Intelligent Sensor Attack Detection and Identification for Automotive Cyber-Physical Systems

Jongho Shin, Youngmi Baek, Yongsoon Eun, Sang Hyuk Son
Department of Information and Communication Engineering
DGIST
Daegu, Republic of Korea
{shinhapp1, ymbaek, yeun, son}@dgist.ac.kr

Abstract— **This paper addresses the problem of detection and identification of the sensor attacks when most sensors are attacked. Sensors can play a key role to improve safety and convenience in automotive Cyber-Physical Systems (CPS). A dramatic increase in connectivity and openness of the automotive CPS brings high security risks. If multiple and heterogeneous sensors equipped for braking and steering provides false sensing information for their controllers under deception attacks, it might cause catastrophic situations during driving. If the existing machine learning approaches are applied for sensor attacks while the majority of sensors is attacked, it cannot guarantee to identify deceptions as cyber-physical attacks. To address this problem, we propose an intelligent sensor attack detection and identification method based on Deep Neural Network (DNN) techniques, called deep learning, without a prior knowledge about the deception attacks modifying sensing data in time. We investigate an autonomous vehicle with Inertial Measurement Unit (IMU) and wheel encoder sensors under conditions of uncertainty and nonlinearity during driving. We firstly identify all possible attacks category on the sensors of it, choose what model to use and then systematically design its architecture on which the performance of deep learning highly depends. We train and then validate the proposed method's performance on real measurement data obtained from an unmanned ground vehicle. Finally, we show analytically the superiority of our method in terms of accuracy, precision, and computation time, including the worst situation where two among three sensors are simultaneously attacked.**

Keywords— Automotive cyber-physical systems, Heterogeneous sensors, Attack detection, Long Short-Term Memory, Gated Recurrent Unit

I. INTRODUCTION

Cyber-Physical Systems (CPS) require control, computation, and communication capability to interact and deal with physical elements [1], [2]. CPS have been already applied to various domains including power stations, health care systems, smart grids, industrial control, and automobiles [3] – [8]. Recently, evolution of the automotive CPS with seamless connectivity and interaction with its external environment has exposed underlying vulnerabilities that enables new threats against its security and safety, such as unknown attacks through network links. In addition, during driving, there may be anomalies caused by uncertainty of dynamic environments, and faults may occur on a system over time. A fault is an abnormal state which might lead to *errors* or *failure* of the system, including permanent, transient, and intermittent faults [9], [10]. Note that except in the case of permanent faults, the other faults can arise by cyber-attacks, which consequently might impact on systems physically. These faults can naturally happen on any sensor or actuator in a given system. Specially, the automotive CPS are equipped with many different sensors such as Global Positioning System (GPS), Inertial Measurement Unit (IMU), ultrasonic sensor, vision, and wheel encoders to enhance safety of passengers in autonomous driving. For the reason mentioned above, the sensors of the automotive CPS also become an attractive target for cyber-physical attacks. For example, if an attacker spoofs GPS of an autonomous vehicle or an unmanned aerial vehicle in cyber space, the physical damage could happen since it cannot provide exact information about the position of the vehicle [11] – [14]. This attack might also appear to be a fault.

In terms of cyber-physical attacks, because the line between faults and attacks is not clear, it is not easy to distinguish the cause of such an abnormal situation in a given system. Both faults and attacks can be a major cause of damage to the system by making it not work normally. In this regard, to distinguish faults from attacks is beyond of the scope of this paper, and we treat a term of a fault the same as an *attack* in this paper. In order to disrupt the normal operation or control of physical systems by exploiting cyber space, all of the deliberate attempts to change, steal or insert data is referred to as *deception attacks*, which are conducted from outside by adversaries.

An important issue in cyber-physical attacks related to heterogeneous sensors of the automotive CPS concerns a constraint on sensor attack detection. If the majority of equipped sensors is attacked simultaneously, these types of attacks are typically hard to detect without a prior knowledge of attacks. In other words, the performance of attack detection methods is limited to detect complicated, concurrent attacks and anomalies, or relies on known attacks. It is important to exploit a large amount of historic data that should accumulate sensor information to detect these attacks as anomalies in the automotive CPS under an uncertain condition. Deep Neural Network (DNN) techniques, called deep learning, can be good solutions to distinguish anomaly from normal situations without prior knowledge on the attack features. This is because those models are capable of discovering complicated relationship functions between input and output by high-level abstractions

978-1-5386-2726-6/17/$31.00 ©2017 IEEE

without expert knowledge even though the majority of sensors has the problems.

Therefore, we focus on a novel way to detect sensor attacks and to identify attacked sensors based on the deep learning technique. We consider an autonomous vehicle with heterogeneous sensors and use real data obtained from a mobile robot under the real uncertain and nonlinear condition for training and validation. The proposed method focuses on only deception attacks that conduct to modify sensing data in time. By these attacks, two wheel encoders, and one IMU equipped on the automotive CPS are mainly targeted during driving in this work. We firstly identify all possible attacks category on the sensors of it, and then choose the deep learning model and design its architecture. This is because the performance of a detection method based on the deep learning technique is dependent upon choosing what model to use related to input data and the selected model's architecture. Finally, we validate the performance of the presented attack detection and identification method, comparing with other techniques.

The rest of this paper is organized as follows. Section II provides background and related work detailed in the literature. In Section III, the overview of the proposed system is presented. Evaluation and analysis of our system are discussed in Section IV. Finally, we draw conclusions in Section V.

II. Background and Related Work

A. Specific Types of Attacks on Cyber-Physical Systems

In recent literature, some researchers have addressed security issues in respect of faults and attacks. Fawzi *et al.* investigated a high-level system design to improve the security of a general control system [15]. They have solved these issues without consideration to requirements of the targeted applications since their system is not designed to target particular applications in specific domains (e.g., an autonomous vehicle or industrial process). Sabaliauskaite *et al.* studied the robust detection system independent of cyber-space of CPS against cyber-attacks [16]. They designed the *watchdog* system which has the smart sensors to monitor and check the physical values. According to pre-defined rules, the system detects attacks. It is applicable only to a particular application with defined critical parameters to be monitored.

In order to detect transient faults caused by the uncertainty of the changeable environment during driving of autonomous vehicles, Jo *et al*. developed the adaptive fault model which investigates the correlation between sensor data [10]. They could not identify the faults when the majority of sensors is faulty simultaneously. In other words, their model allows the attack to be detected only in the condition where one sensor is only attacked or faulty when the autonomous vehicle has three sensors for a controller.

Therefore, sensors in an autonomous vehicle are attacked over half of heterogeneous sensors, there is no help for attack detection without a prior knowledge of attacks.

There are three types of attacks, physical attack, Denial of Service (DoS) attack and deception attack [7], [17]. The physical attack aims to result in a physical breakdown by physically disrupting or damaging the system or the components of it such as cutting wires of a sensor, putting MCU board of plant into the water, and shattering a sensor. The physical attack can be detected easily since the physical space of CPS is only targeted by an adversary.

The DoS attack is a cyber-attack exploiting a communication function of CPS and causes a deficiency of availability of CPS by exhausting the system and network resource such as overloading the network traffic. To monitor and identify the traffic pattern while the networks are flooded with many requests is helpful to detecting the DoS attack. There is an intrusion detection system as a representative countermeasure against the DoS attack.

Finally, the deception attack aims to violate the integrity that represents the reliability of sensors and data in a given system. It modifies data or injects false data so that it provides wrong information of time stamp, sensor values or sender identification for a controller. Eventually, this attack intends to make a wrong decision or failure for controlling the targeted system. Without noticing the deception attack in safety-critical applications, it should cause a dangerous accident. In this regard, the deception attack is one of cyber-physical attacks exploiting both vulnerabilities of cyber and physical spaces of CPS.

Our efforts focus on developing a detection system against this deception attack as a cyber-physical attack. Unfortunately, even although the system is capable of self-monitoring, it is not easy to detect possible deception attacks if an adversary has the deep knowledge of the targeted system. Therefore, we assume that the adversary can get only a little information about a type of sensors equipped in CPS.

B. Techniques of Sensor Attack Detection

Although there are a lot of different methodologies in sensor attack detection technologies, we focus on quantitative methods. Among quantitative methods based on analytical redundancy as an alternative approach to hardware redundancy, a model-based method is identified as one of promising approaches since the 1990s. It focuses on generating and evaluating a residual being a good representative factor to detect attacks on sensors. The residual is generated from the difference between a sensor value and an estimated value. Specially, state estimation for a residual generation has been widely employed as one of a powerful approach to attack detection. Unfortunately, many of these attack detection methods are developed based on a linear system. For example, the Kalman filter uses dynamics and modeling of a system and requires additional equations to predict or estimate sensor values [18], [19]. It is not suitable for a nonlinear condition such as an autonomous vehicle because the Kalman filter for the linear time-invariant system is assumed [20]. In [21], The observer-based method is also not appropriate in a nonlinear system although an observer is added and structure of a control system is more complex. Further, the Kalman filter and observer-based method would be not applicable to the automotive CPS which performs a number of safety-critical functions since the performance of this method is not necessarily guaranteed due to environmental uncertainty, noise, and modeling errors. Recent literature addresses these nonlinearity and uncertainty issues but there still is a discrepancy between the real system and mathematical model [22].

Another of quantitative methods is a data-based method which only requires a number of data to train an input monitoring model of a given system [23]. Data-based methods are mainly adopted to model a complex nonlinear system so as to carry out attack detection. Especially, Neural Network (NN) and Support Vector Machine (SVM) have been applied to detect attacks on CPS among existing machine learning methods because of their high performance related to training speed, memory usage, and classification accuracy.

Although Santos e*t al.* suggested the method using SVM to detect and classify faults of a wind turbine [24], their method is inefficient under a growing number of features. This is because the high dimension has highly effect on the lack of needed data or data distribution of it, related to the empty space. To address this issue, some researchers have tried to reduce the dimension by Principal Component Analysis (PCA) prior to SVM being used [25], [26]. Although linear dimension reduction is executed, the combination of PCA and SVM could not lead to improved performance because important features are also removed [27].

In order to detect attack for large wind farms, Wang *et al.* designed the architecture using multilayer perceptron of NN. [28]. However, NN does not consider sequential information of data and use only present information.

C. Recurrent Neural Network for Attack Detection

RNNs (Recurrent Neural Networks) refer to neural networks to process historic information among deep learning techniques. A standard RNN generates a hidden state h_t with input data x_t at time t as shown in Fig. 1. The RNN is a network that has feedback loops where information of one step should flow to the next connected step. In other words, the previous information connects to the current task. The equation of the RNN is as follows:

$$h_t = act(W \cdot [h_{t-1}, x_t] + b) \qquad (1)$$

where act is an activation function. W and b are a weight matrix and bias term, respectively.

The RNN defined in equation (1) is mainly used for classification or prediction. Backpropagation algorithm is generally used to optimize and learn a model after classification or prediction [29]. Although RNNs can treat a very long sequence of data, it is a very time-consuming process and the required information might be enormous.

There are Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU) as specific types of RNNs. Specially, LSTM is capable of addressing the vanishing gradient problem of the RNN, which is related that the impact of input on the hidden states dynamically vanishes as the recurrent connection of the RNN network is continuously repeated [30]. The memory block of LSTM consists of an internal memory cell and some added gates (the input, output, and forgot gates), which can solve a problem of long dependency [31]. The internal memory cell manages the flow of information and a forget gate chooses to discard information in the internal memory cell. An input gate determines whether to accept new information as input or not and an output gate decides whether or not to use the values [31]. GRU also solves the long dependency and has some gates. GRU is modified version from LSTM by reducing the number of gates and removing the internal memory cell in the memory block [32].

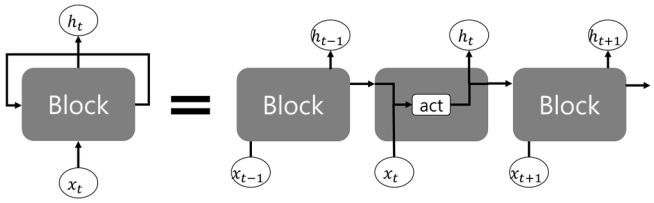

Fig. 1. The Behavior of Recurrent Neural Network.

Largely due to the memory block, LSTM and GRU is widely known to have better performance if there is the meaningful interdependency in the sequence of data. To detect sensor attacks without pre-defined knowledge (rules or patterns) of specific attacks, it is worth to explore the relationship between time-series data obtained from various sensors and time-series data facilities prediction for outliers to be identified. In this regard, we try to find an effective way to make a current decision using the recent information as well as information over very long periods. Thus, we employ each of LSTM and GRU improved from the standard RNN to have high accuracy and low computational complexity in automotive CPS to solve nonlinearity issue. LSTM and GRU consider historic data and find important features without feature extraction and selection. [31], [32].

III. Intelligent Sensor Attack Detection and Identification Strategy

In this section, we propose an intelligent strategy to use the LSTM and GRU to detect coinstantaneous attacks of sensors because they are fast and precise with time series data. Our approach to sensor attack detection is first to collect and monitor continuous data and then generate a detection model representing normality of sensors. Finally, sensor attacks are

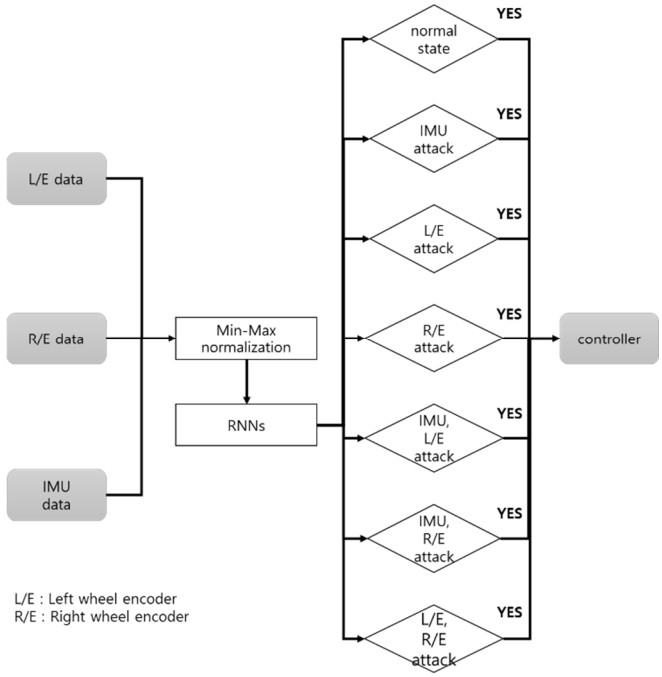

Fig. 2. Flowchart of the proposed detection strategy.

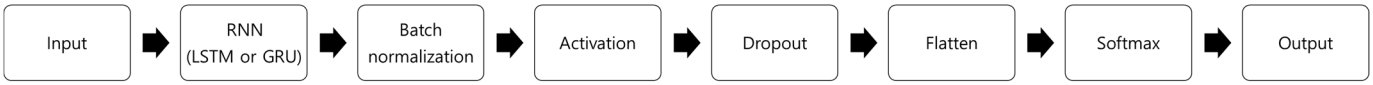

Fig. 3. LSTM/GRU Architecture for Attack Detection and Identification.

naturally detected and identified by using the learned detection model whenever it identifies the deviation from normality. The flow of the proposed detection and identification strategy is summarized in Fig. 2. For our work, the IMU, left and right wheel encoder sensors are employed. Data obtained from the sensors are normalized at a preprocessing phase. After each of LSTM and GRU optimizes weights of a model with training data at a learning phase, the learned model classifies the sensors into 7 states with test data at a prediction phase.

A. Min-Max Normalization

The scale of variables is different from a sensor to a sensor. If the scale of variables is different, a model of a system is learned by focusing on large-scale variables and then small-scale variables are not effectively learned. Finally, the model is not optimized effectively and does not have good performance. Normalization is needed to make the variables distributed evenly in a specific range. Especially, it is possible to adjust the scale of all the variables via Min-Max Normalization [33].

$$x'_t = (S_{max} - S_{min}) \times \frac{x_t - V_{min}}{V_{max} - V_{min}} + S_{min}, \quad x_t \in \mathrm{X} \tag{2}$$

where x'_t is the normalized sensor value at time t, and S_{max} and S_{min} are the maximum and minimum values of the range to be normalized. x_t is the raw sensor value at time t, corresponding to each variable of sensors, V_{max} and V_{min} are the maximum and minimum values of x_t in sensor specification. X is a set of all values to variables measured from all sensors. The number of variables is 8. The variables consist of acceleration x/y/z, angular velocity x/y/z of the IMU sensor and individual velocities of left and right encoders. The raw values to all of the 8 variables obtained from IMU, left and right encoders are adjusted to the same range. max_n and min_n are set to each 1 and -1. Since all the variables of the sensors are time series data, they are calculated sequentially.

B. Attack Detection and Identification

As mentioned above, we generate architectures based on RNN so as to develop a sensor attack detection with high accuracy and low computational complexity in automotive CPS to solve nonlinearity issue. The deeper the architecture of a model is, the greater computational complexity is. Deep network refers to a network to have multiple hidden layers [34]. For this reason, It is important to design a lightweight architecture using the minimum layers.

As shown in Fig. 3, we design an overall architecture using each of LSTM and GRU through empirical experiments, which consists of 3 layers including LSTM/GRU layer, a batch normalization layer and a Softmax layer. First, both LSTM and GRU process time series data after preprocessing. In this architecture, the normalized data are calculated by LSTM as following the equation:

$$\begin{aligned} f_t &= \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \\ i_t &= \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \\ \widetilde{C_t} &= act(W_C \cdot [h_{t-1}, x_t] + b_C) \\ C_t &= f_t * C_{t-1} + i_t * \widetilde{C_t} \\ o_t &= \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \\ h_t &= o_t * act(C_t) \end{aligned} \tag{3}$$

where σ is the sigmoid layer. h_t is the hidden state and x_t is input data at time t. f_t, i_t and o_t are the forget, input and output gate at time t, respectively. C_t and $\widetilde{C_t}$ are each internal memory cell and temporary value to make new internal memory cell at time t. Fig. 4 shows the complex behavior of LSTM.

The memory block of GRU is simpler than that of LSTM. GRU replaces the forget, input and output gate with an update and a reset gate. Also, GRU combines the internal memory cell and hidden state [32]. That is,

$$\begin{aligned} z_t &= \sigma(W_z \cdot [h_{t-1}, x_t] + b_z) \\ r_t &= \sigma(W_r \cdot [h_{t-1}, x_t] + b_r) \\ \widetilde{h_t} &= act(W \cdot [r_t * h_{t-1}, x_t] + b_h) \\ h_t &= (1 - z_t) * h_{t-1} + z_t * \widetilde{h_t} \end{aligned} \tag{4}$$

where z_t and r_r are the update gate and reset gate at time t, respectively. $\widetilde{h_t}$ is a temporary value to make new hidden state at time t. GRU has two sigmoid layers and no cell state as shown in Fig. 5.

Both batch normalization and dropout are added before and after activation function, as regularization methods to reduce

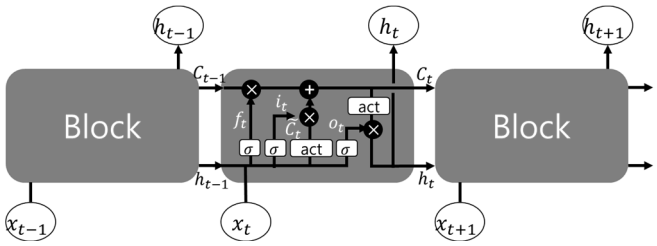

Fig. 4. The Behavior of Long Short-Term Memory.

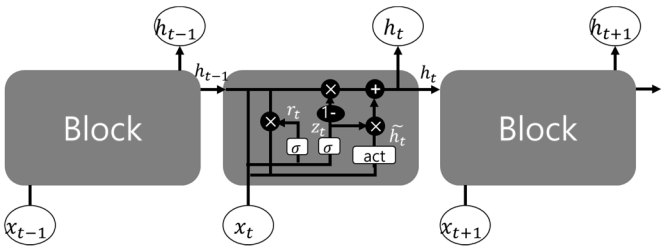

Fig. 5. The Behavior of Gated Recurrent Unit.

Fig. 6. Jackal.

overfitting of the model of the system [35], [36]. We choose ReLu (Rectified Linear unit) as a nonlinear activation function [37]. Flatten converts matrix data into vector data. Softmax classifies the vector data by using cross entropy loss [38]. Adam is employed to optimize the RNN models [39]. Adam finds a minimum point of loss by calculating features and weights.

IV. Experiments

A. Data Acquisition

IMU, left and right wheel encoder sensors are embedded in the mobile robot platform called Jackal and are used to detect attacks as shown in Fig. 6. Data of these sensors are obtained by driving Jackal on the 220-meter straight road at the several constant velocities (0.4, 0.7, 1.0, 1.3 and 1.6 m/s). The data of the different speeds include different properties depending on speeds. Moreover, an environmental uncertainty that people cannot think about is considered. Since we obtain individual 10 sets at each velocity, a total of 50 datasets for training and testing consist of 124550 sensor data. Among them, 99640 data (80%) are used for training data and other 24910 (20%) are used for test data. IMU and wheel encoder sensors have the different sampling frequency. Thus, the raw sensor data should be obtained with the lowest frequency among the sensors. Since the left and right encoder sensors have 20 Hz and IMU has about 73 Hz, the sampling frequency of the IMU sensor is adjusted.

The input data consist of data corresponding to 8 variables (i.e., acceleration x/y/z and angular velocity x/y/z from IMU and left/right velocity from left/right encoders) and we construct 90 samples as one unit in a time sequence as presented in Fig. 7. The output is classified into 7 classes as shown in the Table I. The result of Softmax classifier can indicate one of Class 1, 2 and 3 when only one sensor is attacked. The result of that can be mapped to one of Class 4, 5 and 6 when two sensors are attacked simultaneously. An attack against IMU is a constant bias attack that makes acceleration values of x-axis become constant,

TABLE I. 7 Classes

Label	State
Class 0	Normal operation
Class 1	IMU sensor attack
Class 2	Left encoder sensor attack
Class 3	Right encoder sensor attack
Class 4	IMU and left encoder sensors attack
Class 5	IMU and right encoder sensors attack
Class 6	Left and right encoder sensors attack

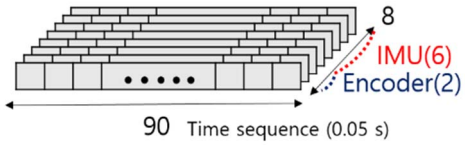

Fig. 7. Input data.

regardless of the sensor values. The signal from a sensor x_t is defined as follows.

$$x_t = a_t \tag{5}$$

where at time t, s_t is a useful signal and a_t is a deception attack of the sensor. The acceleration values follow the normal distribution and the majority of data are distributed near average value of acceleration x because of constant velocity. The values are injected in the range that the sensors can have. These injections make a vehicle system recognize these attacks as natural behaviors although the attacks happen. Table II shows average values of acceleration x at each velocity and the constant values are injected to cause confusion. An additive attack to add a specific value to the output of the encoder sensor is performed. The attacked signal x_t is defined as following the equation.

$$x_t = s_t \pm 0.3 \tag{6}$$

where at time t, s_t is a useful signal. The value of 0.3 m/s is added to each velocity value at 0.4, 0.7, 1.0 and 1.3 m/s except in one case. To the maximum velocity, the value of -0.3 m/s is added. For example, right velocity obtained at 0.4m/s is about 0.4. If right encoder sensor is attacked, the sensor output about 0.7m/s. It confuses the normal value at 0.7m/s.

Each dataset includes data of all the class evenly to learn the model equally and classify every state properly at a prediction phase. In other words, each class has 1/7 ratio of a dataset. Training is run with training data mentioned above via a random search which has parameters of a model set to a random value. The random search is repeated two hundred twenty times. Training time of proposed architecture of LSTM is 939.95 seconds. In the case of GRU, training time is 747.42 seconds.

B. Accuracy and Precision

In terms of accuracy, the performance of the GRU/LSTM model is tested and the results are compared with that of RNN and NN. The detail parameter settings for individual models are summarized in Table III. These values indicated in Table III is

TABLE II. IMU Attack

Velocity (m/s)	Injected constant value (m/s)
0.4	-0.080528566
0.7	-0.150470588
1.0	-0.018102026
1.3	-0.158619575
1.6	-0.068407156

cached only when each result of tested models is the best. The number of hidden neurons of all the networks is set to 90. The dropout is applied to only deep learning methods. All of the learning rate, learning decay and batch size follow the values of Table III. The epochs of all the networks have the same value.

In SVM, RBF (Radial Basis Function) is used as a kernel function and penalty parameter of the error term and gamma is set to 1000 and 1, respectively. The number of components to be kept in PCA is six. Five-fold cross validation is used for optimization in all the models. All the methods are designed and trained in scikit-learn, Keras and Theano library environments.

Fig. 8 presents the results (accuracy) of SVM, combination of PCA and SVM, NN, conventional RNN, GRU, and LSTM. The accuracy of LSTM is the highest among the detection and identification methods. LSTM is well capable of accommodating nonlinearity of automotive CPS because the internal memory cell of LSTM is more complex than that of RNN and GRU. The accuracy of NN is also high but is still lower than RNN, LSTM and GRU since it does not check the correlation between time series. Both SVM and the combination of PCA and SVM have low accuracies. It seems that it is difficult for SVM to process a large amount of data (about 100,000) during learning. Furthermore, those do not consider time series of data, along with NN. As a result, It would be better for RNN, GRU and LSTM to be applied to detect attacks because the total classification rates of them are higher than 95%.

Table IV(a) and (b) show low precision of Class 2 (left encoder sensor attack) and Class 6 (left and right encoder sensors attack), respectively, in comparison with the others of Table IV, NN, RNN, GRU and LSTM have high precisions of Class 2 and Class 6 over 90%. These networks automatically find important features without feature generation. In Table IV(a), (b), and (c), the machine learning methods suffer from classifying data as Class 0 (normal operation). Although the number of data of Class 0 is increased and total data are divided into 103,200 training data and 25,800 test data, the precisions of NN, SVM and the combination PCA and SVM is still around 82%. The Class 0 of NN has the lowest precision among all the classes in Table IV(d). On the other hand, the precisions of Class 0 of RNN, GRU and LSTM exceed 92%. In the case of Class 5, the results of RNN, GRU and LSTM are the same as 99.9%. The precision of Class 6 of RNN is relatively low. On the contrary, the high precisions of Class 6 of GRU and LSTM are presented in Table IV(e) and (f). In terms of simultaneous attacks, the precision of class 6 (simultaneous attack) in RNN are not good, while the precisions of all of class 4, 5 and 6 in LSTM and GRU are higher than 95%. Furthermore, since the detection rate is related to the life of persons, the system could be safer if the accuracy and precision of the model are close to 100%.

TABLE III. PARAMETER SETTINGS

Model	Hidden neuron	Dropout	Learning rate	Learning decay	Batch size	Epochs
GRU	90	0.077	0.0009	1.47e-5	128	50
LSTM	90	0.224	0.0007	6.15e-6	32	50
RNN	90	0.126	0.001	3.49e-6	32	50
NN	90	-	0.01	5.23e-5	16	50

C. Computational Cost

In this paper, the experiments have been implemented with Intel Core i7-7700K CPU @ 4.20 GHz and 16 GB RAM on 64 bit Ubuntu 14.04 PC. The average execution time of LSTM is 0.002104 seconds, which is real time data preprocessing and prediction. In the case of GRU, the average execution time is 0.001645 seconds. This is because the proposed method detects and classifies test data after the parameters are trained in offline. Besides, it is evident that the architectures of GRU and LSTM are light. The execution time of the model is much faster than sampling time (0.05 seconds). The calculation time of GRU is faster than that of LSTM but it seems that LSTM is applicable to attack detection and identification of sensors on automotive CPS. These results showed a new possibility of DNN at a simple (constant and straight) vehicle condition, although training was run in offline.

V. CONCLUSION

We addressed the problem of attack detection and identification when the majority of multiple sensors is attacked in an automotive CPS. LSTM and GRU detected and identified attacks by considering sequential information with real data. We demonstrated that the accuracy of LSTM is the highest among data-based methods (i.e., Neural Network, SVM, simple RNN, GRU and LSTM). The accuracy of LSTM is followed the accuracy of GRU. Especially, LSTM and GRU have the superior ability to detect coinstantaneous attacks. LSTM and GRU showed high performance in identification of Class 2, 3, 4, 5 and 6. Although calculation time of GRU is faster than that of LSTM, it is no matter to detect the attacks of the sensors on a general computer to use a CPU.

If the detection system is applied to real vehicles, it will need to have a model to learn the change of various speeds and require online method. Thus, we plan to consider the online sensor attack detection using LSTM and study a new model or architecture in a real vehicle. Additionally, we have a plan to develop a resilient system in the more complex driving condition which includes turning, stop, slope and road surfaces as our future work.

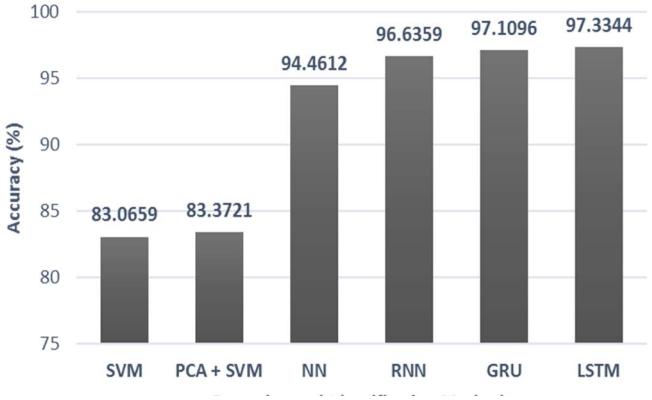

Fig. 8. The accuracies of the detection and identification methods.

TABLE IV. CONFUSION MATRIX

(a) SVM

Actual class	Predicted class							
	0	1	2	3	4	5	6	Total
0	**3056**	251	0	0	0	0	371	3686
1	464	**3041**	0	2	1	0	178	3686
2	0	0	**2720**	494	423	49	0	3686
3	1	0	220	**3097**	31	337	0	3686
4	0	0	72	273	**3339**	2	0	3686
5	0	0	44	431	10	**3198**	3	3686
6	557	140	0	1	0	6	**2980**	3684
Pre (%)	*82.9*	82.5	*73.9*	84.0	90.6	86.8	*80.9*	25800

(b) PCA and SVM

Actual class	Predicted class							
	0	1	2	3	4	5	6	Total
0	**3010**	252	7	1	0	0	416	3686
1	432	**3078**	1	2	1	0	172	3686
2	0	0	**2813**	385	432	56	0	3686
3	1	0	227	**3086**	31	341	0	3686
4	0	0	83	164	**3436**	3	0	3686
5	0	0	35	434	7	**3207**	3	3686
6	664	134	0	1	0	5	**2880**	3684
Pre (%)	*81.7*	83.5	*76.3*	83.7	93.2	87.0	*78.2*	25800

(c) NN

Actual class	Predicted class							
	0	1	2	3	4	5	6	Total
0	**3004**	37	7	2	0	0	636	3686
1	1	**3676**	0	0	0	0	9	3686
2	1	0	**3512**	110	47	15	1	3686
3	1	0	182	**3438**	15	50	0	3686
4	0	0	3	1	**3682**	0	0	3686
5	1	2	1	12	0	**3669**	1	3686
6	238	56	0	0	0	0	**3390**	3684
Pre (%)	*81.5*	**99.7**	95.3	93.3	**99.9**	99.5	92.0	25800

(d) RNN

Actual class	Predicted class							
	0	1	2	3	4	5	6	Total
0	**2625**	171	0	0	0	0	0	2796
1	171	**3453**	2	0	0	0	60	3686
2	0	4	**3595**	25	60	2	0	3686
3	0	0	5	**3661**	5	15	0	3686
4	0	0	57	9	**3619**	0	1	3686
5	0	0	0	1	0	**3684**	1	3686
6	184	48	0	0	3	14	**3435**	3684
Pre (%)	93.9	93.7	97.5	**99.3**	98.2	**99.9**	*93.2*	24910

(e) GRU

Actual class	Predicted class							
	0	1	2	3	4	5	6	Total
0	**2654**	142	0	0	0	0	0	2796
1	177	**3389**	1	0	0	0	119	3686
2	0	3	**3600**	14	69	0	0	3686
3	0	0	15	**3639**	2	29	1	3686
4	0	0	4	1	**3674**	0	7	3686
5	0	0	0	0	2	**3683**	1	3686
6	85	47	0	0	0	1	**3551**	3684
Pre (%)	94.9	91.9	97.7	98.7	**99.7**	**99.9**	96.4	24910

(f) LSTM

Actual class	Predicted class							
	0	1	2	3	4	5	6	Total
0	**2597**	199	0	0	0	0	0	2796
1	158	**3445**	13	0	0	0	70	3686
2	0	3	**3668**	0	15	0	0	3686
3	0	0	14	**3668**	0	4	0	3686
4	0	0	19	14	**3652**	0	1	3686
5	0	0	0	0	1	**3684**	1	3686
6	126	23	0	0	0	3	**3532**	3684
Pre (%)	92.9	93.5	**99.5**	**99.5**	**99.1**	**99.9**	95.9	24910

ACKNOWLEDGMENT

This research was supported in part by the Global Research Laboratory Program (2013K1A1A2A02078326) through NRF, the ICT R&D program of MSIP/IITP (2014-0-00065, Resilient Cyber-Physical Systems Research) and the DGIST Research and Development Program (CPS Global Center) funded by the Ministry of Science, ICT & Future Planning.

REFERENCES

[1] Shi, Jianhua, et al. "A survey of cyber-physical systems." *Wireless Communications and Signal Processing (WCSP), 2011 International Conference on.* IEEE, 2011.

[2] Mitchell, Robert, and Ing-Ray Chen. "A survey of intrusion detection techniques for cyber-physical systems." *ACM Computing Surveys (CSUR)* 46.4 (2014): 55.

[3] Cárdenas, Alvaro A., et al. "Attacks against process control systems: risk assessment, detection, and response." *Proceedings of the 6th ACM symposium on information, computer and communications security*. ACM, 2011.

[4] del Pino, MA Pérez, et al. "Towards self-organizing maps based Computational Intelligent System for denial of Service Attacks Detection." *2010 IEEE 14th International Conference on Intelligent Engineering Systems*. IEEE, 2010.

[5] Sridhar, Siddharth, Adam Hahn, and Manimaran Govindarasu. "Cyber–physical system security for the electric power grid." *Proceedings of the IEEE* 100.1 (2012): 210-224.Sdfsdfs

[6] Mo, Yilin, et al. "Cyber–physical security of a smart grid infrastructure." *Proceedings of the IEEE* 100.1 (2012): 195-209.

[7] Kwon, Cheolhyeon, Weiyi Liu, and Inseok Hwang. "Security analysis for cyber-physical systems against stealthy deception attacks." *2013 American Control Conference*. IEEE, 2013.

[8] Bezzo, Nicola, et al. "Attack resilient state estimation for autonomous robotic systems." *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2014.

[9] Isermann, Rolf. *Fault-diagnosis systems: an introduction from fault detection to fault tolerance*. Springer Science & Business Media, 2006.

[10] Jo, Minsu, et al. "Adaptive Transient Fault Model for Sensor Attack Detection." *Cyber-Physical Systems, Networks, and Applications (CPSNA), 2016 IEEE 4th International Conference on*. IEEE, 2016.

[11] Kerns, Andrew J., et al. "Unmanned aircraft capture and control via GPS spoofing." *Journal of Field Robotics* 31.4 (2014): 617-636.

[12] Shepard, Daniel P., et al. "Evaluation of smart grid and civilian UAV vulnerability to GPS spoofing attacks." *Proceedings of the ION GNSS Meeting*. Vol. 3. 2012.

[13] Koscher, Karl, et al. "Experimental security analysis of a modern automobile." *2010 IEEE Symposium on Security and Privacy*. IEEE, 2010.

[14] Rutkin, Aviva Hope. "spoofers use fake GPS signals to knock a yacht off course." *MIT Technology Review* (2013).

[15] Fawzi, Hamza, Paulo Tabuada, and Suhas Diggavi. "Security for control systems under sensor and actuator attacks." *2012 IEEE 51st IEEE Conference on Decision and Control (CDC)*. IEEE, 2012.

[16] Sabaliauskaite, Giedre, and Aditya P. Mathur. "Intelligent checkers to improve attack detection in cyber physical systems." *Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC), 2013 International Conference on*. IEEE, 2013.

[17] Amin, Saurabh, Alvaro A. Cárdenas, and S. Shankar Sastry. "Safe and secure networked control systems under denial-of-service attacks." *International Workshop on Hybrid Systems: Computation and Control*. Springer Berlin Heidelberg, 2009.

[18] Zhang, Youmin, and Jin Jiang. "Bibliographical review on reconfigurable fault-tolerant control systems." *Annual reviews in control* 32.2 (2008): 229-252.

[19] Gao, Zhiwei, Carlo Cecati, and Steven X. Ding. "A survey of fault diagnosis and fault-tolerant techniques—Part I: Fault diagnosis with model-based and signal-based approaches." *IEEE Transactions on Industrial Electronics* 62.6 (2015): 3757-3767.

[20] Isermann, Rolf, and Peter Balle. "Trends in the application of model-based fault detection and diagnosis of technical processes." *Control engineering practice* 5.5 (1997): 709-719.

[21] Amin, Saurabh, et al. "Cyber security of water SCADA systems—Part II: Attack detection using enhanced hydrodynamic models." *IEEE Transactions on Control Systems Technology* 21.5 (2013): 1679-1693.

[22] Samy, Ihab, Ian Postlethwaite, and Da-Wei Gu. "Detection and accommodation of sensor faults in UAVs-a comparison of NN and EKF based approaches." *Decision and Control (CDC), 2010 49th IEEE Conference on*. IEEE, 2010.

[23] Venkatasubramanian, Venkat, et al. "A review of process fault detection and diagnosis: Part III: Process history based methods." *Computers & chemical engineering* 27.3 (2003): 327-346.

[24] Santos, Pedro, et al. "An SVM-based solution for fault detection in wind turbines." *Sensors* 15.3 (2015): 5627-5648.

[25] Widodo, Achmad, and Bo-Suk Yang. "Support vector machine in machine condition monitoring and fault diagnosis." *Mechanical systems and signal processing* 21.6 (2007): 2560-2574.

[26] Cao, L. J., et al. "A comparison of PCA, KPCA and ICA for dimensionality reduction in support vector machine." *Neurocomputing* 55.1 (2003): 321-336.

[27] Harmouche, Jinane, Claude Delpha, and Demba Diallo. "Incipient fault detection and diagnosis based on Kullback–Leibler divergence using principal component analysis: Part II." *Signal Processing* 109 (2015): 334-344.

[28] Wang, Qing, et al. "A sensor network modeling and fault detection method for large wind farms by using neural networks." *Control & Automation (ICCA), 11th IEEE International Conference on*. IEEE, 2014.

[29] Werbos, Paul J. "Backpropagation through time: what it does and how to do it." *Proceedings of the IEEE* 78.10 (1990): 1550-1560.

[30] Bengio, Yoshua, Patrice Simard, and Paolo Frasconi. "Learning long-term dependencies with gradient descent is difficult." *IEEE transactions on neural networks* 5.2 (1994): 157-166.

[31] Hochreiter, Sepp, and Jürgen Schmidhuber. "Long short-term memory." *Neural computation* 9.8 (1997): 1735-1780.

[32] Cho, Kyunghyun, et al. "Learning phrase representations using RNN encoder-decoder for statistical machine translation." *arXiv preprint arXiv:1406.1078* (2014).

[33] Hussain, Saed, Maizura Mokhtar, and Joe M. Howe. "Sensor failure detection, identification, and accommodation using fully connected cascade neural network." *IEEE Transactions on Industrial Electronics* 62.3 (2015): 1683-1692.

[34] Delalleau, Olivier, and Yoshua Bengio. "Shallow vs. deep sum-product networks." *Advances in Neural Information Processing Systems*. 2011.

[35] Ioffe, Sergey, and Christian Szegedy. "Batch normalization: Accelerating deep network training by reducing internal covariate shift." *International Conference on Machine Learning*. 2015.

[36] Srivastava, Nitish, et al. "Dropout: a simple way to prevent neural networks from overfitting." *Journal of Machine Learning Research* 15.1 (2014): 1929-1958.

[37] Nair, Vinod, and Geoffrey E. Hinton. "Rectified linear units improve restricted boltzmann machines." *Proceedings of the 27th international conference on machine learning (ICML-10)*. 2010.

[38] Wan, Li, et al. "Regularization of neural networks using dropconnect." *Proceedings of the 30th international conference on machine learning (ICML-13)*. 2013.

[39] Kingma, Diederik, and Jimmy Ba. "Adam: A method for stochastic optimization." *arXiv preprint arXiv:1412.6980* (2014)