A learning-based short-term wind speed forecasting approach through spiking neural networks

Jing Hu*

Shanghai Minghua Electric Power Science & Technology Co., Ltd.,
Shanghai 200090, China
Email: hjwuhee@163.com
*Corresponding author

Lili Xie, Xinyi Chen, Weidong Liu and Xingpeng Zhang

Shanghai Electric Power New Energy Development Co., Ltd.,
Shanghai 200010, China
Email: xiell@xny.shanghaipower.com
Email: chenxy@xny.shanghaipower.com
Email: liuwd@xny.shanghaipower.com
Email: zhangxp@xny.shanghaipower.com

Dianwei Qian

School of Control and Computer Engineering,
North China Electric Power University,
Changping District, Beijing 102206, China
Email: dianwei.qian@ncepu.edu.cn

Abstract: In real-world applications, the index of wind speed is concerned to many fields. This index plays an extremely important role in wind power systems. Unfortunately, it is hard enough to accurately measure the wind speed. Its forecasting undoubtedly becomes harder and more challenging. This paper focuses on the problem of short-term wind speed forecasting. It is too complex to model the wind speed by mathematical formulas. The technique of neural networks is a learning-based approach. By this technique, the method of spiking neural networks is one of the most successful methods to fulfil the modelling of complex dynamics and the exploitation of learning ability. This paper investigates a spiking-neural-network-based structure, designs a hybrid learning algorithm that combines the adaptive learning rate and the momentum term and implements them for the short-term wind speed forecasting. Experiments and comparisons are illustrated to show the effectiveness and feasibility of this learning-based forecasting approach.

Keywords: wind speed; forecasting; short-term; spiking neural networks; SNNs; learning algorithm; modelling; power generation.

Reference to this paper should be made as follows: Hu, J., Xie, L., Chen, X., Liu, W., Zhang, X. and Qian, D. (2020) ‘A learning-based short-term wind speed forecasting approach through spiking neural networks’, *Int. J. Advanced Mechatronic Systems*, Vol. 8, No. 1, pp.26–35.

Biographical notes: Jing Hu is currently a Senior Engineer in the Shanghai Minghua Electric Power Science & Technology Co. Ltd, Shanghai, China. She received her BE degree from Wuhan University, Wuhan, China, in 2001. She also received her Master’s degree from the Wuhan University, Wuhan, China, in 2004. Her research draws on process control of clean energy generation systems.

Lili Xie is currently a Deputy General Manager in the Shanghai Electric Power New Energy Development Co. Ltd. She is also the Secretary of CPC Committee in this corporation. She graduated from the Shanghai University of Electric Power in 1998. She received her Master of Business Administration from the East China University of Science and Technology in 2009. Her research covers operation and management of new energy power generation systems.

Xinyi Chen is the Director in the Production and Technology Department, Shanghai Electric Power New Energy Development Co. Ltd. She received her Bachelor’s in Energy Engineering and Automation from the North China Electric Power University in 2015. She focuses on photovoltaic power generation system and power station design.

Weidong Liu is the Regional Manager (Northwest China and Zhejiang Province), Shanghai Electric Power New Energy Development Co. Ltd. He concentrates on operation and management of renewable power systems.

Xingpeng Zhang is the Director Assistant in the Production & Technology Department, Shanghai Electric Power New Energy Development Co. Ltd. He is also an Engineer in the department. He received his Bachelor of Engineering from the School of Mechatronic Engineering, China University of Mining and Technology, in 2002. His research field covers forecasting of wind energy and management of renewable power systems.

Dianwei Qian is currently an Associate Professor in the School of Control and Computer Engineering, The North China Electric Power University, Beijing, China. He received his BE degree from the Hohai University (HHU), Nanjing, China, in 2003, and PhD degree from the Institute of Automation, Chinese Academy of Sciences (CAS), Beijing, China, in 2008. He also received his Master's degree from the Northeastern University (NEU), Shenyang, China, in 2005. His research draws on the diverse control methods to aid in the analysis and design of complex dynamical systems.

1 Introduction

On the surface of the Earth, wind is made up of the bulk movement of air. Such movement contains the kinetic energy called wind energy. Since power is energy per unit time, wind power in an open air stream is proportional to the third power of the wind speed, that is, the double wind speed will lead to the eightfold increase of the wind power (Zhu and Zhang, 2019). This fact indicates that the index of wind speed becomes a pivotal role in the field of wind generation. Wind energy is sustainable and renewable, and has a much smaller impact on the environment compared to burning fossil fuels. With the development of the wind energy technologies, such an index has been paid more and more attention in both the academic and engineering communities (Li et al., 2017).

However, wind speed is not only a physical quantity, but also it is atmospheric. Inherently, wind is caused by air moving from high to low pressure on account of the changes in temperature and its direction is usually parallel to isobars due to the Earth's rotation (Alharthi et al., 2018). Wind speed is affected by a number of factors and situations, including but not limited to pressure gradient, local weather conditions and terrain conditions (Zheng et al., 2018). Concerning wind generation, it has a direct function of wind speed that is not easily dispatchable on demand. Consequently, the problem of the wind speed forecasting rises up (Astolfi et al., 2018). This problem is interesting and challenging because wind is fluctuant, intermittent and stochastic.

Dependent on different time scales, wind speed forecasting can be roughly categorised into the short-term forecasting, the medium-term forecasting and the long-term forecasting (Xu et al., 2018). The medium-term and long-term scales usually refer to forecasting of the monthly or annual available wind energy resource and they are often employed to power-system management, wind farms planning and energy trading. Mostly, some typical and conventional operations of wind generation include wind-turbine active control, dispatch planning and maintenance scheduling and regulation and so on (Qian et al., 2013). Their temporal resolutions of wind speed forecasting ranges between several minutes and several hours. Such type of wind speed forecasting calls for the short-term forecasting (Al-Falahi et al., 2017).

Since the short-term forecasting of wind speed is concerned to the economy efficiency and reliability of wind generation very much, this forecasting problem has been paid more and more attention. Due to the complexity of wind speed, it is rather hard to forecast it by the first principle methods. From the aspect of statistics, some forecasting methods have been reported in recent years. Erdem and Shi (2011) developed an autoregressive moving average (ARMA) model, which contained the wind speed and direction information. Another autoregressive integrated moving average (ARIMA) model (Aasim and Singh, 2019) was investigated by Aasim and Singh. Hu and Wang (2015) applied the Gaussian process regression to forecast the short-term wind speed. According to the multivariate probabilistic method and multiple linear regressions, Casella (2019) investigated the forecasting problem of short-term wind speed. The statistical forecasting methods are based on historical values of wind speed, as well as historical and forecast values of meteorological variables (Qian et al., 2015). Unfortunately, there will always be some inherent and irreducible uncertainties in every prediction so that the forecasting accuracy of wind speed can hardly be refined (Qian et al., 2016).

On the other hand, with the advent and emerging of artificial intelligence, some intelligent forecasting models of wind speed have recently been addressed, where neural networks and support vector machines are two main tools to deal with the forecasting problem of wind speed (Marugan et al., 2018). Ma et al. (2019) utilised a negative correlation learning neural network-based hybrid model to describe and forecast wind speed features. Ding et al. (2019) proposed the bidirectional gated recurrent unit neural networks-based error correction model to correct error of numerical weather prediction and applied the neural networks to forecast short-term wind power. The support vector machine-based model used is employed to calculate the regression relationships between the historical data and forecasting data of wind speed (Wang et al., 2018). A hybrid wind speed forecasting model was developed by Wu and Lin (2019) where the least squares support vector machine was established to forecast the wind speed series and the machine parameters were optimised by a bat algorithm. Some other reports about this filed can also be found in (Wang et al., 2016; Liu et al., 2014; Huang and Kuo, 2018; Yu et al., 2018).

This paper focuses on the neural-network-based forecasting of wind speed. The structures, algorithms and connections of neural networks are various. They are deeply dependent on the biological background of neural works. Among the diversity of neural networks, spiking neural networks (SNNs) are advocated because the kind of networks is more closely mimic natural neural networks, that is, the structure of SNNs is designed to describe realistic brain-like information processing (Bohte et al., 2002; Shrestha and Song, 2017; Saunders et al., 2019). The SNNs-based modelling becomes popular and well-reputed. The SNNs not only can have the ability of capturing informational dynamics observed among real biological neurons, but also they can represent and integrate several information dimensions into a single model. So far, it is reported that the SNNs-based modelling has been successfully applied to electrical load forecasting (Kulkarni et al., 2013), carbon price forecasting (Sun et al., 2016) and air pollution prediction (Maciag et al., 2019). The applications indicate that the SNNs could be a potential solution for the forecasting problem of short-term wind speed. To some extent, the coming of SNNs in this field is also propelled and steered by the need of forecasting the short-term wind speed.

In order to explore the SNNs for the forecasting problem of short-term win speed, this paper proposes a learning-based approach. According to the spiking-neural-network-based structure, the proposed approach combines the adaptive learning rate and the momentum term. Such combination can contribute to refine the forecasting accuracy and expedite the learning speed. Compared to some traditional neural networks such as the back-propagation (BP) neural networks and the radial-basis-function (RBF) neural networks, the numerical results demonstrate the effectiveness and feasibility of the proposed learning-based approach.

The remainder is presented as follows. Section 2 presents the spiking-neural-network-based structure and designs the hybrid learning algorithm. Section 3 introduces the data and data-preparation. In Section 4, the proposed approach is carried out and some numerical results and comparisons are illustrated. Finally, conclusions are drawn in Section 5. The highlights of this paper are listed as follows:

- a spiking-neural-network-based forecasting structure is addressed
- the hybrid learning algorithm is designed, which combines the adaptive learning rate and the momentum term
- numerical experiments and comparisons are demonstrated to support the learning-based approach.

2 Design of SNNs

2.1 Spiking neurons

As other neural networks, the SNNs consist of spiking neurons and weights, where the neurons are connected with each other by the weighted connections. Different from other types of neurons, a spiking neuron does not fire at each propagation cycle, but it fire only when a membrane potential – an intrinsic quality of the neuron related to its membrane electrical charge – reaches a threshold. Once the neuron fires, it generates a signal that travels to other neurons that, in turn, increase or decrease their potentials in accordance with this signal.

The spiking neuron model (Bohte et al., 2002) is just a mathematical description of the biological properties of the spiking neuron. Some relevant models of spiking neurons include Hodgkin-Huxley (HH) model, leaky integrate-and-fire (LIF) model and Izhikevich model, where the LIF model becomes popular for its computational effectiveness. A generalisation of the LIF model is called spike response model (SRM). In the SRM, a neuron each time receives an input from a previous neuron, its internal state (membrane potential) changes. When the voltage passes its threshold, its action potentials are generated. Note that the threshold is not fixed but it depends on the time since the last spike. A simplified version of the SRM is named SRM0. Such a simplified mode includes an independence of the response kernel upon the time since the last spike.

The variant SRM0 is easier to fit to experimental data than the full SRM, since it needs less data. Therefore, this paper adopts the SRM0 as the spiking neurons. Considering such a neuron N_j, it receives input spikes from a series of presynaptic neurons in the set Γ_j. Here Γ_j is an index set, which contains each firing at times $t_i : i \in \Gamma_j$. N_j will fire at time $t = t_j$ whenever its membrane potential $u_j(t)$ reach a threshold θ, that is,

$$t_j = t^{(f)} : u_j\left(t^{(f)}\right) = \theta \wedge \dot{u}_j\left(t^{(f)}\right) > 0 \tag{1}$$

In equation (1), $\wedge$ indicates a logical AND operator. Each input spike from a presynaptic neuron induces a post synaptic potential via multiple synaptic contacts. The magnitude of the post synaptic potential due to the k^{th} synaptic connection depends on synaptic weight $\omega_{ji}^{(k)}$ and synaptic delay of the related synapse $d_{ji}^{(k)}$. The total of different post synaptic potentials is the membrane potential of N_j, formulated by

$$u_j(t) = \sum_{i \in \Gamma_j} \sum_k \omega_{ji}^{(k)} \varepsilon\left(t - t_{\mathrm{i}} - d_{ji}^{(k)}\right) = \sum_{i \in \Gamma_j} \sum_k \omega_{ji}^{(k)} y_{ji}^{(k)}(t) \tag{2}$$

In equation (2), $y_{ji}^{(k)}(t) = \varepsilon(t - t_{\mathrm{i}} - d_{ji}^{(k)})$. The function $\underline{\varepsilon}(\cdot)$ is a spike response kernel that the delegates normalised post synaptic potential. In this paper, we choose the spike response kernel as

$$\varepsilon(s)=\frac{s}{\tau_s}\exp\left(1-\frac{s}{\tau_s}\right)\cdot H(s) \tag{3}$$

Here τ_s is a synaptic time constant and $H(s)$ is a Heaviside step function, also named unit step function, defined by

$$H(s)=\begin{cases}1 & s>0^+\\ \int \sigma ds & 0^- \le s \le 0^+\\ 0 & s<0^-\end{cases} \tag{4}$$

Here σ means delta function.

2.2 Network structure

Shown in Figure 1, the left part is the network structure and the right part is the spiking neuron. From Figure 1, the topological structure of the SNNs is with a three-layer fully connected feed-forward topology. The structure of this networks looks like the BP neural networks. But they are different.

Illustrated in Figure 1, I, H and O indicate the set of input-layer, hidden-layer and output-layer neurons, respectively. $N_i^{(I)}: i\in I,\ N_h^{(H)}: h\in H\ N_o^{(O)}: o\in O$ denotes the neurons located at the input, hidden and output layers, respectively. $|I|$, $|H|$ and $|O|$ describe the number of the neurons located at the input, hidden and output layers, respectively. In brevity, $N_i^{(I)}, N_h^{(H)}$ and $N_o^{(O)}$ are abbreviated by N_i, N_h and N_o to describe the neurons in the input, hidden and output layers. Accordingly, the firing time in the layers are depicted by t_i, t_h and t_o. $|K|$ means the delay number of synaptic connections.

The firing time vectors of the input, hidden and output layers have the forms of

$$\begin{aligned}\boldsymbol{t}_I&=\begin{bmatrix}t_1 & t_2 & \cdots & t_i & \cdots & t_{|I|}\end{bmatrix}^T\in\Re^{|I|}\\ \boldsymbol{t}_H&=\begin{bmatrix}t_1 & t_2 & \cdots & t_h & \cdots & t_{|H|}\end{bmatrix}^T\in\Re^{|H|}\\ \boldsymbol{t}_O&=\begin{bmatrix}t_1 & t_2 & \cdots & t_o & \cdots & t_{|O|}\end{bmatrix}^T\in\Re^{|O|}\end{aligned} \tag{4}$$

The weight matrix between the output and hidden layers is written by

$$\boldsymbol{W}_{OH}\in\Re^{|O|\times|H||K|} \tag{5}$$

where the element of this matrix is written by $(\omega_{oh}^{(k)})_{o,h+(k-1)|H|}$. And the weight matrix between the hidden and input layers has the form of

$$\boldsymbol{W}_{HI}\in\Re^{|H|\times|I||K|} \tag{6}$$

where the element of this matrix is described by $(\omega_{hi}^{(k)})_{h,i+(k-1)|I|}$.

Further, the spike response matrix between the hidden and output layers can be obtained as

$$\boldsymbol{Y}_{OH}\in\Re^{|O|\times|H||K|} \tag{7}$$

where the element of the matrix is gotten by $(y_{oh}^{(k)}(t_o))_{o,h+(k-1)|H|}$. And the spike response matrix between the hidden and input layers has the form of

$$\boldsymbol{Y}_{HI}\in\Re^{|H|\times|I||K|} \tag{8}$$

where the element of the matrix is depicted by $(y_{hi}^{(k)}(t_h))_{h,i+(k-1)|I|}$.

Then, the membrane potential vectors of the hidden and output layers have the forms of

$$\boldsymbol{u}_H=\operatorname{diag}\left(\boldsymbol{W}_{HI}\boldsymbol{Y}_{HI}\left(\boldsymbol{t}_I\right)^T\right)\in\Re^{|H|} \tag{9}$$

and

$$\boldsymbol{u}_O=\operatorname{diag}\left(\boldsymbol{W}_{OH}\boldsymbol{Y}_{OH}\left(\boldsymbol{t}_H\right)^T\right)\in\Re^{|O|} \tag{10}$$

A nonlinear function (11) is defined in order to map the membrane potential of the neuron N_j into the firing time t_j, that is,

$$f\left(u_j\right)=t_j \tag{11}$$

Accordingly, there exists a vector function

$$f(u)=\left[f\left(u_1\right), f\left(u_2\right),\cdots f\left(u_j\right),\cdots f\left(u_n\right)\right]^T\in\Re^n \tag{12}$$

Actually, equation (1) has mapped the membrane potential to the spiking neuron firing time. Note that equation (1) contains the logic operator and it is an event mapping function. In order to design the BP for the SNNs in Figure 1, a continuous mapping function and its corresponding vector function is needed. For this purpose, equations (11) and (12) are defined. However, both equations (11) and (12) have no explicit form expression about equation (11). It is hard to obtain the explicit function and its derivative when the BP is deduced. Fortunately, the explicit function and its derivative are not inevitable due to the existence of the SpikeProp approximation (Bohte et al., 2002).

Now, the operation in the hidden layer can be written by

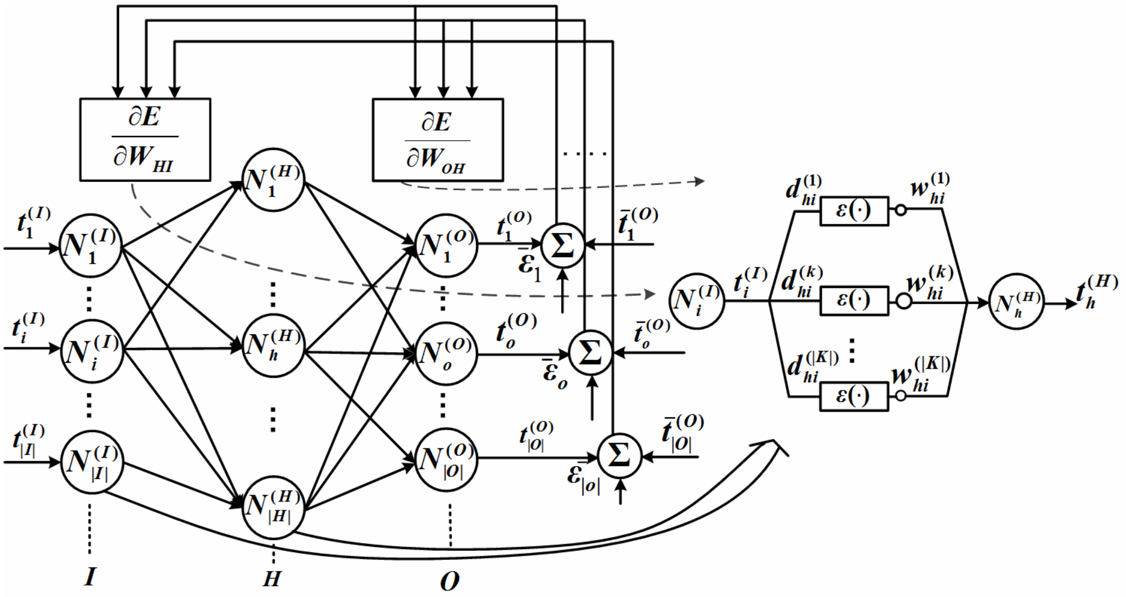
$$\boldsymbol{t}_H=\boldsymbol{f}\left(\operatorname{diag}\left(\boldsymbol{W}_{HI}\boldsymbol{Y}_{HI}\left(\boldsymbol{t}_I\right)^T\right)\right)=\boldsymbol{f}_d\left(\boldsymbol{W}_{HI}\boldsymbol{Y}_{HI}\left(\boldsymbol{t}_I\right)^T\right) \tag{13}$$

Here $\boldsymbol{f}_d$ is the map from $\Re^{n\times n}$ to $\Re^n$.

At last, with the topological structure in Figure 1, the input-output mapping of the SNNs can have the form of

$$\boldsymbol{t}_O=\boldsymbol{f}_d\left(\boldsymbol{W}_{OH}\boldsymbol{Y}_{OH}\left(\boldsymbol{f}_d\left(\boldsymbol{W}_{HI}\boldsymbol{Y}_{HI}\left(\boldsymbol{t}_I\right)^T\right)\right)^T\right) \tag{14}$$

Figure 1 Topological structure of the SNN



2.3 Hybrid learning rule

Assuming that there exist the optimal weight $\bar{\omega}_{ji}^{(k)}$ and the optimal weight matrix $\bar{\boldsymbol{W}}_{ji}$, the membrane potential of the optimal neuron can be written as

$$\bar{u}_j(t) = \sum_i \sum_k \bar{\omega}_{ji}^{(k)} y_{ji}^{(k)}(t) \tag{15}$$

Accordingly, the optimal firing times of the SNNs output can have the form of

$$\bar{\boldsymbol{t}}_O = \boldsymbol{f}_d\left(\bar{\boldsymbol{W}}_{OH}\boldsymbol{Y}_{OH}\left(\boldsymbol{f}_d\left(\bar{\boldsymbol{W}}_{HI}\boldsymbol{Y}_{HI}\left(\boldsymbol{t}_I\right)^T\right)\right)^T\right) \tag{16}$$

From equation (16), the system error can be obtained by the optimal firing times minus the firing times, that is,

$$\boldsymbol{e}_O = \bar{\boldsymbol{e}}_O - \boldsymbol{t}_O \tag{17}$$

According to equation (17), the learning cost function of the SNNs can be described by

$$E = \frac{1}{2}\boldsymbol{e}_O^T \boldsymbol{e}_O \tag{18}$$

The gradient descent weight update has the form of

$$\Delta\boldsymbol{W}_{ji} = -\boldsymbol{\eta}_j \frac{\partial E}{\partial \boldsymbol{W}_{ji}} \tag{19a}$$

$$\Delta\boldsymbol{W}_{OH} = -\boldsymbol{\eta}_o \frac{\partial E}{\partial \boldsymbol{W}_{OH}} \tag{19b}$$

$$\Delta\boldsymbol{W}_{HI} = -\boldsymbol{\eta}_h \frac{\partial E}{\partial \boldsymbol{W}_{HI}} \tag{19c}$$

Here $\boldsymbol{\eta}_j$, $\boldsymbol{\eta}_o$ and $\boldsymbol{\eta}_h$ indicate learning rates, it is positive definite matrices and their dimensions are in accordance with the change of the neurons but its elements are kept unchanged. This paper defines $\boldsymbol{\eta}_j$, $\boldsymbol{\eta}_o$ and $\boldsymbol{\eta}_h$ as diagonal matrices in the input, hidden and output layers, described by

$$\boldsymbol{\eta}_j = \operatorname{diag}\begin{pmatrix}\eta_1 & \eta_2 & \cdots & \eta_{|j|}\end{pmatrix}$$

$$\boldsymbol{\eta}_h = \operatorname{diag}\begin{pmatrix}\eta_1 & \eta_2 & \cdots & \eta_{|H|}\end{pmatrix}$$

$$\boldsymbol{\eta}_o = \operatorname{diag}\begin{pmatrix}\eta_1 & \eta_2 & \cdots & \eta_{|O|}\end{pmatrix}$$

Lemma 1 (Shrestha and Song, 2017): Define $y = f(\operatorname{diag}(\boldsymbol{AB})) = f(\boldsymbol{x}) : \Re^n \to \Re$, where $\boldsymbol{A} \in \Re^{n\times m}, \boldsymbol{B} \in \Re^{m\times n}$ and $\boldsymbol{x} = \operatorname{diag}(\boldsymbol{AB}) \in \Re^n$. Then, we have $\frac{\partial y}{\partial \boldsymbol{A}} = \operatorname{diag}\left(\frac{\partial y}{\partial \boldsymbol{x}}\right)\boldsymbol{B}^T$.

According to Lemma 1, the gradient of the

$$\begin{aligned}\frac{\partial E}{\partial \boldsymbol{W}_{OH}} &= \operatorname{diag}\left(\frac{\partial \boldsymbol{t}_O}{\partial \boldsymbol{u}_O}\frac{\partial E}{\partial \boldsymbol{t}_O}\right)\boldsymbol{Y}_{OH} \\ &= \operatorname{diag}\left(\begin{bmatrix}\frac{e_1}{\dot{u}_1(t_1)} & \cdots & \frac{e_o}{\dot{u}_o(t_o)} & \cdots & \frac{e_{|O|}}{\dot{u}_{|O|}(t_{|O|})}\end{bmatrix}^T\right)\boldsymbol{Y}_{OH}\end{aligned} \tag{20}$$

Considering the SpikeProp approximation around the neuron firing time, we have

$$\frac{\partial f(u_j(t_j))}{\partial u_j(t_j)} = \frac{\partial t_j}{\partial u_j(t_j)} = \frac{\partial t_j}{\partial u_j(t_j)}\Big|_{u_j=\theta} = \frac{-1}{\dot{u}_j(t_j)} \tag{21}$$

Since the firing time of a spiking neuron only depends on its membrane potential, we can have the following intermediate variable, defined by

$$\boldsymbol{\delta}_O = \frac{\partial \boldsymbol{t}_O}{\partial \boldsymbol{u}_O}\frac{\partial E}{\partial \boldsymbol{t}_O} \tag{22}$$

where $\delta_o = \frac{e_o}{\dot{u}_o(t_o)}$.

Substituting equation (22) into equation (20) yields

$$\frac{\partial E}{\partial \boldsymbol{W}_{OH}} = \operatorname{diag}\left(\boldsymbol{\delta}_o\right) \boldsymbol{Y}_{OH} \tag{23}$$

In the hidden layer, the gradient of the weights has the form of

$$\frac{\partial E}{\partial \boldsymbol{W}_{HI}} = \operatorname{diag}\left(\boldsymbol{\delta}_H\right) \boldsymbol{Y}_{HI} \tag{24}$$

In equation (24), $\boldsymbol{\delta}_H$ is determined by

$$\boldsymbol{\delta}_H = \frac{\partial \boldsymbol{t}_H}{\partial \boldsymbol{u}_H} \frac{\partial E}{\partial \boldsymbol{t}_H} = \frac{\partial \boldsymbol{t}_H}{\partial \boldsymbol{u}_H} \frac{\partial \boldsymbol{u}_O}{\partial \boldsymbol{t}_H} \boldsymbol{\delta}_O \tag{25}$$

where

$$\begin{aligned} \delta_h &= \frac{\sum_o \delta_o \sum_k \omega_{oh}^{(k)} \dfrac{\theta \partial y_{oh}^{(h)}\left(t_o\right)}{\partial t_o}}{\dot{u}_h\left(t_h\right)} = -\frac{1}{\dot{u}_h\left(t_h\right)} \frac{\partial \boldsymbol{u}_O}{\partial t_h} \boldsymbol{\delta}_O \\ &= \frac{1}{\dot{u}_h\left(t_h\right)} \frac{\partial \boldsymbol{t}_O}{\partial t_h} \boldsymbol{e}_O \end{aligned} \tag{26}$$

Finally, the weight update rule based on the SpikeProp algorithm at the p^{th} iteration is formulated by

$$\boldsymbol{W}_{ji}(p+1) = \boldsymbol{W}_{ji}(p) - \boldsymbol{\eta}_j \operatorname{diag} \boldsymbol{Y}_{ji} \tag{27}$$

As far as each spiking neuron is concerned, the weight update rule via the SpikeProp algorithm is

$$\omega_{ji}(p+1) = \omega_{ji}(p) - \eta_j \delta_j y_{ji} \tag{28}$$

Similarly, the weight update rule between the output and hidden layers at the p^{th} iteration can be written by

$$\boldsymbol{W}_{OH}(p+1) = \boldsymbol{W}_{OH}(p) - \boldsymbol{\eta}_O \operatorname{diag}\left(\boldsymbol{\delta}_O\right) \boldsymbol{Y}_{OH} \tag{29}$$

Concerning each connection, this weight update rule is

$$\omega_{oh}(p+1) = \omega_{oh}(p) - \eta_o \delta_o y_{oh} \tag{30}$$

With regard to the weight up rule between the hidden and input layers, we can have

$$\boldsymbol{W}_{HI}(p+1) = \boldsymbol{W}_{HI}(p) - \boldsymbol{\eta}_h \operatorname{diag}\left(\boldsymbol{\delta}_h\right) \boldsymbol{Y}_{HI} \tag{31}$$

$$\omega_{hi}(p+1) = \omega_{hi}(p) - \eta_h \delta_h y_{hi} \tag{32}$$

Inherently, the SpikeProp algorithm is similar to the error BP. It characteristic is that every connection between two layers consists of a fixed number of delayed synaptic potentials. There are many techniques to enhance the generalisation performance and the convergence speed, such as additional learning rules, having a momentum term and so on.

For this purpose, this paper employs the adaptive learning rate technique and adds a momentum term at the p^{th} iteration. Note that equations (27), (29) and (31) are equivalent to equations (28), (30) and (32), respectively. Thus, equations (28), (30) and (32) are listed to illustrate the designed hybrid algorithm. Finally, the weight update rule has the form of

$$\begin{aligned} \omega_{ji}(p+1) &= \omega_{ji}(p) - \eta_j(p+1)\delta_j y_{ji} + \alpha_j \Delta\omega_{ji}(p-1) \\ \omega_{oh}(p+1) &= \omega_{oh}(p) - \eta_o(p+1)\delta_o y_{oh} + \alpha_o \Delta\omega_{ji}(p-1) \\ \omega_{hi}(p+1) &= \omega_{hi}(p) - \eta_h(p+1)\delta_h y_{hi} + \alpha_h \Delta\omega_{hi}(p-1) \end{aligned} \tag{33}$$

Here $\eta_j(p + 1)$, $\eta_o(p + 1)$ and $\eta_h(p + 1)$ are adaptive learning rates and they are adjusted in light of the delta-bar-delta rule; α_j, α_o and α_h are momentum coefficients.

So far, the SNNs in Figure 1 have been structured with the SpikeProp-based hybrid learning rule.

3 Wind speed date

3.1 Dataset

The data of wind speed are gotten from a Canadian website 'http://climate.weather.gc.ca/'. The data are measured in Regina, Saskatchewan. This city is in Canada and its geographical coordinates are located at 5°°25'56” North and 104°39'58” West. This paper selects the data of wind speed between July 1, 2015 and July 31, 2015. The dataset contains a full month and it is able to represent the characteristics of wind speed in this short-term period. In this dataset, the sampling period is 1 hour so that 24 samples can be obtained in one day and there are 744 samples in this dataset. For each sample, the wind speed is not only contained, but also the wind direction, temperature, humidity and air pressure are included.

In practice, some bad data may be included in this dataset because of measuring error and mis-registration. In order to utilise the dataset, the availability of these data has to be checked. The purpose of such pre-processing is to eliminate the adverse effect of these bad data. Then, the ARMA method is adopted to calculate new data so that the number of data in this set is kept unchanged as 744. These data are illustrated in Figures 2–6.

Figure 2 Hourly data of wind speed from July 1 to July 31 in 2015 (see online version for colours)

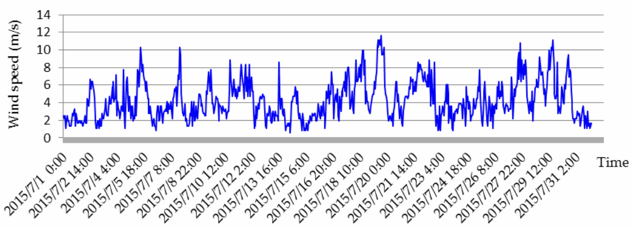


Figure 3 Hourly data of wind direction from July 1 to July 31 in 2015 (see online version for colours)

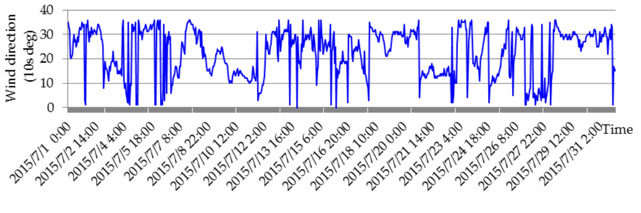


Figure 4 Hourly data of air pressure from July 1 to July 31 in 2015 (see online version for colours)

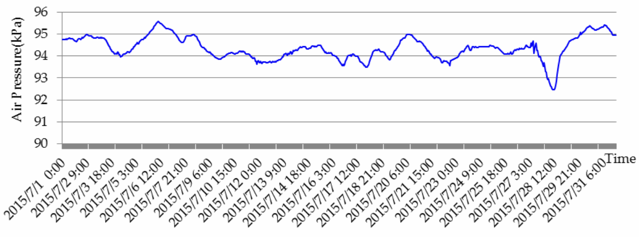


Figure 5 Hourly data of temperature from July 1 to July 31 in 2015 (see online version for colours)

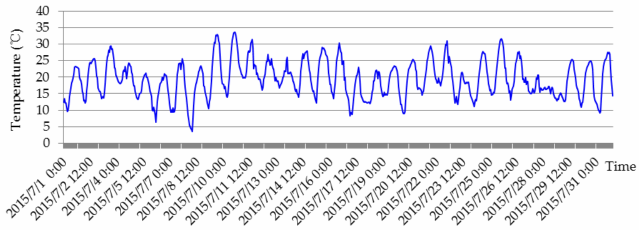
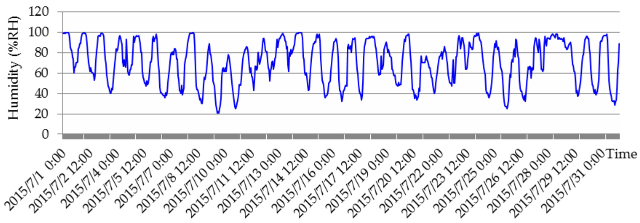


Figure 6 Hourly data of humidity from July 1 to July 31 in 2015 (see online version for colours)



3.2 Data analysis

Consider the variables of wind speed, wind direction, temperature, humidity and air pressure in Figures 2–6. The units of these variables are completely different from each other so that it is hard to analyse them by the statistic method. In order to deal with this issue, all the data of these variables are non-dimensionalised and normalised. For each variable, the normalised formula is described by

$$\vartheta_m = \frac{\Theta_m - \min(\Theta)}{\max(\Theta) - \min(\Theta)} \tag{34}$$

Here Θ indicates the dataset of a variable; Θ_m is the m^{th} element in this dataset and ϑ_m is its normalised one; min() and max() mean the maximum and minimum elements in this dataset.

Take two datasets into consideration. Their correlation coefficient can be formulated by

$$\rho_{\Phi\Psi} = \frac{\operatorname{cov}(\Phi, \Psi)}{\sqrt{D(\Phi)}\sqrt{D(\Psi)}} \tag{35}$$

Here Φ and Ψ indicate two datasets; cov() means their covariance; D() delegates the variance of each set.

Finally, the correlation coefficients of wind speed with respect to wind direction, air pressure, temperature and humidity are calculated and listed by Table 1.

Table 1 Some correlation coefficients

Wind speed with respect to	*Correlation coefficients*
Wind direction	0.82
Air pressure	–0.52
Temperature	0.44
Humidity	–0.47

From Table 1, it is apparent that the variable of wind speed is strong direct relationship with respect to the variable of wind direction, it is moderate with respect to the variable of temperature, and it is moderately negative correlation with respect to the variables of air pressure and humidity. Since the wind speed modelling is rather difficult and complex by the first principle method, this paper explores the learning-based modelling for the wind speed. The model is intelligent on basis of SNNs.

Concerning the SNNs method, an important step is to decide the input variables. Redundant data cannot benefit the modelling accuracy and they will definitely increase the computational burden. According to the results in Table 1, the four variables, that is, wind direction, air pressure, temperature, and humidity, are associated with the variable of wind speed very much. Consequently, the wind direction, air pressure, temperature and humidity will be employed as the input data of the SNNs besides the wind speed, which can contribute to the modelling accuracy of the wind speed.

4 Comparisons

In the dataset, we have 744 samples, which are divided into two groups. The first group containing the ahead 720 data is treated as the training data. The second group contains the last 24 data that are adopted as the testing data. Such division indicates the number of neurons at the output layer of the SNNs should be 24.

Concerning the computational burden, the 24 time-series data of the wind speed are designed as the inputs. Considering the effect of the wind direction, air pressure, temperature and humidity, the averages of the wind speed and the wind direction, air pressure, temperature and humidity at the 24 samples are also employed as the inputs in order to refine the modelling accuracy.

Such structure suggests that the 24 time-series data and the averages of the wind speed and the wind direction, air pressure, temperature and humidity are utilised to forecast the subsequent 24 data of wind speed.

4.1 Parameters of the SNNs

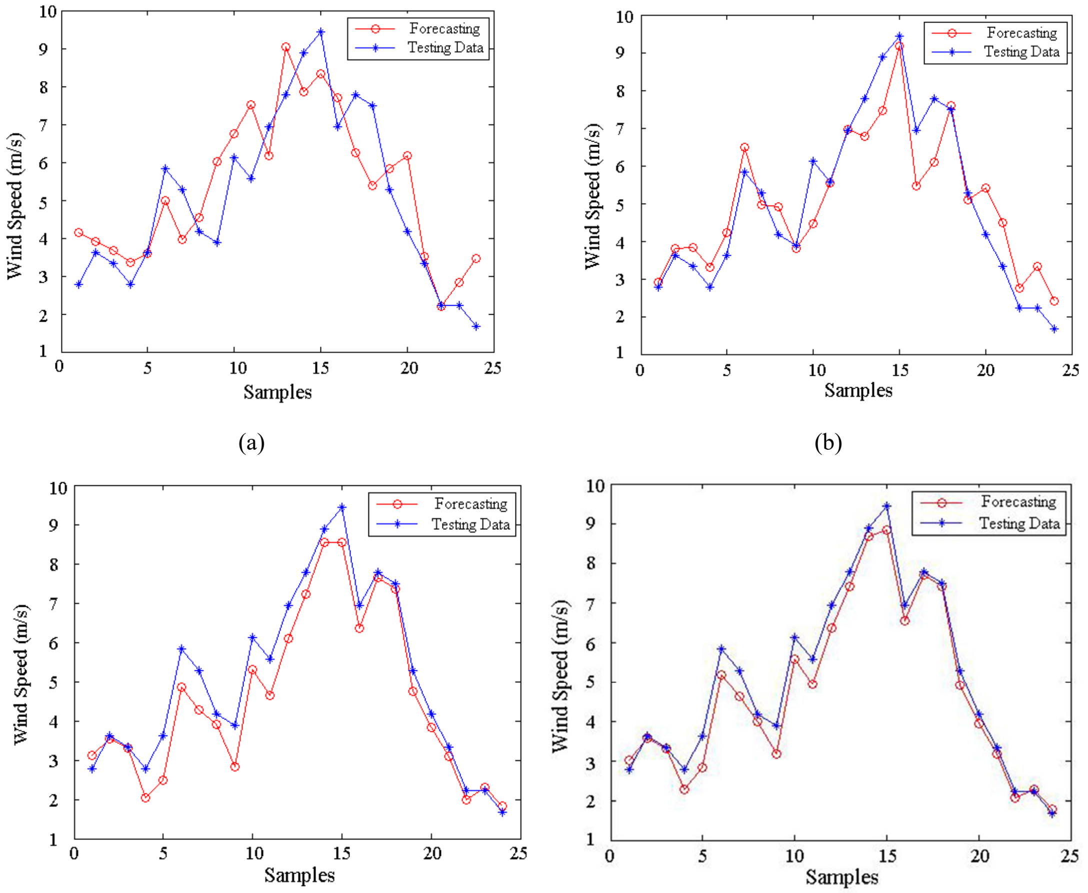
According to the hardware configuration of the computer, the number of neurons at the hidden layer is set to 40. Similarly, the number of synaptic connections between the input and hidden layers is set to 16 and the number of synaptic connections between the hidden and output layers is set to 16. The three numbers are selected by trial and error and they are picked up by the trade-off between the modelling accuracy and the computational time. For the synaptic delay, its synaptic delay is set to 1 ms. The synaptic time constant τ_s is set to 5 ms. The threshold θ is 1 mV. The momentum coefficients α_j, α_o and α_h are set to 0.1. The initial values of η_j, η_o and η_h are set to 0.05 and they are they are adjusted in light of the delta-bar-delta rule during the learning process. The initial values of ω_{ji}, ω_{oh} and ω_{hi} are set to the random number in the closed interval [0, 1] and they are updated by the formulas (33).

4.2 Results

The ahead 720 data, that is, the data from July 1 to July 30, are utilised as training data. These data are used to train all the four neural networks. Once the neural networks have been trained, all the weights and parameters are kept fixed, that is, the learning-based forecasting model of wind speed has been fulfilled. The last step is to verify the modelling accuracy via some testing data. In this paper, the data from the last day, July 31, are employed to verify the performance of these neural networks. The forecasting performance of the four types of neural networks is demonstrated in Figure 7.

From Figure 7, it is apparent that both the SNN-based approaches in Figures 7(a)–7(b) have the better performance. To some extent, the BP and RBF neural networks in Figures 7(c)–7(d) could be available to model the wind speed, but they are not accurate enough and their errors are worse than the SNN-based approaches. As far as the applications in the real-world are concerned, the wind-speed forecasting is usually the first step for the subsequent dispatch. Therefore it is expected to forecast the accuracy as well as possible so that the SNN-based approaches are more potential in reality, compared to the modelling by the BP and RBF networks.

Figure 7 Comparisons of the performance by four kinds of neural networks, (a) BP neural networks (b) RBF neural networks (c) SNNs by the basic SpikeProp (d) SNNs by the designed hybrid algorithm (see online version for colours)



(a)

(b)

(c)

(d)

Further, for the purpose of measuring the modelling accuracy, some indexes are introduced, calculated and listed by Table 2. The indexes in Table 2 are defined as follows. MAE, MAPE and RMSE are short for mean absolute error, mean absolute percent error and root mean square error, respectively. They are defined by

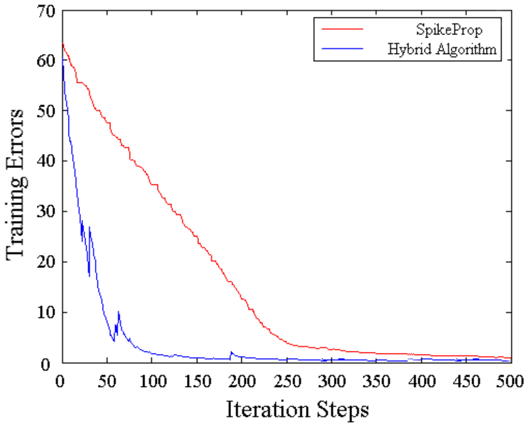
$$\begin{aligned} \text{MAE} &= \frac{1}{\bar{N}} \sum_{i=1}^{\bar{N}} \left| \tilde{v}_i - v_i \right| \\ \text{MAPE} &= \frac{1}{\bar{N}} \sum_{i=1}^{\bar{N}} \frac{\left| \tilde{v}_i - v_i \right|}{v_i} \times 100\% \\ \text{RMSE} &= \sqrt{\frac{1}{\bar{N}} \sum_{i=1}^{\bar{N}} \left(\tilde{v}_i - v_i \right)^2} \end{aligned} \tag{36}$$

Here $\bar{N}$ is the number of samples, $\tilde{v}_i$ is the forecasting wind speed of the i^{th} sample and v_i is the true wind speed of the i^{th} sample.

Table 2 Comparisons from the aspect of statistic indexes

Indexes / *Approaches*	*MAE*	*MAPE*	*RMSE*
BP neural networks	0.98	22	1.18
RBF neural networks	0.68	16.2	0.87
SNNs by the basic SpikeProp by Shrestha and Song (2017)	0.52	11.1	0.63
SNNs by the hybrid algorithm	0.35	7.5	0.42

Figure 8 Comparison of error convergence by the SNNs with two kinds of learning algorithms (see online version for colours)



Finally, Figure 8 illustrates the error convergence of the two SNN-based approaches. From Figure 8, the delta-bar-delta rule and the momentum term in the designed hybrid algorithm can effectively expedite the error convergence and apparently shorten the number of iterations. From Table 2, the SNNs by the hybrid algorithm have the slight better performance than the SNNs by the basic SpikeProp (Shrestha and Song, 2017). In Figure 8, such a hybrid algorithm can apparently refine the convergence.

5 Conclusions

Concerning the forecasting problem of wind speed, this paper has developed a learning-based method. Such a method adopts the SNNs with the adaptive learning rate and the momentum term. The hourly wind speed data from the city of Regina in Canada are utilised as the training and testing dataset. The data not only set contains the wind speed, but also it has wind direction, air pressure, temperature and humidity. Through the data pre-processing, we get 744 data from July 1, 2005 to July 31, 2005. The correlation coefficients of these facts with respect to wind speed, it reveals that wind speed has a strong direct relationship with wind direct. Among the 744 data, the head 720 data are adopted as the training data and the other 24 data are considered as the testing data. In order to verify the feasibility and validity of the SNNs with the hybrid algorithm, other three neural networks, that is, the BP neural networks, the RBF neural networks and the SNNs with the basic SpikeProp, are taken into consideration as comparisons. Firstly, these four kinds of neural networks are trained by the training data. Then, the well-trained neural networks take charge of forecasting the testing data. The forecasting results by the four kinds of neural networks have been illustrated. Some statistic indexes such as MAE, MAPE and RMSE are calculated. Compared with the forecasting results by the BP neural networks and the RBF neural networks, the forecasting results by two kinds of SNNs have higher accuracy. Further, the results by the SNNs with the hybrid algorithm are more accurate than the results by the SNNs with the basic SpikeProp. However, this learning-based forecasting approach depends on the pre-processed training data. In reality, any measured data must contain noise or error. Currently, we are studying this issue and theoretically analysing the adverse effect of the bad data.

References

Aasim, S.N. and Singh, A.M. (2019) 'Repeated wavelet transform based ARIMA model for very short-term wind speed forecasting', *Renew. Energy*, Vol. 136, pp.758–768.

Al-Falahi, M.D.A., Jayasinghe, S.D.G. and Enshaei, H. (2017) 'A review on recent size optimization methodologies for standalone solar and wind hybrid renewable energy system', *Energy Conv. Manag.*, Vol. 143, pp.252–274.

Alharthi, Y.Z., Siddiki, M.K. and Chaudhry, G.M. (2018) 'Resource assessment and techno-economic analysis of a grid-connected solar PV-wind hybrid system for different locations in Saudi Arabia', *Sustainability*, Vol. 10, No. 10, p.3690.

Astolfi, D., Castellani, F., Berno, F. and Terzi, L. (2018) 'Numerical and experimental methods for the assessment of wind turbine control upgrades', *Appl. Sci.-Basel*, Vol. 8, No. 12, p.2639.

Bohte, S.M., Kok, J.N. and La Poutre, H. (2002) 'Error-backpropagation in temporally encoded networks of spiking neurons', *Neurocomputing*, Vol. 48, pp.17–37.

Casella, L. (2019) 'Wind speed reconstruction using a novel multivariate probabilistic method and multiple linear regression: advantages compared to the single correlation approach', *J. Wind Eng. Ind. Aerodyn.*, Vol. 191, pp.252–265.

Ding, M., Zhou, H., Xie, H., Wu, M., Nakanishi, Y. and Yokoyama, R. (2019) 'A gated recurrent unit neural networks based wind speed error correction model for short-term wind power forecasting', *Neurocomputing*, Vol. 365, pp.54–61.

Erdem E. and Shi, J. (2011) 'ARMA based approaches for forecasting the tuple of wind speed and direction', *Appl. Energy*, Vol. 88, No. 4, pp.1405–1414.

Hu, J.M. and Wang, J.Z. (2015) 'Short-term wind speed prediction using empirical wavelet transform and Gaussian process regression', *Energy*, Vol. 93, pp.1456–1466.

Huang, C.J. and Kuo, P.H. (2018) 'A short-term wind speed forecasting model by using artificial neural networks with stochastic optimization for renewable energy systems', *Energies*, Vol. 11, No. 10, p.2777.

Kulkarni, S., Simon, S.P. and Sundareswaran, K. (2013) 'A spiking neural network (SNN) forecast engine for short-term electrical load forecasting', *Appl. Soft. Comput.*, Vol. 13, No. 8, pp.3628–3635.

Li, C.D., Wang, L., Zhang, G.Q., Wang, H.D. and Shang, F. (2017) 'Functional-type single-input-rule-modules connected neural fuzzy system for wind speed prediction', *IEEE/CAA J. of Autom. Sinica*, Vol. 4, No. 4, pp.751–762.

Liu, D., Niu, D.X., Wang, H. and Fan, L.L. (2014) 'Short-term wind speed forecasting using wavelet transform and support vector machines optimized by genetic algorithm', *Renew. Energy*, Vol. 62, pp.592–597.

Ma, T., Wang, C., Wang, J.Z., Cheng, J.J. and Chen, X.Y. (2019) 'Particle-swarm optimization of ensemble neural networks with negative correlation learning for forecasting short-term wind speed of wind farms in western China', *Inf. Sci.*, Vol. 505, pp.157–182.

Maciag, P.S., Kasabov, N., Kryszkiewicz, M. and Bembenik, R. (2019) 'Air pollution prediction with clustering-based ensemble of evolving spiking neural networks and a case study for London area', *Environ. Modell. Softw.*, Vol. 118, pp.262–280.

Marugan, A.P., Marquez, F.P.G., Perez, J.M.P. and Ruiz-Hernandez, D. (2018) 'A survey of artificial neural network in wind energy systems', *Appl. Energy*, Vol. 228, pp.1822–1836.

Qian, D.W., Tong, S.W. and Li, C.D. (2016) 'Leader-following formation control of multiple robots with uncertainties through sliding mode and nonlinear disturbance observer', *ETRI J*, Vol. 38, No. 5, pp.1008–1018.

Qian, D.W., Tong, S.W. and Yi, J.Q. (2013) 'Adaptive control based on incremental hierarchical sliding mode for overhead crane systems', *Appl. Math. Inf. Sci.*, Vol. 7, No. 4, pp.1359–1364

Qian, D.W., Tong, S.W., Yang, B.B. and Lee, S.G. (2015) 'Design of simultaneous input-shaping-based SIRMs fuzzy control for double-pendulum-type overhead cranes', *Bull. Pol. Acad. Sci-Tech. Sci.*, Vol. 63, No. 4, pp.887–896.

Saunders, D.J., Patel, D., Hazan, H., Siegelmann, H.T. and Kozma, R. (2019) 'Locally connected spiking neural networks for unsupervised feature learning', *Neural Netw.*, Vol. 119, pp.332–340.

Shrestha, S.B. and Song, Q. (2017) 'Robust learning in SpikeProp', *Neural Netw.*, Vol. 86, pp.54–68.

Sun, G.Q., Chen, T., Wei, Z.N., Sun, Y.H., Zang, H.X. and Chen, S. (2016) 'A carbon price forecasting model based on variational mode decomposition and spiking neural networks', *Energies*, Vol. 9, No. 1, DOI: 10.3390/en9010054.

Wang, C., Wu, J., Wang, J.Z. and Hu, Z.J. (2016) 'Short-term wind speed forecasting using the data processing approach and the support vector machine model optimized by the improved cuckoo search parameter estimation algorithm', *Math. Probl. Eng.*, DOI: 10.1155/2016/4896854.

Wang, Y.L., Zhou, X., Liang, L.K., Zhang, M.J., Zhang, Q. and Niu, Z.Q. (2018) 'Short-term wind speed forecast based on least squares support vector machine', *J. Inf. Process. Syst.*, Vol. 14, No. 6, pp.1385–1397.

Wu, Q.L. and Lin, H.X. (2019) 'Short-term wind speed forecasting based on hybrid variational mode decomposition and least squares support vector machine optimized by bat algorithm model', *Sustainability*, Vol. 11, No. 3, p.652.

Xu, L., Li, C., Xie, X. and Zhang, G. (2018) 'Long-short-term memory network based hybrid model for short-term electrical load forecasting', *Information*, Vol. 9, No. 7, p.165.

Yu, C.J., Li, Y.L., Xiang, H.Y. and Zhang, M.J. (2018) 'Data mining-assisted short-term wind speed forecasting by wavelet packet decomposition and Elman neural network', *J. Wind Eng. Ind. Aerodyn.*, Vol. 175, pp.136–143.

Zheng, C.W., Xiao, Z.N., Peng, Y.H., Li, C.Y. and Du, Z.B. (2018) 'Rezoning global offshore wind energy resources', *Renew. Energy*, Vol. 129, pp.1–11.

Zhu, J.Z. and Zhang, Y. (2019) 'Probabilistic load flow with correlated wind power sources using a frequency and duration method', *IET Gener. Transm. Distrib.*, Vol. 13, No. 18, pp.4158–4170.