grouped by location and timing to reduce external variables such as traffic. Sliding window and the overlapping technique were also considered. Average accuracy of 82.3% was achieved for five-driver identification. Moreover, by analyzing driving skill and aggression, Chowdhury et al. concluded that skilled-driver identification tends to provide higher accuracy.

Phumphuang et al. proposed a driver identification model that utilized only acceleration data [18]. The data had been collected for 3 months and involved 5 drivers. Descriptive statistics such as average, standard deviation, differential, and correlation were computed from the lateral and longitudinal axis of the acceleration data. Then, the PCA (Principle Component Analysis) method was used to select proper variables. With some criteria and a set of variables, this method can correctly identify all 5 drivers. This result shows a possibility to identify an individual driver using only acceleration data.

Enev et al. [10] who used multisensors from the CAN bus to identify the driver also mentioned the result using only the single top sensor. The utilization of the brake position data that is the top sensor can identify the driver with the accuracy of 87.33% from the first part of the data and 100% for the second part of the data.

In our previous work [19], a driver identification method using only acceleration data has been proposed. The data involved 13 drivers who drove their own campus buses for 10 months. The histogram of acceleration illustrated that it can represent the unique driving behavior of an individual driver. The experiment was conducted on four periods of time of a day to reduce the traffic variable. By using the histogram of acceleration as the feature of the model, up to 88% of accuracy was achieved. However, our previous work has not considered the unique characteristic of public transport. In particular, public transport drivers normally stop their buses at every bus station. This should be taken into consideration for the data preprocessing module. Moreover, in our previous work, a number of aspects were overlooked including the sliding window size, overlapping technique, and amount of data. In this paper, all of the abovementioned issues have been considered and accuracy of up to 99% has been achieved. The system architecture has been proposed especially for public transport. Table 1 shows comparison between our proposed system and other existing work.

3. System Architecture

Our system architecture consists of three main modules. The first module is the data collection module, which is packed in one box as shown in Figure 1. We collaborated with the campus buses' company to attach each of our boxes in 13 buses. Each driver is assigned to drive a single bus. The buses are electrical vehicles as shown in Figure 2, supplying power directly to the box. Figure 3 depicts our system architecture overall. The data collection module sends the data obtained from the sensor to our server through 3G. The data is stored in the database system before being processed to the second module. The second module is the data preprocessing module, which is responsible for cleaning raw data and extracting features from it. Then, the third module, the driver identification module, identifies the driver from those processed data. The details of each module are explained in the following subsections.

Table 1: Comparison between our proposed system and other existing work.

Author	Sensor used for driver identification	Data collection period	Number of drivers	Features	Inactive-driving filtering	Classifier	Accuracy of driver identification (%)	Practicability for impostor detection
Our proposed system with the new data preprocessing	Acceleration sensor	10 months	3-13	Histogram of acceleration	✔	Neural network	94-99	✔ F1 score 0.87 using KNN classifier
Our previous work [19]	Acceleration sensor	10 months	3-13	Histogram of acceleration	✘	Neural network	74-88	✘
Phumphuang et al. [18]	Acceleration sensor	3 months	5	Statistical features with PCA	✘	✘	60-100	✘
Chowdhury et al. [20]	GPS	2 months (1223 hours)	4-5	Statistical features	✘	Random Forest	~ 82.3	✘
Enev et al. [10]	Brake paddle position	~ 3 hours/driver	15	Statistical, descriptive, and frequency features	✘	Random Forest	87-100	✘

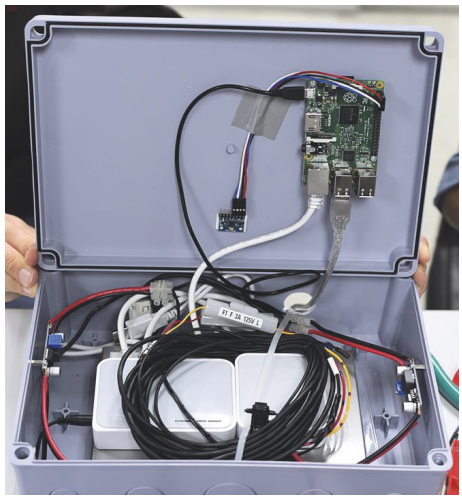

Figure 1: Equipment box attached in each vehicle.

Figure 2: Campus buses used in the experiment.

3.1. Data Collection Module. Before processing and analyzing data, the data collection module is considered for ensuring integrity. The overall data collection module is shown in Figure 4. The data collection module can be divided into 2 main processes: data producing and data processing:

(i) *Data producing* consists of 2 sensors, which are the MPU-6050 module [21] and GPS module. The MPU-6050 sensor module contains a 3-axis accelerometer. The accelerometer allows 1 kHz sampling rates from 2g to -2g with the sensitivity of 16,384 LBS/g. This sensor module is horizontally placed in the box at the middle front panel of the bus. As illustrated in Figure 5, the longitudinal, lateral, and vertical movement are recorded on the x-, y-, and z-axis, respectively. Next, the GPS module provides the location of the shuttle bus. The GPS module is placed in the center of the metal roof of the shuttle bus, which is away from radio antennas to avoid interference. Note that GPS data is not used for driver identification. It is only used for determining the stopping period of the bus in the data preprocessing module

(ii) *Data processing* is done by Raspberry Pi, which is a low-cost and high-performance computer. It is equipped with an acceleration sensor module and GPS module for processing the raw data. The sampling rate of the acceleration sensor data and that of geolocation data are 90 Hz and 1 Hz, respectively. The sampling data is sent to the server via a 3G connection using a SIM card. Raspberry Pi communicates with the server by using the HTTP protocol. The data is stored in a structure database. In the storing process, data from the acceleration sensor module will be concatenated with the data from the GPS module. As the sampling rate of the data from the acceleration sensor module is more than the sampling rate of the data from the GPS module, the concatenated data will use the latest updated geolocation data with the time stamp

3.2. Data Preprocessing Module. The data preprocessing module is used to process raw acceleration data before being used as the input to the driver identification module. This module consists of two main submodules, which are the inactive-period filtering module and feature extraction module.

3.2.1. Inactive-Period Filtering Module. The characteristics of a public transport vehicle are different from a private vehicle. Normally, a public bus stops for a period in order to pick up passengers at bus stops. In this period, the driver does not expose any driving behavior through the acceleration data. We consider this period as the inactive period and it should be removed. We visualize a histogram of the stopping period as shown in Figure 6 to understand the characteristic of the public vehicle data. This figure shows the frequency of each stopping period. As illustrated in Figure 6, the bus mostly stops for less than five seconds. This indicates normal driving behaviors in light traffic. On the other hand, the bus hardly stops for longer than five seconds. As it is the long period, it takes a significant proportion in the overall data and should be removed as mentioned. Therefore, the inactive period is defined by the period in which location data is the same for more than five seconds. Algorithm 1 shows our proposed inactive-period filtering algorithm. After removing the inactive period, the remaining data will be called active driving data. This is a crucial step since it significantly increases the accuracy comparing to our previous work. Details of the results are provided in Section 5.1.

3.2.2. Feature Extraction Module. The input of the feature extraction module is the active driving data. Each axis of the acceleration is processed to a histogram. The histograms are represented in relative frequency so that they can be compared to one another. In our previous work, a histogram was built within four durations: morning, late-morning, afternoon, and evening. Each duration was evaluated separately to reduce traffic effect. However, in this paper, a

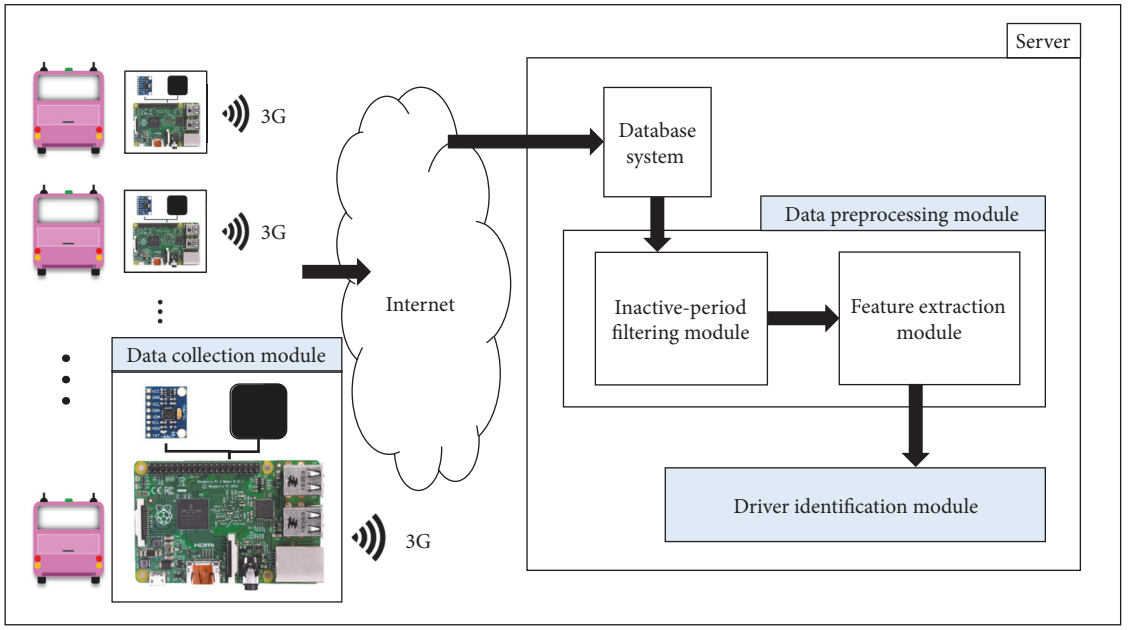

Figure 3: System architecture.

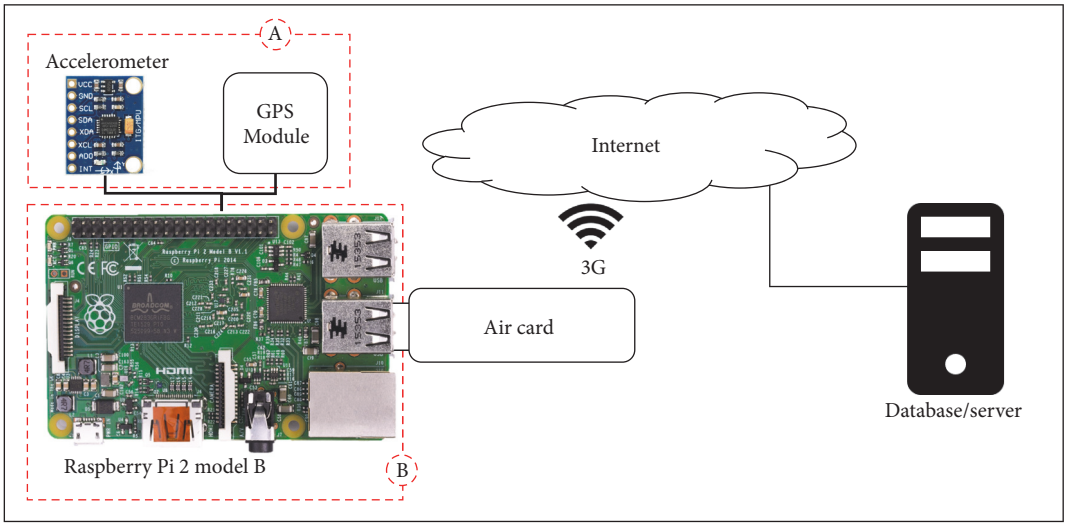

Figure 4: The data collection module, where A is a data producing part and B is a data processing part.

histogram is built from the acceleration data of a single driver within a sliding window. Data from a random time of the day was evaluated together so the system can be used more practically and realistically. To choose the sliding window size, the acceleration data should be collected for a proper period to allow the histogram representing the driving behavior. Moreover, the sliding window can be overlapped. This is also a crucial technique to improve the accuracy. Figure 7 depicts histogram extraction from the acceleration data. More details of the sliding window and overlapping technique will be evaluated and discussed in Section 5.3. The histogram extraction algorithm is shown in Algorithm 2. The min value, max value, and number of bins need to be set in the same way for all histograms. The min and max values should be tested to cover most of the acceleration values. In this paper, the range of histogram value is set to -3.6 to 3.6 m/s^2. The histogram has 100 bars as it is a proper value that can represent the curve of the histogram. Too low a value would be so rough that it cannot represent the behavior. Too high a value would cause unnecessary computation. The number of bars can also be optimized for a specific dataset.

However, after extracting the histograms, some of them provide uncommon curves shown in Figure 8. We called this the glitch of the histogram, which occurs in about 15% of all histograms. This glitch comes from a number of the same identical value of the acceleration data, which locates in a bin of the histogram. This happens because the acceleration sensor or Raspberry Pi may freeze and then record the same

Input: acceleration *acc_raw*,
latitude *lat*,
longitude *lng*
Output: acceleration (without inactive period) *acc*
for each *acc_raw* **do**:
  Initialize *next_lat* to next five-second of *lat*
  Initialize *rateOfChange_lat* to (*next_lat* - *lat*)/*lat*
  Initialize *next_lng* to next five-second of *lng*
  Initialize *rateOfChange_lng* to (*next_lng* - *lng*)/*lng*
  if *rateOfChange_lat* is zero **and** *rateOfChange_lng* is zero **do:**
    remove *acc_raw*
  end if
end for
return *acc_raw* as *acc*

Algorithm 1: Inactive-period filtering algorithm.

Input: acceleration *acc*,
sliding window size *wind*,
overlap percent *ovrp*
Output: histogram of acceleration *hist*
while *acc* remain:
  Initialize *hist* to histogram of *acc* for *wind* minutes
  using fixed min, max, and bin size
  Change *hist* frequency to relative frequency
  Skip *acc* to the next (1-*ovrp*)∗*wind* minutes
end while
return *hist*

Algorithm 2: Feature extraction algorithm.

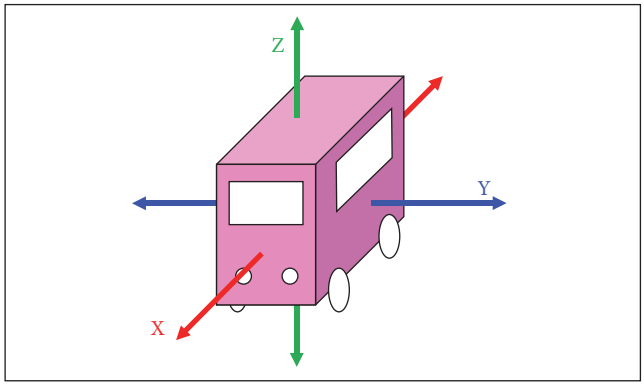

Figure 5: Vehicle movement represented in a coordinate system.

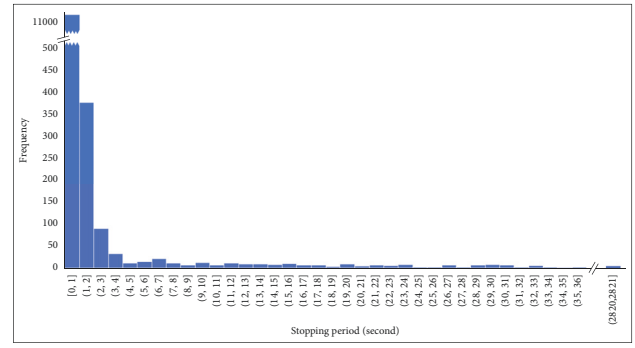

Figure 6: Histogram of inactive period.

acceleration value for a period of time. To solve this problem, the same identical value is removed from the histogram. The resulting histogram without the glitch is shown in Figure 9. The histogram is now ready for the next process.

3.3. Driver Identification Module. This section overviews the driver identification proposed in our previous work. After the histogram is properly built, it is used as an input of the classifier in order to identify the driver. In this paper, we mainly focus on the neural network algorithm, but other classifier algorithms will also be discussed in Section 5.5. To construct the neural network model for training and testing, the *neuralnet* package in the Comprehensive R Archive Network (CRAN) is used. To configure the model, each layer of the neural network is set as follows:

(i) The number of nodes in the input layer depends on the involved axis of acceleration. The histogram, which is the input of the neural network, has 100 bars for one axis. If one axis is involved, the input layer will

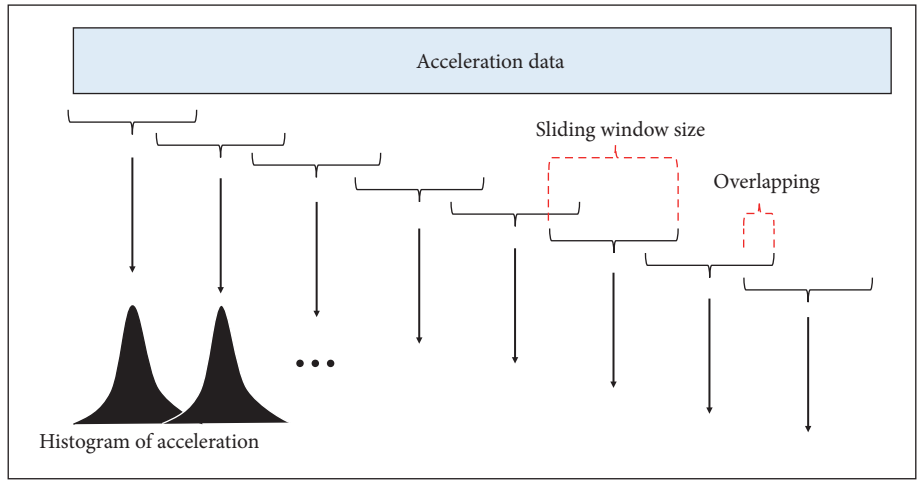

Figure 7: Histogram extraction from the acceleration data.

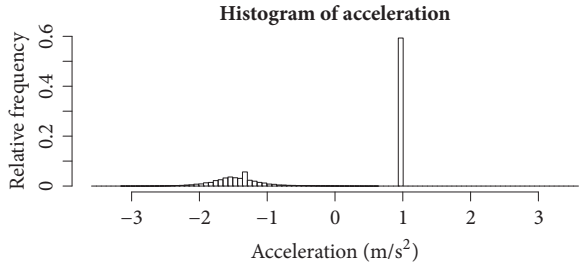

Figure 8: Example of a histogram that contains a glitch.

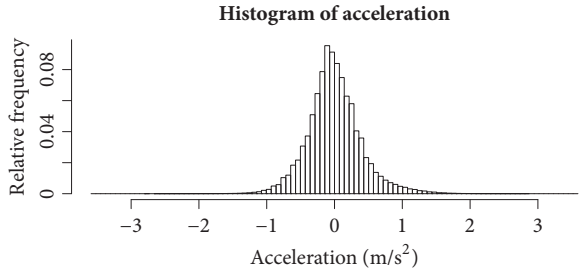

Figure 9: Example of a normal histogram.

have 100 nodes. The input layer will have 200 or 300 nodes if two or three axes are involved, respectively

(ii) The number of hidden layers and number of nodes in the hidden layer need to be tuned. According to our experiment, increasing the number of hidden layers does not provide significant improvement in terms of accuracy. Moreover, it leads to more computation time. Therefore, in this paper, only one hidden layer is used. In the case of hidden nodes, according to our experiment, the optimal number of hidden nodes is one-third of the number of input nodes. To be specific, if three axes are involved, the number of hidden nodes will be 100

(iii) The output layer represents the identified driver. Thus, the number of output nodes depends on the number of drivers. Specifically, if there are 13 drivers involved, the number of output nodes will be 13. When testing, the output node that has the highest value will be selected as the identified driver

To illustrate, we show the example of the neural network model that involves three axes and 13 drivers in Figure 10.

4. Case Study: Impostor Detection (Open Set Classification)

This section describes one use case of driver identification, which is the car thief or carjacking detection. Most driver identification researches only focus on identifying a driver in a closed set. This means an impostor driver who has never been trained in the model will be identified as one of the trained drivers. This is essential in a carjacking or car thief detection application because we have never had data of a thief to train the model beforehand. Thus, we extend the limitation of our model by adjusting only the identification module in order to identify the impostor. In a closed set of drivers, the output node of the neural network that has the maximum value will be selected as the identified driver. However, in an open set, a threshold needs to be calculated to separate an impostor from a genuine driver. The threshold is calculated for each driver by using validation data (not testing data and training data). After trying different techniques, the threshold that provides the highest accuracy is the 20th percentile of the output value in the validation data of the correctly identified drivers. Thus, in the testing step, the output value of the selected driver that is lower than the threshold will be rejected and judged to be an impostor.

However, the value from the output nodes of the neural network might not be suitable for calculating the threshold because the output value does not directly represent the

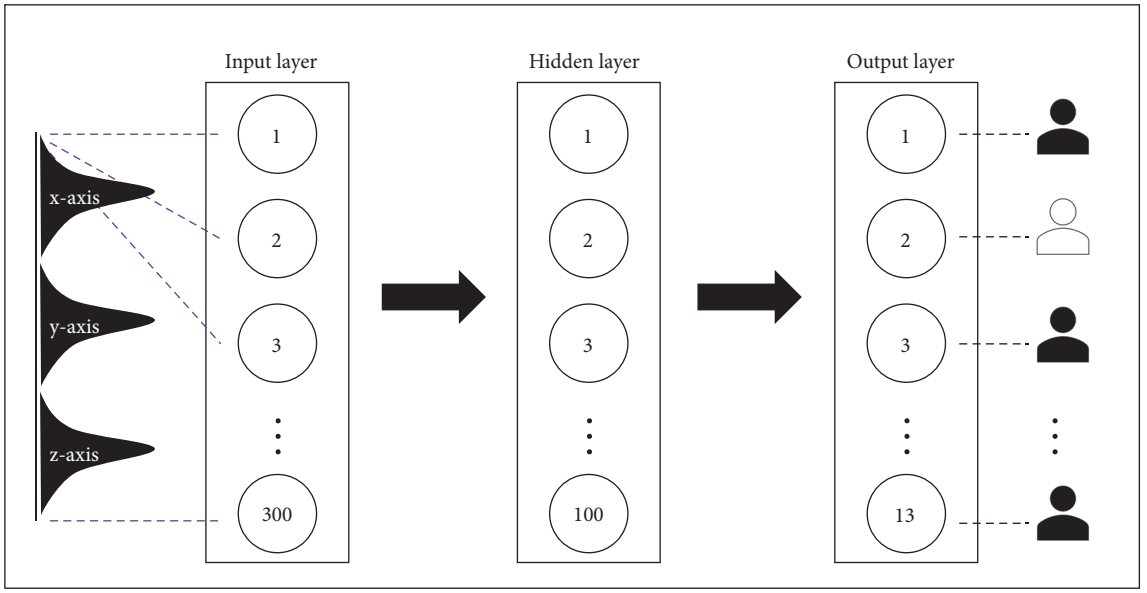

Figure 10: Example of the neural network model.

Input: k constant in KNN k
validation data *validateData*
training data *trainData*
Output: threshold for each driver $threshold_d$
1: **for each** driver d **do**:
2: $validateData_d$ = *validateData* that keep only driver d
3: **for each** $validateData_d$ **do**:
4: $kList_d$ = find k nearest neighbor in *trainData*
5: $topkList_d$ = keep maximum distance from $kList_d$
6: **end for**
7: $threshold_d$ = average distance in $topkList_d$
8: **end for**

Algorithm 3: Threshold calculation for each driver using the KNN.

difference for each histogram. This leads to low impostor detection accuracy, the details of which will be provided in Section 5.7. Hence, we try to utilize the K-nearest neighbors (KNN) algorithm, which can specify the difference of histograms directly by calculating the distance between them. Like the utilization of the neural network, the validation data is still used to calculate the threshold for each driver. To do so, each histogram of the validation data will be used to calculate the Euclidian distance to other histograms in the training data to find the k-nearest neighbors. In this work, using k = 17 provides the highest accuracy. The threshold for each driver is calculated by averaging the maximum distance among the 17 nearest neighbors. Then, in the testing step, 17 new nearest neighbors will be also calculated. The majority of the drivers of those neighbors will be selected as the predicted driver if the closest distance is less than the driver's threshold. The threshold calculation is shown in Algorithm 3 and the impostor detection using the predefined threshold is shown in Algorithm 4.

5. Results and Discussion

This section aims at evaluating the driver identification system in many aspects including parameter tuning and robustness testing of the system. In Section 5.1, our proposed inactive-period filtering module is evaluated to show the improvement of the accuracy after the proposed module is applied. In Section 5.2, the importance of each axis is described together with the system evaluation. In Section 5.3, the optimal values of sliding window and overlapping percentage for the system are determined. Section 5.4 studies the effect of the number of drivers. In Section 5.5, the comparison between the neural network classifier and other classifier algorithms is shown and discussed. Section 5.6 studies the effect of the driving period. Section 5.7 shows the result when using our system to identify the impostor.

We evaluated the system by using the 10-fold cross validation, which divides all data to 10 different training and testing portions. The accuracy would be averaged across 10 different

Input: testing histogram *testData*
training data *trainData*
k constant in KNN *k*
threshold for driver d $treshold_d$
Output: identification result (driver label or impostor) *result*
1: $kList_d$ = from *testData* find *k* nearest neighbor in *trainData*
2: *nearest* = minimum distance in $kList_d$
3: *predictedDriver* = driver label that is the majority in $kList_d$
4: **if** *nearest* is more than $treshold_{predictedDriver}$ **do:**
5: *result* = "impostor"
6: **else do:**
7: *result* = *predictedDriver*

Algorithm 4: Impostor detection using the KNN.

Table 2: Confusion matrix (2x2).

Total population		True condition	
		Condition positive	Condition negative
Predicted condition	Predicted condition positive	True positive	False positive
	Predicted condition negative	False negative	True negative

testing portions. Few histograms from the training portion need to be removed as it would overlap the testing portion. Data of each driver were mixed equally. The main metric is accuracy, which is calculated as shown in (1). All of the accuracies in this section will be calculated by averaging value from the 10-fold cross validation. However, to evaluate the case study of impostor detection, false positive and false negative results are also important. To understand these values, the confusion matrix is illustrated in Table 2. According to the confusion matrix, the three following important metrics are introduced. First, the precision or positive predictive value is calculated as shown in (2). The precision can show how many predicted impostors are correct or it is actually another driver. Second, recall, true positive rate, or sensitivity is calculated as shown in (3). The recall can show how many times the impostor is predicted correctly or it is predicted as other drivers. Precision and recall measures different perspectives. Third, the F1 score or F measure is calculated from the harmonic mean of precision and recall as shown in (4). The F1 score combines information of both the precision metric and recall metric.

$$Accuracy = \frac{Nc}{Na}. \quad (1)$$

When N_c is the number of correctly identified histograms, N_a is number of all testing histograms.

$$precision = \frac{\Sigma\, True\ \ positive}{\Sigma\, Predicted\ \ condition\ \ positive} \quad (2)$$

$$Recall = \frac{\Sigma\, True\ \ positive}{\Sigma\, Condition\ \ Positive} \quad (3)$$

$$F1\ \ score = \frac{2}{1/precision + 1/recall}. \quad (4)$$

5.1. Performance Evaluation of the Inactive-Period Filtering Module. This section aims to show the achieved accuracy after applying the inactive-period filtering module. This module is one of the most important modules, which helps the accuracy significantly increase from our previous work [19]. In our previous work, the accuracy was evaluated in three combinations of the axes shown in Figure 11 colored in blue with a diagonal strip. In this paper, as we proposed the system that applies the inactive-period filtering module, it helps increase the accuracy up to 18% as shown in Figure 11 colored in orange with a white dotted line. From the result of our previous work, using more axes obviously increases the accuracy. However, with the new filtering module, using only the longitudinal axis can achieve more than 90%, which is higher than those of the previous work's results and high enough to be used alone without other axes. The effect of involved axes will be explained in the next subsection.

5.2. Performance Evaluation on Axis Involved. As the acceleration has three axes, this section aims to show the result when using data obtained from a different axis and variation of axes involved. From our previous work, the result was shown in three combinations of the axes which are the x-axis only, the x- and y-axes, and all three axes. The result was already shown again in the previous subsection. In this section, we evaluated our new system using all possible combinations of all axes. Figure 12 shows the accuracy results when combining data from different axes. As can be seen, the x-axis (the longitudinal axis) provides the highest accuracy followed by the y-axis and z-axis, respectively. This is because each axis represents different moving directions of the vehicle. The x-axis is the longitudinal axis, representing the behavior in frontal movement. The y-axis is the lateral axis, representing the behavior when changing lanes or turning. The z-axis is the vertical axis, representing the behavior when driving

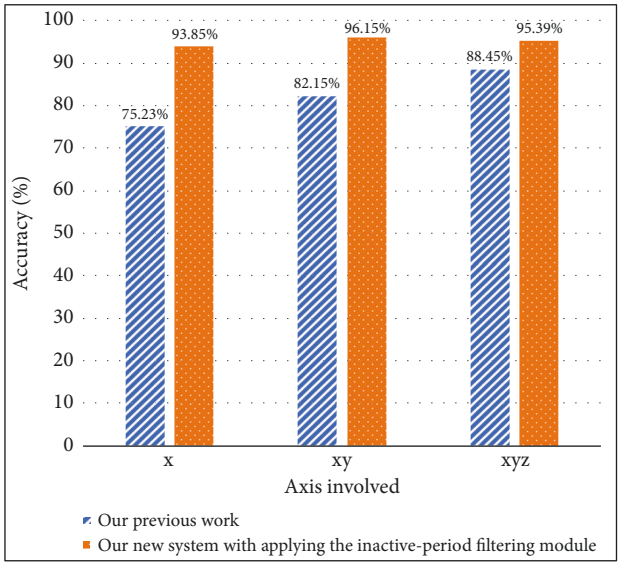

Figure 11: Plot of the accuracy with and without applying the inactive-period filtering module.

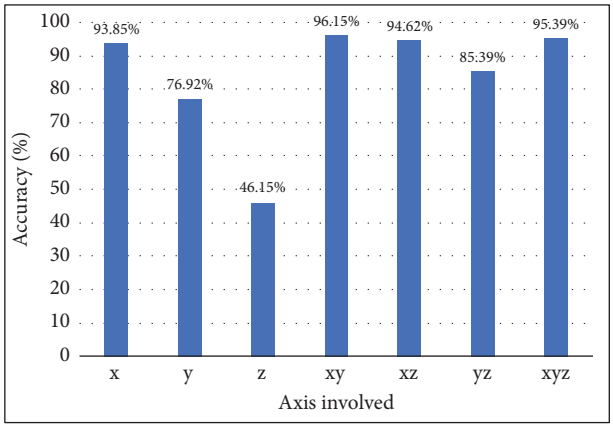

Figure 12: Plot of the accuracy across the axis involved.

across a slope way, such as small bridge, lamp, or speed bumper. Moreover, as can be seen in Figure 12, the results when utilizing data from the xy-axes, xz-axes, and yz-axes are 96.15%, 94.62%, and 85.39%, respectively. With the inactive-period filtering module, utilizing only data obtained from the x-axis in our system can provide accuracy of 93.85%. When combining the data obtained from the x-axis with that from the y-axis and/or z-axis, the accuracy is slightly improved. This reveals the fact that the data from the x-axis is the most important to achieve high accuracy. The idea of axis combination is not limited to the acceleration data. For other driver identification systems, this idea can be applied to other vehicular information that has more than one axis such as velocity or jerk.

5.3. Performance Evaluation of Feature Extraction Module. To represent the behavior of the driver, the acceleration data should be collected for a proper duration. The longer duration

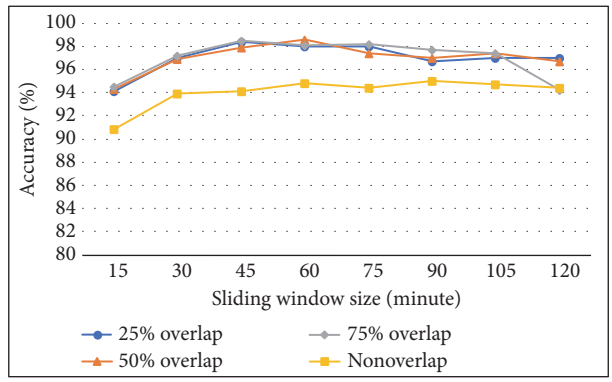

Figure 13: Plot of the accuracy across sliding window size and overlapping percentage.

normally provides more accuracy as the system has more behavior information. However, it leads to longer time to identify the driver, which might not be practical for some applications. Therefore, the duration that is the size of the sliding window should be tested for the optimal value. In our experiment, we varied the sliding window sizes from 15 to 120 minutes. At each sliding window size, we also tested with different overlapping percentages. Figure 13 shows accuracy results from the experiment (note that the y-axis of the figure starts at 80 percent to clarify the difference). As can be seen, the sliding window sizes that provide the highest accuracy are 45 and 60 minutes. With the sliding window size of 45 minutes, the system would identify the driver every 45 minutes, which is not very practical. However, with the overlapping technique, the system can identify the driver more frequently. To choose the overlapping percentage value, it can be seen from Figure 13 that different overlapping percentages do not provide significant difference in accuracy. However, for every tested sliding window size, our system with the overlapping technique apparently yields higher accuracy than that without the overlapping technique. This is because there is probably important driving behavior information at the edge of the nonoverlapping histogram. Using the overlapping technique could help gather such information, leading to higher accuracy. Thus, to achieve the highest accuracy, our system will be configured to use 45 minutes of the sliding window size and 75 percent of the histogram overlapping. With this configuration, the system can identify the driver every 11 minutes. Nevertheless, for some applications that need to identify the driver faster than every 11 minutes, a shorter sliding window size can be chosen. For example, according to Figure 12, if the sliding window size and the overlapping percentage are configured to 15 minutes and 75 percent, respectively, our system can identify the driver every 3-4 minutes with accuracy of 94%. The sliding window and overlapping technique can be applied to most of the driver identification systems as they normally utilize time-series data from the sensors in the vehicle.

5.4. Performance Study on the Effect of the Number of Drivers. Driver identification is a multiclass classification problem.

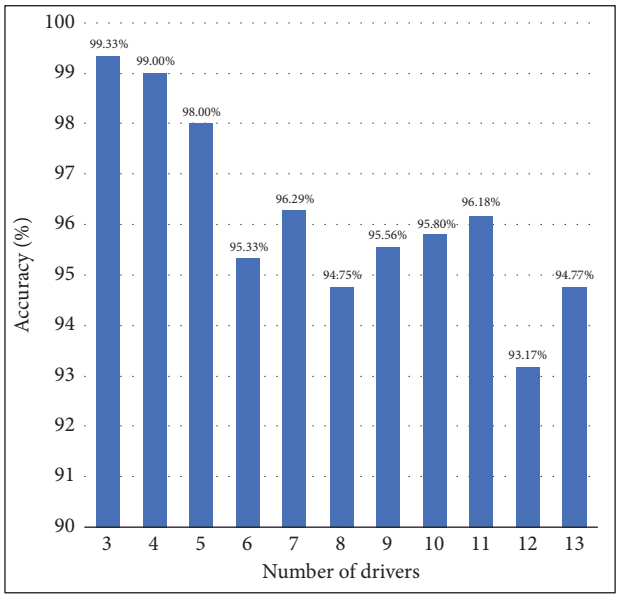

Figure 14: Plot of the accuracy across set of drivers.

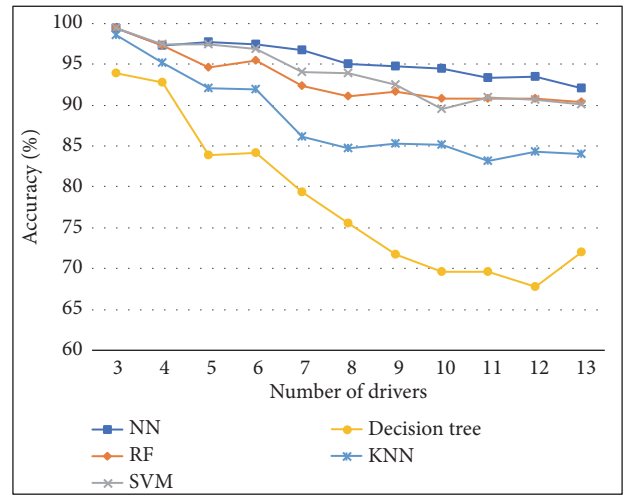

Figure 15: Plot of the accuracy across algorithms and number of drivers.

The number of drivers in this case is the number of classes, which clearly has impact on the classification result. Thus, in our experiment, we vary the number of drivers to be identified from 3 to 13 drivers. According to Section 5.2, we will use only the data obtained from the x-axis, which is the most important axis. Additionally, the sliding window size and overlapping percentage are set to 45 minutes and 75 percent because these parameters provide the highest accuracy as revealed in Section 5.3. The accuracy result is shown in Figure 14 of which the x-axis is the total number of drivers to be identified. Note that the y-axis of the figure starts at 90. As can be seen, the lower the number of drivers to be identified, the higher the accuracy that can be achieved. Specifically, our system can achieve an accuracy of 99.33% in case of identifying 3 drivers and 94.77% in case of identifying 13 drivers.

5.5. Performance Comparison with Other Classifier Algorithms. The classifier algorithm plays an important role in the driver identification system. From our previous work, we originally used a neural network (NN) in our driver identification module. However, in this paper, we aim to show the possibility of using other classifier algorithms together with our proposed histogram feature. The decision tree, Support Vector Machine (SVM), Random Forest (RF), and k-nearest neighbors (KNN) are selected to be evaluated in this experiment. All classifier algorithms are implemented by using the library from CRAN. The decision tree, RF, and SVM are implemented by using *dtree*, *randomForest*, and *e1071*, respectively. They are configured with the default configuration. The NN is configured as explained in Section 3.3, the driver identification module. KNN is configured as explained in Section 4, the case study of impostor detection. Figure 15 illustrates the accuracy achieved by each classifier algorithm. Note that the y-axis starts at 60% to clearly show the difference. The x-axis of the figure denotes the number of drivers to be identified from 3 to 13 drivers. This can show the accuracy of each algorithm affected by a different number of drivers. As can be seen, the NN achieved higher accuracy than other classifier algorithms with every number of drivers. However, the complexity of the NN is the highest among these classifier algorithms. As a result, using the NN requires training time longer than other algorithms. In the aspect of prediction time, however, using the NN or other algorithms approximately requires the same period of time. Thus, for each driver identification application, the classifier algorithms should be evaluated and selected based on the requirement of complexity and accuracy.

5.6. Performance Study on the Effect of Driving Period. This section aims to show the identification accuracy affected by different driving periods. The amount of data, or the driving period, is crucial as it represents the driving hours required for the system to be operated for the first time. The training period should not be too short or too long. On one hand, a too short training period may not represent the driving behavior in the long run. On the other hand, a too long training period may not be practical because it takes time that is too long to collect data. Therefore, we evaluate our system performance with varied driving periods to ensure the integrity of our system. Figure 16 shows the accuracy when the active driving period is varied from 3 to 60 hours. We use 45 minutes of sliding window size with 75 percent overlapping as explained in Section 5.3. As can be seen, the highest accuracy is 94%, achieved when the driving period is set to 10 or 15 driving hours. The accuracy drops when having a driving period shorter than 10 hours. The accuracy also drops when having too much driving data, but it is still above 80% for all cases. In the case of using 10 hours of driving period, if the driver drives 2 hours per day, this will require 5 training days to collect the 10 hours of driving data before the system can identify the driver with 94% accuracy.

Table 3: Summation of the confusion matrix from 10-fold cross validation in the impostor detection experiment when utilizing the neural network.

		True driver			
		Driver A	Driver B	Driver C	Impostor
Predicted driver	Driver A	131	1	0	10
	Driver B	0	132	0	11
	Driver C	0	3	136	35
	Impostor	59	54	54	140

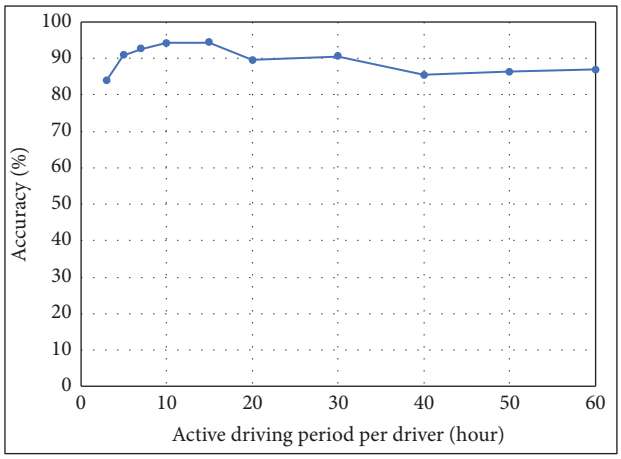

Figure 16: Plot of the accuracy across active driving period per driver.

5.7. Evaluation in the Case Study on Our Proposed Impostor Detection. According to Section 4, we have already explained our modified system for detecting an impostor. To evaluate our impostor detection system, we set a situation where 3 drivers share a vehicle. Therefore, from our 13 drivers, we randomly select 3 drivers to be the genuine drivers. Then, impostors are selected and mixed randomly from the remaining 10 drivers. Then, in the 10-fold cross validation process, we keep 1/10 of the data to be the testing data. We also keep another 1/10 of the data to be the validation data. This validation data is used to calculate the threshold for each driver, which has already been explained in Section 4. Thus, in each fold, the portion of training data would be 8/10. In the system training process, we only input the genuine driver's data, because, in the real situation, we do not have the impostor's data for the first place. Then, we mix the impostor's data, with the same amount of one genuine driver, in the testing data. We evaluate our impostor detection system by considering the precision, recall, and F1 score calculated as shown in (2), (3), and (4), respectively. The value of these metrics only ranges from 0 to 1, representing the performance of the system from the lowest performance to the highest performance. Drivers including impostors have their own precision, recall, and F1 score. We also define an additional metric called overall F1 score. As data across all drivers and impostor is balanced, the overall F1 score is calculated using the macro-average for simplicity as shown in (5). Figure 17 shows the result of each metric when utilizing our proposed

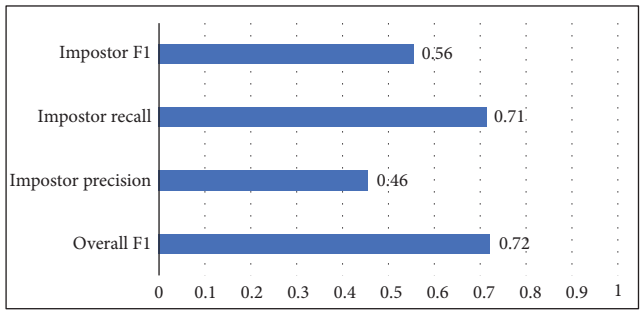

Figure 17: Plot of overall F1, impostor precision, impostor recall, and impostor F1 when utilizing the neural network.

neural network. As can be seen, the overall F1 score is 0.72, which is acceptable. However, if considering only the impostor, the F1 score is only 0.56, which is caused by the low impostor's precision score. The precision is only 0.46, because impostor's histograms were detected as the genuine driver many times. For more clarification of the result, the confusion matrix is also illustrated in Table 3. As can be seen in the first row, 142 histograms are identified as driver A. 131 of them are correctly identified. However, 1 of them is actually driver B and 10 of them are actually the impostor. The second row and the third row can be explained in the same direction. For the fourth row, 304 histograms are identified as the impostor. 140 of them are correctly identified. However, 59 of them are driver A, 54 of them are driver B, and 54 of them are driver C.

$$Overall\ F1 = \frac{F1_{d1} + F1_{d2} + F1_{d3} + F1_{i}}{4}. \quad (5)$$

When $F1_{d1}$ is the F1 score of the first driver, $F1_{d2}$ is the F1 score of the second driver, $F1_{d3}$ is the F1 score of the third driver, and $F1_{i}$ is the F1 score of the impostor.

As can be seen from the result, the performance of the system is not promising. This is because the output nodes of the NN are not suitable for calculating the threshold. Thus, we newly propose an algorithm using the KNN as explained in Section 4. The KNN algorithm involves the utilization of distance function. Thus, the KNN is more appropriate for threshold calculation. Figure 18 shows the precision, recall, and F1 score of the impostor detection system when utilizing the KNN. As can be seen, our system utilizing the KNN outperforms that utilizing the NN in all metrics. Specifically, the overall F1 score increases from 0.72 to 0.87. Considering only the impostor, our system with the KNN can achieve

Table 4: Summation of the confusion matrix from 10-fold cross validation in the impostor detection experiment when utilizing the K-nearest neighbors.

		True driver			
		Driver A	Driver B	Driver C	Impostor
Predicted driver	Driver A	169	1	0	20
	Driver B	0	175	0	0
	Driver C	0	0	170	23
	Impostor	21	14	20	153

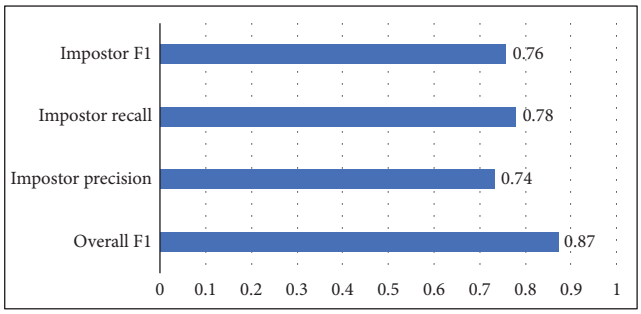

Figure 18: Plot of overall F1, impostor precision, impostor recall, and impostor F1 when utilizing the k-nearest neighbors.

0.76 of the F1 score. This is because the precision and recall increase to 0.74 and 0.78, respectively. For more clarification of the result, the confusion matrix is also illustrated in Table 4. As can be seen in the first row, 190 histograms are predicted as driver A. 169 of them are correctly identified. However, 1 of them is actually driver B and 20 of them are actually the impostor. The second row and the third row can be explained in the same direction. For the fourth row, 208 histograms are predicted as the impostor, 153 of which are correctly identified. However, 21 of them are predicted as driver A, 14 of them are predicted as driver B, and 20 of them are predicted as the impostor.

6. Conclusions

In this paper, we have proposed a public transport driver identification system architecture which utilized only a single acceleration sensor. The system architecture aims to improve the accuracy and the practical usage of the driver identification system for public transport. The system consists of three main modules. The first module, data collection, involved the installation of sensors and data sending. The second module, data preprocessing, involved the data cleaning, inactive-period filtering, and acceleration histogram extraction. The last module, driver identification module, involved the machine learning, which is the neural network. Our system achieved an accuracy of up to 99%. In order to show that our system is practical, the performance evaluation considered several important factors including the inactive-period filtering module, acceleration axis involved, sliding window size and the overlapping technique, classifier algorithms, number of drivers, and driving period. Moreover, by modifying the identification module, we expanded the capability of our model to identify impostors with an overall F1 score of 0.87. Our system architecture design and consideration of aspects on the performance evaluation could be used as a guideline for real public transport driver identification system and further driver identification researches.

Data Availability

The data used to support the findings of this study are owned by a company, but it can be provided from the corresponding author upon request.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This research was supported by Chula Computer Engineering Graduate Scholarship for CP Alumni, Chulalongkorn University. The authors would like to thank all members of ISEL, family, and friends for their help.

References

[1] Statista, *Internet of Things (IoT) connected devices installed base worldwide from 2015 to 2025 (in billions)*, 2016, https://www.statista.com/statistics/471264/iot-number-of-connected-devices-worldwide/.

[2] Gartner, *Gartner Says 8.4 Billion Connected "Things" Will Be in Use in 2017, Up 31 Percent From 2016*, 2017, https://www.gartner.com/newsroom/id/3598917.

[3] Ericsson, "Internet of Things forecast," 2018, https://www.ericsson.com/en/mobility-report/internet-of-things-forecast.

[4] Gartner, *Gartner Says By 2020, a Quarter Billion Connected Vehicles Will Enable New In-Vehicle Services and Automated Driving Capabilities*, 2015, https://www.gartner.com/newsroom/id/2970017.

[5] C. Marina Martinez, M. Heucke, F.-Y. Wang, B. Gao, and D. Cao, "Driving style recognition for intelligent vehicle control and advanced driver assistance: a survey," *IEEE Transactions on Intelligent Transportation Systems*, vol. 19, no. 3, pp. 666–676, 2018.

[6] INTERPOL, *Vehicle crime*, 2017, https://www.interpol.int/Crime-areas/Vehicle-crime/Database-statistics.

[7] C. Miyajima, Y. Nishiwaki, K. Ozawa et al., "Driver modeling based on driving behavior and its evaluation in driver identification," *Proceedings of the IEEE*, vol. 95, no. 2, pp. 427–437, 2007.

[8] A. Wahab, C. Quek, C. K. Tan, and K. Takeda, "Driving profile modeling and recognition based on soft computing approach," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 20, no. 4, pp. 563–582, 2009.

[9] I. Del Campo, R. Finker, M. V. Martinez, J. Echanobe, and F. Doctor, "A real-time driver identification system based on artificial neural networks and cepstral analysis," in *Proceedings of the 2014 International Joint Conference on Neural Networks, IJCNN 2014*, pp. 1848–1855, China, July 2014.

[10] M. Enev, A. Takakuwa, K. Koscher, and T. Kohno, "Automobile driver fingerprinting," *Proceedings on Privacy Enhancing Technologies*, vol. 2016, no. 1, 2016.

[11] D. Hallac, A. Sharang, R. Stahlmann et al., "Driver identification using automobile sensor data from a single turn," in *Proceedings of the 19th IEEE International Conference on Intelligent Transportation Systems, ITSC 2016*, pp. 953–958, Brazil, November 2016.

[12] M. V. Martínez, J. Echanobe, and I. Del Campo, "Driver identification and impostor detection based on driving behavior signals," in *Proceedings of the 19th IEEE International Conference on Intelligent Transportation Systems, ITSC 2016*, pp. 372–378, Brazil, November 2016.

[13] N. C. Fung, B. Wallace, A. D. C. Chan et al., "Driver identification using vehicle acceleration and deceleration events from naturalistic driving of older drivers," in *Proceedings of the 12th IEEE International Symposium on Medical Measurements and Applications, MeMeA 2017*, pp. 33–38, USA, May 2017.

[14] S. Ezzini, I. Berrada, and M. Ghogho, "Who is behind the wheel? Driver identification and fingerprinting," *Journal of Big Data*, vol. 5, no. 1, p. 9, 2018.

[15] F. Martinelli, F. Mercaldo, A. Orlando, V. Nardone, A. Santone, and A. K. Sangaiah, "Human behavior characterization for driving style recognition in vehicle system," *Computers and Electrical Engineering*, 2018.

[16] L. Moreira-Matias and H. Farah, "On developing a driver identification methodology using in-vehicle data recorders," *IEEE Transactions on Intelligent Transportation Systems*, vol. 18, no. 9, pp. 2387–2396, 2017.

[17] S. Jafarnejad, G. Castignani, and T. Engel, "Towards a real-time driver identification mechanism based on driving sensing data," in *Proceedings of the 20th IEEE International Conference on Intelligent Transportation Systems, ITSC 2017*, pp. 1–7, Japan, October 2017.

[18] P. Phumphuang, P. Wuttidittachotti, and C. Saiprasert, "Driver identification using variance of the acceleration data," in *Proceedings of the 19th International Computer Science and Engineering Conference, ICSEC 2015*, pp. 1–6, IEEE, Thailand, 2015.

[19] N. Virojboonkiate, P. Vateekul, and K. Rojviboonchai, "Driver identification using histogram and neural network from acceleration data," in *Proceedings of the 17th IEEE International Conference on Communication Technology, ICCT 2017*, pp. 1560–1564, China, October 2017.

[20] A. Chowdhury, T. Chakravarty, A. Ghose, T. Banerjee, and P. Balamuralidhar, "Investigations on Driver Unique Identification from Smartphone's GPS Data Alone," *Journal of Advanced Transportation*, vol. 2018, Article ID 9702730, 11 pages, 2018.

[21] InvenSense, *MPU-6050 Six-Axis (Gyro + Accelerometer) MEMS MotionTracking™ Devices*, 2013, https://www.invensense.com/products/motion-tracking/6-axis/mpu-6050/.