This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TII.2017.2777460, IEEE Transactions on Industrial Informatics

Levenberg-Marquardt Backpropagation Training of Multilayer Neural Networks for State Estimation of A Safety Critical Cyber-Physical System

Chen Lv, *Member, IEEE*, Yang Xing, Junzhi Zhang, Xiaoxiang Na, Yutong Li, Teng Liu, Dongpu Cao, *Member, IEEE* and Fei-Yue Wang, *Fellow, IEEE*

***Abstract*— As an important safety critical cyber-physical system (CPS), the braking system is essential to the safe operation of the electric vehicle. Accurate estimation of the brake pressure is of great importance for automotive CPS design and control. In this paper, a novel probabilistic estimation method of brake pressure is developed for electrified vehicles based on multilayer Artificial Neural Networks (ANN) with Levenberg-Marquardt Backpropagation (LMBP) training algorithm. Firstly, the high-level architecture of the proposed multilayer ANN for brake pressure estimation is illustrated. Then, the standard backpropagation (BP) algorithm used for training of the feed-forward neural network (FFNN) is introduced. Based on the basic concept of backpropagation, a more efficient training algorithm of LMBP method is proposed. Next, real vehicle testing is carried out on a chassis dynamometer under standard driving cycles. Experimental data of the vehicle and the powertrain systems are collected, and feature vectors for FFNN training collection are selected. Finally, the developed multilayer ANN is trained using the measured vehicle data, and the performance of the brake pressure estimation is evaluated and compared with other available learning methods. Experimental results validate the feasibility and accuracy of the proposed ANN-based method for braking pressure estimation under real deceleration scenarios.**

***Index Terms*— Cyber-Physical System, Safety Critical System, Artificial Neural Networks, LMBP, Brake Pressure Estimation, Electric Vehicle.**

I. Introduction

CYBER physical systems, which are distributed, networked systems that fuse computational processes with the physical world exhibiting a multidisciplinary nature, have recently become a research focus [1-4]. As a typical application of CPS in green transportation, electric vehicles have been widely studied with different topics by researchers and engineers from academia, industry and governmental organizations [5-11]. In an electric vehicle (EV), the cyber world of control and communication, the physical plant of electric powertrain, the human driver, and the driving environment, are tightly coupled and dynamically interacted, determining the overall system's performance jointly [12]. These complex subsystems with multi-disciplinary interactions, strong uncertainties, and hard nonlinearities make the estimation, control and optimization of electric vehicles very difficult [13]. Thus, there are still a number of fundamental issues and critical challenges varying in their importance from convenience to safety of EV remained open [14-17].

Among all those concerns in EV CPS, a key one is safety. Safety critical systems are those ones whose failure or malfunction may result in serious injury or severe damage to people, equipment, or environment [18]. As one of the most important safety critical systems in EV, the correct functioning of braking system is essential to the safe operation of the vehicle [19]. There are a variety of safety standards, control algorithms, and developed devices helping guarantee braking safety for current EVs. However, with increasing degrees of electrification, control authority and autonomy of automotive CPS, safety critical functions of braking system are also required to evolve to keep pace [20].

In the braking system of a passenger car, the braking torque is generated by the hydraulic pressure applied in the brake cylinder. Thus, the accurate measurement of the brake pressure through a pressure sensor is of great importance for various braking control functions and chassis stability logics. However, failures of the brake pressure measurement, which may be caused by software discrepancies or hardware problems, could result in vehicle's critical safety issues. Thus, high-precision estimation of brake pressure become a hot research area in automotive CPS design and control. Moreover, in order to handle the trade-offs between performance and cost, sensor-less observation is required. This makes the study of brake pressure estimation highly motivated.

Based on advanced theories and algorithms from the aspect of control engineering, observation methods of braking pressure for vehicles have been investigated by researchers worldwide. In [21], a recursive least square algorithm for estimation of brake cylinder pressure was proposed based on the pressure response characteristics of anti-lock braking

C. Lv, Y. Xing, and D. Cao are with Advanced Vehicle Engineering Centre, Cranfield University, Bedford, MK43 0AL, UK (email:{c.lyu,y.xing, d.cao}@cranfield.ac.uk).

J. Zhang and Y. Li are with the Department of Automotive Engineering, Tsinghua University, Beijing 100084, P.R. China (email:jzhzhang@tsinghua.edu.cn, wilson420813@gmail.com).

X. Na is with the Department of Engineering, University of Cambridge, CB2 1PZ, United Kingdom (e-mail: xnhn2@eng.cam.ac.uk).

T. Liu and F.-Y. Wang are with the State Key Laboratory of Management and Control for Complex Systems, Institute of Automation, Chinese Academy of Sciences, Beijing 100190, P.R. China (email: tengliu17@gmail.com, feiyue@ieee.org).

(C. Lv and Y. Xing contributed equally to the work. Corresponding authors are J. Zhang and D. Cao)

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TII.2017.2777460, IEEE Transactions on Industrial Informatics

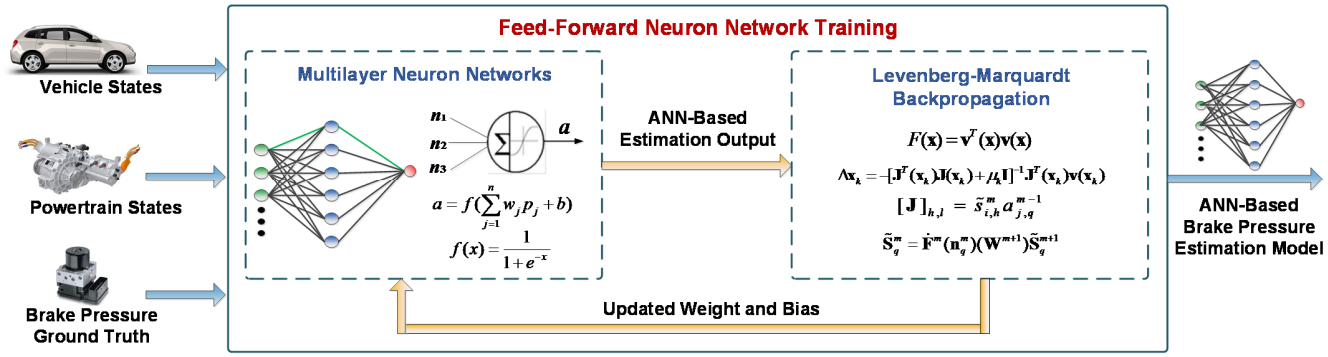


Fig. 1 High-level architecture of the proposed brake pressure estimation algorithm based on multilayer Artificial Neural Networks.

system (ABS). In [22], an extended-kalman-filter-based estimation algorithm was developed considering hydraulic model and tyre dynamics. In [23], an algorithm for online observation of brake pressure was designed through a developed inverse model, and the algorithm was verified in the vehicle's electronic stability program. In [24], the models of brake pressure increase, decrease and hold are proposed, respectively, by using the experimental data. And the models can be used for fast online observation of hydraulic brake pressure. In [25], a brake pressure estimation algorithm was proposed for ABS considering the hydraulic fluid characteristics. In [26], the estimation algorithm was performed by calculating the volume of fluid flowing through the valve. The amount of fluid is a function of the pressure differential across the valve and the actuation time of the valve. Nevertheless, the existing research on brake pressure estimation was mainly investigated from the perspective of control engineering, and an approach with the probabilistic method, such as machine learning, has rarely been seen.

In this paper, an Artificial-Neural-Network-based estimation method is studied for accurately observing the brake pressure of an electric passenger car. The main contribution of this work lies in the following aspects: 1) an ANN-based machine learning framework is proposed to quantitatively estimate the brake pressure of an EV; 2) The proposed approach is implemented with experimental data obtained via vehicle testing, and compared with other methods; 3) The proposed approaches has a great potential to achieve a sensorless design of the braking control system, removing the brake pressure sensor existing in the current products and largely reducing the cost of the system. Moreover, it also provides an additional redundancy for the safety-critical braking functions.

The rest of this paper is organized as follows. Section II describes the high-level architecture of the proposed multilayer ANN for brake pressure estimation. Section III briefly introduces the standard backpropagation algorithm and illustrate the notations and basic concepts demanded in the Levenberg-Marquardt algorithm. Section IV presents details of the application of the LMBP method to training the feed-forward neural networks. In Section V, experiment implementations including feature selection, data collection and preprocessing are presented. Section VI reports the experimental results of the proposed brake pressure estimation algorithm including performance comparison to other approaches. Finally, conclusions are made in Section VII.

II. Multilayer Artificial Neural Networks Architecture

In order to achieve the objective of brake pressure estimation, multilayer artificial neural networks are firstly constructed with the input of vehicle and powertrain states. Details of the high-level system architecture and structure of the component are described in this section.

A. System Architecture

The system architecture with proposed methodology is shown in Fig. 1. The multilayer artificial neural network receives state variables of the vehicle and the electric powertrain system as inputs, and then yields the estimation of the brake pressure through the activation function. The Levenberg-Marquardt Backpropagation algorithm is then operated with the performance function, which is a function of the ANN-based estimation and the ground truth of brake pressure. The weight and bias variables are adjusted according to Levenberg-Marquardt method, and the backpropagation algorithm is used to calculate the Jacobian matrix of the performance function with respect to the weight and bias variables. With updated weights and biases, the ANN further estimates the brake pressure at the next time step. On the basis of the above iterative processes, the ANN-based brake pressure estimation model is well trained. The detailed method and algorithms are introduced in the following subsection.

B. Multilayer Feed-Forward Neural Network

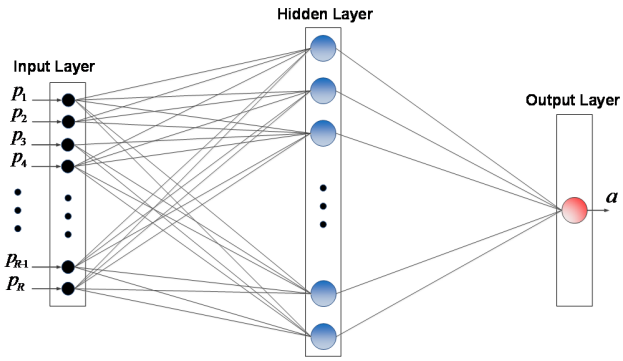


Fig. 2 Structure of the multilayer feed-forward neural network.

In this work, a multilayer feed-forward neural network is chosen to estimate brake pressure. A FFNN is composed of one

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TII.2017.2777460, IEEE Transactions on Industrial Informatics

input layer, one or more hidden layers and one output layer. Since a neural network with one hidden layer has the capability to handle most of the complex functions, in this work the FFNN with one hidden layer is constructed. Fig. 2 shows the structure of a multilayer FFNN with one hidden layer.

The basic element of a FFNN is the neuron, which is a logical-mathematical model that seeks to simulate the behavior and functions of a biological neuron [27]. Fig. 3 shows the schematic structure of a neuron. Typically, a neuron has more than one input. The elements in the input vector $\mathbf{p}=[p_1, p_2, \ldots, p_R]$ are weighted by elements $w_1, w_2, \ldots, w_j$ of the weight matrix $\mathbf{W}$ respectively.

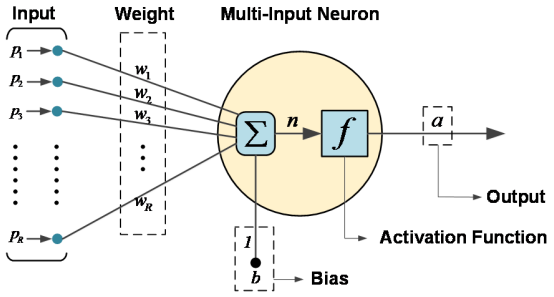


Fig. 3 Structure of the multi-input neuron.

The neuron has a bias b, which is summed with the weighted inputs to form the net input n, which can be expressed by

$$n=\sum_{j=1}^{R} w_j p_j + b = \mathbf{W}\mathbf{p} + b \tag{1}$$

Then the net input n passes through an active function f, which generates the neuron output a.

$$a = f(n) \tag{2}$$

In this study, the log-sigmoid activation function is adopted. It can be given by the following expression:

$$f(x)=\frac{1}{1+e^{-x}} \tag{3}$$

Thus, the multi-input FFNN in Fig. 2 implements the following equation

$$a^2 = f^2\left(\sum_{i=1}^{S} w_{1,i}^2 f^1\left(\sum_{j=1}^{R} w_{i,j}^1 p_j + b_i^1\right) + b^2\right) \tag{4}$$

where a^2 denotes the output of the overall networks. R is the number of inputs, S is the number of neurons in the hidden layer, and p_j indicates the jth input. f^1 and f^2 are the activation functions of the hidden layer and output layer, respectively. b_i^1 represents the bias of the ith neuron in the hidden layer, and b^2 is the bias of the neuron in the output layer. $w_{i,j}^1$ represents the weight connecting the jth input and the ith neuron of the hidden layer, and $w_{1,i}^2$ represents the weight connecting the ith source of the hidden layer to the output layer neuron.

III. Standard Backpropagation Algorithm

In order to train the established FFNN, the backpropagation algorithm can be utilized [28]. Considering a multilayer feedforward neural network, such as the one with three-layer shown in Fig. 2, its operation can be described using the following equation:

$$\mathbf{a}^{m+1} = \mathbf{f}^{m+1}(\mathbf{W}^{m+1}\mathbf{a}^m + \mathbf{b}^{m+1}) \tag{5}$$

where $\mathbf{a}^m$ and $\mathbf{a}^{m+1}$ are the outputs of the m-th and (m+1)-th layers of the networks, respectively. $\mathbf{b}^{m+1}$ is the bias vector of (m+1)-th layers of the networks. $m=0,1,\ldots,M-1$, where M is the number of layers of the neural network. The neurons of the first layer obtain inputs:

$$\mathbf{a}^0 = \mathbf{p} \tag{6}$$

Eq. (6) provides the initial condition for Eq. (5). The outputs of the neurons in the last layer can be seen as the overall networks' outputs:

$$\mathbf{a} = \mathbf{a}^M \tag{7}$$

The task is to train the network with associations between a specified set of input-output pairs $\{(\mathbf{p}_1,\mathbf{t}_1),(\mathbf{p}_2,\mathbf{t}_2),\ldots,(\mathbf{p}_Q,\mathbf{t}_Q)\}$, where $\mathbf{p}_q$ is an input to the network, and $\mathbf{t}_q$ is the corresponding target output. As each input is applied to the network, the network output is compared to the target.

The backpropagation algorithm uses mean square error as the performance index, which is to be minimized by adjusting the network parameters, as shown in Eq. (8).

$$F(\mathbf{x}) = E[\mathbf{e}^T\mathbf{e}] = E[(\mathbf{t}-\mathbf{a})^T(\mathbf{t}-\mathbf{a})] \tag{8}$$

where $\mathbf{x}$ is the vector matrix of network weights and biases. Using the approximate steepest descent rule, the performance index $F(\mathbf{x})$ can be approximated by

$$\hat{F}(\mathbf{x}) = (\mathbf{t}(k)-\mathbf{a}(k))^T(\mathbf{t}(k)-\mathbf{a}(k)) = \mathbf{e}^T(k)\mathbf{e}(k) \tag{9}$$

where the expectation of the squared error in Eq. (8) has been replaced by the squared error at iteration step k.

The steepest descent algorithm for the approximate mean square error is

$$w_{i,j}^m(k+1) = w_{i,j}^m(k) - \alpha\frac{\partial\hat{F}}{\partial w_{i,j}^m} \tag{10}$$

$$b_i^m(k+1) = b_i^m(k) - \alpha\frac{\partial\hat{F}}{\partial b_i^m} \tag{11}$$

where α is the learning rate.

Based on the chain rule, the derivatives in Eq. (10) and Eq. (11) can be calculated as:

$$\frac{\partial\hat{F}}{\partial w_{i,j}^m} = \frac{\partial\hat{F}}{\partial n_i^m}\cdot\frac{\partial n_i^m}{\partial w_{i,j}^m}, \quad \frac{\partial\hat{F}}{\partial b_i^m} = \frac{\partial\hat{F}}{\partial n_i^m}\cdot\frac{\partial n_i^m}{\partial b_i^m} \tag{12}$$

We now define s_i^m as the sensitivity of $\hat{F}$ to changes in the ith element of the net input at layer m.

$$s_i^m \equiv \frac{\partial\hat{F}}{\partial n_i^m} \tag{13}$$

Using the defined sensitivity, then the derivatives in Eq. (12) can be simplified as

$$\frac{\partial\hat{F}}{\partial w_{i,j}^m} = s_i^m a_j^{m-1} \tag{14}$$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TII.2017.2777460, IEEE Transactions on Industrial Informatics

$$\frac{\partial \hat{F}}{\partial b_i^m} = s_i^m \tag{15}$$

Then the approximate steepest descent algorithm can be rewritten in matrix form as:

$$\mathbf{W}^m(k+1) = \mathbf{W}^m(k) - \alpha \mathbf{s}^m (\mathbf{a}^{m-1})^T \tag{16}$$

$$\mathbf{b}^m(k+1) = \mathbf{b}^m(k) - \alpha \mathbf{s}^m \tag{17}$$

where

$$\mathbf{s}^m \equiv \frac{\partial \hat{F}}{\partial \mathbf{n}^m} = [\frac{\partial \hat{F}}{\partial n_1^m}, \frac{\partial \hat{F}}{\partial n_2^m}, ..., \frac{\partial \hat{F}}{\partial n_{S^m}^m}]^T \tag{18}$$

To derive the recurrence relationship for the sensitivities, the following Jacobian matrix is utilized.

$$\frac{\partial \mathbf{n}^{m+1}}{\partial \mathbf{n}^m} \equiv \begin{bmatrix} \frac{\partial n_1^{m+1}}{\partial n_1^m} & \frac{\partial n_1^{m+1}}{\partial n_2^m} & \cdots & \frac{\partial n_1^{m+1}}{\partial n_{S^m}^m} \\ \frac{\partial n_2^{m+1}}{\partial n_1^m} & \frac{\partial n_2^{m+1}}{\partial n_2^m} & \cdots & \frac{\partial n_2^{m+1}}{\partial n_{S^m}^m} \\ \vdots & \vdots & & \vdots \\ \frac{\partial n_{S^{m+1}}^{m+1}}{\partial n_1^m} & \frac{\partial n_{S^{m+1}}^{m+1}}{\partial n_2^m} & \cdots & \frac{\partial n_{S^{m+1}}^{m+1}}{\partial n_{S^m}^m} \end{bmatrix} \tag{19}$$

Consider the i, j element in the matrix:

$$\frac{\partial n_1^{m+1}}{\partial n_1^m} = w_{i,j}^{m+1} \frac{\partial a_j^m}{\partial n_j^m} = w_{i,j}^{m+1} \dot{f}^m(n_j^m) \tag{20}$$

Thus, the Jacobian matrix can be rewritten as

$$\frac{\partial \mathbf{n}^{m+1}}{\partial \mathbf{n}^m} = \mathbf{W}^{m+1} \dot{\mathbf{F}}^m(\mathbf{n}^m) \tag{21}$$

where

$$\dot{\mathbf{F}}^m(\mathbf{n}^m) = \begin{bmatrix} \dot{f}^m(n_1^m) & 0 & \cdots & 0 \\ 0 & \dot{f}^m(n_2^m) & & 0 \\ \vdots & \vdots & & \vdots \\ 0 & 0 & \cdots & \dot{f}^m(n_{S^m}^m) \end{bmatrix} \tag{22}$$

Then the recurrence relation for the sensitivity can be obtained by using the chain rule:

$$\begin{aligned} \mathbf{s}^m &= \frac{\partial \hat{F}}{\partial \mathbf{n}^m} = \left(\frac{\partial \mathbf{n}^{m+1}}{\partial \mathbf{n}^m} \right)^T \frac{\partial \hat{F}}{\partial \mathbf{n}^{m+1}} \\ &= \dot{\mathbf{F}}^m(\mathbf{n}^m)(\mathbf{W}^{m+1})^T \mathbf{s}^{m+1} \end{aligned} \tag{23}$$

This recurrence relation is initialized at the final layer as

$$\begin{aligned} s_i^M &= \frac{\partial \hat{F}}{\partial n_i^M} = \frac{\partial ((\mathbf{t}-\mathbf{a})^T(\mathbf{t}-\mathbf{a}))}{\partial n_i^M} = \frac{\partial \sum_{j=1}^{S^M} (t_j - a_j)^2}{\partial n_i^M} \\ &= -2(t_i - a_i) \frac{\partial a_i}{\partial n_i^M} = -2(t_i - a_i) \dot{f}^m(n_i^m) \end{aligned} \tag{24}$$

Thus the recurrence relation of the sensitivity matrix can be expressed as

$$\mathbf{s}^M = -2\dot{\mathbf{F}}^M(\mathbf{n}^M)(\mathbf{t}-\mathbf{a}) \tag{25}$$

The overall BP learning algorithm is now finalized and can be summarized as the following steps: 1) firstly, propagate the input forward through the network; 2) secondly, propagate the sensitivities backward through the network from the last layer to the first layer; 3) finally, update the weights and biases using the approximate steepest descent rule.

IV. Levenberg-Marquardt Backpropagation

While backpropagation is a steepest descent algorithm, the Levenberg-Marquardt algorithm is derived from Newton's method that was designed for minimizing functions that are sums of squares of nonlinear functions [29, 30].

Newton's method for optimizing a performance index $F(\mathbf{x})$ is

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \mathbf{A}_k^{-1} \mathbf{g}_k \tag{26}$$

$$\mathbf{A}_k \equiv \nabla^2 F(\mathbf{x})|_{\mathbf{x}=\mathbf{x}_k} \tag{27}$$

$$\mathbf{g}_k \equiv \nabla F(\mathbf{x})|_{\mathbf{x}=\mathbf{x}_k} \tag{28}$$

where $\nabla^2 F(\mathbf{x})$ is the Hessian matrix and $\nabla F(\mathbf{x})$ is the gradient.

Assume that $F(\mathbf{x})$ is a sum of squares function:

$$F(\mathbf{x}) = \sum_{i=1}^{N} v_i^2(\mathbf{x}) = \mathbf{v}^T(\mathbf{x})\mathbf{v}(\mathbf{x}) \tag{29}$$

then the gradient and Hessian matrix are

$$\nabla F(\mathbf{x}) = 2\mathbf{J}^T(\mathbf{x})\mathbf{v}(\mathbf{x}) \tag{30}$$

$$\nabla^2 F(\mathbf{x}) = 2\mathbf{J}^T(\mathbf{x})\mathbf{J}(\mathbf{x}) + 2\mathbf{S}(\mathbf{x}) \tag{31}$$

where $\mathbf{J}(\mathbf{x})$ is the Jacobian matrix

$$\mathbf{J}(\mathbf{x}) = \begin{bmatrix} \frac{\partial v_1(\mathbf{x})}{\partial x_1} & \frac{\partial v_1(\mathbf{x})}{\partial x_2} & \cdots & \frac{\partial v_1(\mathbf{x})}{\partial x_n} \\ \frac{\partial v_2(\mathbf{x})}{\partial x_1} & \frac{\partial v_2(\mathbf{x})}{\partial x_2} & \cdots & \frac{\partial v_2(\mathbf{x})}{\partial x_n} \\ \vdots & \vdots & & \vdots \\ \frac{\partial v_N(\mathbf{x})}{\partial x_1} & \frac{\partial v_N(\mathbf{x})}{\partial x_2} & \cdots & \frac{\partial v_N(\mathbf{x})}{\partial x_n} \end{bmatrix} \tag{32}$$

and

$$\mathbf{S}(\mathbf{x}) = \sum_{i=1}^{N} v_i(\mathbf{x}) \nabla^2 v_i(\mathbf{x}) \tag{33}$$

If $\mathbf{S}(\mathbf{x})$ is assumed to be small then the Hessian matrix can be approximated as

$$\nabla^2 F(\mathbf{x}) \cong 2\mathbf{J}^T(\mathbf{x})\mathbf{J}(\mathbf{x}) \tag{34}$$

Substituting Eq. (30) and Eq. (34) into Eq. (26), we achieve the Gauss-Newton method as:

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TII.2017.2777460, IEEE Transactions on Industrial Informatics

$$\Delta \mathbf{x}_k = -[\mathbf{J}^T(\mathbf{x}_k)\mathbf{J}(\mathbf{x}_k)]^{-1}\mathbf{J}^T(\mathbf{x}_k)\mathbf{v}(\mathbf{x}_k) \tag{35}$$

One problem with the Gauss-Newton method is that the matrix may not be invertible. This can be overcome by using the following modification to the approximate Hessian matrix:

$$\mathbf{G} = \mathbf{H} + \mu\mathbf{I} \tag{36}$$

This leads to the Levenberg-Marquardt algorithm [31]:

$$\Delta \mathbf{x}_k = -[\mathbf{J}^T(\mathbf{x}_k)\mathbf{J}(\mathbf{x}_k) + \mu_k\mathbf{I}]^{-1}\mathbf{J}^T(\mathbf{x}_k)\mathbf{v}(\mathbf{x}_k) \tag{37}$$

Using this gradient direction, and recompute the approximated performance index. If a smaller value is yield, then the procedure is continued with the μ_k divided by some factor $\vartheta > 1$. If the value of the performance index is not reduced, then μ_k is multiplied by ϑ for the next iteration step.

The key step in this algorithm is the computation of the Jacobian matrix. The elements of the error vector and the parameter vector in the Jacobian matrix (32) can be expressed as

$$\mathbf{v}^T = [v_1 \;\; v_2 \;\; \dots \;\; v_N] = [e_{1,1} \;\; e_{2,1} \;\; \dots \;\; e_{S^M,1} \;\; e_{1,2} \dots \;\; e_{S^M,Q}] \tag{38}$$

$$\mathbf{x}^T = [x_1 \;\; x_2 \;\; \dots \;\; x_N] = [w^1_{1,1} \;\; w^1_{1,2} \;\; \dots \;\; w^1_{S^1,R} \;\; b^1_1 \dots \;\; b^1_{S^1} \;\; w^2_{1,1} \dots \;\; b^M_{S^M}] \tag{39}$$

where the subscript N satisfies:

$$N = Q \times S^M \tag{40}$$

and the subscript n in the Jacobian matrix satisfies:

$$n = S^1(R+1) + S^2(S^1+1) + \dots + S^M(S^{M-1}+1) \tag{41}$$

Making these substitutions into Eq. (32), then the Jacobian matrix for multilayer network training can be expressed as

$$\mathbf{J}(\mathbf{x}) = \begin{bmatrix} \frac{\partial e_{1,1}}{\partial w^1_{1,1}} & \frac{\partial e_{1,1}}{\partial w^1_{1,2}} & \dots & \frac{\partial e_{1,1}}{\partial w^1_{S^1,R}} & \frac{\partial e_{1,1}}{\partial b^1_1} & \dots \\ \frac{\partial e_{2,1}}{\partial w^1_{1,1}} & \frac{\partial e_{2,1}}{\partial w^1_{1,2}} & \dots & \frac{\partial e_{2,1}}{\partial w^1_{S^1,R}} & \frac{\partial e_{2,1}}{\partial b^1_1} & \dots \\ \vdots & \vdots & & \vdots & \vdots & \\ \frac{\partial e_{S^M,1}}{\partial w^1_{1,1}} & \frac{\partial e_{S^M,1}}{\partial w^1_{1,2}} & \dots & \frac{\partial e_{S^M,1}}{\partial w^1_{S^1,R}} & \frac{\partial e_{S^M,1}}{\partial b^1_1} & \dots \\ \frac{\partial e_{1,2}}{\partial w^1_{1,1}} & \frac{\partial e_{1,2}}{\partial w^1_{1,2}} & \dots & \frac{\partial e_{1,2}}{\partial w^1_{S^1,R}} & \frac{\partial e_{1,2}}{\partial b^1_1} & \dots \\ \vdots & \vdots & & \vdots & \vdots & \end{bmatrix} \tag{42}$$

In standard backpropagation algorithm, the terms in the Jacobian matrix is calculated as

$$\frac{\partial \hat{F}(\mathbf{x})}{\partial x_l} = \frac{\partial \mathbf{e}_q{}^T \mathbf{e}_q}{\partial x_l} \tag{43}$$

For the elements of the Jacobian matrix, the terms can be calculated by

$$[\mathbf{J}]_{h,l} = \frac{\partial v_h}{\partial x_l} = \frac{\partial e_{k,q}}{\partial w_{i,j}} \tag{44}$$

Thus in this modified Levenberg-Marquardt algorithm, we compute the derivatives of the errors, instead of the derivatives of the squared errors as adopted in standard backpropagation.

Using the concept of sensitivities in the standard backpropagation process, here we define a new Marquardt sensitivity as

$$\tilde{s}^m_{i,h} \equiv \frac{\partial v_h}{\partial n^m_{i,q}} = \frac{\partial e_{k,q}}{\partial n^m_{i,q}} \tag{45}$$

where $h = (q-1)S^M + k$.

Using the Marquardt sensitivity with backpropagation recurrence relationship, the elements of the Jacobian can be further calculated by

$$[\mathbf{J}]_{h,l} = \frac{\partial e_{k,q}}{\partial w^m_{i,j}} = \frac{\partial e_{k,q}}{\partial n^m_{i,q}} \frac{\partial n^m_{i,q}}{\partial w^m_{i,j}} = \tilde{s}^m_{i,h} a^{m-1}_{j,q} \tag{46}$$

if x_l is a bias,

$$[\mathbf{J}]_{h,l} = \frac{\partial e_{k,q}}{\partial b^m_i} = \frac{\partial e_{k,q}}{\partial n^m_{i,q}} \frac{\partial n^m_{i,q}}{\partial b^m_i} = \tilde{s}^m_{i,h} \tag{47}$$

The Marquardt sensitivities can be computed using the same recurrence relations as the one used in the standard BP method, with one modification at the final layer. The Marquardt sensitivities at the last layer can be given by

$$\begin{aligned} \tilde{s}^M_{i,h} &= \frac{\partial e_{k,q}}{\partial n^M_{i,q}} = \frac{\partial (t_{k,q} - a^M_{k,q})}{\partial n^M_{i,q}} = -\frac{\partial a^M_{k,q}}{\partial n^M_{i,q}} \\ &= \begin{cases} -\dot{f}^M(n^M_{i,q}) & \text{for } i = k \\ 0 & \text{for } i \neq k \end{cases} \end{aligned} \tag{48}$$

After applying the $\mathbf{p}_q$ to the network and computing the corresponding output $\mathbf{a}^M_q$, the LMBP algorithm can be initialized by

$$\tilde{\mathbf{S}}^M_q = -\dot{\mathbf{F}}^M(\mathbf{n}^M_q) \tag{49}$$

Each column of the matrix should be backpropagated through the network so as to generate one row of the Jacobian matrix. The columns can also be backpropagated together using

$$\tilde{\mathbf{S}}^m_q = \dot{\mathbf{F}}^m(\mathbf{n}^m_q)(\mathbf{W}^{m+1})\tilde{\mathbf{S}}^{m+1}_q \tag{50}$$

The entire Marquardt sensitivity matrices for the overall layers are then obtained by the following augmentation

$$\tilde{\mathbf{S}}^m = \left[\tilde{\mathbf{S}}^m_1 \,|\, \tilde{\mathbf{S}}^m_2 \,|\, \dots \,|\, \tilde{\mathbf{S}}^m_Q\right] \tag{51}$$

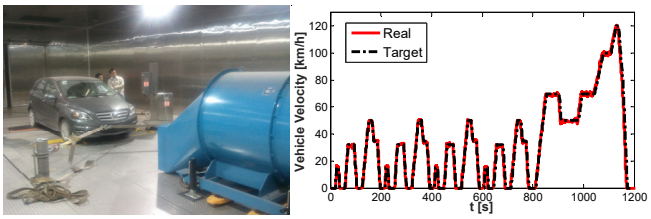
V. Experimental Testing and Data Collection

In order to train the FFNN model with the LMBP algorithm proposed above and validate its effectiveness for brake pressure estimation, real vehicle driving data is needed. Thus, experiments using an electric passenger car are carried out on a chassis dynamometer. The testing vehicle together with the testing scenarios, selected feature vectors, data collection and data pre-processing are described as follows.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TII.2017.2777460, IEEE Transactions on Industrial Informatics

A. Testing Vehicle and Scenario

The experiment is implemented on a chassis dynamometer with an electric passenger car, as shown in Fig. 4(a). The utilized electric vehicle is driven by a permanent-magnet synchronous motor, which is able to work in either driving or regenerating mode. The battery pack is connected to the electric motor via D.C. bus, releasing or absorbing power during driving and regenerative braking processes, respectively. Key parameters of the test vehicle are presented in Table 1.



(a) (b)

Fig. 4. (a) The testing vehicle operating on a chassis dynamometer; (b) Speed profile of the NEDC driving cycle.

TABLE 1
KEY PARAMETERS OF THE ELECTRIC VEHICLE.

Parameter	Value	Unit
Total vehicle mass	1360	kg
Wheel base	2.50	m
Frontal area	2.40	m^2
Nominal radius of tyre	0.295	m
Coefficient of air resistance	0.32	—
Motor peak power	45	kW
Motor maximum torque	144	Nm
Motor maximum speed	9000	rpm
Battery voltage	326	V
Battery capacity	66	Ah

To set up the testing scenario on a chassis dynamometer, standard driving cycles can be utilized. In this study, the New European Drive Cycle (NEDC) which consists of four repeated ECE-15 urban driving cycles and one Extra Urban Driving Cycle (EUDC) is adopted [32]. As Fig. 4(b) shows, the four successive ECE-15 driving cycles in the first section of the NEDC represent urban driving with low operating speed while the second section, i.e. the EUDC driving cycle, indicates a highway driving scenario with the vehicle speed up to 120 km/h.

B. Data Collection and Processing

Vehicle data and powertrain states on CAN bus are collected with a sampling frequency of 100 Hz. Finally, experimental data of 6327 seconds containing six NEDC driving cycles in total are recorded. The vehicle speed and brake pressure of the collected testing data during the four successive ECE-15 driving cycles are presented in Fig. 5.

In order to achieve a better training performance of the FFNN model with machine learning methods, the raw experimental data are smoothed at first using the following equation:

$$d_t = \frac{1}{N}\sum_{n}^{N} d_{tn} \tag{52}$$

where d_t is the value of a signal at time t, d_{tn} is the n-th sampled value of signal d at time step t, and N is the total amount of samples within each second.

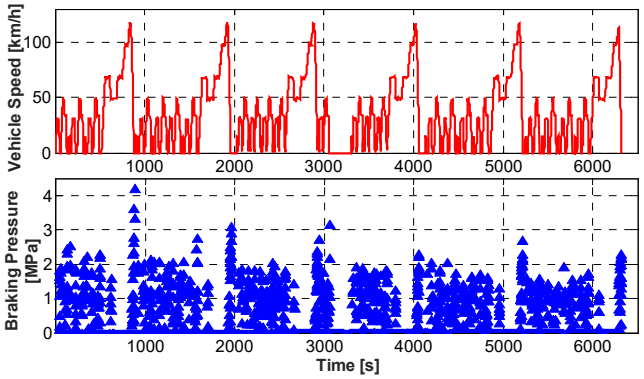


Fig. 5 Collected data of the vehicle speed and corresponding brake pressure.

Then, in order to eliminate the effect brought by different units of signals utilized, the input signals are scaled to be in the range of 0 to 1.

C. Feature Selection and Model Training

In this work, the important vehicle and powertrain state variables are selected for the training of the multilayer ANN model for brake pressure estimation, while the real value of the brake pressure is utilized as a ground truth during the training process. When the electric vehicle is decelerating, the electric motor operates as a generator, recapturing vehicle's kinetic energy. During this period, the values of the motor and battery current change from positive to negative, indicating that the battery is charged by regenerative braking energy. Thus, apart from the vehicle states, the signals of motor speed and torque, battery current and voltage, state of charge (SoC) are also chosen as features, i.e. the input vector of the FFNN. The data of some of the selected feature variables during one driving cycle are shown in Fig. 6.

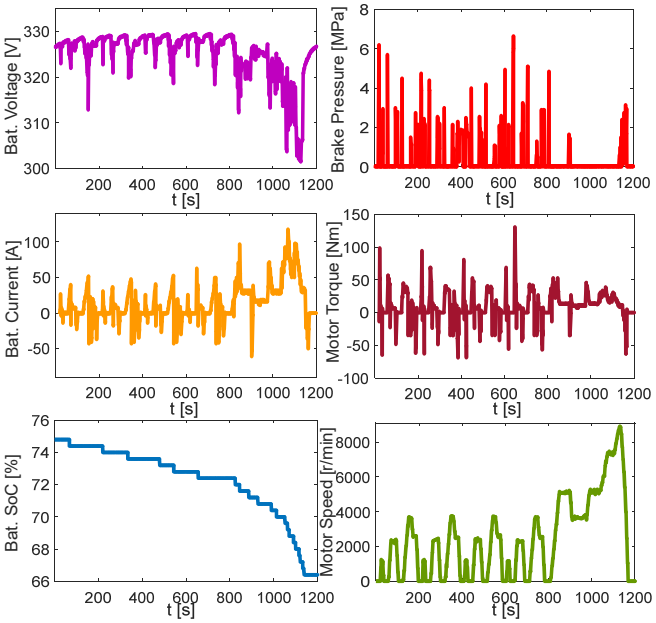


Fig. 6 Experimental data of selected features during one driving cycle.

Besides, statistical information, including the mean value, maximum value, and standard deviation (STD) of some of the vehicle states in the past few seconds are also adopted in this

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TII.2017.2777460, IEEE Transactions on Industrial Informatics

work. The features used for model training are listed below in Table 2.

TABLE 2
SELECTED FEATURES FOR FFNN MODEL TRAINING

No.	Signal	Unit
1	Vehicle Velocity	*km/h*
2	Mean Value of Velocity	*km/h*
3	STD of Velocity	*km/h*
4	Maximum Value of Velocity	*km/h*
5	Vehicle Acceleration	m/s^2
6	Motor Speed	*rad/s*
7	Motor Torque	Nm
8	Battery Current	A
9	Battery Voltage	V
10	Battery SoC	%
11	Gradient of Bat. Voltage	V/s
12	Gradient of Bat. Current	A/s

After determining the feature vectors, the regression model of the FFNN is trained. To modulate and evaluate the model performance, the *K*-fold cross validation approach is adopted [33]. In this method, among the *K* folds divided, (*K*-1) ones are utilized to train the model, and the rest one fold is adopted for testing. Thus, the overall recorded data are divided into two sets, namely the training set and the testing one. The testing set, which is used for model validation, contains 1400 samples chosen randomly from the raw data, and rest of the data are allocated to the training set. The final evaluation of the model performance is carried out based on the *K* test results. In this work, the value of *K* is set as 5. Then, with the 5-fold cross validation, the constructed FFNN is trained using the fast LMBP algorithm developed in Section IV. Some key parameter of FFNN are illustrated below.

TABLE 3
KEY PARAMETERS OF THE FFNN MODEL

Parameter	Value
Maximum number of epochs to train	1000
Performance goal	0
Maximum validation failures	6
Minimum performance gradient	$1e^{-7}$
Initial μ	0.001
μ decrease factor	0.1
μ increase factor	10
Maximum μ	$1e^{10}$
Epochs between displays	25
Maximum time to train in seconds	Infinite

VI. EXPERIMENT RESULTS AND DISCUSSIONS

In this section, results of the estimated ANN-based brake pressure with LMBP learning algorithm are presented and discussed. The algorithms are implemented in a computer with the MATLAB 2017a platform. The processor of the computer is an Intel Core i7-4710MQ CPU which supports 4 cores and 8 threads parallel computing, while the RAM equipped is a 32G one. The time consuming for the FFNN training varies with the number of the hidden neurons selected. In this study, since the range of hidden neurons number is from 10 to 100, thus the training time for FFNN varies from 0.6s to 10s, and the average training time cost for the FFNN with 70 neurons is 3.4s.

A. Results of the ANN-based Braking Pressure Estimation

To quantitatively evaluate the estimation performance, two commonly used indicies, namely the coefficient of determination R^2 and the root-mean-square-error (RMSE), are adopted. The definitions of the R^2 and RMSE are presented as follows. Suppose the reference data is $T = \{t_1 \ldots t_N\}$, and the predicted value is $Y = \{y_1 \ldots y_N\}$. Then R^2 can be calculated as:

$$R^2 = 1 - \frac{E_{res}}{E_{tot}} \tag{53}$$

$$E_{res} = \sum_i^N (t_i - y_i)^2 \tag{54}$$

$$E_{tot} = \sum_i^N (t_i - \bar{T})^2 \tag{55}$$

where E_{res} is the residual sum of square, E_{tot} is the total sum of square, and $\bar{T}$ is the mean value of the reference data.

The RMSE can be obtained by:

$$RMSE = \sqrt{\frac{\sum_i^N (t_i - y_i)^2}{N}} \tag{56}$$

Firstly, the impact of the neuron number on the brake pressure estimation performance is analyzed. Considering the complexity of the problem, the estimation performance is tested under different number of neurons ranging from 10 to 100. According to Fig. 7, as the number of neurons changes, the estimation accuracy of the FFNN varies slightly. The best prediction performance is yield by FFNN with the number of neurons at 70.

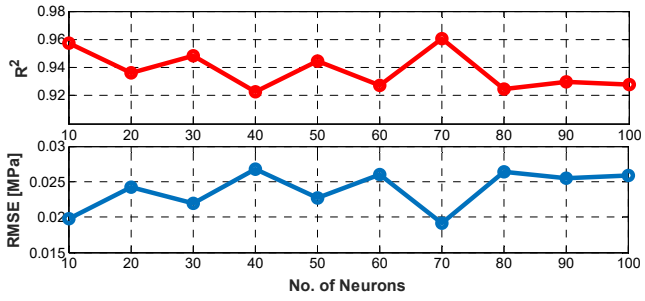


Fig. 7 Estimation performance of FFNN with different number of neurons.

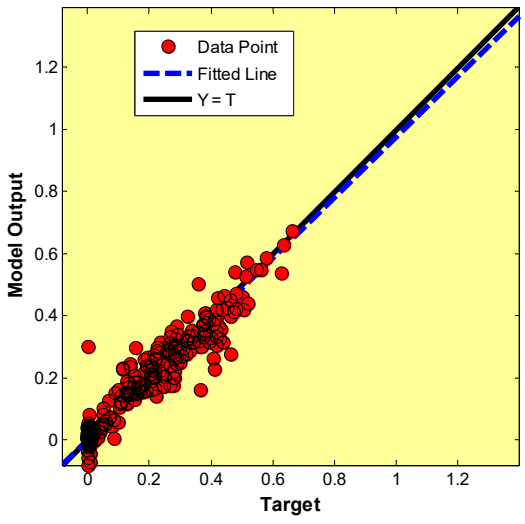


Fig. 8 Regression performance of the FFNN model with 70 neurons.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TII.2017.2777460, IEEE Transactions on Industrial Informatics

Then, the linear regression performance of the trained model is investigated. Based on the linear regression result shown in Fig. 8, the test regression result R is of 0.96677, indicating the FFNN model with 70 neurons can accurately estimate the braking pressure through selected features.

Fig. 9 shows the brake pressure estimation result in time domain. The x-axis presents the 1400 samples of the testing data set, and the y-axis shows the estimation results of the scaled brake pressure. Since the input and output data for model training is scaled to the range of [0, 1], the model testing output is then falling within the range between 0 and 1 accordingly. Based on the results, the FFNN model achieves high-precision regression performance, and the RMSE is around 0.1 MPa, demonstrating the feasibility and effectiveness of the developed method.

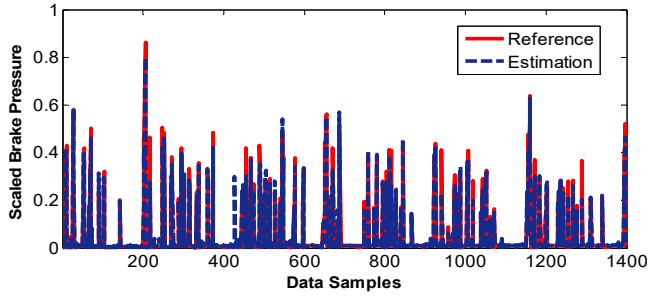


Fig. 9 ANN-based braking pressure estimation results with 1400 testing data points.

B. *Importance Analysis of the Selected Features*

Besides, the utilized feature variables are further investigated through analyzing the importance of predictors [34]. A larger value of the predictor importance indicates that the feature variable has a greater effect on the model output.

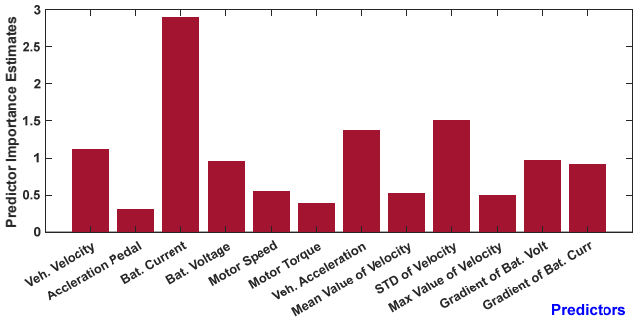


Fig. 10 The predictor importance estimation results.

Fig. 10 illustrates the estimation results of the predictor importance. Based on the results, the most important feature in the model is the battery current, followed by STD of velocity, vehicle velocity, and acceleration. Besides, the battery voltage, the gradients of the battery voltage and current also exert impacts on the model estimation performance.

C. *Comparison of Estimation Results with different Learning Methods*

The developed ANN-based approach is compared with other machine learning methods, including regression decision tree, Quadratic support vector machine (SVM), Gaussian process model, and regression Random Forest. These models are also trained and tested with the 5-fold cross validation method. Apart from R^2 and *RMSE*, other two evaluation parameters, i.e. the training time and the testing time, are also utilized to assess the performance of different models.

Detailed results of the comparison are shown in Table 4. According to the results, the single decision tree algorithm gives much shorter training time and a much faster testing speed in comparison to the other algorithms. In terms of real-time application, the regression decision tree could be a good candidate because of its simplicity and high computation efficiency. However, with respect to the brake pressure estimation accuracy (both R^2 and *RMSE*), the developed ANN algorithm yields the best performance with acceptable training time and testing speed.

TABLE 4
COMPARISON OF BRAKING PRESSURE ESTIMATION PERFORMANCE

Method	R^2	RMSE (MPa)	Training Time (s)	Testing Speed(obs/s)
Decision Tree	0.912	0.133	1.092	~240000
Quadratic SVM	0.867	0.188	141.93	~46000
Gaussian Process model	0.921	0.125	156.89	~8100
Random Forest	0.903	0.104	3.79	~36000
ANN	0.935	0.101	3.42	~82000

VII. CONCLUSIONS

In this paper, a novel probabilistic estimation method of brake pressure is developed for a safety critical automotive CPS based on multilayer ANN with LMBP training algorithm. The high-level architecture of the proposed multilayer ANN for brake pressure estimation is illustrated at first. Then, the standard BP algorithm used for training of FFNN is introduced. Based on the basic concept of BP, a more efficient algorithm of LMBP method is developed for model training. The real vehicle testing is carried out on a chassis dynamometer under NEDC driving cycles. Experimental data of the vehicle and powertrain systems is collected, and feature vectors for FFNN training collection are selected. With the vehicle data obtained, the developed multilayer ANN is trained. The experimental results show that the developed ANN model, which is trained by LMBP, can accurately estimate the brake pressure, and its performance is advantageous over other learning-based methods with respect to estimation accuracy, demonstrating the feasibility and effectiveness of the proposed algorithm.

Further work can be carried out in the following areas: the proposed algorithm will be further refined with onboard road testing; intelligent control algorithms of braking system will be designed based on state estimation.

REFERENCES

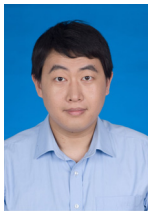
[1] Alam, B., J. Wu, G. Wang, and J. Cao. "Sensing and decision making in cyber-physical systems: the case of structural event monitoring." IEEE Transactions on Industrial Informatics (2016), in press.
[2] Wang, F-Y. "The emergence of intelligent enterprises: From CPS to CPSS." IEEE Intelligent Systems 25, no. 4 (2010): 85-88.
[3] Gong, H., Li, R., An, J., Chen, W. and Li, K. "Scheduling Algorithms of Flat Semi-Dormant Multi-Controllers for a Cyber-Physical System." IEEE Transactions on Industrial Informatics (2017), in press.
[4] Wang, F-Y. "Control 5.0: from Newton to Merton in popper's cyber-social-physical spaces." IEEE/CAA Journal of Automatica Sinica 3, no. 3 (2016): 233-234.

[5] Zhou, Q., Zhang, W., Cash, S., Olatunbosun, O., Xu, H. and Lu, G. "Intelligent sizing of a series hybrid electric power-train system based on Chaos-enhanced accelerated particle swarm optimization." Applied Energy 189 (2017): 588-601.
[6] Lv, C., Wang, H. and Cao, D. "High-Precision Hydraulic Pressure Control Based on Linear Pressure-Drop Modulation in Valve Critical Equilibrium State." IEEE Transactions on Industrial Electronics, 2017.
[7] Kisacikoglu, M., Erden, F. and Erdogan, N. "Distributed Control of PEV Charging Based on Energy Demand Forecast." IEEE Transactions on Industrial Informatics (2017), in press.
[8] Qin, Y., Langari, R., Wang, Z., Xiang, C. and Dong, M., "Road excitation classification for semi-active suspension systemwith deep neural networks." Journal of Intelligent & Fuzzy Systems Preprint (2017): 1-12.
[9] Wang, S., Dong, Z.Y., et al. "Stochastic Collaborative Planning of Electric Vehicle Charging Stations and Power Distribution System." IEEE Transactions on Industrial Informatics (2017), in press.
[10] Lv, C., Liu, Y., Hu, X., Cao, D., et al. Simultaneous Observation of Hybrid States for Cyber-Physical Systems: A Case Study of Electric Vehicle Powertrain, IEEE Transactions on Cybernetics (2017), in press.
[11] Mirzaei, M.J., Kazemi, A. and Homaee, O. "A probabilistic approach to determine optimal capacity and location of electric vehicles parking lots in distribution networks." IEEE Transactions on Industrial Informatics 12, no. 5 (2016): 1963-1972.
[12] Lv, C., Zhang J., and Li Y. "Extended-Kalman-filter-based regenerative and friction blended braking control for electric vehicle equipped with axle motor considering damping and elastic properties of electric powertrain." Vehicle System Dynamics 52, no. 11 (2014): 1372-1388.
[13] Lv, C., Wang, H., Zhao, B., et al. "Cyber-Physical System Based Optimization Framework for Intelligent Powertrain Control." SAE International Journal of Commercial Vehicles 10, no. 2017-01-0426 (2017): 254-264.
[14] Huang, Y., Khajepour, A. et al. "A supervisory energy-saving controller for a novel anti-idling system of service vehicles." IEEE/ASME Transactions on Mechatronics 22, no. 2 (2017): 1037-1046.
[15] Hu, X., Moura, S.J., Murgovski, N., Egardt, B. and Cao, D. "Integrated optimization of battery sizing, charging, and power management in plug-in hybrid electric vehicles." IEEE Transactions on Control Systems Technology 24, no. 3 (2016): 1036-1043.
[16] Cena, G., Bertolotti, I. C., et al. "CAN with eXtensible in-frame Reply: Protocol Definition and Prototype Implementation." IEEE Transactions on Industrial Informatics (2017), in press.
[17] Lv, C., Zhang, J., Li, Y. and Yuan, Y. "Novel control algorithm of braking energy regeneration system for an electric vehicle during safety–critical driving maneuvers." Energy conversion and management 106 (2015): 520-529.
[18] Martins, L., and Gorschek, T.. "Requirements Engineering for Safety-Critical Systems: Overview and Challenges." IEEE Software (2017).
[19] Ames, A.D., Xu, X., Grizzle, J.W. and Tabuada, P. "Control Barrier Function Based Quadratic Programs for Safety Critical Systems." IEEE Transactions on Automatic Control (2016).
[20] Shoukry, Y., Nuzzo, P., Puggelli, A., Sangiovanni-Vincentelli, A.L., et al. "Secure state estimation for cyber physical systems under sensor attacks: a satisfiability modulo theory approach." IEEE Transactions on Automatic Control (2017).
[21] Ding, N., and Zhan, X. "Model-based recursive least square algorithm for estimation of brake pressure and road friction." In Proceedings of the FISITA 2012 World Automotive Congress, pp. 137-145. Springer, Berlin, Heidelberg, 2013.
[22] Jiang, G., Miao, X., Wang, Y., et al. "Real-time estimation of the pressure in the wheel cylinder with a hydraulic control unit in the vehicle braking control system based on the extended Kalman filter." Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering, 2016, in press.
[23] Li, L., Song, J. and Han, Z. Hydraulic model and inverse model for electronic stability program online control system. Chinese Journal of Mechanical Engineering, 44, no. 2 (2008): 139.
[24] Zhang, J., Lv, C., Gou, J. Kong, D. "Cooperative control of regenerative braking and hydraulic braking of an electrified passenger car." Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering 226, no. 10 (2012): 1289-1302.
[25] Yao, J., Zhang, Y. and Wang, J. "Research on algorithm of braking pressure estimating for anti-lock braking system of motorcycle." In Aircraft Utility Systems (AUS), IEEE International Conference on, pp. 586-591. IEEE, 2016.
[26] O'Dea, K. Anti-lock braking performance and hydraulic brake pressure estimation. No. 2005-01-1061. SAE Technical Paper, 2005.
[27] Demuth, H.B., Beale, M.H., De Jess, O. and Hagan, M.T. Neural network design, 2014.
[28] Soualhi, A., Makdessi, M., German, R., et al. "Heath Monitoring of Capacitors and Supercapacitors Using Neo Fuzzy Neural Approach." IEEE Transactions on Industrial Informatics (2017).
[29] Bishop, C. M. Pattern recognition and machine learning. springer, 2006.
[30] Dreyfus, G. Neural networks: methodology and applications. Springer Science & Business Media, 2005.
[31] Hagan, M. T., and Menhaj, M. B. "Training feedforward networks with the Marquardt algorithm." IEEE transactions on Neural Networks 5, no. 6 (1994): 989-993.
[32] Lv, C., Zhang, J., Li, Y. and Yuan, Y. "Mechanism analysis and evaluation methodology of regenerative braking contribution to energy efficiency improvement of electrified vehicles." Energy Conversion and Management 92 (2015): 469-482.
[33] Refaeilzadeh, P., Tang, L. and Liu, H. "Cross-validation." In Encyclopedia of database systems, pp. 532-538. Springer US, 2009.
[34] Budescu, David V. "Dominance analysis: A new approach to the problem of relative importance of predictors in multiple regression." Psychological bulletin 114, no. 3 (1993): 542.
[35] Lv, C., Zhang, L., Li, Y., Yuan, Y. "Mode-switching-based active control of a powertrain system with non-linear backlash and flexibility for an electric vehicle during regenerative deceleration." Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering 229, no. 11 (2015): 1429-1442.
[36] Bagdadi, O. "Assessing safety critical braking events in naturalistic driving studies." Transportation research part F: traffic psychology and behaviour 16 (2013): 117-126.
[37] Lv, C., Zhang, L., Li, Y., et al. "Hardware-in-the-loop simulation of pressure-difference-limiting modulation of the hydraulic brake for regenerative braking control of electric vehicles." Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering 228, no. 6 (2014): 649-662.
[38] Kim, M.H., Lee, S. and Lee, K.C. "Kalman predictive redundancy system for fault tolerance of safety-critical systems." IEEE Transactions on Industrial Informatics 6, no. 1 (2010): 46-53.

Chen Lv is currently a Research Fellow at Advanced Vehicle Engineering Center, Cranfield University, UK. He was with the Department of Automotive Engineering, Tsinghua University, China. He received the Ph.D. degree at Department of Automotive Engineering, Tsinghua University, China in 2016. From 2014 to 2015, he was a joint PhD researcher at EECS Dept., University of California, Berkeley. His research focuses on cyber-physical system, hybrid system, advanced vehicle control and intelligence, where he has contributed over 40 papers and obtained 11 granted China patents. Dr. Lv serves as a Guest Editor for IEEE/ASME Transactions on Mechatronics, IEEE Transactions on Industrial Informatics and International Journal of Powertrains, and an Associate Editor for International Journal of Electric and Hybrid Vehicles, International Journal of Vehicle Systems Modelling and Testing, International Journal of Science and Engineering for Smart Vehicles, and Journal of Advances in Vehicle Engineering. He received the Highly Commended Paper Award of IMechE UK in 2012, the National Fellowship for Doctoral Student in 2013, the NSK Outstanding Mechanical Engineering Paper Award in 2014, the Tsinghua University Graduate Student Academic Rising Star Nomination Award in 2015, the China SAE Outstanding Paper Award in 2015, the 1st Class Award of China Automotive Industry Scientific and Technological Invention in 2015, and the Tsinghua University Outstanding Doctoral Thesis Award in 2016.

Yang Xing received his B.S. in Automatic Control from Qingdao University of Science and Technology, Shandong, China, in 2012. He then received his Msc. with distinction in Control Systems from the department of Automatic Control and System Engineering, The University of Sheffield, UK, in 2014. Now he is a Ph. D. candidate for Transport Systems, Cranfield University, UK. His research interests include driver behaviour modelling, driver-vehicle interaction, and advance driver assistance systems. His work focuses on the understanding of driver behaviours and identification of driver intentions using machine-learning methods for intelligent and automated vehicles.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TII.2017.2777460, IEEE Transactions on Industrial Informatics



Xiaoxiang Na received the B.Sc. and M.Sc. degrees in automotive engineering from the College of Automotive Engineering, Jilin University, Changchun, China, in 2007 and 2009, respectively, and the Ph.D. degree in driver-vehicle dynamics from the Department of Engineering, University of Cambridge (CUED), Cambridge, U.K., in 2014. He is currently a Research Associate with CUED. His main research interests include driver–vehicle dynamics and vehicle in-service monitoring.

Yutong Li is a Ph.D. candidate at Tsinghua Univ., China. He received the bachelor's degree in internal combustion engine engineering at the Jilin University in 2008. He is currently working in the Vehicle Dynamics & Control Lab at Univ. of California, Berkeley, under supervision of Prof. J. Karl Hedrick. His research interest covers nonlinear control, receding horizon control and its application to vehicle engineering. He received Tsinghua first class scholarship for doctoral student in 2013 and 2015, and the China SAE Outstanding Paper Award in 2015.

Teng Liu received the B.S. degree in mathematics from Beijing Institute of Technology, Beijing, China, 2011. He received his Ph.D. degree in the vehicle engineering from Beijing Institute of Technology (BIT), Beijing, in 2017. His Ph.D. dissertation, under the supervision of Dr. Fengchun Sun, was entitled "Reinforcement learning-based energy management for hybrid electric vehicles." Dr. Liu is now a postdoctoral research fellow at Institute of Automation, Chinese Academy of Sciences, China.

Dr. Liu has more than 6 years' research and working experience in new energy vehicle and control. His current research focuses on parallel driving, parallel reinforcement learning, automated driving, and energy management of electrified vehicles. He has published over 20 papers in these areas.

Dongpu Cao received the Ph.D. degree from Concordia University, Canada, in 2008. He is currently a Senior Lecturer at Advanced Vehicle Engineering Center, Cranfield University, UK. His research focuses on vehicle dynamics and control, automated driving and parallel driving, where he has contributed more than 100 publications and 1 US patent. He received the ASME AVTT'2010 Best Paper Award and 2012 SAE Arch T. Colwell Merit Award. Dr. Cao serves as an Associate Editor for IEEE TRANSACTIONS ON INTELLIGENT TRANSPORTATION SYSTEMS, IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY, IEEE TRANSACTIONS ON INDUSTRIAL ELECTRONICS, IEEE/ASME TRANSACTIONS ON MECHATRONICS and ASME JOURNAL OF DYNAMIC SYSTEMS, MEASUREMENT, AND CONTROL. He has been a Guest Editor for VEHICLE SYSTEM DYNAMICS, and IEEE TRANSACTIONS ON HUMAN-MACHINE SYSTEMS. He serves on the SAE International Vehicle Dynamics Standards Committee and a few ASME, SAE, IEEE technical committees.

Fei-Yue Wang (S'87-M'89-SM'94-F'03) received his Ph.D. in Computer and Systems Engineering from Rensselaer Polytechnic Institute, Troy, New York in 1990. He joined the University of Arizona in 1990 and became a Professor and Director of the Robotics and Automation Lab (RAL) and Program in Advanced Research for Complex Systems (PARCS). In 1999, he founded the Intelligent Control and Systems Engineering Center at the Institute of Automation, Chinese Academy of Sciences (CAS), Beijing, China, under the support of the Outstanding Overseas Chinese Talents Program from the State Planning Council and "100Talent Program" from CAS, and in 2002, was appointed as the Director of the Key Lab of Complex Systems and Intelligence Science, CAS. From 2006 to 2010, he was Vice President for Research, Education, and Academic Exchanges at the Institute of Automation, CAS. In 2011, he became the State Specially Appointed Expert and the Director of the State Key Laboratory of Management and Control for Complex Systems.

Dr. Wang's current research focuses on methods and applications for parallel systems, social computing, and knowledge automation. He was the Founding Editor-in-Chief of the International Journal of Intelligent Control and Systems (1995-2000), Founding EiC of IEEE ITS Magazine (2006-2007), EiC of IEEE Intelligent Systems (2009-2012), and EiC of IEEE Transactions on ITS (2009-2016). Currently he is EiC of IEEE Transactions on Computational Social Systems, Founding EiC of IEEE/CAA Journal of Automatica Sinica, and Chinese Journal of Command and Control. Since 1997, he has served as General or Program Chair of more than 20 IEEE, INFORMS, ACM, and ASME conferences. He was the President of IEEE ITS Society (2005-2007), Chinese Association for Science and Technology (CAST, USA) in 2005, the American Zhu Kezhen Education Foundation (2007-2008), and the Vice President of the ACM China Council (2010-2011). Since 2008, he has been the Vice President and Secretary General of Chinese Association of Automation. Dr. Wang has been elected as Fellow of IEEE, INCOSE, IFAC, ASME, and AAAS. In 2007, he received the National Prize in Natural Sciences of China and was awarded the Outstanding Scientist by ACM for his research contributions in intelligent control and social computing. He received IEEE ITS Outstanding Application and Research Awards in 2009, 2011 and 2015, and IEEE SMC Norbert Wiener Award in 2014.