TẠP CHÍ KHOA HỌC & CÔNG NGHỆ ĐẠI HỌC DUY TÂN
DTU Journal of Science and Technology 02(57) (2023) 3-9

TRƯỜNG ĐẠI HỌC DUY TÂN
DUYTAN UNIVERSITY

Training artificial neural network regression based on the generalized delta rule: a case study in modeling the compressive strength of concrete

Huấn luyện mạng nơ-ron thần kinh nhân tạo dùng cho phân tích hồi quy dựa trên quy tắc delta khái quát: ứng dụng cho mô phỏng cường độ chịu nén của bê tông

Nhat-Duc Hoang[a,b*], Quoc-Lam Nguyen[b], Quang-Nhat Pham[b]
Hoàng Nhật Đức[a,b*], Nguyễn Quốc Lâm[b], Phạm Quang Nhật[b]

[a]*Institute of Research and Development, Duy Tan University, Da Nang, 550000, Vietnam*
[a]*Viện Nghiên cứu và Phát triển Công nghệ Cao Trường Đại học Duy Tân, Đà Nẵng, Việt Nam*
[a]*Faculty of Civil Engineering, Duy Tan University, Da Nang, 550000, Vietnam*
[b]*Khoa Xây dựng, Trường Đại học Duy Tân, Đà Nẵng, Việt Nam*

(Ngày nhận bài: 19/12/2022, ngày phản biện xong: 06/3/2023, ngày chấp nhận đăng: 14/3/2023)

Abstract

This paper presents the algorithms for training an artificial neural network (ANN) for regression analysis; the algorithm is based on the generalized delta rule. The training method of a simple neuron model and an ANN model are presented and generalized. The models are then programed in Visual C# .NET and applied to predict the compressive strength of concrete mixes. Three datasets, collected from the literature, are used to demonstrate the applications of the models.

Keywords: Artificial neural network; Regression analysis; Concrete strength; Generalized delta rule.

Tóm tắt

Bài báo trình bày các thuật toán dùng để huấn luyện một mạng nơ-ron thần kinh nhân tạo dùng cho phân tích hồi quy. Các thuật toán này được dựa trên quy tắc delta khái quát. Cách thức huấn luyện một nơ-ron thần kinh và một mạng nơ-ron thần kinh nhân tạo được trình bày và khái quát hóa. Các mô hình trên được lập trình với ngôn ngữ Visual C# và được ứng dụng để dự báo cường độ chịu nén của các mẫu bê tông. Bài báo sử dụng ba bộ dữ liệu, được thu thập từ các tài liệu tham khảo, để minh họa cho tính ứng dụng của các mô hình.

Từ khóa: Mạng nơ-ron thần kinh nhân tạo; Phân tích hồi quy; Cường độ chịu nén của bê tông; Quy tắc delta khái quát.

1. Introduction

The goal of the training phase of an ANN model is to minimize the output errors on a dataset via the adaptation of the network weights (Bullinaria, 2004). For regression analysis problems, the squared error loss (SEL) function is often used (Aggarwal, 2018). Using this function, an algorithm based on a stochastic gradient-descent can be formulated to train the neural network. This algorithm

[*]*Corresponding Author:* Hoang Nhat Duc, Institute of Research and Development, Duy Tan University, Da Nang, 550000, Vietnam; Faculty of Civil Engineering, Duy Tan University, Da Nang, 550000, Vietnam.
Email: hoangnhatduc@duytan.edu.vn

basically performs gradient-descent updates with respect to a set of randomly initialized the network weights. Based on the method of error back-propagation and the chain rule (which is used to express and compute the derivative of the composition of differentiable functions), each element in the network's weight matrices can be updated to fit a collected dataset at hand. This method can be neatly summarized and presented by the generalized delta rule (Kim, 2017), which revises a weight element as follows:

$$w = w + \alpha \times \Delta \times z \quad (1)$$

where $\Delta = \varphi'(v_j^{(i)}) \times e_j^{(i)}$; α denotes the learning rate; $e_j^{(i)}$ is the error of the neuron j at the layer i. $\varphi'(v_j^{(i)})$ is the derivative of the activation function with respect to the weighted sum of the inputs $v_j^{(i)}$.

In addition, the generalized delta rule (Kim, 2017) greatly eases the formulation of the error back-propagation by employing the following rule: The error of the neuron j at the layer i ($e_j^{(i)}$) is computed by the errors of the previous neurons multiplied by their connecting weights.

This paper first formulates the training algorithm via the method of error back-propagation and the chain rule. Subsequently, the equivalent training algorithm based on the generalized delta rule is presented. The current work focuses on the application of ANN model in regression analysis because this task is ubiquitous in civil engineering. Regression models are capable of analyzing past data records and providing prediction results that help the decision-making processes in many phases of a construction project such as planning (Lishner & Shtub, 2022), design (Gandomi, Yun, & Alavi, 2013), and management (Cheng, Hoang, & Wu, 2015). The models, based on the formulation of the generalized delta rule, are constructed in Visual C# .NET. Those models are then applied to predict the compressive strength of concrete. It is because it is a crucial task in construction engineering that helps reduce time and cost in concrete mix design.

2. Training one neuron for regression analysis

The model of one neuron for performing regression analysis is demonstrated in Fig. 2.1. Herein, for the ease of illustration, a neuron with two inputs (x_1 and x_2) is presented in the figure. The input data is represented as a vector $X = [x_1, x_2, \ldots, D_x]$ where D_x is the number of factors (e.g. the quantity of cement or curing age) that influence the target output (e.g. the compressive strength of concrete). The matrix W is the weight of the neuron. y is the modeled output.

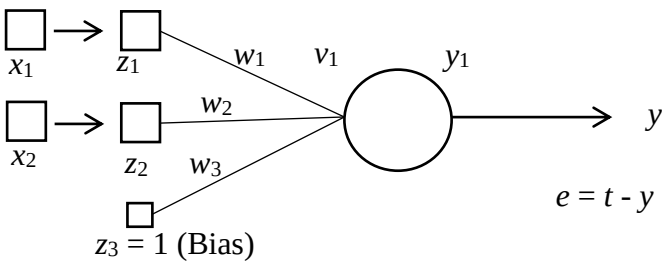

Fig. 2.1. The model of one neuron used for regression analysis

Given the input vector X, the input of the neuron is constructed by adding an element of 1 as the bias as follows:

$$Z = [z_1, z_2, z_3] = [x_1, x_2, 1]^T \quad (2)$$

Accordingly, the weighted input of the neuron is given by:

$$v_1 = z_1 \times w_1 + z_2 \times w_2 + z_3 \times w_3 = \sum_{d=1}^{D=3} x_d \times w_d \quad (3)$$

where $D = D_x + 1$.

Considering a simple linear transformation between the input and output of the neuron, the output of a neuron is simply computed as $y_1 = \varphi(v_1) = v_1$, where $\varphi()$ denotes an activation function. Hence, the final output of the neuron is denoted as $y = y_1$. Given the loss function L() defined in Eq. (1), the derivatives of L with respect to each element of the weight matrix W are given by:

$$\frac{\partial L}{\partial w_1} = \frac{\partial L}{\partial y}\frac{\partial y}{\partial y_1}\frac{\partial y_1}{\partial v_1}\frac{\partial y_1}{\partial w_1} = -(t-y)\times 1\times 1\times x_1 = -(t-y)\times x_1 = -e\times x_1 \quad (4)$$

$$\frac{\partial L}{\partial w_2} = \frac{\partial L}{\partial y}\frac{\partial y}{\partial y_1}\frac{\partial y_1}{\partial v_1}\frac{\partial y_1}{\partial w_2} = -(t-y)\times 1\times 1\times x_2 = -(t-y)\times x_2 = -e\times x_2 \quad (5)$$

$$\frac{\partial L}{\partial w_3} = \frac{\partial L}{\partial y}\frac{\partial y}{\partial y_1}\frac{\partial y_1}{\partial v_1}\frac{\partial y_1}{\partial w_3} = -(t-y)\times 1\times 1\times x_3 = -(t-y)\times 1 = -e\times 1 \quad (6)$$

In general, the weight of the neuron model is revised according to the chain rule as follows:

$$w_d = w_d - \alpha\frac{\partial L}{\partial w_d} = w_d + \alpha\times(t-y)\times x_d = w_d + \alpha\times e\times x_d \quad (7)$$

Applying the generalized delta rule, we first compute $\Delta = \varphi'(v)\times e$. Subsequently, the weight element can be adapted as follows:

$$w_d = w_d + \alpha\times\Delta\times z_d \quad (8)$$

Since $\varphi'(v) = 1$ (because $\varphi(v) = v$), we have $\Delta = 1\times e = e$. Thus, the updating equation based on the generalized delta rule can be stated as follows: $w_d = w_d + \alpha\times e\times z_d$. This equation is identical to Eq. (7), which is derived from the chain rule.

3. Training an Artificial Neural Network consisting of one hidden layer

The model of an ANN used for regression analysis is illustrated in Fig. 2.2. Herein, the network consists of three layers: input, hidden, and output layer. For the ease of illustration, a neural network with two inputs (x_1 and x_2) and two hidden units is presented in the figure. There are two weight matrices in this network: $W^{(1)}$ (which is the weights connecting the input to the hidden layer) and $W^{(2)}$ (which connects the hidden to the output layer). The training phase aims at adapting these two weight matrices to fit the collected dataset at hand. Herein, the sigmoid function is used as the activation function $\varphi()$ in the hidden layer.

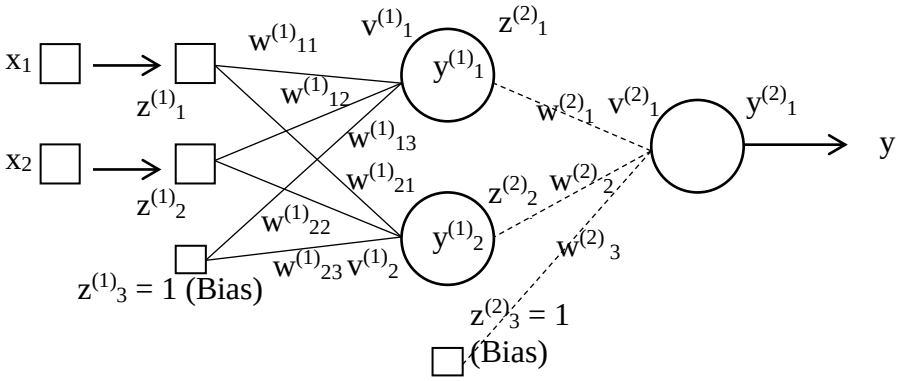

Fig. 2.2. The model of an ANN used for regression analysis

The two weight matrices in the network are given by:

$$W^{(1)} = \begin{bmatrix} w_{11}^{(1)} & w_{12}^{(1)} & w_{13}^{(1)} \\ w_{21}^{(1)} & w_{22}^{(1)} & w_{23}^{(1)} \end{bmatrix} \quad \text{and } W^{(2)} = \begin{bmatrix} w_1^{(2)} & w_2^{(2)} & w_3^{(2)} \end{bmatrix} \tag{9}$$

Given an input signal X, the input $Z^{(1)}$, weighted input $V^{(1)}$, and output $Y^{(1)}$ of the 1st layer are presented as follows:

$$Z^{(1)} = [z_1^{(1)}, z_2^{(1)}, z_3^{(1)}] = [x_1, x_2, 1]^T \tag{10}$$

$$V^{(1)} = [v_1^{(1)}, v_2^{(1)}] \tag{11}$$

where $v_1^{(1)} = z_1^{(1)} \times w_{11}^{(1)} + z_2^{(1)} \times w_{12}^{(1)} + z_3^{(1)} \times w_{13}^{(1)} = \sum_{d=1}^{D} z_d^{(1)} \times w_{1d}^{(1)}$

and $v_2^{(1)} = z_1^{(1)} \times w_{21}^{(1)} + z_2^{(1)} \times w_{22}^{(1)} + z_3^{(1)} \times w_{23}^{(1)} = \sum_{d=1}^{D} z_d^{(1)} \times w_{2d}^{(1)}$

$$Y^{(1)} = [y_1^{(1)}, y_2^{(1)}] \tag{12}$$

where $y_1^{(1)} = \varphi(v_1^{(1)}) \qquad y_2^{(1)} = \varphi(v_2^{(1)})$

The input $Z^{(2)}$, weighted input $V^{(2)}$, and output $Y^{(2)}$ of the 2nd layer are given by:

$$Z^{(2)} = [z_1^{(2)}, z_2^{(2)}, z_3^{(2)}] = [z_1^{(2)}, z_2^{(2)}, 1] \tag{13}$$

where $z_1^{(2)} = y_1^{(1)} \qquad z_2^{(2)} = y_2^{(1)} \qquad z_3^{(2)} = 1$

$$V^{(2)} = [v_1^{(2)}]$$

where $v_1^{(2)} = z_1^{(2)} \times w_1^{(2)} + z_2^{(2)} \times w_2^{(1)} + z_3^{(2)} \times w_3^{(1)} = \sum_{u=1}^{U} z_u^{(2)} \times w_u^{(2)}$

$$Y^{(2)} = [y_1^{(2)}] \tag{14}$$

where $y_1^{(2)} = v_1^{(2)},$

Finally, the output of the ANN is given by: $y = y_1^{(2)}$. Accordingly, the weight matrix $W^{(2)}$ is updated according to the chain rule as follows:

$$\frac{\partial L}{\partial w_1^{(2)}} = \frac{\partial L}{\partial y}\frac{\partial y}{\partial y_1^{(2)}}\frac{\partial y_1^{(2)}}{\partial v_1^{(2)}}\frac{\partial v_1^{(2)}}{\partial w_1^{(2)}} = -(t-y)\times 1\times 1\times z_1^{(2)} = -(t-y)\times z_1^{(2)} = -e\times y_1^{(1)} \tag{15}$$

$$\frac{\partial L}{\partial w_2^{(2)}} = \frac{\partial L}{\partial y}\frac{\partial y}{\partial y_1^{(2)}}\frac{\partial y_1^{(2)}}{\partial v_1^{(2)}}\frac{\partial v_1^{(2)}}{\partial w_2^{(2)}} = -(t-y)\times 1\times 1\times z_2^{(2)} = -(t-y)\times z_2^{(2)} = -e\times y_2^{(1)} \tag{16}$$

$$\frac{\partial L}{\partial w_3^{(2)}} = \frac{\partial L}{\partial y}\frac{\partial y}{\partial y_1^{(2)}}\frac{\partial y_1^{(2)}}{\partial v_1^{(2)}}\frac{\partial v_1^{(2)}}{\partial w_3^{(2)}} = -(t-y)\times 1\times 1\times z_3^{(2)} = -(t-y)\times z_3^{(2)} = -e\times 1 \tag{17}$$

$$\frac{\partial L}{\partial w_u^{(2)}} = \frac{\partial L}{\partial y}\frac{\partial y}{\partial y_1}\frac{\partial y_1}{\partial v_1}\frac{\partial y_1}{\partial w_u^{(2)}} = -e\times z_u^{(2)} \tag{18}$$

$$w_u^{(2)} = w_u^{(2)} - \alpha \times \frac{\partial L}{\partial w_u^{(2)}} = w_u^{(2)} + \alpha \times e \times z_u^{(2)} \tag{19}$$

The generalized delta rule for this layer is presented as follows:

Let $\Delta^{(2)} = \varphi^{(2)}{}'(v^{(2)}) \times e^{(2)}$ (20)

$e^{(2)} = e = t - y$ (21)

$\varphi^{(2)}{}'(v^{(2)}) = 1$ because $y_1^{(2)} = v_1^{(2)}$

$w_u^{(2)} = w_u^{(2)} + \alpha \times \Delta^{(2)} \times z_u^{(2)} = w_u^{(2)} + \alpha \times e \times z_u^{(2)}$ (22)

The weight matrix $W^{(1)}$ is updated based on the chain rule as follows:

$$\frac{\partial L}{\partial w_{11}^{(1)}} = \frac{\partial L}{\partial y}\frac{\partial y}{\partial y_1^{(2)}}\frac{\partial y_1^{(2)}}{\partial v_1^{(2)}}\frac{\partial v_1^{(2)}}{\partial z_1^{(2)}}\frac{\partial z_1^{(2)}}{\partial y_1^{(1)}}\frac{\partial y_1^{(1)}}{\partial v_1^{(1)}}\frac{\partial v_1^{(1)}}{\partial w_{11}^{(1)}} = -(t-y)\times 1\times 1\times w_1^{(2)}\times 1\times [y_1^{(1)}\times(1-y_1^{(1)})]\times z_1^{(1)}$$

$$= -(t-y)\times 1\times 1\times w_1^{(2)}\times 1\times [y_1^{(1)}\times(1-y_1^{(1)})]\times x_1$$

$$= -e\times w_1^{(2)}\times [y_1^{(1)}\times(1-y_1^{(1)})]\times x_1 \quad (23)$$

Because the sigmoid function is used as the activation function $\varphi()$ in the hidden layer, we have $\frac{\partial y_1^{(1)}}{\partial v_1^{(1)}} = y_1^{(1)} \times (1 - y_1^{(1)})$.

$$\frac{\partial L}{\partial w_{12}^{(1)}} = -e\times w_1^{(2)}\times [y_1^{(1)}\times(1-y_1^{(1)})]\times x_2 \quad (24)$$

$$\frac{\partial L}{\partial w_{13}^{(1)}} = -e\times w_1^{(2)}\times [y_1^{(1)}\times(1-y_1^{(1)})]\times 1 \quad (25)$$

Similarly,

$$\frac{\partial L}{\partial w_{21}^{(1)}} = -e\times w_2^{(2)}\times [y_2^{(1)}\times(1-y_2^{(1)})]\times x_1 \quad (26)$$

$$\frac{\partial L}{\partial w_{22}^{(1)}} = -e\times w_2^{(2)}\times [y_2^{(1)}\times(1-y_2^{(1)})]\times x_2 \quad (27)$$

$$\frac{\partial L}{\partial w_{23}^{(1)}} = -e\times w_2^{(2)}\times [y_2^{(1)}\times(1-y_2^{(1)})]\times 1 \quad (28)$$

In summary, we can state that:

$$\frac{\partial L}{\partial W^{(1)}} = \begin{bmatrix} -e\times w_1^{(2)}\times [y_1^{(1)}\times(1-y_1^{(1)})]\times x_1, & -e\times w_1^{(2)}\times [y_1^{(1)}\times(1-y_1^{(1)})]\times x_2, & -e\times w_1^{(2)}\times [y_1^{(1)}\times(1-y_1^{(1)})]\times 1 \\ -e\times w_2^{(2)}\times [y_2^{(1)}\times(1-y_2^{(1)})]\times x_1, & -e\times w_2^{(2)}\times [y_2^{(1)}\times(1-y_2^{(1)})]\times x_2, & -e\times w_2^{(2)}\times [y_2^{(1)}\times(1-y_2^{(1)})]\times 1 \end{bmatrix} \quad (29)$$

Thus, $w_{11}^{(1)} = w_{11}^{(1)} + \alpha \times e \times w_1^{(2)} \times [y_1^{(1)} \times (1 - y_1^{(1)})] \times x_1$ (30)

In general, $w_{uv}^{(1)} = w_{uv}^{(2)} - \alpha \times \frac{\partial L}{\partial w_{uv}^{(1)}}$ (31) and $\frac{\partial L}{\partial w_{uv}^{(1)}} = -e \times w_u^{(2)} \times [y_u^{(1)} \times (1 - y_u^{(1)})] \times z_v^{(1)}$ (32)

The generalized delta rule for revising $W^{(1)}$ is presented as follows:

Let $\Delta_1^{(1)} = \varphi^{(1)}{}'(v_1^{(1)}) \times e_1^{(1)}$ and $\Delta_2^{(1)} = \varphi^{(1)}{}'(v_2^{(1)}) \times e_2^{(1)}$ (33)

where $e_1^{(1)} = e^{(2)} \times w_1^{(2)}$ and $e_2^{(1)} = e^{(2)} \times w_2^{(2)}$

The $\Delta_1^{(1)}$ and $\Delta_2^{(1)}$ of the neuron 1 and 2 are given by:

$$\Delta_1^{(1)} = [y_1^{(1)} \times (1 - y_1^{(1)})] \times e_1^{(1)} = [y_1^{(1)} \times (1 - y_1^{(1)})] \times e^{(2)} \times w_1^{(2)} = [y_1^{(1)} \times (1 - y_1^{(1)})] \times e \times w_1^{(2)} \quad (34)$$

$$\Delta_2^{(1)} = [y_2^{(1)} \times (1 - y_2^{(1)})] \times e_2^{(1)} = [y_2^{(1)} \times (1 - y_2^{(1)})] \times e^{(2)} \times w_2^{(2)} = [y_2^{(1)} \times (1 - y_2^{(1)})] \times e \times w_2^{(2)} \quad (35)$$

The 1st element of this weight matrix is updated as follows:

$$w_{11}^{(1)} = w_{11}^{(1)} + \alpha \times \Delta_1^{(1)} \times z_1^{(1)} = w_{11}^{(1)} + \alpha \times [y_1^{(1)} \times (1 - y_1^{(1)})] \times e \times w_1^{(2)} \times z_1^{(1)} \quad (36)$$

$$w_{11}^{(1)} = w_{11}^{(1)} + \alpha \times [y_1^{(1)} \times (1 - y_1^{(1)})] \times e \times w_1^{(2)} \times x_1$$

In general, the equation used for updating an element at u^{th} row and v^{th} column of $W^{(1)}$ is as follows:

$$w_{uv}^{(1)} = w_{uv}^{(1)} + \alpha \times [y_u^{(1)} \times (1 - y_u^{(1)})] \times e \times w_u^{(2)} \times z_v^{(1)} \quad (37)$$

As can be observed, the Eq. (37) derived from the generalized delta rule is similar to the Eq. (31) and (32) derived from the chain rule.

4. Model application

In this section, the regression models, which are presented in the previous section, are used to predict the compressive strength of concrete in three datasets. These datasets are collected from published papers. The general information of the datasets is as follows:

Dataset 1 (Farooq et al., 2021): The compressive strength of self-compacting concrete modified with fly ash is modeled. The input factors include the contents of the cement, water–binder ratio, coarse aggregate, fine aggregate, fly ash, and superplasticizer. There are 300 samples in this dataset.

Dataset 2 (Pazouki, Golafshani, & Behnood, 2022): The compressive strength of self-compacting concrete containing class F fly ash are the dependent variable. The concrete age and the contents of mixes are used as input variables. There are 327 samples in this dataset.

Dataset 3 (Hoang, 2022): The compressive strength of rice husk ash blended mixes are the dependent variable. The input variables consider the concrete age and the contents of mixes. There are 527 samples in this dataset.

Table 1. The prediction results of the models

Dataset	Phase	Indices	One neuron	ANN
1	Training	RMSE	5.04	4.34
		MAPE (%)	12.55	10.66
		R^2	0.88	0.92
	Testing	RMSE	5.64	5.33
		MAPE (%)	13.41	13.50
		R^2	0.82	0.88
2	Training	RMSE	13.00	8.94
		MAPE (%)	48.57	28.01
		R^2	0.07	0.73
	Testing	RMSE	12.94	10.37
		MAPE (%)	41.72	28.68
		R^2	0.30	0.65
3	Training	RMSE	12.72	8.33
		MAPE (%)	26.41	17.37
		R^2	0.37	0.82

Dataset	Phase	Indices	One neuron	ANN
	Testing	RMSE	12.62	8.27
		MAPE (%)	27.53	16.14
		R^2	0.48	0.81

The models are trained over 100 epochs and the learning rate is fixed to be 0.01. In the ANN model, 10 neurons are used in the hidden layer. The models' prediction results are summarized in Table 1. Herein, the root mean square error (RMSE), mean absolute percentage error (MAPE), and coefficient of determination (R^2) (Nguyen, Cao, Tran, Tran, & Hoang, 2022) are used for model assessment. As can be seen from this table, the 1st model (consisting of one neuron) can relatively well predict the samples in the Dataset 1 with $R^2 = 0.82$. However, this linear model obtains poor results in the other two datasets with $R^2 < 0.5$. In addition, the ANN model shows good performance with $R^2 = 0.88$, 0.65, 0.81 for the Dataset 1, 2, and 3, respectively.

5. Conclusion

Regression analysis is a crucial task in construction engineering. This paper presents the generalized delta rule for training the ANN model. The formulation derived from the chain rule is reviewed, followed by the presentation of the generalized delta rule. The formulation based on the generalized delta rule helps to simplify the expression of the training process of an ANN and facilitate the development of computer program based on the ANN. The developed models have been built in Visual C# .NET and applied to predicting the compressive strength of concrete in three datasets. Experimental results show good prediction performance of the models. Future extension of the current study includes the use of the generalized delta rule in training deep neural networks, which consists of more than one hidden layer.

References

[1] Aggarwal, C. C. (2018). *Neural Networks and Deep Learning*. United States. Springer International Publishing.

[2] Bullinaria, J. A. (2004). Introduction to Neural Networks. *Lecture Note, School of Computer Science - University of Birmingham <https://www.cs.bham.ac.uk/~jxb/NN/>*.

[3] Cheng, M.-Y., Hoang, N.-D., & Wu, Y.-W. (2015). Cash flow prediction for construction project using a novel adaptive time-dependent least squares support vector machine inference model. *Journal of Civil Engineering and Management, 21*(6), 679-688. doi: 10.3846/13923730.2014.893906

[4] Farooq, F., Czarnecki, S., Niewiadomski, P., Aslam, F., Alabduljabbar, H., Ostrowski, K. A., Śliwa-Wieczorek, K., Nowobilski, T., & Malazdrewicz, S. (2021). A Comparative Study for the Prediction of the Compressive Strength of Self-Compacting Concrete Modified with Fly Ash. *Materials, 14*(17), 4934. doi:10.3390/ma14174934

[5] Gandomi, A. H., Yun, G. J., & Alavi, A. H. (2013). An evolutionary approach for modeling of shear strength of RC deep beams. *Materials and Structures, 46*(12), 2109-2119. doi: 10.1617/s11527-013-0039-z

[6] Hoang, N.-D. (2022). Compressive Strength Estimation of Rice Husk Ash-Blended Concrete Using Deep Neural Network Regression with an Asymmetric Loss Function. *Iran. J. Sci. Technol. - Trans. Civ. Eng.* doi: 10.1007/s40996-022-01015-4

[7] Kim, P. (2017). MatLab Deep Learning with Machine Learning, Neural Networks and Artificial Intelligence. *Apress, ISBN: 1484228448*.

[8] Lishner, I., & Shtub, A. (2022). Using an Artificial Neural Network for Improving the Prediction of Project Duration. *Mathematics, 10*(22), 4189. doi: 10.3390/math10224189

[9] Nguyen, H., Cao, M.-T., Tran, X.-L., Tran, T.-H., & Hoang, N.-D. (2022). A novel whale optimization algorithm optimized XGBoost regression for estimating bearing capacity of concrete piles. *Neural Computing and Applications*. doi: 10.1007/s00521-022-07896-w

[10] Pazouki, G., Golafshani, E. M., & Behnood, A. (2022). Predicting the compressive strength of self-compacting concrete containing Class F fly ash using metaheuristic radial basis function neural network. *Structural Concrete, 23*(2), 1191-1213. doi: https://doi.org/10.1002/suco.202000047.