Hindawi
Journal of Advanced Transportation
Volume 2017, Article ID 6057830, 13 pages
https://doi.org/10.1155/2017/6057830

WILEY | Hindawi

Research Article

Combining Unsupervised Anomaly Detection and Neural Networks for Driver Identification

Thitaree Tanprasert,[1] Chalermpol Saiprasert,[2] and Suttipong Thajchayapong[2]

[1]*Department of Computer Science, Harvey Mudd College, Claremont, CA, USA*
[2]*National Electronic and Computer Technology Center (NECTEC), National Science and Technology Development Agency (NSTDA), Pathum Thani, Thailand*

Correspondence should be addressed to Suttipong Thajchayapong; suttipong.thajchayapong@nectec.or.th

Received 30 June 2017; Accepted 7 September 2017; Published 16 October 2017

Academic Editor: Andrea Monteriù

This paper proposes an algorithm for real-time driver identification using the combination of unsupervised anomaly detection and neural networks. The proposed algorithm uses nonphysiological signals as input, namely, driving behavior signals from inertial sensors (e.g., accelerometers) and geolocation signals from GPS sensors. First anomaly detection is performed to assess if the current driver is whom he/she claims to be. If an anomaly is detected, the algorithm proceeds to find relevant features in the input signals and use neural networks to identify drivers. To assess the proposed algorithm, real-world data are collected from ten drivers who drive different vehicles on several routes in real-world traffic conditions. Driver identification is performed on each of the seven-second-long driving behavior signals and geolocation signals in a streaming manner. It is shown that the proposed algorithm can achieve relatively high accuracy and identify drivers within 13 seconds. The proposed algorithm also outperforms the previously proposed driver identification algorithms. Furthermore, to demonstrate how the proposed algorithm can be deployed in real-world applications, results from real-world data associated with each operation of the proposed algorithm are shown step-by-step.

1. Introduction

Recently, current technological advancements such as cutting edge sensor network, high speed wireless infrastructure, and Internet of Things (IoT) technologies have propelled autonomous driving into reality [1, 2]. Autonomous driving has been implemented as part of the advanced driver assistance system (ADAS) to improve road safety as found in Tesla autopilot driving mode [3]. However, one of the well known issues associated with connected and autonomous vehicles is the security concerns related to vulnerability of getting wirelessly hacked, which would allow hackers to take full control of the vehicle [4, 5]. Automatic driver identification can be utilized to detect whether an autonomous vehicle has been hacked. A change in steering wheel control, brake, and accelerator pedals positions would indicate a shift in driving patterns which deviate from normal driving behavior of the car's owner. One way to detect these changes is to use a driver identification approach [6] to perform proactive measures including providing real-time warning, slowing down the vehicle, and eventually switching off the engine for the safety of the driver and passengers.

Accurate driver identification could improve vehicle's security and traffic safety control measures as well as providing more insights into physical and psychological aspects related to individual's driving behaviors. Modeling, recognition, and classification of driving behaviors have been studied using various statistical and computing methods [7, 8]. With the rise of sensor technologies embedded in smartphones, driving behavior signals can be collected and assessed more readily and conveniently. Therefore, accurate real-time low-cost driver identification can be realized by utilizing these driving behavior signals.

Unfortunately, data from direct human actions such as turning angles of steering wheels and the pattern of stepping on brake and acceleration pedals require specialized in-vehicle sensors. Consequently, these signals cannot be easily obtained in large scale. Instead, driving behavior signals from inertial sensors (e.g., lateral and longitudinal accelerations from accelerometers) and geolocation signals from GPS

sensors are commonly used for driver identification as they can be measured using conventional sensors embedded in widely used portable devices (e.g., smartphones) [6]. Thus, developing algorithms for these input signals would widen the deployment of driver identification capability itself.

This paper proposes a driver identification algorithm designed to use driving behavior and geolocation signals for real-time applications (e.g., for instantaneous traffic safety purpose [8]). First, an anomaly detection is performed to preliminarily assess if the current driving behavior signals deviate from the expected patterns. Then, driver identification is activated where feature extraction and construction take into account both the qualities of input signals (characteristics of patterns and noise) and physical characteristics of driving behaviors (e.g., aggressiveness and steadiness) before being fed into neural networks. Using real-world data, it is shown that only seven-second-long driving behavior signals and geolocation signals are used to identify drivers. Furthermore, the proposed algorithm achieves relatively high accuracy and also outperforms the previously proposed algorithms [6, 7, 9, 10]. Lastly, for implementation purposes, real-world results associated with each operation of the proposed algorithm are shown step-by-step.

The novelty of the proposed algorithm can be summarized as follows:

(1) *Using unsupervised anomaly detection to minimize the necessity of assessing every input signal*: driver identification usually involves recognition of numerous different patterns and previously proposed algorithms often find it necessary to recognize and assess every instant of input signals [7, 11–13]. However, for real-time applications, recognizing patterns by assessing every instant of input signal may not be practical as it can incur delays and also consume computational resources. Unlike previous algorithms, the proposed algorithm uses anomaly detection as a preliminary operation where an anomaly would indicate that the person driving the vehicle may not be whom we expect to be. The driver identification operation is then activated only if an anomaly is detected, thus minimizing the necessity of assessing every input signal.

(2) *Identifying drivers based on low-level signals using neural network with feature extraction and model construction*: the proposed algorithms focus on using low-level signals from inertial sensors found in off-the-shelf devices (e.g., smartphones) unlike previous algorithms which require specialized sensors and/or driving simulators [7, 8, 11–14]. The challenge is that, before being effectively deployed in real-world scenarios, driver identification algorithms will have to learn many distinct behaviors. Consequently, it is more likely that similar behaviors are observed from different drivers, which can reduce the accuracy of driver identification algorithms. To address this challenge, the proposed algorithm incorporates feature extraction and model construction to make driving behavior signals as distinguishable as possible, and, subsequently, enhance the neural networks by facilitating fast model construction and more accurate driver identification.

The remainder of this paper is organized as follows. Section 2 discusses previous researches involving driving behavior classification methods, the use of driving signals from inertial sensors, and previous approaches to similar problems. Section 3 presents the challenges and problems observed in the input data, illustrates the proposed algorithm and the assumption behind it, and provides details on identification process. Section 4 describes data collection process from real-world situation and experimental setups. The resulting accuracy and those associated with each operation of the proposed algorithm are then discussed in Section 5. Finally, Section 6 concludes this paper.

2. Related Work

2.1. Modeling of Driving Behaviors. Many signals could be used to represent or to model driving behaviors. Previous studies have been conducted on using direct human behavior signals, which are the record of activities performed by the drivers, including acceleration and brake pedal operations and steering angle of the steering wheel. The work in [11] proposed a method to identify drivers based on Artificial Neural Network and Cepstral Analysis using two driving signals collected from a specialized sensor-equipped vehicle which are gas pedal pressure and brake pedal pressure. They found that the identification rates decrease as the number of drivers to be identified increases. A success rate of 84% was reported for the case of 3 drivers. Although their method achieves a high identification rate, the data set used was only for 3 drivers. As will be seen later in this paper, our proposed approach is capable of producing comparable accuracy with higher number of drivers.

Few other studies also combine vehicle-to-vehicle (V2V) signals, which is the distance between vehicles in car-following maneuver [7, 8]. In both studies, the relationship between following distance and vehicle's velocity, along with the patterns of pedal activities, is modeled using Gaussian Mixture Model (GMM). The model in [7] is even tested in real-world situation with the result of 76.8% accuracy. However, input signals representing car-following maneuver in [7, 8] cannot be collected via widely used on-board devices (e.g., smartphones), which limits large-scale deployment.

The algorithm proposed in [15] uses neural network in an attempt to classify driving styles into 5 categories. The paper shows a potential in using neural network for driving behavior classification. However, the behavior signals collected in the experiment include pedal operations and yaw rate, which cannot be collected via inertial sensors. Moreover, the algorithm could only identify driving styles, not the drivers themselves.

2.2. Driver Identification in Real-World Situations. A number of previous studies have shown that drivers can be uniquely identified using driving behavior signals [7, 8, 11, 16, 17]. A variety of sensors can be found in off-the-shelf devices

available in current market. Amongst them are inertial and GPS sensors, which have useful signals for processing and interpreting driving behaviors.

In [18], driving behavior signals from smartphones are used in combination with dynamic time warping (DTW) in order to classify driving behaviors into typical and aggressive categories. This study shows that signals from inertial sensors can represent classifiable patterns of driving behaviors, so, it can potentially be used in more complex classification process such as driver identification. The algorithm proposed in this paper is built up from this idea in order to identify individuals according to their driving behaviors. In respect to the previous study in [15], the two main challenges of the proposed algorithm in this paper are (1) the same driver may have various driving behavior patterns and (2) the possibility of having two or more drivers who have very similar driving behavior patterns.

Deployment of anomaly detection has also been mentioned in [17], but it is used primarily for driving behavior identification, not driver identification. Unlike [17], the proposed algorithm in this paper applies anomaly detection before identifying the drivers themselves.

Gaussian Mixture Model (GMM) is amongst the most widely used algorithms for driver identification [7]. Using GMM for feature extraction, the algorithm proposed in [9] assumes that the frequencies of the amount each driver presses his/her feet on a pedal can be used to identify drivers. In this paper, GMM is used as a benchmark algorithm for performance comparisons.

More recently proposed algorithms use Support Vector Machine (SVM) and are more focused in using driving behavior signals as inputs [10, 16]. In [16], signals from inertial sensors, namely, accelerometer and gyroscope, are processed in order to detect accelerating, braking, and wheel-turning events. The patterns of said events are analyzed further with K-means clustering and SVM to identify the drivers. Data collection in the experiment is done in natural, real-world situation. However, the classification is only assessed on two drivers on the same vehicle. In this paper, SVM is also used as one of the benchmark algorithms for performance comparisons.

We have previously proposed a driver identification algorithm in [6]. This algorithm adapts the principal component analysis (PCA) approach, using only lateral and longitudinal accelerations. The identification of driver is decided based on the angle between the principal components of the test data and the principal components of all training data. The algorithm proposed in this paper is a significant enhancement of our previous algorithm in [6] as a new and more rigorous approach is used. Experiments are also conducted with more real-world drivers.

3. Methodology

3.1. Overview of the Proposed Algorithm. The proposed algorithm is based on the following assumptions:

(1) Every driver exhibits unique driving characteristics which can be recognized and classified by driving signals from inertial and GPS sensors (e.g., acceleration, instantaneous speed, and headings) [7].

(2) Every driver may exhibit different driving characteristics in different traffic conditions and on different routes. A driving pattern from one trip of a driver does not necessarily provide complete model of the driver's behavior [19]. This assumption is crucial in both data preprocessing and experimental setup.

(3) The algorithm should be able to work with common characteristics (e.g., sampling rates) of input signals measured by off-the-shelf devices, so that it can be practically and widely used in real-time applications even on devices with limited efficiency.

The proposed algorithm can be divided into 4 consecutive stages as shown in Figure 1: anomaly detection, data preprocessing, classification model using Feed-Forward Neural Network, and decision making. Firstly, anomaly detection is performed on the raw driving signals in order to see if driving behavior deviates from the usual patterns. If an anomaly is detected, the algorithm proceeds to driver identification. The purpose of the preprocessing is to create time-series of input signals that represent driving behaviors over the optimal span of time to enhance the classification capability. The classification model is then constructed using neural network to try to distinguish drivers as much as possible. Finally, the output of the classification model is interpreted with the decision making algorithm to make the final identification.

3.2. Input Description. Low-level driving signals used in the algorithm are obtained from inertial sensors (e.g., accelerometers) and GPS sensors. These are standard sensors on most portable devices, making it low-cost and convenient to collect driving signals from drivers in real-world environment.

(1) The accelerometer measures acceleration in lateral and longitudinal directions separately, every 200 milliseconds, which equates to frequency of measurement of 5 Hz. The x direction data represents the vehicle's lateral acceleration, which results from the turning the steering wheel. The y direction data represents the vehicle's longitudinal acceleration, indicating braking and accelerating.

(2) The GPS receiver measures 5 signals: latitude, longitude, instantaneous speed, heading, and altitude. The measurement is done every 1 second which equates to frequency of measurement of 1 Hz.

The input data from both the accelerometer and GPS receiver have time stamps. Therefore, we can match the data that are measured at the same time and use the correlation of data from both sensors to classify and identify drivers.

3.3. Anomaly Detection. Anomaly detection is carried out prior to driver identification process. The main purpose is to detect whether driving behavior deviates from the norm, which is defined as a series of patterns that occur most frequently for each driver. A deviation will indicate

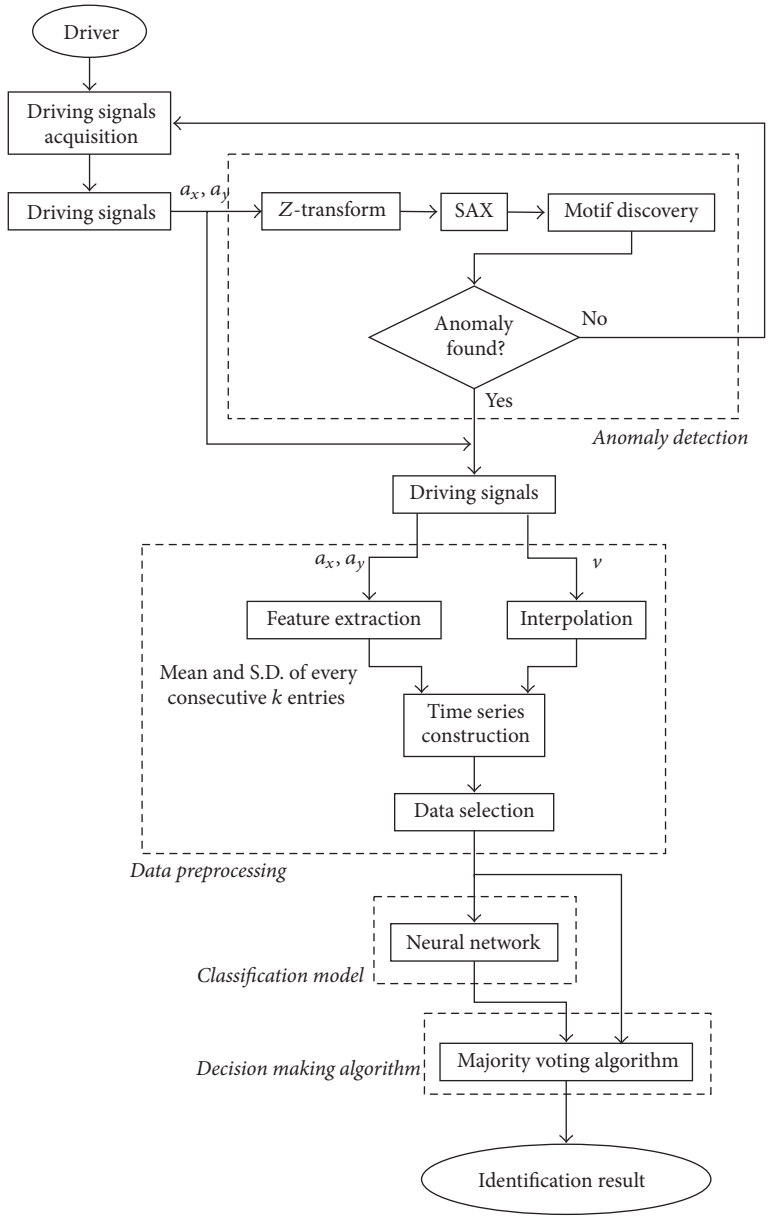

Figure 1: Proposed algorithm for constructing classification model and driver identification.

an anomaly which implies that a driver currently driving the vehicle may not be whom we expect to be, and further rigorous identification operations are needed. The anomaly detection operation is adopted from [20].

3.3.1. Z-Transform. The purpose of this stage is to normalize acceleration signals.

3.3.2. Discretization Using SAX. At this stage, continuous time-series acceleration signal is discretized into alphabetical symbols using Symbolic Aggregate approXimation (SAX) proposed in [21]. Using SAX, continuous acceleration signal is approximated into sequence of alphabets with each alphabet representing predefined set of breakpoints which divides the distribution space into equiprobable regions. The length of a sequence is determined by a sliding window. In each window, users can specify the alphabet size which tells us how many levels of alphabet to choose from and also the number of segments which each sliding window is to be broken down into. For example, using alphabet size as 3 and number of segment as 6, a time-series signal A_1 can be discretized into $A_1 = cba_1bba_2abb_3cba_4bba_5cba_6$. For this instance, a could refer to low acceleration values, b being medium and c being high. Values of SAX parameters are data dependent. One obvious benefit of discretizing a time-series is that it reduces the computation time especially when raw data is being collected at a high sampling frequency for long period of time. This makes it suitable for real-time implementation.

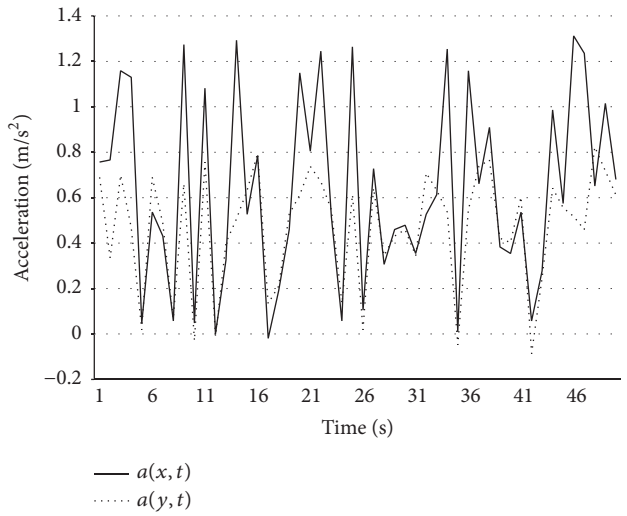

Figure 2: Acceleration in x and y direction of a real-world driver measured during 10 seconds while the vehicle is at rest.

3.3.3. Finding the Norm Using Motif Discovery. The next stage is to establish normal driving behavior for each driver. We achieve that by keeping records of which sequence is repeated most in the discretized data. Taking the example above, for A_1, the most repeated sequence is *cba* which appears 3 times. Finding the most repeated patterns or sequence is known as Motif Discovery [22]. Therefore, we can characterize each driver with the top n most repeated patterns. A tool proposed in [22] was adopted to find the most repeated sequence in a sample dataset with driving signals. Raw driving signals which produce discretized sequences that deviate from the most repeated patterns of a designated driver will be classified as an anomaly. Once an anomaly is detected, the driver identification operation begins.

3.4. Data Preprocessing. Data preprocessing is performed on the acceleration and speed signals, collected from accelerometer and GPS sensors, respectively. It is necessary to eliminate unnecessary or misleading data and transform the data such that the individual driving behavior is most accurately portrayed for effective driver identification. As shown in Figure 1, there are four operations in data preprocessing, as follows:

3.4.1. Feature Extraction on Acceleration Data. Preliminary analysis of raw data collected from the accelerometers reveals that there is a considerable amount of noise in the signals. This is due to the sensitivity of the hardware itself. Figure 2 shows 10 seconds of measured acceleration in x and y direction from a real-world driver, while the speed information indicates that the car is not moving. A classification model should not be constructed based on the patterns of these noises, since they are from uncertainties of the sensors or the vibration from the vehicles, not part of driving behaviors. Moreover, the noise adds to the acceleration value which subsequently causes the measured value to be higher or lower than the actual value. Therefore, it is important to remove these noises as much as possible in order to model driving behavior accurately, leaving only significant signals which relate to what the driver actually does.

In order to reduce these noises, the mean of acceleration in a specific interval of time is computed. Furthermore, the standard deviation of the same interval is computed because it can capture the sudden change in acceleration, which is significant for classification of driving behaviors. To preserve as much essential information from initial acceleration, the mean and standard deviation are calculated in a streaming manner for every consecutive entries k to $k + 4$, for $k = 0, 1, 2, \ldots, K - 5$, where K is the number of entries. The consecutive acceleration values are measured 200 milliseconds apart, so increasing k by 1 causes a 200-millisecond shift in the calculation; that is, the consecutive pairs of mean and standard deviation calculated represent behaviors, which are 200 millisecond apart. We note that even though this calculation is for the 5-Hz sampling frequency of the accelerometers currently available on many personal devices in the market (e.g., smartphones), it can be adjusted for other devices with different sampling frequencies.

3.4.2. Speed Interpolation. According to the assumption that drivers may have different behaviors in different traffic conditions, we aim to match the behaviors represented by acceleration data to the specific external conditions in which they are measured. The rate at which the traffic flows, which is part of traffic condition, could be inferred by the instantaneous speed signals. Thus, the acceleration data should be processed with the instantaneous speed measured at the same moment in time, so that the correlation between driving behaviors and external traffic conditions can be observed. However, the maximum sampling frequencies of accelerometers and GPS sensors in majority of devices are usually not in the same range. For example, an accelerometer might be sampled at 5 Hz while a GPS sensor is sampled at 1 Hz. Therefore, an interpolation is deployed to cope with this challenge. An example of this operation is illustrated in Section 5.2.

3.4.3. Time-Series Construction. Time-series are constructed to represent a brief pattern of driving behavior, which consist of changes in accelerations in x and y direction, absolute value of the instantaneous speed, and also the relationship between the said variables. The optimal size of time-series is determined empirically, which is explained in Section 5.2.

3.4.4. Data Selection. The input from accelerometer and GPS receiver is from the entire trip. It has the information when the vehicle is at rest, due to congestion or traffic lights. Since the drivers do not drive when the vehicle is at rest, the information from this period should not be considered in construction of the classification model and identification process. Therefore, the time-series, which have the data from when speed of the vehicle = 0, would be eliminated in the preprocessing.

3.5. Classification Model Using Feed-Forward Neural Network (FFNN). In order to classify several drivers, we need several

classification hyperplanes. The hyperplanes form nonlinear boundary, which separates multidimensional driving behavior signals of the drivers from each other. For the proposed algorithm, a Feed-Forward Neural Network (FFNN) is selected because it is one of the most computationally simple neural network models, which enables fast execution for real-time identification. It is customized to learn the training data and perform identification for each time-series in the testing data.

Based on preliminary assessment of the algorithm with the training data set, FFNN with one hidden layer is used. Therefore, our FFNN consists of three layers: the input layer, the hidden layer, and the output layer. In the proposed algorithm, FFNN is used to learn a classification function: $C : \underline{X} \rightarrow \underline{Y}$, where $\underline{X} = \{x_t^i\}$ is a set of input to the ith neuron in the input layer of FFNN at time t, and $\underline{Y} = \{y_t^k\}$ is an output from the kth neuron in the output layer at time t. The sigmoid transfer function, $\mathrm{SIG}(x) = 1/(1 + e^{-x})$, used in the proposed algorithm can be represented as shown in (1), where w^{ij} is the weight for the connection from node i in the input layer to node j in the hidden layer and v^{jk} is the weight for the connection from node j in the hidden layer to node k in the output layer:

$$y_t^k = \mathrm{SIG}\left(\sum_j v^{jk} p_t^j\right), \quad p_t^j = \mathrm{SIG}\left(\sum_i w^{ij} x_t^i\right). \tag{1}$$

The number of neurons in the hidden layer depends on the number of hyperplanes needed to construct the boundaries, which classify the input. In this paper, the number of neurons is chosen with the aim of balancing between not having too few neurons (e.g., the capability of modeling nonlinear mapping) and not having too many neurons (e.g., the problems of excessive time-consuming and having too many local minimums). Starting from two neurons, the number is multiplied by two each time as long as the average accuracy of the identification increases, which is similar to binary search. It is found that 10 hidden neurons provide the highest average accuracy.

In this paper, we set the number of training cycle to be 500 and the error epsilon to be $1.0E-5$. These two parameters affect both the running time to create the classification model and the accuracy of the identification process. Details on how the FFNN operates with real-world input data are discussed in Section 5.3.

3.6. Decision Making. Outputs from the classification model on every time-series input signal are taken into account to a make final decision on the identity of the driver. Majority voting algorithm is used to make the final decision. This operation is illustrated when the algorithm is applied to real-world data in Section 5.3.

4. Experimental Setup Using Real-World Data

4.1. Real-World Data Collection. The data in the experiment is collected from real-world driving scenarios, using accelerometers and GPS sensors on smartphones. Part of this data set was used in the experiment in [6]. The data is collected over a 2-month period from 10 school bus drivers, designated as drivers $1, 2, \ldots, 10$, respectively. Each vehicle was assigned with one driver. On each vehicle, a smartphone is rigidly placed horizontally above driver console facing the direction of travel. The routes chosen are approximately 30 km long on average. These routes are on the outskirt of Bangkok consisting of local streets with low traffic volume, collector roads, and arterial roads with high traffic volume to cover scenarios with varying road and traffic conditions. Each driver takes different route, although some routes may overlap. The routes for each driver are also different in the morning and in the afternoon.

As the data collection was not performed in a controlled environment, large amount of training data is crucial to ensure the occurrence of driving behaviors in as many different scenarios as possible. We increase variety of inputs by using data from various trips, where a trip means a journey starting from where each vehicle was originally parked to the driver's designated destination. In this experiment, 2 morning trips and 2 afternoon trips from each driver are used. Therefore, the total number of trips analyzed is 40 trips, as shown in Table 1.

The information from each trip is divided into n intervals of information, referred to as windows, as shown in Figure 3. Following the second assumption in Section 3.1, several windows from the same trip ensure similarities between driving behaviors in training and testing data. The length of the interval is represented by the variable m, as shown in Figure 3. For each experiment, window k from trip j of each driver is the testing data. The remaining $n - 1$ windows from trip j and all windows from the other 3 trips are training data. In this case, with 4 trips per driver, $4n$ different tests are created.

The number of windows n and the number of input rows in each window m determine the length of time in which the driving signals have to be collected to make accurate classification model and results. However, note that consecutive input rows are not necessary from consecutive time stamps in the original data, since the data from when the vehicle is at rest are eliminated. Finally, this input data is fed into the proposed algorithm in a streaming manner to assess driving behavior signals and GPS signals.

4.2. Selection of Benchmark Algorithms. We have considered a number of previously proposed algorithms based on the possibilities of being implemented for comparisons [7, 8, 16, 18]. The algorithm proposed in [16] processes data from inertial sensors. However, it needs yaw angular velocity in order to model steering event. Some cars are equipped with the sensor that can detect the angle, but we cannot collect that information from accelerometer and GPS sensors on smartphones. The algorithms proposed in [7, 8] are based on car-following scenarios. Hence, they require the distance between the interested vehicle and the one in front of it. The velocity of the vehicle in front is also needed for algorithm in [8], while pedal operation patterns are needed for algorithm in [7]. The algorithm proposed in [18] requires behavior signals from gyroscope and magnetometer, in addition to GPS sensors and accelerometers. Moreover, in order to remove

Table 1: This table shows the information of 40 trips which are used in the algorithm performance evaluation.

Drivers #	Date	Start time	End time	Start coordinate	End coordinate
(1)	18/07/14	14:21	14:29	(13.82415, 100.4342)	(13.82545, 100.4125)
(1)	05/08/14	08:27	09:14	(13.82355, 100.4329)	(13.82409, 100.437)
(1)	22/08/14	14:16	14:32	(13.82431, 100.4323)	(13.83162, 100.4131)
(1)	27/08/14	08:45	09:53	(13.83159, 100.4136)	(13.82403, 100.4369)
(2)	16/07/14	07:14	08:10	(13.81396, 100.4573)	(13.80158, 100.4372)
(2)	17/07/14	07:19	08:11	(13.81397, 100.4572)	(13.80159, 100.4371)
(2)	18/07/14	15:44	16:02	(13.87538, 100.4524)	(13.81977, 100.4493)
(2)	23/07/14	16:00	16:29	(13.83166, 100.4477)	(13.82306, 100.4744)
(3)	14/08/14	14:24	14:31	(13.82414, 100.443)	(13.8246, 100.4573)
(3)	02/09/14	07:50	08:05	(13.82276, 100.4745)	(13.82287, 100.432)
(3)	04/09/14	07:48	08:07	(13.82273, 100.4744)	(13.82217, 100.4275)
(3)	09/09/14	14:29	14:43	(13.82734, 100.4524)	(13.82302, 100.4745)
(4)	17/07/14	16:05	16:23	(13.78181, 100.4428)	(13.82096, 100.4487)
(4)	30/07/14	07:46	07:51	(13.82092, 100.4417)	(13.82095, 100.4416)
(4)	04/08/14	07:33	08:13	(13.82356, 100.4325)	(13.82373, 100.4324)
(4)	05/08/14	15:27	15:51	(13.82091, 100.4491)	(13.78172, 100.443)
(5)	16/07/14	07:24	07:31	(13.78171, 100.4431)	(13.82092, 100.4487)
(5)	24/07/14	14:09	14:43	(13.82405, 100.4323)	(13.78131, 100.4745)
(5)	15/08/14	07:29	07:43	(13.78164, 100.4431)	(13.82096, 100.4487)
(5)	21/08/14	14:35	14:43	(13.82095, 100.4491)	(13.78164, 100.4431)
(6)	04/08/14	14:26	15:02	(13.82089, 100.4492)	(13.80154, 100.4372)
(6)	05/08/14	14:21	15:22	(13.82296, 100.4346)	(13.76514, 100.4447)
(6)	08/08/14	07:36	07:48	(13.78169, 100.4428)	(13.82087, 100.4488)
(6)	19/08/14	07:24	08:06	(13.76529, 100.4446)	(13.8231, 100.4331)
(7)	28/07/14	15:15	15:24	(13.83164, 100.4136)	(13.83169, 100.4474)
(7)	14/08/14	15:13	15:25	(13.83169, 100.4136)	(13.83172, 100.4131)
(7)	19/08/14	08:14	08:35	(13.83165, 100.4135)	(13.8241, 100.4369)
(7)	25/08/14	08:21	10:02	(13.83158, 100.4136)	(13.82317, 100.4441)
(8)	15/07/14	15:13	15:28	(13.82359, 100.4323)	(13.83173, 100.4474)
(8)	29/07/14	15:11	15:29	(13.82397, 100.4322)	(13.83169, 100.4474)
(8)	19/08/14	08:18	08:31	(13.8318, 100.4136)	(13.82278, 100.4309)
(8)	03/09/14	08:24	08:41	(13.83188, 100.4136)	(13.82276, 100.4317)
(9)	29/07/14	14:10	14:27	(13.82356, 100.4324)	(13.8316, 100.4474)
(9)	01/08/14	08:01	09:31	(13.83179, 100.4477)	(13.82294, 100.4277)
(9)	04/08/14	08:00	08:23	(13.83171, 100.4477)	(13.82558, 100.4396)
(9)	07/08/14	14:11	14:27	(13.82368, 100.4324)	(13.83163, 100.4474)
(10)	31/07/14	07:47	08:03	(13.79481, 100.3993)	(13.82276, 100.4308)
(10)	31/07/14	14:55	15:13	(13.81317, 100.4226)	(13.79481, 100.3993)
(10)	05/08/14	07:44	08:01	(13.79491, 100.3993)	(13.82268, 100.4297)
(10)	05/08/14	14:25	15:06	(13.82419, 100.4467)	(13.79482, 100.3993)

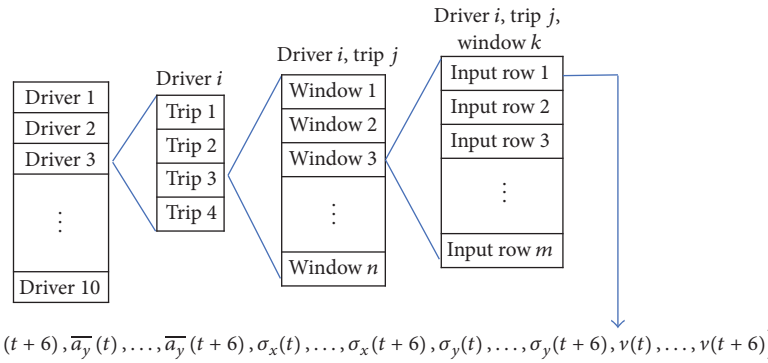

Figure 3: Input setup for the performance evaluation. Each input row is represented as a vector of time-series data, where a denotes the mean of acceleration, σ denotes the standard deviation of acceleration, and v denotes instantaneous speed.

noise from acceleration data, the data has to be compared to that of device on Controlled-area Network (CAN) bus for filtering. Due to certain limitations in this experiment, we are not able to obtain these input signals.

Therefore, for benchmarking, we select three previously proposed algorithms, namely, *Benchmark 1* which is our previously proposed algorithm in [6], *Benchmark 2* which is Gaussian Mixture Model (GMM) [7, 9], and *Benchmark 3* which is based on Support Vector Machine (SVM) [10]. Benchmark 1 is chosen because it is designed specifically to use similar input signals as the algorithm proposed in this paper [6]. This algorithm adapts the principal component analysis (PCA) approach, using only the acceleration data in x and y directions. The identification of driver is then determined by the angle between the principal components of the test data and the principal components of all training data.

Benchmark 2 is chosen as another benchmark algorithm because GMM is commonly used for recognizing drivers by previously proposed algorithms [7, 9]. Furthermore, GMM is also suitable for input signals related to accelerating and braking behaviors as it can capture the frequencies of certain amounts of pressure on the pedals. These frequencies are likely to be distinct amongst individual drivers [9].

Support Vector Machine (SVM) is chosen as Benchmark 3 because it often constitutes the core of recently proposed driver identification algorithms as well as in other classification problems [10, 16]. In SVM-based algorithms, an optimal hyperplane is found to separate drivers using support vector methods. For driver identification, the algorithms proposed in [10] are based on SVM and use driving behavior signals including acceleration and braking as their primary input signals. We note that all three benchmark algorithms are assessed using the same set of data as the proposed algorithm.

5. Results and Discussions

This section focuses on discussing results associated with each operation of the proposed algorithm from the experiments with real-world data. First, Section 5.1 shows how to discover normal patterns of driving behaviors of each driver using the operation described in Section 3.3, which in turn can be used to detect anomalies. Then, in Section 5.2, the operations in data preprocessing (described in Section 3.4) are illustrated. Results on driver identification are discussed in Section 5.3. Finally, comparisons with benchmark algorithm are shown in Section 5.4.

5.1. Assessment on Anomaly Detection. The goal of anomaly detection is not yet to uniquely identify driver, but rather to detect if the driver is actually whom he/she claims to be in each of these 20 trips. Anomaly detection is performed by detecting if the most repeated patterns of each trip deviate from the expected ones. In our context, when a driver i is expected to drive the vehicle on a trip j, an anomaly is detected if the most repeated pattern of trip j is found to deviate from the expected patterns of driver i. The anomaly detection part can be viewed as a screening operation so driver identification part using the neural networks (in Section 3.5) is not activated unnecessarily.

Table 2: Initial results of applying motif discovery algorithm to driving signals. It was found that each of the three driver has his own characteristic as they all have unique most repeated patterns.

Driver	Trip number	Most repeated patterns
(1)	(1)	*acbb cbba*
(1)	(2)	*acbb cbba*
(1)	(3)	bacb bcbb
(1)	(4)	*acbb cbba*
(1)	(5)	bcbb babb
(2)	(1)	babc accb
(2)	(2)	babc accb
(2)	(3)	babc accb
(2)	(4)	baac aacc
(2)	(5)	ccba caab
(3)	(1)	abbc bbca
(3)	(2)	**abca bcaa**
(3)	(3)	**abca bcaa**
(3)	(4)	***bcaa caab***
(3)	(5)	***bcaa caab***

Performance evaluation is conducted on the raw acceleration signals from three different drivers. The signals are separated into training and testing sets consisting of 35% and 65% of the data, respectively. On the training phase, the proposed algorithm assesses the input signals and identifies most repeated patterns as follows. Figure 4 shows how the proposed algorithm assesses input signals in a streaming manner, where the most repeated patterns are highlighted. In this example, the pattern *bccabcaa* highlighted in blue would be considered the primary normal driving behavior patterns for this driver. We note that even though the highlighted six patterns in Figure 4 do not appear similar originally, they constitute the same pattern *bccabcaa* when discretized using SAX as described in Section 3.3.

Table 2 shows initial results from applying the proposed algorithm to find the most repeated patterns. The results look promising as each driver is characterized by his/her own unique driving patterns distinguishable from one another. Cells with different font styles (bolds, italics, and underlines) are most repeated patterns which belong to each of the three drivers which can be used to uniquely signify his/her identities. Therefore, an anomaly detection can be operated by designating these most repeated patterns as normal driving behaviors of these drivers and detect the ones that deviate from these normal patterns.

Once the normal patterns of individual drivers are identified, the anomaly detection part is ready to be deployed under the assumption that any deviation from these normal patterns is identified as an anomaly. The testing set consists of 20 driving trips from three drivers where each trip comprises approximately 17,600 data points, equivalent to 65% of the data set used in this analysis. Table 3 shows that the proposed

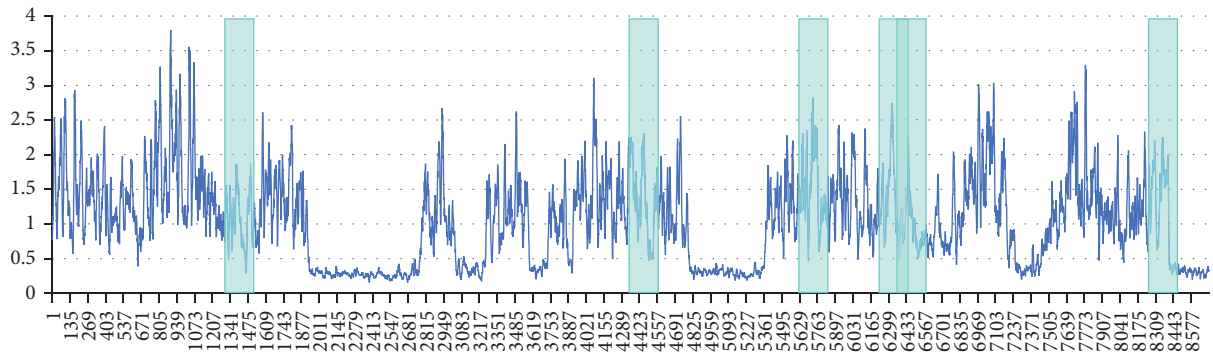

Figure 4: A time-series of acceleration signals. It is found that the most repeated pattern was *bccabcaa* which occurs 6 times.

Table 3: Performance evaluation of the anomaly detection operation using confusion matrix.

Actual	Detected		Total
	Normal	Anomaly	
Normal	7	3	10
Anomaly	0	10	10
Total	7	13	$N = 20$

algorithm performs quite well, where the results can be interpreted as follows.

(i) For overall performance, accuracy is 85% meaning that we expect the anomaly detection operation to be correct most of the time whether it detect anomalies (the driver is not whom we expect to be) or normal cases (the driver is the one we expect to be).

(ii) Focusing on anomaly detection only from the second row of the confusion matrix in Table 3, all 10 anomalies are correctly detected. Therefore, the detection rate (true positive rate or recall) is 100%. In other words, the algorithm can detect all anomalies in the data set that need to be detected.

(iii) Focusing on the activation of neural network operation, we focus on the right column of the confusion matrix in Table 3 where the precision is approximately 77%. This is equivalent to the times when the neural network operation is expected to be necessarily activated.

5.2. Examples of Data Preprocessing Operations. This section discusses outputs associated with the data preprocessing operation described in Section 3.4. We deduce that it is important to show the data preprocessing operation in detail for two main reasons. Firstly, the proposed algorithm is designed to address real-world issues regarding the characteristics of input signals from different sensors (e.g., different sampling rates). Secondly, as the proposed algorithm is developed for streaming operations, it is essential to empirically select suitable window size or the length of each time-series signal to be assessed. Associated examples are described in Sections 5.2.1 and 5.2.2, respectively.

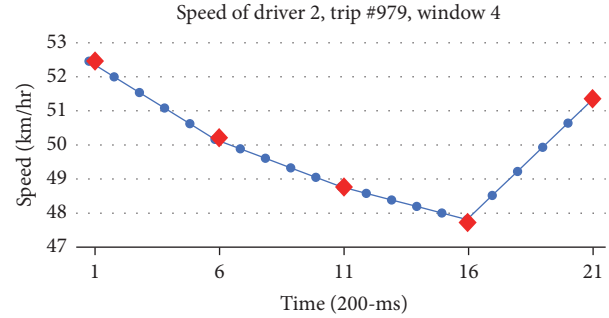

Figure 5: The red diamond dots show the original speed information, each 1 second apart. The line graph shows result of linear interpolation between each original dot.

5.2.1. Speed Interpolation. As shown in Figure 5, in order to match the instantaneous speed to the corresponding mean and standard deviation of accelerations, the speed samples are interpolated to match acceleration data for every time t, every 200 milliseconds apart. Because the time difference between consecutive speeds is only 1 second, the slope in that interval can be approximated with linear relationship. Having corresponding data for every t is necessary for the construction of time-series input. Figure 6 shows acceleration and the interpolated speed data being in sync in sampling frequency.

5.2.2. Selecting Optimal Length of Input Time-Series Signals. The purpose of creating time-series of data is to make recognizable patterns of driving behaviors out of the features extracted from the original driving signal. From observation, some driving behaviors can be classified, more correctly, with short patterns, while others are with longer ones. It is found that lane changing activity usually occurs in 2-3 seconds while turning usually occurs in 5–10 seconds. Hence, time-series of length between 3 and 9 seconds are assessed. Taking machine's efficiency into consideration, the optimal length of time-series in this experiment is the shortest one, while providing acceptable accuracy.

Empirical assessment shows that the optimal length of time-series according to this experiment is 7 seconds, which implies that the uniqueness of each driver's behavior is most

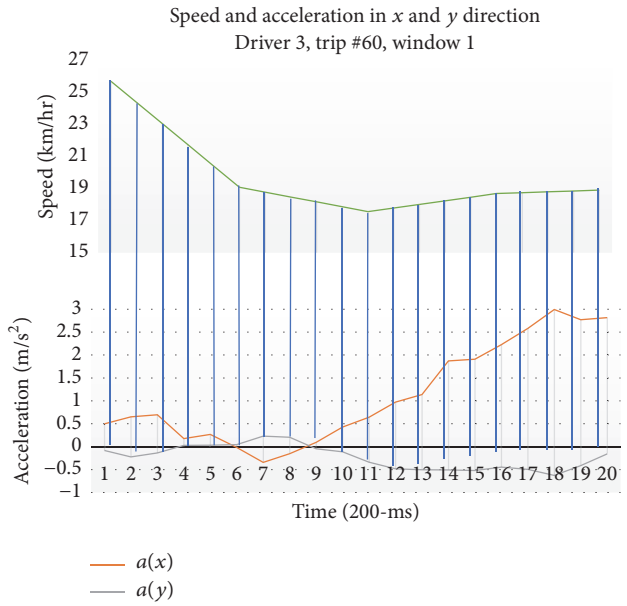

FIGURE 6: The two line graphs below represent the mean of acceleration in x and y direction, computed as described in Section 3.3. The top line graph represents the interpolated speed. The vertical lines show how each speed information corresponds with the acceleration.

clearly represented in a seven-second-long signals. Therefore, each input rows include seven seconds of mean and standard deviation of acceleration data in both x and y direction and seven seconds of speed data, resulting in a total of 35 attributes.

Another factor to consider in constructing time-series is the overlapping period between consecutive series. This parameter does not affect the testing data, since the model makes identification for input row, independently from each other. However, for the training data, large overlapping time, that is, small shift between series, could provide larger number of input rows from the original data than small overlapping time. It also helps to preserve the driving behavior, which occurs during a short span of time, by increasing the number of patterns that represent that behavior. If the overlapping time is small, we may lose some patterns of the identifiable behavior in the gap between two input rows.

5.3. Results and Discussions on Driver Identification. Table 4 shows how driver identification is performed on the real-world data using neural network-based classification and decision making models described in Sections 3.5 and 3.6, respectively. The columns of the table represent correct drivers while the rows are identification results from the proposed algorithm. The numbers in column and row labels denote individual drivers. The numbers in each cell indicate result of identification using the proposed algorithm. For instance, the cell in the column "Actual Driver 3" and row "Identified Driver 9" indicates the number of time-series of input from driver 3, but they are identified by the proposed algorithm to be driver 9. The numbers in the diagonal cells (from upper left to lower right which are marked with bold) indicate correct classifications.

Based on the notations used in Figure 3 (in Section 4.1), the number of windows per trip, n, is set to be 10 and the number of input rows per window, m, is 30. Higher values of both parameters are likely to give better performance, but will also take longer time to construct classification models and run the tests. Therefore, the test case consists of 10 windows of data (1 window per driver), each of which consists of 30 time-series. Hence, 30 identifications are made for each driver, that is, the highest possible number in a cell is 30. The identification for driver i is correct if the number in the cell at column i and row j is the highest number compared to all other cells in column i. With this method, all 30 input rows contribute to the final identification. In the example in Table 4, the proposed algorithm gives correct result for all drivers except driver 2. In the case where the maximum number appears in more than one cell in a column, the highest number cannot be clearly determined and the result will be regarded as incorrect identification.

In each round of experiment, 39 windows of time-series from each driver are used to train the FFNN model and construct the classification model, while 1 window, also from each driver, is used as testing data. Therefore, the proposed algorithm is trained with 390 windows and tested with 10 windows (97.5% training data and 2.5% testing data). The experiment is performed for 20 rounds, each time with a randomly selected window as the testing data. The testing data are equally distributed amongst 4 sample trips; that is, 5 windows from each trip are randomly selected throughout the whole experiment.

Table 5 shows the accuracy of the proposed driver identification algorithm. Each column represents the driver which we are trying to identify while the row represents the identification tests performed on the 10 drivers. As the test windows are randomly selected, the test numbers shown in Table 5 are not in any particular order. The number in each cell indicates the number of correct identifications out of a maximum of 30. Incorrect identifications are marked with bold where the number of correct identification is not the highest number. The same test is used to assess the identification of all 10 drivers. For 100% accuracy, all the cell should have the maximum value of 30. For example, if we take the first row with Test number 1, there are 9 correct identifications and one driver incorrectly identified (Driver 2).

The overall accuracy of the proposed algorithm is 81% ± 7.9%. Further analysis also shows that the proposed algorithm can identify drivers within 13 seconds on average, which shows its potential for real-time applications. Also, it should be noted that the numerical results shown in Table 5 are subjected to randomization of the order in which the data is submitted for training the FFNN model. Therefore, the results do not imply that the proposed algorithm always has more correct identifications on certain drivers (e.g., Drivers 5 and 6) than some drivers (e.g., Drivers 8 and 10).

5.4. Comparisons with Benchmark Algorithms. Table 6 shows that the proposed algorithm achieves much higher overall

Table 4: Example of how the proposed algorithm identifies each driver.

Identified drivers	Actual drivers									
	(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	(10)
(1)	**18**	0	0	0	0	0	7	0	0	6
(2)	0	**8**	0	0	1	0	7	0	0	0
(3)	0	0	**20**	0	0	0	0	0	0	3
(4)	0	17	0	**22**	0	0	0	0	0	1
(5)	5	0	0	3	**27**	0	0	0	0	0
(6)	0	0	0	2	0	**30**	0	0	0	0
(7)	4	0	0	3	0	0	**16**	0	0	8
(8)	0	0	0	0	0	0	0	**28**	0	0
(9)	0	0	10	0	0	0	0	2	**30**	0
(10)	3	5	0	0	2	0	0	0	0	**12**

Table 5: The number in each cell indicates the number of input rows, out of total 30 in the testing window, that result in correct identification. The cells in bold are incorrect identifications, that is, the window where the number of correct identification is not the highest number.

Test number.	Driver										Correct identification
	(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	(10)	
(1)	30	**10**	23	15	23	30	28	20	30	22	9
(3)	17	27	30	19	30	30	**8**	16	30	25	9
(5)	30	30	23	29	25	17	16	30	23	**11**	9
(6)	25	**3**	24	29	27	11	**11**	27	30	19	8
(7)	**2**	21	28	27	30	30	30	**6**	13	30	8
(9)	25	21	30	20	25	22	18	**0**	14	21	9
(11)	30	30	**3**	28	23	**0**	26	**0**	30	30	7
(13)	30	29	30	30	**0**	30	12	**0**	18	30	8
(15)	30	17	26	30	15	15	9	**4**	**4**	30	8
(17)	30	19	**0**	30	12	18	**2**	30	21	30	8
(19)	11	**3**	30	19	24	30	**0**	20	**0**	30	7
(21)	19	**1**	30	**0**	30	30	15	19	30	27	8
(23)	30	13	30	**8**	30	30	**4**	19	30	23	8
(25)	23	**12**	30	24	30	30	15	26	30	15	9
(26)	27	9	30	30	30	30	26	30	30	**2**	9
(27)	21	**0**	30	19	30	30	20	17	30	28	9
(29)	19	**0**	30	19	30	30	24	**0**	30	22	8
(31)	**0**	14	18	21	30	27	**5**	15	22	**0**	7
(33)	19	30	**0**	**9**	30	21	26	25	23	**0**	7
(38)	16	18	**2**	30	20	30	23	**4**	30	**0**	7
				Average accuracy							*81%*

Table 6: Comparisons between the proposed algorithm and the benchmark algorithms in identifying drivers individually.

Drivers	Proposed Algorithm	Benchmark 1 [6]	Benchmark 2 [7, 9]	Benchmark 3 [10, 16]
(1)	90%	50%	25%	80%
(2)	65%	75%	12%	78%
(3)	80%	0%	34%	67%
(4)	85%	50%	19%	76%
(5)	95%	0%	7%	78%
(6)	95%	50%	57%	95%
(7)	70%	50%	19%	67%
(8)	65%	0%	8%	60%
(9)	90%	0%	64%	61%
(10)	75%	75%	23%	64%
Average	*81%*	*35%*	*27%*	*73%*

accuracy than all three benchmark algorithms. First, to give more insightful understanding of the benchmark algorithm, we discuss the results of each benchmark algorithm in comparison with one another. Then, we discuss the results of the proposed algorithm compared to those of benchmark algorithms.

It can be seen that Benchmark 1 can either identify a driver with more than 50% accuracy or fail to identify at all (Drivers 3, 5, 8, and 9 with 0% accuracy). As Benchmark 1 is based on principle component analysis (PCA), the transformation of the input signals into their principal axes domain does not actually utilize the physical characteristics of driving behavior signals themselves. It is found that these principal axes may be very close or even overlapped with increasing number of drivers. This limitation also reduces the capability of recognizing patterns of driving behaviors that span shorter or longer than one second.

For Benchmark 2, to capture the characteristics of the amount of individual drivers pressing the pedals [9], input signals including the statistics of accelerometer signals and instantaneous speeds as shown in Figure 3 are used. As shown in Table 6, Benchmark 2 also performs poorly in identifying drivers compared to the proposed algorithm. However, Benchmark 2 can identify more drivers than Benchmark 1 with certain degrees of accuracy. Based on GMM, Benchmark 2 provides a more flexible approach compared to Benchmark 1 as GMM takes into account all the instants in the input signals and even allows some instants to belong to more than one classes with certain probabilities. However, as accelerometer input signals consist of a number of alternations (e.g., a lot of ups and downs as seen in Figure 4), large number of instants of these input signals are more likely to belong to several drivers. Subsequently, Benchmark 2 cannot completely distinguish individual drivers, which leads to low identification accuracies for most of the drivers.

It can be seen in Table 6 that Benchmark 3 achieves higher overall accuracy than Benchmark 1 and Benchmark 2. Except for Drivers 9 and 10, Benchmark 3 can also generally identify drivers with higher accuracy than Benchmarks 1 and 2. In addition, individual accuracies of Benchmark 3 are more persistent than those of Benchmarks 1 and 2. The main reason behind Benchmark 3's better performance resides in the ability of SVM to find an optimal hyperplane to identify individual drivers.

The proposed algorithm achieves higher overall accuracy because it addresses the limitations discussed above. First, input driving behavior signals are assessed in much finer time scales (e.g., 200 milliseconds) so more fine grained information can be extracted from input signals. Second, the proposed algorithm uses more information than Benchmark 1 because every instant of the input signals is taken into account as described in Section 3.4 and the associated results in Section 5.2. Also, the use of nonlinear mapping capability in neural networks and decision making enables the proposed algorithm to distinguish instants that belong to several drivers more effectively than Benchmark 2. Furthermore, compared to Benchmark 3, the proposed algorithm achieves higher accuracy because it also incorporates variability information by calculating and taking into account the standard deviations of acceleration in the input signals (described in Section 4.1). The inclusion of this variability information provides additional dimensions which further enhance the identification capabilities.

6. Conclusions

In this paper, we have developed an algorithm for driver identification using acceleration and instantaneous speed of vehicles from accelerometers and GPS sensors. First, an anomaly detection is performed to preliminarily assess if the input driving behaviors deviate from expected ones. This triggers the driver identification operation where seven-second-long time-series inputs from these sensors are assessed to identify drivers in a streaming manner. Performance evaluations are conducted using data collected from ten drivers with forty trips in real-world situations. Outputs associated with each operations are shown step-by-step. The proposed algorithm is able to identify the drivers with 13-second-long testing data and with 81% accuracy, regardless of routes and traffic conditions. The proposed algorithm also outperforms three previously proposed algorithms.

Further investigations can be performed in two aspects. Firstly, the amount and degree of noise may vary according to the types of devices. The algorithm should have a more effective preprocessing method to extract the significant feature from the raw data. Secondly, it would be interesting to assess if the proposed algorithm can identify larger numbers of drivers, where more data need to be collected.

Conflicts of Interest

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

References

[1] P. Falcone, F. Borrelli, J. Asgari, H. E. Tseng, and D. Hrovat, "Predictive active steering control for autonomous vehicle systems," *IEEE Transactions on Control Systems Technology*, vol. 15, no. 3, pp. 566–580, 2007.

[2] Q. Li, L. Chen, M. Li, S.-L. Shaw, and A. Nüchter, "A sensor-fusion drivable-region and lane-detection system for autonomous vehicle navigation in challenging road scenarios," *IEEE Transactions on Vehicular Technology*, vol. 63, no. 2, pp. 540–555, 2014.

[3] Premium electric vehicle, https://www.tesla.com/.

[4] T. Zhang, H. Antunes, and S. Aggarwal, "Defending connected vehicles against malware: Challenges and a solution framework," *IEEE Internet of Things Journal*, vol. 1, no. 1, pp. 10–21, 2014.

[5] A. I. Peschansky and A. I. Kovalenko, "A semi-Markov model for an unreliable single-line loss queueing system with different restoration types," *Automation and Remote Control*, vol. 77, no. 12, pp. 2193–2204, 2016.

[6] C. Saiprasert and S. Thajchayapong, "Remote Driver Identification Using Minimal Sensory Data," *IEEE Communications Letters*, vol. 19, no. 10, pp. 1706–1709, 2015.

[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] T. Wakita, K. Ozawa, C. Miyajima et al., "Driver identification using driving behavior signals," *IEICE Transactions on Information and Systems*, vol. E89-D, no. 3, pp. 1188–1194, 2006.

[9] 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*, vol. 20, no. 4, pp. 563–582, 2009.

[10] H. Qian, Y. Ou, X. Wu, X. Meng, and Y. Xu, "Support Vector Machine for Behavior-Based Driver Identification System," *Journal of Robotics*, vol. 2010, pp. 1–11, 2010.

[11] 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, chn, July 2014.

[12] C. Miyajima, Y. Nishiwaki, K. Ozawa, T. Wakita, K. Itou, and K. Takeda, "Cepstral analysis of driving behavioral signals for driver identification," in *Proceedings of the 2006 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2006*, pp. V921–V924, fra, May 2006.

[13] K. Igarashi, C. Miyajima, K. Itou, K. Takeda, F. Itakura, and H. Abut, "Biometric identification using driving behavioral signals," in *Proceedings of the 2004 IEEE International Conference on Multimedia and Expo (ICME)*, pp. 65–68, twn, June 2004.

[14] 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, bra, November 2016.

[15] C. MacAdam, Z. Bareket, P. Fancher, and R. Ervin, "Using neural networks to identify driving style and headway control behavior of drivers," *Vehicle System Dynamics*, vol. 29, pp. 143–160, 1998.

[16] M. Van Ly, S. Martin, and M. M. Trivedi, "Driver classification and driving style recognition using inertial sensors," in *Proceedings of the 2013 IEEE Intelligent Vehicles Symposium, IEEE IV 2013*, pp. 1040–1045, aus, June 2013.

[17] D. Filev, J. Lu, K. Prakah-Asante, and F. Tseng, "Real-time driving behavior identification based on driver-in-the-loop vehicle dynamics and control," in *Proceedings of the 2009 IEEE International Conference on Systems, Man and Cybernetics, SMC 2009*, pp. 2020–2025, usa, October 2009.

[18] D. A. Johnson and M. M. Trivedi, "Driving style recognition using a smartphone as a sensor platform," in *Proceedings of the 14th IEEE International Intelligent Transportation Systems Conference (ITSC '11)*, pp. 1609–1615, Washington, DC, USA, October 2011.

[19] H. Ohta, "Individual differences in driving distance headway," *Vision in vehicles*, vol. 4, pp. 91–100, 1993.

[20] P. Senin, J. Lin, X. Wang et al., "Time series anomaly discovery with grammar-based compression," in *Proceedings of the 18th International Conference on Extending Database Technology, EDBT 2015*, pp. 481–492, bel, March 2015.

[21] P. Patel, E. Keogh, J. Lin, and S. Lonardi, "Mining motifs in massive time series databases," in *Proceedings of the 2nd IEEE International Conference on Data Mining, ICDM '02*, pp. 370–377, jpn, December 2002.

[22] P. Senin, J. Lin, X. Wang et al., "GrammarViz 2.0: A tool for grammar-based pattern discovery in time series," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 8726, no. 3, pp. 468–472, 2014.