Journal Pre-proof

A probability density function generator based on neural networks

Chi-Hua Chen, Fangying Song, Feng-Jang Hwang, Ling Wu

PII: S0378-4371(19)31871-0
DOI: https://doi.org/10.1016/j.physa.2019.123344
Reference: PHYSA 123344

To appear in: *Physica A*

Received date : 16 May 2019
Revised date : 28 August 2019

Please cite this article as: C.-H. Chen, F. Song, F.-J. Hwang et al., A probability density function generator based on neural networks, *Physica A* (2019), doi: https://doi.org/10.1016/j.physa.2019.123344.

A Probability Density Function Generator Based on Neural Networks

Chi-Hua Chen[1], Fangying Song[1,*], Feng-Jang Hwang[2], Ling Wu[1]
[1]College of Mathematics and Computer Science, Fuzhou University, China
[2] School of Mathematical and Physical Sciences, Transport Research Centre, University of Technology Sydney, Australia

Abstract—In order to generate a probability density function (PDF) for fitting the probability distributions of practical data, this study proposes a deep learning method which consists of two stages: (1) a training stage for estimating the cumulative distribution function (CDF) and (2) a performing stage for predicting the corresponding PDF. The CDFs of common probability distributions can be utilised as activation functions in the hidden layers of the proposed deep learning model for learning actual cumulative probabilities, and the differential equation of the trained deep learning model can be used to estimate the PDF. Numerical experiments with single and mixed distributions are conducted to evaluate the performance of the proposed method. The experimental results show that the values of both CDF and PDF can be precisely estimated by the proposed method.

Keywords—probability density function, cumulative distribution function, neural networks.

1. Introduction

For explaining trends or phenomena, the statistic tools and probability models are usually applied to analyse practical data. Various common probability distributions (e.g. the exponential distribution (ED), normal distribution (ND), log-normal distribution (LD), gamma distribution (GD), etc.) are extensively used as the assumptions about the distribution of practical data [1]. The CDF and PDF of practical data, however, could be complex and irregular, so serious errors would be incurred by the naïve assumptions. This study proposes a deep learning method [2-4] that develops a neural network to learn the CDF of practical data and fit the corresponding PDF (shown in Figure 1). In the training stage, the CDF of practical data is collected, and the proposed deep neural network model with designed activation functions is trained to learn the CDF. In the performing stage, the differentiation of CDF in the trained deep neural network model is conducted to estimate the PDF of practical data for various sorts of probability-based applications.

The contributions of this study are highlighted as follows.

(1). A combination of several probability functions is utilised as activation functions in the proposed deep neural network model for learning the CDF of practical data.

(2). The weight updates of each neuron in the proposed deep neural network model based on gradient descent have been derived and evaluated.

(3). The PDF of practical data in various probability-based applications can be obtained by performing the CDF differentiation for the corresponding trained deep neural network model.

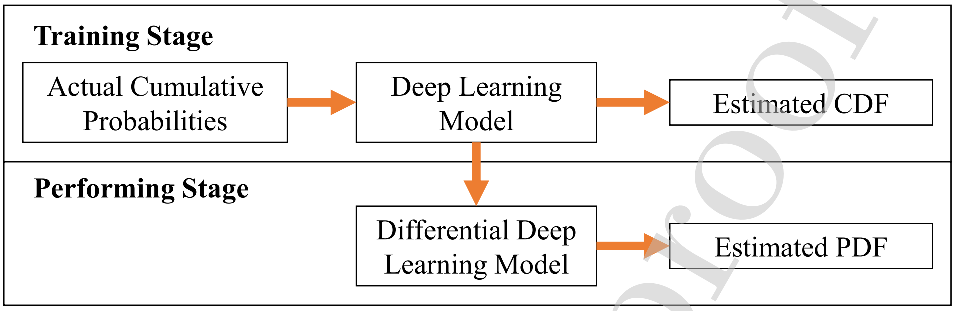


Fig. 1. The proposed deep learning method for estimating practical probability distributions.

In the next section, the literature reviews of neural networks and activation functions are provided. Section 3 proposes a deep neural network model based on designed activation functions to estimate the CDF and PDF of practical data. In Section 4, practical experiments are designed to evaluate the proposed method, and the results of these experiments are also analysed and discussed in this section. The conclusions and future work of this study are presented in Section 5.

2. Literature Reviews

In recent years, the activation functions including linear function (i.e. $a_L(\cdot)$ in Equation (1)) [5-7], rectified linear unit (ReLU) function (i.e. $a_R(\cdot)$ in Equation (2)) [6-14] and sigmoid function (i.e. $a_S(\cdot)$ in Equation (3)) [15-20] have been popularly applied in the neural network models. The graph of linear function is a straight line, so the linear function cannot denote curves or hyperbolae. The ReLU function adopts the positive part of inputs, and the negative part of inputs is denoted as zero for improving the linear function. The sigmoid function can be used to represent curves or hyperbolae for the analyses of nonlinear problems.

$$a_L(z) = z \tag{1}$$

$$a_R(z) = \begin{cases} z, \text{if } z > 0 \\ 0, \text{otherwise} \end{cases} \tag{2}$$

$$a_S(z) = \frac{1}{1+e^{-z}} \tag{3}$$

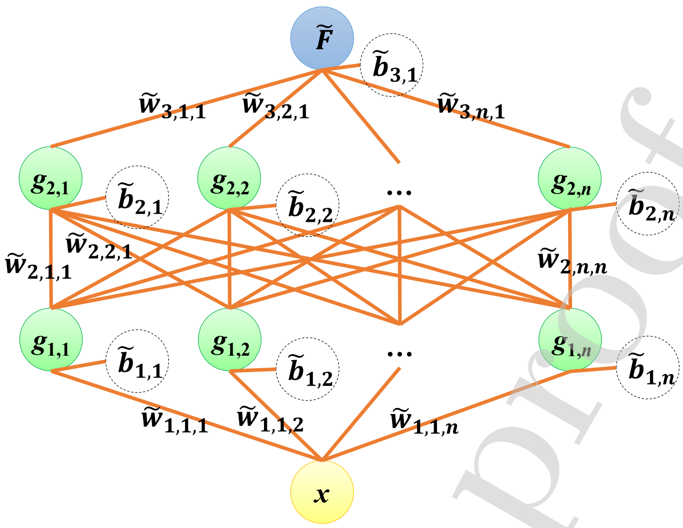


Fig. 2. The neural network method for estimating probability distributions.

A neural network model for estimating probability distribution is shown in Figure 2. The neural network has an input layer, two hidden layers and an output layer. One parameter (i.e. the parameter x) is included in the input layer, and one parameter (i.e. the parameter $\tilde{F}$) is included in the output layer. Each hidden layer has n neurons; for instance, the parameter $g_{k,j}$ denotes the j-th neuron in the k-th layer (shown in Equation (4)), and the parameter $\tilde{w}_{k,i,j}$ denotes the weight between the i-th neuron in the (k-1)-th layer and the j-th neuron in the k-th layer. The function $a(\cdot)$ which denotes the activation function of neuron could be a linear function, ReLU function, or sigmoid function. The minimum square errors (shown in Equation (5)) are analysed according to the true value of output (i.e. the parameter F) for optimising the constructed neural network. In the neural network, the parameter $\sigma_{k,j}$ denotes the error of the j-th neuron in the k-th layer. Therefore, the weights in the neural network can be updated by Equations (6)-(11) based on gradient descent method [21-24].

$$g_{k,j} = a\left(z_{k,j}\right), \text{where } z_{k,j} = \sum_{i=1}^{n} \tilde{w}_{k,i,j} \times g_{k-1,i} + \tilde{b}_{k,j} \tag{4}$$

$$E = \frac{1}{2}\left(\tilde{F} - F\right)^2 = \frac{1}{2}\sigma_{3,1}{}^2 \tag{5}$$

$$\begin{aligned}\frac{\partial E}{\partial \tilde{b}_{3,1}} &= \frac{\partial E}{\partial \sigma_{3,1}}\frac{\partial \sigma_{3,1}}{\partial g_{3,1}}\frac{\partial g_{3,1}}{\partial z_{3,1}}\frac{\partial z_{3,1}}{\partial \tilde{b}_{3,1}} \\ &= \sigma_{3,1}\times 1\times \frac{\partial g_{3,1}}{\partial z_{3,1}}\times 1\end{aligned} \tag{6}$$

$$\begin{aligned}\frac{\partial E}{\partial \tilde{w}_{3,i,1}} &= \frac{\partial E}{\partial \sigma_{3,1}}\frac{\partial \sigma_{3,1}}{\partial g_{3,1}}\frac{\partial g_{3,1}}{\partial z_{3,1}}\frac{\partial z_{3,1}}{\partial \tilde{w}_{3,i,1}} \\ &= \sigma_{3,1}\times 1\times \frac{\partial g_{3,1}}{\partial z_{3,1}}\times g_{2,i}\end{aligned} \tag{7}$$

$$\begin{aligned}\frac{\partial E}{\partial \tilde{b}_{2,i}} &= \frac{\partial E}{\partial \sigma_{3,1}}\frac{\partial \sigma_{3,1}}{\partial g_{3,1}}\frac{\partial g_{3,1}}{\partial z_{3,1}}\frac{\partial z_{3,1}}{\partial g_{2,i}}\frac{\partial g_{2,i}}{\partial z_{2,i}}\frac{\partial z_{2,i}}{\partial \tilde{b}_{2,i}} \\ &= \sigma_{3,1}\times 1\times \frac{\partial g_{3,1}}{\partial z_{3,1}}\times \tilde{w}_{3,i,1}\times \frac{\partial g_{2,i}}{\partial z_{2,i}}\times 1 \\ &= \sigma_{2,i}\times \frac{\partial g_{2,i}}{\partial z_{2,i}}\times 1\end{aligned} \tag{8}$$

$$\begin{aligned}\frac{\partial E}{\partial \tilde{w}_{2,i,j}} &= \frac{\partial E}{\partial \sigma_{3,1}}\frac{\partial \sigma_{3,1}}{\partial g_{3,1}}\frac{\partial g_{3,1}}{\partial z_{3,1}}\frac{\partial z_{3,1}}{\partial g_{2,i}}\frac{\partial g_{2,i}}{\partial z_{2,i}}\frac{\partial z_{2,i}}{\partial \tilde{w}_{2,i,j}} \\ &= \sigma_{3,1}\times 1\times \frac{\partial g_{3,1}}{\partial z_{3,1}}\times \tilde{w}_{3,i,1}\times \frac{\partial g_{2,i}}{\partial z_{2,i}}\times g_{1,j} \\ &= \sigma_{2,i}\times \frac{\partial g_{2,i}}{\partial z_{2,i}}\times g_{1,j}\end{aligned} \tag{9}$$

$$\frac{\partial E}{\partial \tilde{b}_{1,j}} = \sigma_{1,i}\times \frac{\partial g_{1,i}}{\partial z_{1,i}}\times 1 \tag{10}$$

$$\frac{\partial E}{\partial \tilde{w}_{1,i,j}} = \sigma_{1,i}\times \frac{\partial g_{1,i}}{\partial z_{1,i}}\times x \tag{11}$$

When the neural network is used to learn the CDF of data, the value of CDF is recorded as the parameter F. Therefore, the PDF of data can be estimated through performing the CDF differentiation for the trained neural network (shown in Equation (12)). The activation function can be set as a linear function, ReLU function, or sigmoid function, and the differentiation derivation for the trained neural network model with respective activation functions can be found in Equations (13)-(15). The parameter x is missing after the differentiation as shown in Equations (13) and (14), which only include constants. Therefore, this study uses a combination of numerous probability functions as activation functions to build up the robustness of the neural network model.

$$\begin{aligned}\frac{\partial \tilde{F}}{\partial x} &= \frac{\partial \tilde{F}}{\partial g_{3,1}} \frac{\partial g_{3,1}}{\partial z_{3,1}} \sum_{i=1}^{n} \frac{\partial z_{3,1}}{\partial g_{2,i}} \frac{\partial g_{2,i}}{\partial z_{2,i}} \sum_{j=1}^{n} \frac{\partial z_{2,i}}{\partial g_{1,j}} \frac{\partial g_{1,j}}{\partial z_{1,j}} \frac{\partial z_{1,j}}{\partial x} \\ &= 1 \times \frac{\partial g_{3,1}}{\partial z_{3,1}} \times \sum_{i=1}^{n} \tilde{w}_{3,i,1} \times \frac{\partial g_{2,i}}{\partial z_{2,i}} \times \sum_{j=1}^{n} \tilde{w}_{2,j,i} \times \frac{\partial g_{1,j}}{\partial z_{1,j}} \tilde{w}_{1,1,j}\end{aligned} \tag{12}$$

$$\begin{aligned}\frac{\partial \tilde{F}}{\partial x} &= \frac{\partial \tilde{F}}{\partial g_{3,1}} \frac{\partial g_{3,1}}{\partial z_{3,1}} \sum_{i=1}^{n} \frac{\partial z_{3,1}}{\partial g_{2,i}} \frac{\partial g_{2,i}}{\partial z_{2,i}} \sum_{j=1}^{n} \frac{\partial z_{2,i}}{\partial g_{1,j}} \frac{\partial g_{1,j}}{\partial z_{1,j}} \frac{\partial z_{1,j}}{\partial x} \\ &= 1 \times \frac{\partial g_{3,1}}{\partial z_{3,1}} \times \sum_{i=1}^{n} \tilde{w}_{3,i,1} \times \frac{\partial g_{2,i}}{\partial z_{2,i}} \times \sum_{j=1}^{n} \tilde{w}_{2,j,i} \times \frac{\partial g_{1,j}}{\partial z_{1,j}} \tilde{w}_{1,1,j} \\ &\text{where } a(z) = a_L(z) = z \Rightarrow \frac{\partial g_{k,i}}{\partial z_{k,i}} = 1\end{aligned} \tag{13}$$

$$\begin{aligned}\frac{\partial \tilde{F}}{\partial x} &= \frac{\partial \tilde{F}}{\partial g_{3,1}} \frac{\partial g_{3,1}}{\partial z_{3,1}} \sum_{i=1}^{n} \frac{\partial z_{3,1}}{\partial g_{2,i}} \frac{\partial g_{2,i}}{\partial z_{2,i}} \sum_{j=1}^{n} \frac{\partial z_{2,i}}{\partial g_{1,j}} \frac{\partial g_{1,j}}{\partial z_{1,j}} \frac{\partial z_{1,j}}{\partial x} \\ &= 1 \times \frac{\partial g_{3,1}}{\partial z_{3,1}} \times \sum_{i=1}^{n} \tilde{w}_{3,i,1} \times \frac{\partial g_{2,i}}{\partial z_{2,i}} \times \sum_{j=1}^{n} \tilde{w}_{2,j,i} \times \frac{\partial g_{1,j}}{\partial z_{1,j}} \tilde{w}_{1,1,j} \\ &\text{where } a(z) = a_R(z) = \begin{cases} z, \text{if } z > 0 \\ 0, \text{otherwise} \end{cases} \Rightarrow \frac{\partial g_{k,i}}{\partial z_{k,i}} = \begin{cases} 1, \text{if } z_{k,i} > 0 \\ 0, \text{otherwise} \end{cases}\end{aligned} \tag{14}$$

$$\begin{aligned}\frac{\partial \tilde{F}}{\partial x} &= \frac{\partial \tilde{F}}{\partial g_{3,1}} \frac{\partial g_{3,1}}{\partial z_{3,1}} \sum_{i=1}^{n} \frac{\partial z_{3,1}}{\partial g_{2,i}} \frac{\partial g_{2,i}}{\partial z_{2,i}} \sum_{j=1}^{n} \frac{\partial z_{2,i}}{\partial g_{1,j}} \frac{\partial g_{1,j}}{\partial z_{1,j}} \frac{\partial z_{1,j}}{\partial x} \\ &= 1 \times \frac{\partial g_{3,1}}{\partial z_{3,1}} \times \sum_{i=1}^{n} \tilde{w}_{3,i,1} \times \frac{\partial g_{2,i}}{\partial z_{2,i}} \times \sum_{j=1}^{n} \tilde{w}_{2,j,i} \times \frac{\partial g_{1,j}}{\partial z_{1,j}} \tilde{w}_{1,1,j} \\ &\text{where } a(z) = a_S(z) = \frac{1}{1+e^{-z}} \Rightarrow \frac{\partial g_{k,i}}{\partial z_{k,i}} = g_{k,i} \times \left(1\text{-}g_{k,i}\right)\end{aligned} \tag{15}$$

3. The Proposed Method

The proposed deep learning method is comprised of two stages: (1) the training stage for estimating the CDF and (2) the performing stage for predicting the corresponding PDF (shown in Figure 1).

In the training stage, practical data can be collected and analysed for generating actual cumulative probabilities. A deep learning model retaining multiple layers (i.e. an input layer, hidden layers and an output layer) is constructed to learn a CDF in accordance with the actual cumulative probabilities (shown in Figure 3). The input layer includes the random variable x, the actual CDF of which is denoted by F, and the output layer contains the estimated CDF of x, denoted by $\tilde{F}$. The CDFs of common probability distributions can be adopted as the activation functions in the hidden layers. For example, the CDFs of ED, ND, LD, and GD [1] (denoted by f_1, f_2, f_3, and f_4, respectively) can be used as the activation functions (shown respectively in Equations (16)-(19), where $\tilde{\lambda}$, $\tilde{\mu}_1$, $\tilde{s}_1$, $\tilde{\mu}_2$, $\tilde{s}_2$, $\tilde{\alpha}$, and $\tilde{\beta}$ are all parameters). In Equation

(19), $\gamma\left(\tilde{\alpha},\tilde{\beta},x\right)$ is the lower incomplete gamma function, and $\Gamma(\tilde{\alpha})$ is the gamma function. Furthermore, the number of neurons with different activation functions can be extended to n, and the CDF can be estimated by Equation (20) in accordance with a weight set (i.e. $\{\tilde{w}_1,\tilde{w}_2,...,\tilde{w}_n\}$). The loss function E of the proposed deep learning model is defined by Equation (21) according to the estimated CDF (i.e. $\tilde{F}$) and the actual CDF (i.e. F) for minimising the estimation error. The gradient descent method is adopted in the setting of parameters. The partial differential of E with respect to each parameter is calculated (Equations (22)-(29)), and the parameter in the (k+1)-th iteration can be updated in accordance with the aforementioned partial differential in the k-th iteration (Equations (30)-(37), where η is the learning rate). In Equation (28), $\psi(\cdot)$ is the digamma function. The number of hidden layers could be increased to estimate a relatively complex CDF.

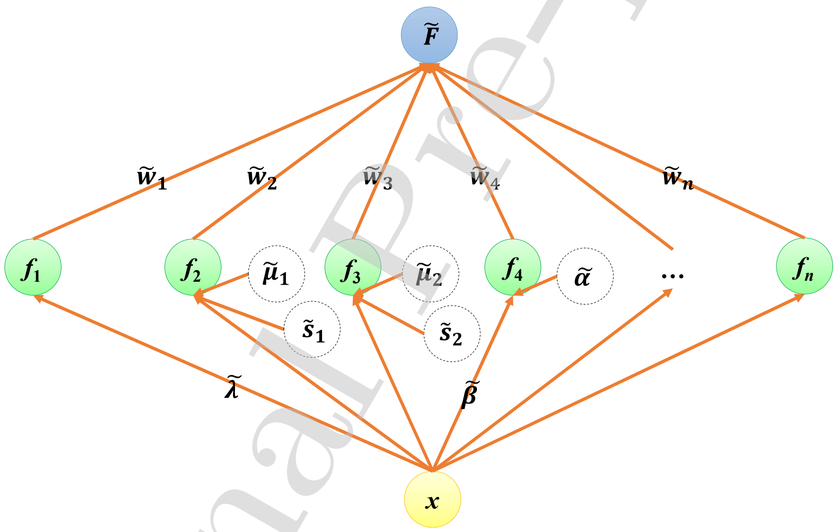


Fig. 3. The structure of the proposed deep learning model.

$$f_1 = 1 - e^{-\tilde{\lambda}x} \tag{16}$$

$$f_2 = \frac{1}{2}\left[1+\operatorname{erf}\left(\frac{x-\tilde{\mu}_1}{\tilde{s}_1\sqrt{2}}\right)\right] \tag{17}$$

$$f_3 = \frac{1}{2}\left[1+\operatorname{erf}\left(\frac{\ln(x)-\tilde{\mu}_2}{\tilde{s}_2\sqrt{2}}\right)\right] \tag{18}$$

$$f_4 = \frac{\gamma\left(\tilde{\alpha},\tilde{\beta},x\right)}{\Gamma(\tilde{\alpha})} \tag{19}$$

$$\tilde{F}=\frac{\sum_{i=1}^{n}\tilde{w}_i f_i}{\sum_{i=1}^{n}\tilde{w}_i} \tag{20}$$

$$E=\frac{1}{2}\left(\tilde{F}-F\right)^2=\frac{1}{2}\sigma^2 \tag{21}$$

$$\frac{\partial E}{\partial \tilde{w}_j}=\frac{\partial E}{\partial \sigma}\frac{\partial \sigma}{\partial \tilde{F}}\frac{\partial \tilde{F}}{\partial \tilde{w}_j}=\sigma\times\frac{\sum_{i=1}^{n}\left[\tilde{w}_i\left(f_j-f_i\right)\right]}{\left(\sum_{i=1}^{n}\tilde{w}_i\right)^2} \tag{22}$$

$$\frac{\partial E}{\partial \tilde{\lambda}}=\frac{\partial E}{\partial \sigma}\frac{\partial \sigma}{\partial \tilde{F}}\frac{\partial \tilde{F}}{\partial f_1}\frac{\partial f_1}{\partial \tilde{\lambda}}=\sigma\times\frac{\tilde{w}_1}{\sum_{i=1}^{n}\tilde{w}_i}\times\left(xe^{-\tilde{\lambda}x}\right) \tag{23}$$

$$\frac{\partial E}{\partial \tilde{\mu}_1}=\frac{\partial E}{\partial \sigma}\frac{\partial \sigma}{\partial \tilde{F}}\frac{\partial \tilde{F}}{\partial f_2}\frac{\partial f_2}{\partial \tilde{\mu}_1}=\sigma\times\frac{\tilde{w}_2}{\sum_{i=1}^{n}\tilde{w}_i}\times\left(\frac{-1}{\sqrt{2\pi\tilde{s}_1^{\,2}}}e^{-\frac{(x-\tilde{\mu}_1)^2}{2\tilde{s}_1^{\,2}}}\right) \tag{24}$$

$$\frac{\partial E}{\partial \tilde{s}_1}=\frac{\partial E}{\partial \sigma}\frac{\partial \sigma}{\partial \tilde{F}}\frac{\partial \tilde{F}}{\partial f_2}\frac{\partial f_2}{\partial \tilde{s}_1}=\sigma\times\frac{\tilde{w}_2}{\sum_{i=1}^{n}\tilde{w}_i}\times\left(\frac{-\left(x-\tilde{\mu}_1\right)}{\sqrt{2\pi\tilde{s}_1^{\,2}}}e^{-\frac{(x-\tilde{\mu}_1)^2}{2\tilde{s}_1^{\,2}}}\right) \tag{25}$$

$$\frac{\partial E}{\partial \tilde{\mu}_2}=\frac{\partial E}{\partial \sigma}\frac{\partial \sigma}{\partial \tilde{F}}\frac{\partial \tilde{F}}{\partial f_3}\frac{\partial f_3}{\partial \tilde{\mu}_2}=\sigma\times\frac{\tilde{w}_3}{\sum_{i=1}^{n}\tilde{w}_i}\times\left(\frac{-1}{\sqrt{2\pi\tilde{s}_2^{\,2}}}e^{-\frac{(\ln(x)-\tilde{\mu}_2)^2}{2\tilde{s}_2^{\,2}}}\right) \tag{26}$$

$$\begin{aligned}\frac{\partial E}{\partial \tilde{s}_2}&=\frac{\partial E}{\partial \sigma}\frac{\partial \sigma}{\partial \tilde{F}}\frac{\partial \tilde{F}}{\partial f_3}\frac{\partial f_3}{\partial \tilde{s}_2}\\&=\sigma\times\frac{\tilde{w}_3}{\sum_{i=1}^{n}\tilde{w}_i}\times\left(\frac{-\left(\ln(x)-\tilde{\mu}_2\right)}{\sqrt{2\pi\tilde{s}_2^{\,2}}}e^{-\frac{(\ln(x)-\tilde{\mu}_2)^2}{2\tilde{s}_2^{\,2}}}\right)\end{aligned} \tag{27}$$

$$\begin{aligned}\frac{\partial E}{\partial \tilde{\alpha}}&=\frac{\partial E}{\partial \sigma}\frac{\partial \sigma}{\partial \tilde{F}}\frac{\partial \tilde{F}}{\partial f_4}\frac{\partial f_4}{\partial \tilde{\alpha}}\\&=\sigma\times\frac{\tilde{w}_4}{\sum_{i=1}^{n}\tilde{w}_i}\\&\times\left[\ln\left(\frac{x}{\tilde{\beta}}\right)\left(\frac{x}{\tilde{\beta}}\right)^{\tilde{\alpha}}e^{-\frac{x}{\tilde{\beta}}}\sum_{l=0}^{\infty}\frac{\left(\frac{x}{\tilde{\beta}}\right)^l}{\Gamma\left(\tilde{\alpha}+l+1\right)}-\left(\frac{x}{\tilde{\beta}}\right)^{\tilde{\alpha}}e^{-\frac{x}{\tilde{\beta}}}\sum_{l=0}^{\infty}\frac{\left(\frac{x}{\tilde{\beta}}\right)^l\psi\left(\tilde{\alpha}+l+1\right)}{\Gamma\left(\tilde{\alpha}+l+1\right)}\right]\end{aligned} \tag{28}$$

$$\frac{\partial E}{\partial \tilde{\beta}} = \frac{\partial E}{\partial \sigma}\frac{\partial \sigma}{\partial \tilde{F}}\frac{\partial \tilde{F}}{\partial f_4}\frac{\partial f_4}{\partial \tilde{\beta}} = \sigma \times \frac{\tilde{w}_4}{\sum_{i=1}^{n}\tilde{w}_i} \times \left[\frac{1}{\tilde{\beta}\Gamma(\tilde{\alpha})}\left(\frac{x}{\tilde{\beta}}\right)^{\tilde{\alpha}} e^{-\frac{x}{\tilde{\beta}}}\right] \tag{29}$$

$$\tilde{w}_j^{(k+1)} = \tilde{w}_j^{(k)} - \eta \frac{\partial E^{(k)}}{\partial \tilde{w}_j^{(k)}} \tag{30}$$

$$\tilde{\lambda}^{(k+1)} = \tilde{\lambda}^{(k)} - \eta \frac{\partial E^{(k)}}{\partial \tilde{\lambda}^{(k)}} \tag{31}$$

$$\tilde{\mu}_1^{(k+1)} = \tilde{\mu}_1^{(k)} - \eta \frac{\partial E^{(k)}}{\partial \tilde{\mu}_1^{(k)}} \tag{32}$$

$$\tilde{s}_1^{(k+1)} = \tilde{s}_1^{(k)} - \eta \frac{\partial E^{(k)}}{\partial \tilde{s}_1^{(k)}} \tag{33}$$

$$\tilde{\mu}_2^{(k+1)} = \tilde{\mu}_2^{(k)} - \eta \frac{\partial E^{(k)}}{\partial \tilde{\mu}_2^{(k)}} \tag{34}$$

$$\tilde{s}_2^{(k+1)} = \tilde{s}_2^{(k)} - \eta \frac{\partial E^{(k)}}{\partial \tilde{s}_2^{(k)}} \tag{35}$$

$$\tilde{\alpha}^{(k+1)} = \tilde{\alpha}^{(k)} - \eta \frac{\partial E^{(k)}}{\partial \tilde{\alpha}^{(k)}} \tag{36}$$

$$\tilde{\beta}^{(k+1)} = \tilde{\beta}^{(k)} - \eta \frac{\partial E^{(k)}}{\partial \tilde{\beta}^{(k)}} \tag{37}$$

In the performing stage, the differential equation of trained deep learning model described by Equation (38) is derived to obtain the estimated PDF (i.e. $\tilde{P}$).

$$\begin{aligned}\tilde{P} &= \frac{\partial \tilde{F}}{\partial x} = \sum_{i=1}^{n}\left(\tilde{w}_i \times \frac{\partial f_i}{\partial x}\right) \\ &= \tilde{w}_1 \times \tilde{\lambda} e^{-\tilde{\lambda} x} + \tilde{w}_2 \times \frac{1}{\sqrt{2\pi \tilde{s}_1^{\,2}}} e^{-\frac{(x-\tilde{\mu}_1)^2}{2\tilde{s}_1^{\,2}}} + \\ &\quad \tilde{w}_3 \times \frac{1}{x\tilde{s}_2\sqrt{2\pi}} e^{-\frac{(\ln(x)-\tilde{\mu}_2)^2}{2\tilde{s}_2^{\,2}}} + \tilde{w}_4 \times \frac{x^{\tilde{\alpha}-1} e^{-\frac{x}{\tilde{\beta}}}}{\tilde{\beta}^{\alpha}\Gamma(\tilde{\alpha})} + \sum_{i=5}^{n}\left(\tilde{w}_i \times \frac{\partial f_i}{\partial x}\right)\end{aligned} \tag{38}$$

4. Practical Experimental Results and Discussions

The experimental environments, experimental results, and relevant comparisons are presented and discussed in the following subsections.

4.1. Experimental Environments

Six test cases, including single ED, single ND, single LD, single GD, and two mixed distributions (MDs) of single ND and single GD with different parameters, were set in the experiments, where the proposed deep learning model for comparison was constructed with f_1, f_2, f_3, and f_4.

(1) Case 1: a single ED with the parameter λ=0.5 was used as the benchmark.

(2) Case 2: a single ND with the parameters σ=1 and μ=0.5 was used as the benchmark.

(3) Case 3: a single LD with the parameters σ=1 and μ=0.5 was used as the benchmark.

(4) Case 4: a single GD with the parameters α=0.1 and β=0.5 was used as the benchmark.

(5) Case 5: a MD of single ND and single GD with the parameters σ=1, μ=0.5, α=0.1 and β=0.5 was used as the benchmark.

(6) Case 6: a MD of single ND and single GD with the parameters σ=1, μ=0.5, α=2 and β=10 was used as the benchmark.

For evaluation of the proposed method, the mean absolute percentage errors (MAPEs) of the estimated CDF (i.e. M_F) is defined as Equation (39) according to $\tilde{F}$ and F, and the MAPE of the estimated PDF (i.e. M_P) is defined as Equation (40) according to $\tilde{P}$ and P.

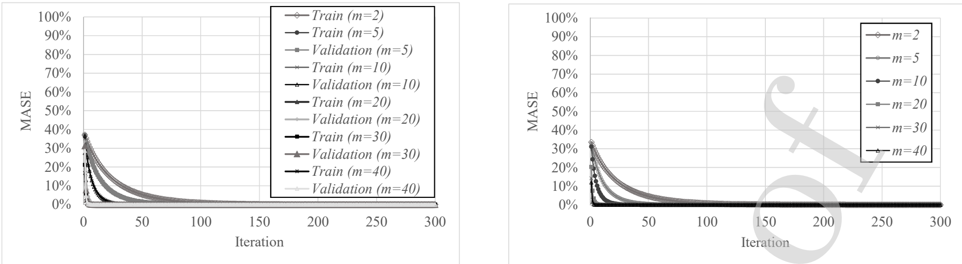
$$M_F = \sum_{k=1}^{m} \frac{\left|F_k - \tilde{F}_k\right|}{F_k} \times \frac{100\%}{m} \tag{39}$$

$$M_P = \sum_{k=1}^{m} \frac{\left|P_k - \tilde{P}_k\right|}{P_k} \times \frac{100\%}{m} \tag{40}$$

In the numerical experiments, single and mixed common probability distributions with the number of sample records denoted by *m* were designed. The values of *m* we set in the experiment include 2, 5, 10, 20, 30, and 40 for accuracy comparisons between sample sizes in the training stage. Furthermore, the *t*-fold cross-validation was considered to evaluate the proposed method and verify the overfitting problems, and the value of *t* is set to be 5. The sample records were randomly distributed in five groups. In the cross-validation evaluation, one group was selected as the validation data, and the other four groups were used as the training data.

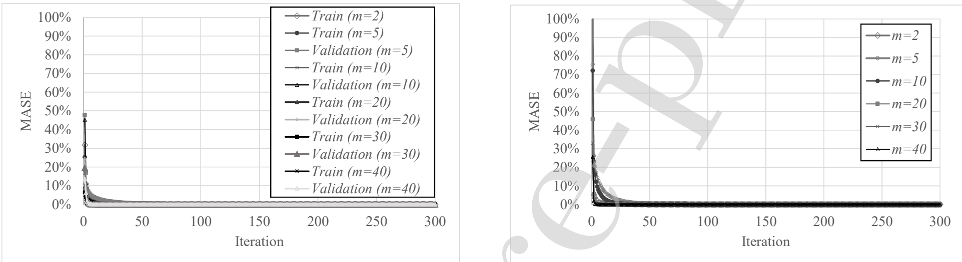
4.2. Experimental Results

Figures 4, 5, 6, 7, 8 and 9 shows the MAPEs of the estimated CDF and PDF in Cases 1, 2, 3, 4, 5 and 6, respectively. The MAPEs converge faster with a larger sample size (i.e. a larger value of *m*) in each case. The values of both M_F and M_P in each case are about 0% when the value of *m* is 40. Furthermore, the MAPEs of the training data are close to the MAPEs of the validation data, so the overfitting problems could not exist in the trained neural network models. Therefore, the proposed method can obtain a precise estimated PDF for each case.



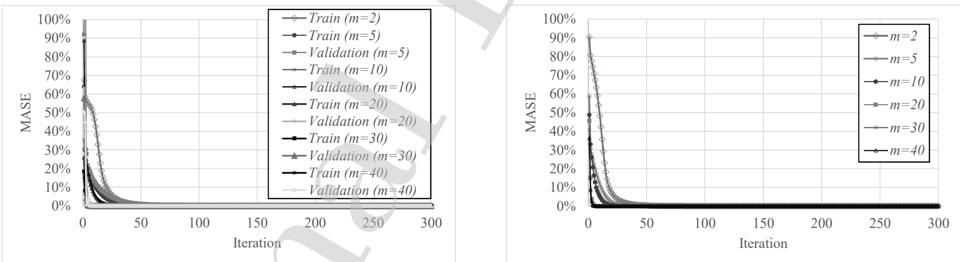
(a). The MAPEs of the estimated CDF (b). The MAPEs of the estimated PDF

Fig. 4. The MAPEs of the estimated CDF and PDF in Case 1 (i.e. a single ED)



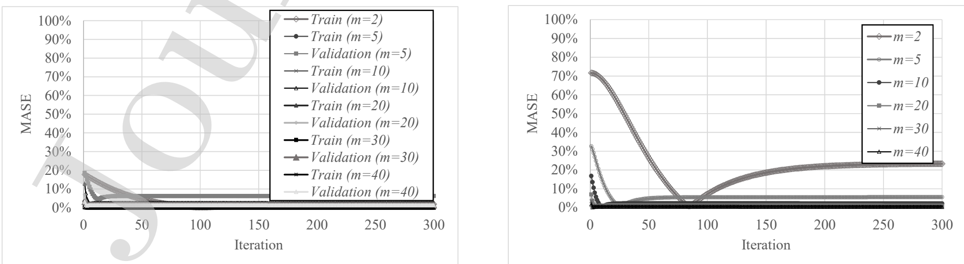
(a). The MAPEs of the estimated CDF (b). The MAPEs of the estimated PDF

Fig. 5. The MAPEs of the estimated CDF and PDF in Case 2 (i.e. a single ND)



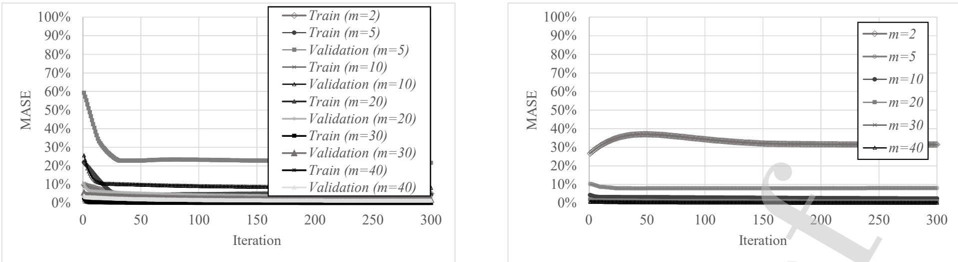
(a). The MAPEs of the estimated CDF (b). The MAPEs of the estimated PDF

Fig. 6. The MAPEs of the estimated CDF and PDF in Case 3 (i.e. a single LD)



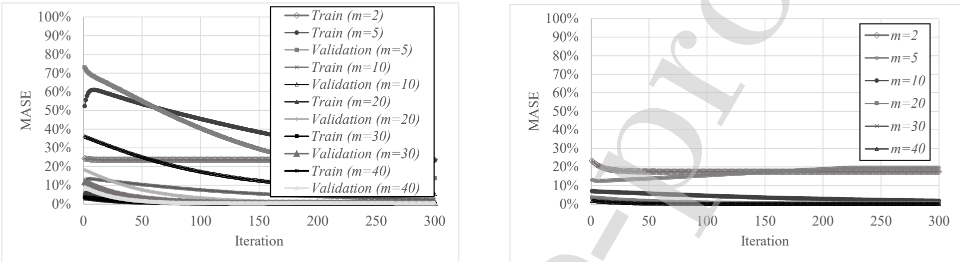
(a). The MAPEs of the estimated CDF (b). The MAPEs of the estimated PDF

Fig. 7. The MAPEs of the estimated CDF and PDF in Case 4 (i.e. a single GD)



(a). The MAPEs of the estimated CDF (b). The MAPEs of the estimated PDF

Fig. 8. The MAPEs of the estimated CDF and PDF in Case 5 (i.e. a MD)



(a). The MAPEs of the estimated CDF (b). The MAPEs of the estimated PDF

Fig. 9. The MAPEs of the estimated CDF and PDF in Case 6 (i.e. a MD)

In addition, it can be found that relatively large errors in complex cases (e.g. Cases 4, 5 and 6) are generated when the sample sizes are too small (e.g. *m*=2 and *m*=5). Therefore, the sample size should be given more than ten for precisely estimating the PDF of practical data.

4.3. Comparisons and Discussions

For comparisons of estimated CDF and PDF, the linear-function-based neural network (shown in Equation (13)), the ReLU-function-based neural network (shown in Equation (14)) and the sigmoid-function-based neural network (shown in Equation (15)) were selected. In this subsection, the value of m was set as 40, and the number of iterations was set as 300. Because the parameter x is missing after the differentiation for the trained neural network models (i.e. the estimated PDF based on the trained neural network), the MAPEs are large in the linear-function-based neural network and the ReLU-function-based neural network. Although the MAPEs of the sigmoid-function-based neural network are smaller, it requires a higher number of iterations. The proposed method can precisely estimate the CDF and PDF both with MAPEs no more than 0.4% for $m = 40$.

TABLE 1

MAPES OF THE ESTIMATED CDFS AND PDFS WITH DIFFERENT ACTIVATION FUNCTIONS

Case	Probability Function	Linear	ReLU	Sigmoid	The proposed method
Case 1: ED (λ=0.5)	CDF	19.70%	17.15%	72.77%	0.00%
	PDF	61.72%	57.32%	88.35%	0.00%
Case 2: ND (σ=1 , μ=0.5)	CDF	10.90%	6.09%	25.19%	0.00%
	PDF	1662.91%	1061.08%	86.44%	0.00%
Case 3: LD (σ=1 , μ=0.5)	CDF	137.85%	112.23%	576.64%	0.00%
	PDF	69.19%	63.22%	86.34%	0.00%
Case 4: GD (α=0.1 , β=0.5)	CDF	9.83%	3.47%	16.22%	0.03%
	PDF	427.27%	152.93%	91.28%	0.40%
Case 5: ND + GD (σ=1 , μ=0.5 , α=0.1 , β=0.5)	CDF	5.03%	2.51%	10.63%	0.07%
	PDF	1831.77%	1171.96%	85.40%	0.30%
Case 6: ND + GD (σ=1 , μ=0.5 , α=2 , β=10)	CDF	10.33%	4.79%	25.91%	0.02%
	PDF	192.96%	117.64%	88.31%	0.06%

5. CONCLUSIONS AND FUTURE WORK

This study has developed a deep learning method to learn the potentially complex and irregular probability distributions, and the formulae yielded from the differentiation derivation for trained deep learning model can be used to estimate the corresponding PDF. The experimental results have demonstrated that the proposed method can precisely estimate the CDFs and PDFs for single and mixed probability distributions.

The further studies could be conducted by assessing the proposed model with practical data. Furthermore, the accurate CDF and PDF of the real-data probability distribution can be produced to improve further analyses with game theory or queueing theory. Various types of probability-based applications (e.g. energy applications [25, 26], security applications [27, 28], etc.) can be designed based on the proposed method to illustrate the probability functions which are fitting the distribution of practical data for improvement.

ACKNOWLEDGMENT

This work was partially supported by the National Natural Science Foundation of China (Nos. 61906043, 61877010, 11501114 and 11901100), Fujian Natural Science Funds (No. 2019J01243), and Fuzhou University (Nos. 510730/XRC-18075, 510809/GXRC-19037, 510649/XRC-18049 and 510650/XRC-18050).

References

[1] King, M.: 'Probability density function' in King, M. (Ed.): 'Statistics for process control engineers: a practical approach', John Wiley & Sons Ltd., New York, 2017.

[2] LeCun, Y., Bengio, Y. Hinton, G.E.: 'Deep learning', *Nature*, 2015, 521, pp. 436-444, doi: 10.1038/nature14539

[3] Schmidhuber, J.: 'Deep learning in neural networks: An overview', *Neural Networks*, 2015, 61, pp. 85-117, doi: 10.1016/j.neunet.2014.09.003

[4] Wongsuphasawat, K., Smilkov, D., Wexler, J., Wilson, J., Mané, D., Fritz, D., Krishnan, D., Viégas, F.B., Wattenberg, M.: 'Visualizing dataflow graphs of deep learning models in TensorFlow', *IEEE Transactions on Visualization and Computer Graphics*, 2018, 24(1), pp. 1-12, doi: 10.1109/TVCG.2017.2744878

[5] Gao, X.H., Li, W., Loomes, M., Wang, L.Y.: 'A fused deep learning architecture for viewpoint classification of echocardiography', *Information Fusion*, 2017, 36, pp. 103-113, doi: 10.1016/j.inffus.2016.11.007

[6] Li, W.F., Su, H., Yang, X., Yan, X.H.: 'Subsurface temperature estimation from remote sensing data using a clustering-neural network method', *Remote Sensing of Environment*, 2019, 229, pp. 213-222, doi: 10.1016/j.rse.2019.04.009

[7] Chen, C.H., Hwang, F.J., Kung, H.Y.: 'Travel time prediction system based on data clustering for waste collection vehicles', *IEICE Transactions on Information and Systems*, 2019, E102D(7), pp. 1374-1383, doi: 10.1587/transinf.2018EDP7299

[8] Zhang, J.Y., Guo, Y., Hu, X., Li, R.Z.: 'Design and implementation of deep neural network for edge computing', *IEICE Transactions on Information and Systems*, 2018, E101D(8), pp. 1982-1996, doi: 10.1587/transinf.2018EDP7044

[9] Lee, S., Jeong, Y., Son, S., Lee, B.: 'A self-predictable crop yield platform (SCYP) based on crop diseases using deep learning', *Sustainability*, 2019, 11(13), article no. 3637, doi: 10.3390/su11133637

[10] Wang, J., Sanchez, J.A., Iturrioz, J.A., Ayesta, I.: 'Geometrical defect detection in the wire electrical discharge machining of fir-tree slots using deep learning techniques', *Applied Sciences*, 2019, 9(1), article no. 90, doi: 10.3390/app9010090

[11] Lee, H., Lee, J., Shin, M.: 'Using wearable ECG/PPG sensors for driver drowsiness detection based on distinguishable pattern of recurrence plots', *Electronics*, 2019, 8(2), article no. 192, doi: 10.3390/electronics8020192

[12] Liu, W., Huang, X., Cao, G., Zhang, J., Song, G., Yang, L.: 'Joint learning of NNeXtVLAD, CNN and context gating for micro-video venue classification', *IEEE Access*, 2019, 7, pp. 77091-77099, doi: 10.1109/ACCESS.2019.2922430

[13] Fan, S., Yu, H., Lu, D., Jiao, S., Xu, W., Liu, F., Liu, Z.: 'CSCC: convolution split compression calculation algorithm for deep neural network', *IEEE Access*, 2019, 7, pp. 71607-71615, doi: 10.1109/ACCESS.2019.2919554

[14] Wu, W., Liu, J., Wang, H., Tang, F., Xian, M.: 'PPolyNets: achieving high prediction accuracy and efficiency with parametric polynomial activations', *IEEE Access*, 2018, 6, pp. 72814-72823, doi: 10.1109/ACCESS.2018.2882407

[15] Ke, X., Zou, J., Niu, Y.: 'End-to-end automatic image annotation based on deep CNN and multi-label data augmentation', *IEEE Transactions on Multimedia*, 2019, 21(8), pp. 2093-2106, doi: 10.1109/TMM.2019.2895511

[16] Wu, L., Chen, C.H., Zhang, Q.: 'A mobile positioning method based on deep learning techniques', *Electronics*, 2019, 8(1), article no. 59, doi: 10.3390/electronics8010059

[17] Ke, X., Shi, L., Guo, W., Chen, D.: 'Multi-dimensional traffic congestion detection based on fusion of visual features and convolutional neural network', *IEEE Transactions on Intelligent Transportation Systems*, 2019, 20(6), pp. 2157-2170, doi: 10.1109/TITS.2018.2864612

[18] Chen, C.H.: 'An arrival time prediction method for bus system', *IEEE Internet of Things Journal*, 2018, 5(5), pp. 4231-4232, doi: 10.1109/JIOT.2018.2863555

[19] Dai, Y., Guo, W., Chen, X., Zhang, Z.: 'Relation classification via LSTMs based on sequence and tree structure', *IEEE Access*, 2018, 6, pp. 64927-64937, doi: 10.1109/ACCESS.2018.2877934

[20] Wang, S., Cai, J., Lin, Q., Guo, W.: 'An overview of unsupervised deep feature representation for text categorization', *IEEE Transactions on Computational Social Systems*, 2019, 6(3), pp. 504-517, doi: 10.1109/TCSS.2019.2910599

[21] Rumelhart, D.E., Hinton, G.E., Williams, R.J.: 'Learning representations by back-propagating errors', *Nature*, 1986, 323, pp. 533–536, doi: 10.1038/323533a0

[22] Cheng, R., Chen, D., Cheng, B., Zheng, S.: 'Intelligent driving methods based on expert knowledge and online optimization for high-speed trains', *Expert Systems with Applications*, 2017, 87, pp. 228-239, doi: 10.1016/j.eswa.2017.06.006

[23] Guo, H., Li, S., Li, B., Ma. Y., Ren, X.: 'A New Learning Automata-Based Pruning Method to Train Deep Neural Networks', *IEEE Internet of Things Journal*, 2018, 5(5), pp. 3263-3269, doi: 10.1109/JIOT.2017.2711426

[24] Torabi, M., Hosseini, M.M.: 'A new descent algorithm using the three-step discretization method for solving unconstrained optimization problems', *Mathematics*, 2018, 6(4), article no. 63. doi: 10.3390/math6040063

[25] Lin, Y.B., Wang, L.C., Lin, P.: 'SES: a novel yet simple energy saving scheme for small cells', *IEEE Transactions on Vehicular Technology*, 2017, 66(9), pp. 8347-8356, doi: 10.1109/TVT.2017.2679027

[26] Chang, C.T., Lee, H.C.: 'Taiwan's renewable energy strategy and energy-intensive industrial policy', *Renewable & Sustainable Energy Reviews*, 2016, 64, pp. 456-465, doi: 10.1016/j.rser.2016.06.052

[27] Chen, C.L., Chiang, M.L., Lin, W.C., Li, D.K.: 'A novel lottery protocol for mobile environments', *Computers & Electrical Engineering*, 2016, 49, pp. 146-160, doi: 10.1016/j.compeleceng.2015.07.016

[28] Tsai, J.F., Lin, M.H., Wang, P.C.: 'An efficient deterministic approach to optimal design of reliable networks', *IEEE Transactions on Reliability*, 2018, 67(2), pp. 598-608, doi: 10.1109/TR.2018.2821669

Chi-Hua Chen serves as a distinguished professor for the College of Mathematics and Computer Science at Fuzhou University in China. He received his Ph.D. degree from the Department of Information Management and Finance of National Chiao Tung University (NCTU) in 2013. He has published over 270 journal articles, conference articles, and patents. His recent research interests include Internet of things, big data, deep learning, cellular networks, data mining and intelligent transportation systems.

Fangying Song is a professor for the College of Mathematics and Computer Science of Fuzhou University. He received his Ph.D. degree from the School of Mathematical Sciences of Xiamen University in 2014. He worked as postdoctoral at Brown University for four years. His recent research interests include machine learning for partial differential equations (PDEs), fractional reaction diffusion equations, multiphase flows and turbulent flows.

Feng-Jang Hwang is the senior lecturer of operational research at the School of Mathematical and Physical Sciences, Transport Research Centre, University of Technology Sydney. He received his Ph.D. degree in Information Management from NCTU in 2011. His research interests centre around operations research, industrial optimization, management information science, data science, intelligent transportation and logistics, and computational intelligence.

Ling Wu received her Ph.D. degree in the School of Economics and Management of Fuzhou University. She is an assistant research fellow for the College of Mathematics and Computer Science of Fuzhou University. Her recent research interests include social networks, grey system, and machine learning.