PLC-BASED ADAPTIVE CONTROLLER FOR STABILITY TANK PRESSURE

Pham Trong Tuong, Le Ngoc Binh, Pham Huy Hoang, Tran Thi Linh Nhi, Van-Phuong Ta
HCMC University of Technology and Education, Vietnam

Received 28/5/2019 , Peer reviewed 21/6/2019, Accepted for publication 30/7/2019

ABSTRACT

In recent years, the trend of applying intelligent controllers into an industrial system has been gaining more and more attention. Artificial neural networks is a must to mention when mentioning intelligent controllers. Not only for it's good performance but also for it's wide range of application. With an adaptive controller, we can save time of recalibrating the controller when load changes. This paper confirms the practical effect of applying Artificial Neural Networks (ANNs) using Radial basis function (RBF) bases on Sliding mode control (SMC) to control nonlinear systems. The proposed algorithm is put into comparison with the super twisting 2-SMC, which was designed to reduce chattering and increase the performance of conventional SMC. The pressure system is controlled by a Programmable Logic Controller (PLC), which is the most commonly used in industry, with a view to applying intelligent controllers in industrial applications to increase quality, productivity and reduce downtime of current systems.

Keyword: *sliding mode control (SMC); pressure control system; adaptive neural controller; radial basis function neural networks; artificial neural networks (ANNs); adaptive neural networks.*

1. INTRODUCTION

Nowadays, proportional–integral–derivative (PID) controller is the most commonly used control algorithms in the industry, particularly in continuous processes due to its functional simplicity, the capability of dealing with the wide spectrum of difficulties and opportunities in manufacturing plants and adaptability to widespread applications [1]. However, in highly-nonlinear uncertain systems or applications requiring accurate control regardless of noise and oscillation, the PID controller is not sufficient enough to fulfill the control requirement [2]. Recalibrating PID gains in those systems will lead to cost overrun and downtime. For instance, in Electro-hydraulic Servo (EHS) system, which is typically a complicated system suffers from disturbances and uncertainties, the PID controller is backward compared with SMC [3].

Sliding mode control (SMC) is a well-known control method for controlling uncertain systems [3]. It uses a switching control beside equivalent control to move the system's state trajectories to an expected position known as sliding surface, and keep it remained on in subsequent time. The main problem of SMC is the chattering phenomenon. Later on in this paper, a novel approach to remove the adverse force of the chattering proposed by Piazza d'Ammi [11] will be mentioned.

Another control method raising awareness is neural networks. Neural networks mimic intelligence of the biological human brain. The massively parallel processing, nonlinear mapping, and self-learning abilities of neural networks have been motivating factors for the development of intelligent control systems [4-6]. Neural networks can automatically tune its parameters itself to optimize the control performance over time [7]. Self-learning ability makes neural networks controller perform outstandingly in highly nonlinear uncertain systems [6]. Moreover, designing a neural networks controller-based SMC doesn't require parameter identification of dynamical system as the SMC does.

2. SLIDING MODE CONTROL

2.1. Conventional sliding mode control

Consider a nonlinear uncertain system

$$\begin{cases} \dot{x}_1 = x_2 \\ \dots \\ \dot{x}_{n-1} = x_n \\ \dot{x}_n = f(x) + g(x).u(t) + d(t) \\ y = x_1 \end{cases} \quad (1)$$

where $g(x) \in R^{nm}$ is the state vector, u(t) is the control input, $f(x) \in R^n$ is a known nonlinear function.

Tracking error is defined as

$$e(t) = y(t) - y_d(t) \quad (2)$$

$$\Delta e(t) = e(t) - e(t-1) \quad (3)$$

Conventional SMC has the following form

$$S = e^{(n-1)} + a_{n-2}e^{(n-2)} + ... + a_1\dot{e} + a_0 e \quad (4)$$

where [a_0, … , a_{n-2}] are pre-chosen constants for which the function (4) meets the Hurwitz requirement (all roots are negative). The object is to achieve S→0 when t→∞, or in other words, e→0 when t→∞. It means that $y(t) \to y_d(t)$ regardless of d(t).

Take the derivative of (4)

$$\dot{S} = e^{(n)} + a_{n-1}e^{(n-1)} + ... + a_1\dot{e} \quad (5)$$

$$= f(x) + g(x)u - y_d^{(n)} + a_{n-1}e^{(n-1)} + ... + a_1\dot{e}$$

The controller u(t) is designed so that in a finite time t_r, the system state trajectories are driven onto a specified sliding surface $S = 0$ and maintained there for all subsequent time. Mathematically, it can be described as below.

$$\text{Target} \begin{cases} S > 0 \to \dot{S} < 0 \\ S < 0 \to \dot{S} > 0 \end{cases} \quad (6)$$

Or, equivalently,

$$S\dot{S} < 0 \Leftrightarrow S.\frac{\delta S}{\delta u}\frac{\delta u}{\delta w}\dot{w} < 0 \quad (7)$$

However, in order to design an effective SMC controller, it demands a comprehensive understanding of the system. This is typically literally difficult in complicated or highly nonlinear uncertain systems. In practice, the SMC controller may not perform fast enough because of the time delay for control, computations and physical limitations of switching devices. As a result, the sliding mode control action can lead to high-frequency oscillations called chattering which may excite unmodeled dynamics, energy loss, and system instability and sometimes it may lead to plant damage [8]. Moreover, presented by Liang et al. (2012) for a class of second-order nonlinear uncertain systems using a sliding mode control strategy, the controller failed to work with unmatched uncertainties. To solve the problem, Dianwei Qian (2011) proposed to use radial basis function neural networks for systems with uncertain dynamics to deal with an unexpected disturbance [9].

2.2. Supertwister 2-SMC

Proposed and experienced by Piazza d'Ammi in [10], the super twisting 2-SMC has the following form:

$$\begin{aligned} F &= F_1 + F_2 \\ F_1 &= -\sqrt{F^*}\sqrt{|S|}\,\text{sgn}(S) \\ F_2 &= -1.1F^*\,\text{sgn}(S) \end{aligned} \quad (8)$$

where the parameter F^* is prechosen constant. F* is adjusted through trial and error to meet control expectation.

With this type of switching control function, the SMC is more flexible and performs smoother when the sliding surface moves gradually to zero. However, the sign function in F_2 causes high-frequency chattering in the control command. In order to minimize the chattering, the sat function is used in replacement of the sign function. The superior practical performance of the sat function was proven in the inverted pendulum system [11]. The sat function has the following form:

$$\mathrm{Sat(S)} = \begin{cases} \mathrm{Sgn(S)} & \text{if } |\mathrm{S}| \geq 1 \\ \mathrm{S} & \text{if } |\mathrm{S}| < 1 \end{cases} \quad (9)$$

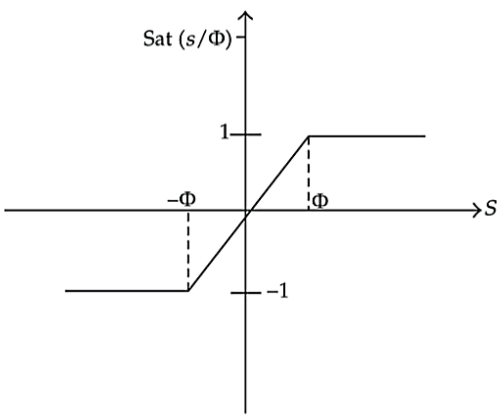

Fig. 1 *Function Sat(S/ϕ) to eliminate chattering*

3. RADIAL BASIS FUNCTION NEURAL NETWORKS (BRFNN)

3.1. Neural Networks Controller

The neural networks controller is famous for its online self-tuning ability. In this paper, the neural networks are only constructed in a simple form to reduce the time of computing, provide faster performance.

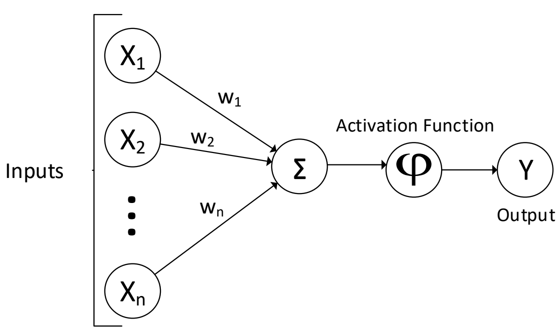

Fig. 2 *General model of Neural Networks*

The individual inputs x_1, x_2, .., x_n are each weighted by corresponding elements w_1, w_2, .., w_n. The nonlinear functions are used as output activation functions. The output value is given as below

$$y = \sum_{i=1}^{p} \mathrm{w}_i x_i + b = \begin{bmatrix} \mathrm{w}^T b \end{bmatrix} \begin{bmatrix} x \\ 1 \end{bmatrix} \quad (10)$$

where b is a biased constant of the inputs.

3.2. Radial basis function Neural Networks (RBFNN)

A radial basis function is a multi-dimensional function that describes the distance between a given input vector and a pre-defined center vector [12]. Normally, an RBFNN has 3 layers: an input layer, hidden layer, and output layer.

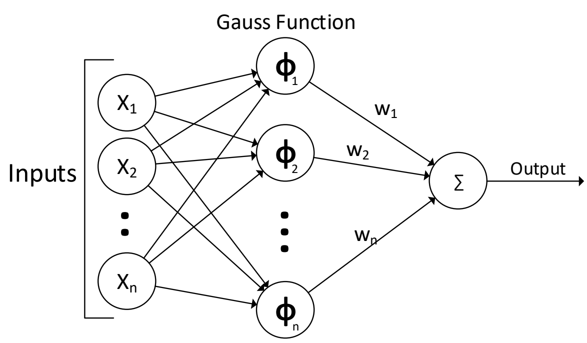

Fig. 3 *Model of RBFNN*

There are lots of radial basis functions such as Gauss, multiquadric, inverse quadratic, polyharmonic spline, etc. In this paper, the Gauss function, which has the following form, is used to examine:

$$\phi_{\mathrm{i}}(\mathrm{r}) = \mathrm{e}^{-(\zeta\,\mathrm{r})^2} = \exp\left(\frac{\|\mathrm{x}-\mathrm{x_i}\|^2}{2\sigma_{\mathrm{i}}^2}\right) \quad (11)$$

where $\mathrm{x_i}$ and σ_{i} denote the center and variance of the i-th node, respectively.

RBFNN model with single output is described as:

$$\mathrm{u_{control}} = \mathrm{f(x)} = \sum_{\mathrm{i}=1}^{\mathrm{m}} \mathrm{w_i} \phi_{\mathrm{i}}(\|\mathrm{x}-\mathrm{x_i}\|) \quad (12)$$

where f(x) is the final output control signal, $\phi_{\mathrm{i}}(.)$ is the radial basis function of i-th neural in the hidden layer, w_i is the weight between the input layer and hidden layer, m is the number of nodes in the hidden layer.

The parameters of w_i, x_i, σ_i RBFNN are always determined so as to minimize the value of $S\dot{S}$ to achieve the condition S=0, satisfy (6).

$$w_i(t+1) = w_i(t) + \dot{w}_i \quad (13)$$

$$x_i(t+1) = x_i(t) + \dot{x}_i \quad (14)$$

$$\sigma_i(t+1) = \sigma_i(t) + \dot{\sigma}_i \quad (15)$$

According to Gradient descent algorithm, the control law can be updated as below

$$\dot{w}_i = -\tau \frac{\delta S(t)\dot{S}(t)}{\delta w_i(t)} \quad (16)$$

$$\dot{x}_i = -\tau \frac{\delta S(t)\dot{S}(t)}{\delta x_i(t)} \quad (17)$$

$$\dot{\sigma}_i = -\tau \frac{\delta S(t)\dot{S}(t)}{\delta \sigma_i(t)} \quad (18)$$

where τ is the adaptive rate. Apply the chain rule, (15) (16) (17) can be rewritten as

$$\dot{w}_i = -\tau \frac{\delta S(t)\dot{S}(t)}{\delta u(t)} \frac{\delta u(t)}{\delta w_i(t)} = \eta_w S(t)\phi_i(s) \quad (19)$$

$$\dot{x}_i = -\tau \frac{\delta S(t)\dot{S}(t)}{\delta u(t)} \frac{\delta u(t)}{\delta \phi(t)} \frac{\delta \phi(t)}{\delta x(t)} = \eta_x S(t)\phi_i(s) w_i(t) \frac{S_i - x_i}{\sigma_i^2} \quad (20)$$

$$\dot{\sigma}_i = -\tau \frac{\delta S(t)\dot{S}(t)}{\delta u(t)} \frac{\delta u(t)}{\delta \phi(t)} \frac{\delta \phi(t)}{\delta \sigma(t)} = \eta_\sigma S(t)\phi_i(s) w_i(t) \frac{(S_i - x_i)^2}{\sigma_i^3} \quad (21)$$

where η_w, η_x, η_σ are learning the rate of weight, bias and spread width of i-th node, respectively.

4. EXPERIMENTAL RESULTS AND DISCUSSION

4.1. Setting the experiment

In order to examine the effectiveness of the RBFNN in industrial control systems, the algorithm is applied to control the pressure control system (PCS) controlled by Programmable Logic Controller (PLC) in RSLogix5000.

The examined system uses PLC as the controller. The value pressure sensor is obtained through ADC of VFD Powerflex700s and transmitted to the controller through the DeviceNet protocol. Ethernet is selected to be the communication protocol between PLC and PC.

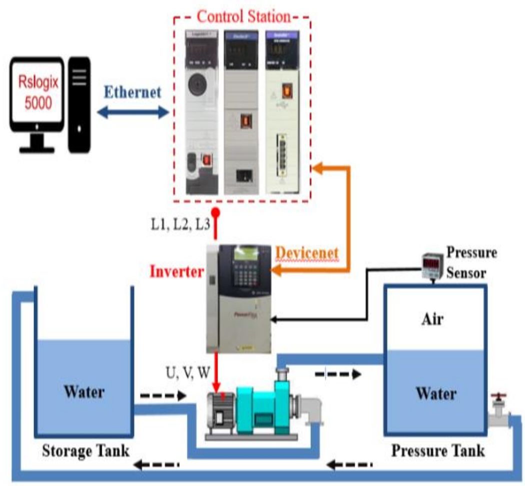

Fig. 4 *Block diagram of the system.*

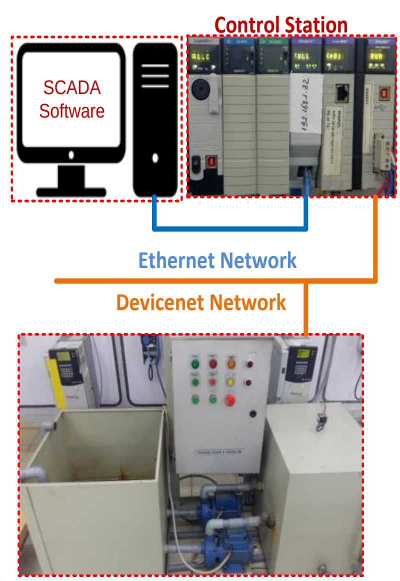

Fig. 5 *A practical system*

4.2. Sliding Mode Control design.

General SMC control form:

$$u_{control} = u_{sw} + u_{eq} \quad (22)$$

Where u_{sw} is switching control and u_{eq} is equivalent control.

The specific parameters of every equipment used in the system are normally hard to identify. In order to model the system, the experiment used a set of 25000 input-output samples, t_{sample} = 20ms, computed by the identification tool of Matlab to obtain the transfer function of the system as below.

$$\ddot{y}(t) + 0.001292\dot{y}(t) + 1.939e-05y(t) = 1.605e\text{-}08u_{eq}(t) \quad (23)$$

The precision of the function is 81,04%, shown in Figure 6. (16) can also be described as:

$$\begin{cases} x^{(n)} = F_0(x) + G_0(x)u_{eq} + L(x) \\ y = x; x = [x^T \dot{x}^T \ldots x^{(n-1)T}]^T \in R^{nm} \end{cases} \quad (24)$$

where $F_0(x), G_0(x)$ are identified parameters, x is the state variable. $L(x)$ is unknown parameter due to system modeling error or changes in system parameters.

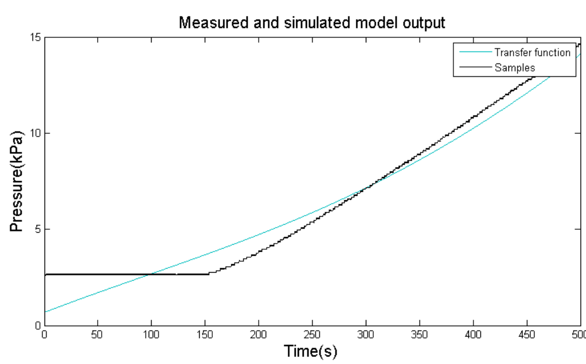

***Fig. 6** Simulation result of the transfer function.*

The switching control u_{sw}

$$u_{sw} = F = F_1 + F_2$$

$$F_1 = -\sqrt{F_1^*}\sqrt{|S|}\,\text{sat}(S) \quad (25)$$

$$F_2 = -1.1F_2^*\,\text{sat}(S)$$

where F_1^* and F_2^* are adjusted constants through trial and error for optimal values to meet the expectation.

4.3. Radial Basis Function Neural Networks design.

The input space S is bounded between [-1,1]. Initial values of the parameters in the networks are defined as below

$$\begin{cases} W = [0{,}1\ \ 0{,}1\ \ 0{,}1\ \ 0{,}1\ \ 0{,}1\ \ 0{,}1\ \ 0{,}1\ \ 0{,}1\ \ 0{,}1\ \ 0{,}1] \\ B = [-1\ \ -0{,}8\ \ -0{,}6\ \ -0{,}4\ \ -0{,}2\ \ 0\ \ 0{,}2\ \ 0{,}4\ \ 0{,}6\ \ 0{,}8\ \ 1] \\ C = [0{,}1\ \ 0{,}1\ \ 0{,}1\ \ 0{,}1\ \ 0{,}1\ \ 0{,}1\ \ 0{,}1\ \ 0{,}1\ \ 0{,}1\ \ 0{,}1] \end{cases}$$

These values are chosen by using the "Trial and error" method – setting parameters based on observation and results from previous experiments on the experimental system. For instance, the weights usually vary in the range [-1 2] so we chose a relatively small number in between. It's simple, easy, and doesn't need a complicated inspection for initial set-up of the controller.

Learning rate $\eta_w = \eta_x = \eta_\sigma = 0{,}5$.

Sample time = 15ms.

4.4. Experimental results

Experimental control results are presented below:

***Table 1.** Desired setpoints during the experiment*

	Stage 1	Stage 2	Stage 3	Stage 4
Current Setpoint (kPa)	6.3	7.5	10.3	7.3
Desired Serpoint (kPa)	7.5	10.3	7.3	9.5

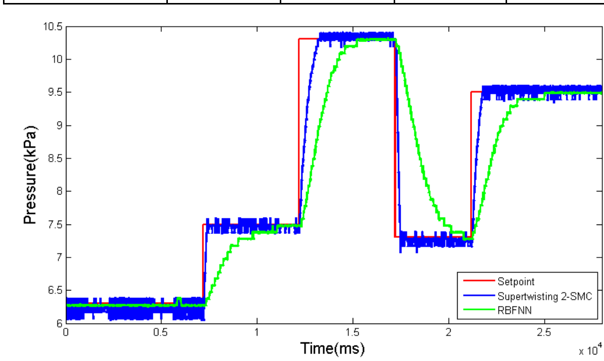

***Fig.7.** Tracking response*

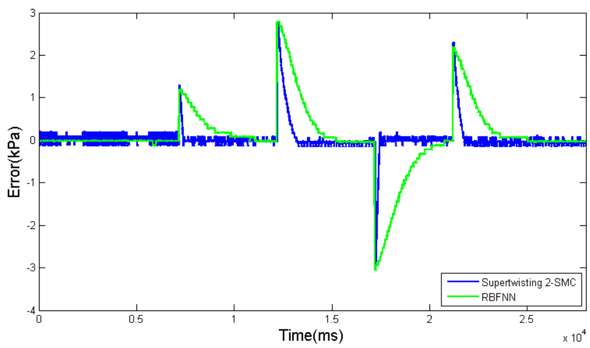

Fig. 8 *Tracking error*

4.5. Discussion

As indicated in figure 7 and table 1, the super twisting 2-SMC controller provides better transition time. However, although it is already designed to reduce chattering, it is unable to thoroughly eliminate the phenomenon. At low values, the chattering is even much worse. Most importantly, in practice, when there is a change in water capacity in the water source tank, the greater the change is, the less accurate the transfer function is. Hence, the system has to be recalibrated.

With RBFNN, we don't need to know the exact elements of f(x) and g(x) in (24) to design a controller. We don't need to spend time on achieving optimal values for the parameters of the system and the adaptation range of the controller is wider. Although it needs time to reach the setpoint, the accuracy is extremely high. The RBFNN does a good job in dealing with disturbance and uncertainties. One interesting advantage of RBFNN is that we can adjust the transition time of every stage as fast or as slow as we want by manually adjusting the learning rate. Of course, we may face with overfitting if the learning rate is too high.

In conclusion, if the system is consistent, require fast transition time and do not require high accuracy, super twisting 2-SMC is a good consideration. In a more complex system or accuracy or wide control range is the first priority, then RBFNN is a better choice.

5. CONCLUSION AND FUTURE WORK

The data exchanged between the control stations and the actuators were transferred via the industrial DeviceNet Networks. This solution not only reduces the effects of noise and signal attenuation during operation but also increase control distance between the control stations and the practical applications.

Moreover, this paper represented a solution to reduce the chattering effect in SMC and applied an intelligent controller with high accuracy to control pressure – Radial Basis Function Neural Networks (RBFNN) based SMC. RBFNN performed excellently regardless of noise. Future works will be focused on how to combine both the advantages of the two controllers – the speed of super twisting 2-SMC and the accuracy of RBFNN. On the other hand, the experiment was conducted in the lab environment, not in the practical field. Applying the controller to the industrial environment, especially in complex dynamical systems, may encounter some problems like bursting, overfitting or overtraining phenomenon. There were several solutions proposed such as dropout [13], use of validation data [14], regularization [15] and so on. Of course, in order to put it into use in industrial applications, there are still dozen of things to work on.

REFERENCES

[1] K. McMillan, 14 Industrial Applications of PID Control.

[2] Amir Gheibi, S. Mohammad Ali Mohammadi, Malihe M. FarsangiGregory, Comparing the performance of PID and fuzzy controllers in the presence of noise for a Photovoltaic System, Journal of mathematics and computer science, 9 (2014), 69-76.

[3] Rozaimi Ghazali, Yahaya Md Sam, Mohd Fua'ad Rahmat, Abd Wahab Ishari Mohd Hashim, Zulfatman, "Performance Comparison between Sliding Mode Control with

PID Sliding Surface and PID Controller for an Electro-hydraulic Positioning System", Hotel Equatorial Bangi-Putrajaya, Malaysia, 14 - 15 January 2011, ISBN 978-983-42366-4-9.

[4] E. Ciupan, F. Lungu, C. Ciupan, "ANN Method for Control of Robots to Avoid Obstacles", International Journal of Computers communication & Control, ISSN 1841-9836, 9(5): 539-554, October 2014.

[5] S. Jung, "Stability Analysis of Reference Compensation Technique for Controlling Robot Manipulators by Neural Network," International Journal of Control, Automation and Systems, Vol. 15, no. 2, pp. 952-958, 2017.

[6] Karas, P., & Kozak, S. (2018). Highly nonlinear process model using an optimal artificial neural network. 2018 Cybernetics & Informatics (K&I). doi:10.1109/cyberi.2018.8337548

[7] Van-Phuong Ta, Xuan-Kien Dang, An Innovative Recurrent Cerebellar Model Articulation Controller for Piezo-Driven Micro-Motion Stage, ICIC International (2018) ISSN 1349-4198, pp.1-1711.041.

[8] Kareem, Abdul, Fuzzy logic based super: twisting sliding mode controllers for dynamic uncertain systems, Department of Electronics and Communication, St. Peter's University, June 2015.

[9] Dianwei Qian, Jianqiang Yi, and Xiangjie Liu, A Robust Sliding Mode Controller Based on RBF Neural Networks for Overhead Crane Systems With Uncertain Dynamics, ICIC International 2011 ISSN 1881-803X, pp. 1995–2000.

[10] Piazza d'Ami. A Quick Introduction To Sliding Mode Control And Its Applications. 09123 Cagliari (I).

[11] Mahmoodabadi, M. J., Bagheri, A., Nariman-zadeh, N., Jamali, A., & Abedzadeh Maafi, R. (2012). Pareto Design of Decoupled Sliding-Mode Controllers for Nonlinear Systems Based on a Multiobjective Genetic Algorithm. Journal of Applied Mathematics, 2012, 1–22. doi:10.1155/2012/639014.

[12] Sun, Tsung-Ying & Liu, Chan-Cheng & Lin, Chun-Ling & Hsieh, Sheng-Ta & Huang, Cheng-Sen. (2009). A Radial Basis Function Neural Network with Adaptive Structure via Particle Swarm Optimization. 10.5772/6763.

[13] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, Ruslan Salakhutdinov, Dropout: A Simple Way to Preven Neural Networks from Overfitting, Journal of Machine Learning Research 15 (2014) 1929-1958.

[14] B. Mounika, G. Raghu, S. Sreelekha and R. Jayanthi, "Neural network based data validation algorithm for pressure processes," 2014 International Conference on Control, Instrumentation, Communication and Computational Technologies (ICCICCT), Kanyakumari, 2014, pp. 1223-1227. doi: 10.1109/ICCICCT.2014.6993147.

[15] Shubham Jain, An Overview of Regularization Techniques in Deep Learning, April 19, 2018.

Corresponding author:
Pham Trong Tuong
HCMC University of Technology and Education
Email: 15151094@student.hcmute.edu.vn