Computers, Materials & Continua

Tech Science Press

DOI: 10.32604/cmc.2024.048787

ARTICLE

Alternative Method of Constructing Granular Neural Networks

Yushan Yin[1], Witold Pedrycz[1,2] and Zhiwu Li[1,*]

[1]School of Electro-Mechanical Engineering, Xidian University, Xi'an, 710071, China

[2]Department of Electrical and Computer Engineering, University of Alberta, Edmonton, AB, T6R 2V4, Canada

*Corresponding Author: Zhiwu Li. Email: zhwli@xidian.edu.cn

Received: 18 December 2023 Accepted: 21 February 2024 Published: 25 April 2024

ABSTRACT

Utilizing granular computing to enhance artificial neural network architecture, a new type of network emerges—the granular neural network (GNN). GNNs offer distinct advantages over their traditional counterparts: The ability to process both numerical and granular data, leading to improved interpretability. This paper proposes a novel design method for constructing GNNs, drawing inspiration from existing interval-valued neural networks built upon NNNs. However, unlike the proposed algorithm in this work, which employs interval values or triangular fuzzy numbers for connections, existing methods rely on a pre-defined numerical network. This new method utilizes a uniform distribution of information granularity to granulate connections with unknown parameters, resulting in independent GNN structures. To quantify the granularity output of the network, the product of two common performance indices is adopted: The coverage of numerical data and the specificity of information granules. Optimizing this combined performance index helps determine the optimal parameters for the network. Finally, the paper presents the complete model construction and validates its feasibility through experiments on datasets from the UCI Machine Learning Repository. The results demonstrate the proposed algorithm's effectiveness and promising performance.

KEYWORDS

Granular neural network; granular connection; interval analysis; triangular fuzzy numbers; particle swarm optimization (PSO)

1 Introduction

Artificial neural networks (ANNs) possess numerical nodes (their connections), enabling nonlinear mapping of numerical data. These networks are also known as numerical neural networks (NNNs). Neural networks with granular connections, capable of nonlinear mapping of nonnumerical (granular) data, represent an extensive application of neural networks. This novel category is termed granular neural networks (GNNs, with granular connections) [1–4]. When processing numerical data, these networks produce granular outputs. GNNs offer distinct advantages over numerical counterparts. Their granular output aligns more closely with reality, aiding in quantifying the network's output quality. For instance, in prediction tasks, the output is a comprehensive, granular prediction rather than

a mere numerical result. Additionally, incorporating information granularity into these new models enhances the interpretability of the outputs (granular data) without compromising model accuracy.

Various types of information granularity, such as fuzzy sets, rough sets, fuzzy rough sets, and interval sets, are applicable in granular connections. A GNN with interval connections was proposed in [5]. The researchers also developed an optimization algorithm for this network. The study in [6] trained neural networks using granulated (interval) data, employing interval optimization algorithms like the generalized bisection method to determine optimal network weights and biases. The research presented in [7] focused on data-driven training algorithms for interval arithmetic vector quantization, applicable both during training and runtime.

Interval value, due to its straightforward processing and capability to represent uncertainty, is a commonly used form of information granularity. In 2013, Song and Pedrycz introduced a GNN with interval-valued connections [8]. Differing from prior methods, they initially trained an NNN with training data. Next, they integrated an information granularity level into the network to convert the nodes of the numerical network into interval values. Two key performance metrics, coverage and specificity, are employed to evaluate the algorithm. Optimizing these metrics yields the best network parameters for constructing the network.

This paper aims to develop a novel GNN related to the interval-valued network previously described. However, it differs because it is not based on an NNN but is an independent structure. Bypassing the initial phase, we directly assign an information granularity level to the weights and biases of the neural network (with undetermined parameters), forming a GNN (with undetermined parameters). We define the product of two indicators: Coverage of numerical data and specificity of information granules as objective functions (performance index). Upon inputting training data into the network, we derive a performance index function dependent on the network parameters. These parameters are then determined by optimizing (using particle swarm optimization (PSO) [9–12]) the objective function, thus constructing the new algorithm [13,14]. GNNs, as an enhancement of the original concept, have broad application potential.

We revisit the earlier GNN model based on an NNN. In the initial development phase, the network's weights are derived from an NNN. By optimizing the objective function, the information granularity level is obtained from these weights. This approach, however, may not yield optimal network parameters. The model proposed in this paper circumvents this limitation, enabling the derivation of all parameters by maximizing the combination of the two performance metrics. These parameters are inherently optimized for the specified performance indicators.

Information granules can be formalized in different ways. Triangular fuzzy numbers, often used in granular computing due to their ability to express uncertainty, are suitable for constructing GNNs with triangular fuzzy number-valued connections.

In this paper, the GNN based on an NNN is designated as Version #1, while the GNN independent of an NNN is termed Version #2. The distinctions between these two approaches are summarized in Table 1. This paper diverges from existing interval-valued neural networks (IVNNs) by developing a method not reliant on an NNN. It introduces triangular fuzzy numbers as network weights. The use of information granules is recognized as a valuable tool in network design, with their optimization contributing to improved network structures. The performance indicator discussed combines the coverage of numerical data and the specificity of information granules, enhancing network quality. The integration of coverage and specificity establishes a robust optimization framework for the network. Fundamentally, introducing this new network model expands the practices of the original networks, presenting theoretically and practically viable algorithms.

Table 1: Comparison of two design methods of constructing GNNs

Version	Information granule	Construction on a basis of a numerical neural network?	Performance index
Version #1	Interval value.	Yes	The coverage of numerical data, the specificity of information granules.
Version #2	Interval value, Triangular fuzzy number.	No	The product of the coverage of numerical data and the specificity of information granules.

The structure of this paper is as follows: Section 2 offers a literature review. In Section 3, we initially outline the operational rules for interval and triangular fuzzy numbers. Subsequently, we describe the structure of a granular (interval-valued) neural network based on an NNN, along with two performance indicators for evaluating the output and optimal allocation of information granularity. Section 4 combines these two performance indicators into a singular metric. Here, we detail a novel method for developing a GNN, culminating in adopting triangular fuzzy numbers as the form of information granularity. Section 5 briefly introduces PSO, utilized in our experiments. Section 6 presents experimental studies. As previously mentioned, in this paper, an interval-valued (triangular fuzzy number) neural network represents a specific type of GNN. Therefore, when referring to a neural network as interval-valued (triangular fuzzy number), these two terms are used interchangeably.

2 Literature Review

ANNs [1–3] exhibit several limitations in information processing. Firstly, the complexity of ANNs increases rapidly with the growth in data dimensionality. Secondly, ANNs struggle to handle non-numeric data, such as qualitative descriptors like "higher," "small," or "around 35 degrees." Thirdly, the "black box" nature of these networks hinders their interpretability. Granular computing, an approach that addresses complex information, involves dividing information into segments according to specific rules, thereby simplifying computations. These segments, known as information granules (e.g., groups, fuzzy numbers, intervals, and classes) [15], enable granular computing to handle non-numeric data. Original data are processed using numeric calculations, while granular data utilize granular computing mechanisms. This method allows for solving problems hierarchically [16–18]. Information granules can enhance the explainability of neural networks and address the challenges of ANNs. An ANN incorporating granular computing to improve existing models is termed a GNN. Examples include neural networks with granular structures and those capable of processing granular data without such structures [1,4].

The initial concept of GNN, combining fuzzy sets with neural networks, emerged in the 1970s [19]. The term "granular neural network" was formally introduced in 2001 by Pedrycz et al. [4]. Since then, GNN research has garnered significant interest, as evidenced in Table 2. Studies on GNNs can be categorized into:

- Type of neural network: Various neural network models, such as radial basis function neural networks [20,21], feedforward neural networks [7,22], multilayer feedforward neural networks [23,24], and functional link ANNs [25], are employed in GNN development.
- Information granules: Granular computing, a paradigm of information processing [8,9], uses granules comprising elements with similar attributes. Various formal frameworks, including fuzzy sets [20,26,27], rough sets [22,28,29], fuzzy rough sets [30], and interval sets [7], can form these granules, leading to diverse GNN types like fuzzy neural networks, rough neural networks, and IVNNs.
- Granulation of input data: Literature generally divides this aspect into two scenarios. In one, the input data are already at a granular level, requiring no further granulation. Conversely, numerical data are converted into forms like interval data or fuzzy numbers, sometimes employing clustering methods for division into several parts.
- Connection: Traditional neural networks use numerical weights to represent their connections. In contrast, GNNs may employ nonnumerical weights, known as granular connections [5]. This distinction prompts researchers to explore modifications to traditional training methods, adapting them to GNNs' unique structures. When GNNs receive numerical input data, they can produce nonnumerical outputs. Furthermore, GNNs can process nonnumerical input data in more complex scenarios. Some studies are investigating the use of primitive neural networks for processing nonnumerical data [31], maintaining the network's original structure in such cases.
- Training and testing. While traditional neural networks process numeric input during training and testing phases, GNNs with granular connections are trained on numerical data and tested on granular data. Additionally, the degree of data integration influences the granularity level; lower integration corresponds to finer granularity and higher integration results in coarser granularity. Notably, networks designed for low-granularity data may sometimes handle high-granularity data. This means the neural networks possess a low-granularity structure but can process high-granularity inputs during testing. A more frequent scenario involves inputting low-granularity data into a network designed for low-granularity processing.

Table 2: Brief review of GNNs

Type of neural network	Radial basis function neural network, feedforward neural network, multilayer feedforward neural network, functional link artificial neural network
Information granules	Fuzzy sets, rough sets, fuzzy rough sets, interval sets
Granulation of input data	Numerical data, granular data
Connection	Numerical connections, granular connections
Training and testing	Type 1: Training (numerical data) and testing (numerical data), Type 2: Training (numerical data) and testing (granular data), Type 3: Training (granular data) and test (numerical data), Type 4: Training (granular data) and testing (granular data)

Numerous GNN models have been developed to address specific challenges. For instance, in [32], researchers devised a nonlinear granularity mapping model with adaptive tuning factors to counteract unknown nonlinearity. In [33], a novel dynamic integration method was introduced to construct reliable prediction intervals for granular data streams, incorporating a new interval value learning algorithm applicable to fuzzy neural networks. In [34], a fuzzy-informed network was developed

specifically for magnetic resonance imaging (MRI) image segmentation, utilizing fuzzy logic to encode input data. Additionally, a method using adaptive granular networks was designed for enhanced accuracy in sludge bulking detection [35]. This approach effectively creates an error compensation model. Furthermore, several GNNs incorporate local models, resulting in what are referred to as crisp neural networks [36,37]. These developments illustrate the versatility and adaptability of GNNs in various applications, ranging from image processing to environmental monitoring.

3 Two-Stage Design Process: Granularity Expansion of NNN

In this section, we initially present operations involving intervals and triangular fuzzy numbers. Subsequently, in Section 3.3, we delve into the existing IVNNs.

3.1 Interval Operations

We consider several interval-valued variables, such as $A = [a_l, a_2]$ and $B = [b_l, b_2]$, and outline their operations as follows [38]:

Addition:

$$A + B = [a_1 + \mathrm{b}_1, a_2 + b_2] \tag{1}$$

Subtraction:

$$A - B = [a_1 - \mathrm{b}_2, a_2 - b_1] \tag{2}$$

Multiplication:

$$A * B = [\min(a_1b_1, a_1b_2, a_2b_1, a_2b_2),\ \max(a_1b_1, a_1b_2, a_2b_1, a_2b_2)] \tag{3}$$

when $a_1 \geq 0, b_1 \geq 0$,

$$A * B = [a_1b_1, a_2b_2] \tag{4}$$

Division (except for cases where the divisor includes 0):

$$\frac{1}{A} = \left[\frac{1}{a_2}, \frac{1}{a_1}\right] \tag{5}$$

$$\frac{B}{A} = B * \frac{1}{A} = [b_1, b_2] * \left[\frac{1}{a_2}, \frac{1}{a_1}\right] \tag{6}$$

In addition to interval operations, we also address the mapping (function) of intervals, particularly focusing on operations for nondecreasing mappings.

$$f(A) = f([a_1, a_2]) = [f(a_1), f(a_2)] \tag{7}$$

For nonincreasing mapping, we obtain:

$$f(A) = f([a_1, a_2]) = [f(a_2), f(a_1)] \tag{8}$$

3.2 Triangular Fuzzy Number

For the sake of space efficiency, we introduce only those parts [39] that are relevant to this paper.

Let $\tilde{A}$ be a convex fuzzy set on the real number set R, if it has such a membership function:

$$\mu_{\tilde{A}}(x) = \begin{cases} 0 & x < a \\ \dfrac{x-a}{b-a} & a < x < b \\ \dfrac{x-c}{b-c} & b < x < c \\ 0 & x > c \end{cases} \tag{9}$$

$\tilde{A}$ is a triangular fuzzy number, where $\mu_{\tilde{A}}(x) : R \to [0, 1]$, a ≤ b ≤ c. If a=b=c, $\tilde{A}$ is a real number.

Given triangular fuzzy numbers $\tilde{A} = (a, b, c)$ and $\tilde{B} = (d, e, f)$ their ∂−cut sets are expressed as $\tilde{A} = \{[(b-a)\,\partial + a, (b-c)\,\partial + c]\}$, $\tilde{B} = \{[(e-d)\,\partial + d, (e-f)\,\partial + f]\}$, which are intervals. Hence, they obey the operation rules applicable to interval operations, implying they can be manipulated like interval values.

3.3 Existing IVNNs

Initially, we construct NNNs in a supervised manner, utilizing a set of input-output pairs (x_k, $target_k$), where $k = 1, 2, \ldots, N$. These networks feature a classic MLP (multilayer perceptron) structure [40], representing a fundamental topology for feedforward neural networks. This paper employs the Levenberg-Marquardt BP (back propagation) learning method [41] as the learning mechanism. Through the learning process, a collection of weights is determined. Succinctly, the network is represented as ($N(\boldsymbol{x}; W)$), with (W) denoting the network's weights, where each weight is represented by (w_i). The subsequent phase involves the granulation of weights. Specifically, an interval is constructed around each (w_i), with the interval's size (or length) determined by a parameter representing an information granularity level. Fig. 1 illustrates the structure of a GNN [7], evolved from a classic feedforward neural network. It consists of three layers: An input layer, where data features are inputted as a vector $\boldsymbol{x} = [x_1, x_2, \ldots, x_n]^T$; a hidden layer, containing n_1 neurons; and an output layer with only one neuron. In an NNN, the weights connecting the input layer to the hidden layer are typically denoted by W_{ji}, and those connecting the hidden layer to the output layer by W_j. Each connection includes a bias B_j. For the GNN, both the connection weights and biases are substituted with intervals $W_{ji} = \left[w_{ji}^-, w_{ji}^+\right]$, $W_j = \left[w_j^-, w_j^+\right]$, and $B_j = \left[b_j^-, b_j^+\right]$. Consequently, when numeric data is inputted into this network, the output is an interval $Y = [y^-, y^+]$.

After constructing the NNN, it is necessary to convert the network nodes into interval values. The procedures are detailed below. In the hidden layer of an NNN, the following is observed:

$$o_j = f_1\left(z_j\right), z_j = \sum_{i=1}^{n} w_{ji} * x_i + b_j, j = 1, 2, \ldots, n_1, i = 1, 2, \ldots, n$$

$$f_1\left(z_j\right) = \frac{1}{1 + e^{-z_j}} \tag{10}$$

For the output layer of the network, we obtain:

$$y = f_2(z), z = \sum_{j=1}^{n_1} w_j * o_j + b, f_2(z) = z \tag{11}$$

where w_{ji} denotes the weight from the input layer to the hidden layer, b_j denotes the bias from the input layer to the hidden layer, b denotes the bias from the hidden layer to the output layer, w_j denotes

the weight from the hidden layer to the output layer, f_1 and f_2 represent the activation functions of the neuron from the input layer to the hidden layer and from the hidden layer to the output layer, respectively. o_j denotes the output of a hidden layer, and y denotes the output of the network.

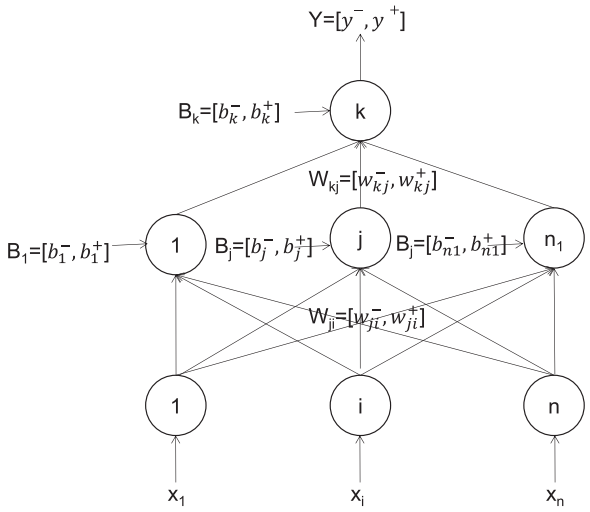


Figure 1: Architecture of a GNN

The subsequent section discusses the granulation of connections. The network is assigned an information granularity level ε, and connections are substituted with interval values. The weights w_{ji}, w_j and biases b_j are replaced by the interval generalizations:

$$\begin{aligned} w^- &= w - \varepsilon_- |w| \\ w^+ &= w + \varepsilon_+ |w| \\ \mathrm{b}_j^- &= b_j - \varepsilon_- \left|b_j\right| \\ b_j^+ &= b_j + \varepsilon_+ \left|b_j\right| \end{aligned} \tag{12}$$

Herein, we adopt a uniform allocation of information granularity to construct interval connections (granular connections) centered on the initial weights and biases. The intervals are symmetrically distributed around the original numeric value of each connection, implying that $\varepsilon_- = \varepsilon_+ = \varepsilon/2$. By substituting $\varepsilon_- = \varepsilon_+ = \varepsilon/2$ into the above equations, we obtain:

$$\begin{aligned} w^- &= w - \varepsilon_- |w| = w - \varepsilon/2 * |w| = \max\left(w\left(1-\varepsilon/2\right), w\left(1+\varepsilon/2\right)\right) \\ w^+ &= w + \varepsilon_+ |w| = w + \varepsilon/2 * |w| = \max\left(w\left(1-\varepsilon/2\right), w\left(1+\varepsilon/2\right)\right) \\ \mathrm{b}_j^- &= \mathrm{b}_j - \varepsilon_- \left|\mathrm{b}_j\right| = \mathrm{b}_j - \varepsilon/2 * \left|\mathrm{b}_j\right| = \min\left(b_j\left(1-\varepsilon/2\right), b_j\left(1+\varepsilon/2\right)\right) \\ b_j^+ &= \mathrm{b}_j + \varepsilon_+ \left|\mathrm{b}_j\right| = \mathrm{b}_j + \varepsilon/2 * \left|\mathrm{b}_j\right| = \max\left(b_j\left(1-\varepsilon/2\right), b_j\left(1+\varepsilon/2\right)\right) \end{aligned} \tag{13}$$

$$\begin{aligned} w^- &= \min\left(w\left(1-\varepsilon/2\right), w\left(1+\varepsilon/2\right)\right) \\ w^+ &= \max\left(w\left(1-\varepsilon/2\right), w\left(1+\varepsilon/2\right)\right) \\ \mathrm{b}_j^- &= \min\left(b_j\left(1-\varepsilon/2\right), b_j\left(1+\varepsilon/2\right)\right) \\ b_j^+ &= \max\left(b_j\left(1-\varepsilon/2\right), b_j\left(1+\varepsilon/2\right)\right) \end{aligned} \tag{14}$$

where ε is within the range of (0,1].

Substituting Eq. (14) into Eqs. (10) and (11) and replacing connections with interval values, we arrive at the following operations.

For the hidden layer:

$$o_j = \left[o_j^-, o_j^+\right] = \left[f_1\left(z_j^-\right), f_1\left(z_j^+\right)\right]$$

$$z_j^- = \sum_{i=1}^{n}\left(\min\left(w_{ji}^- xi, w_{ji}^+ xi\right) + b_j^-\right)$$

$$z_j^+ = \sum_{i=1}^{n}\left(\max\left(w_{ji}^- xi, w_{ji}^+ xi\right) + b_j^+\right), j = 1, 2, \ldots, n_1 \tag{15}$$

For the output layer:

$$Y = [y^-, y^+] = \left[f_2\left(z^-\right), f_2\left(z^+\right)\right]$$

$$z^- = \sum_{i=1}^{n}\left(\min\left(w_j^- o_i^-, w_j^+ o_i^-, w_j^- o_i^+, w_j^+ o_i^+\right) + b_j^-\right)$$

$$z^+ = \sum_{i=1}^{n}\left(\max\left(w_j^- o_i^-, w_j^+ o_i^-, w_j^- o_i^+, w_j^+ o_i^+\right) + b_j^+\right), j = 1, 2, \ldots, n_1 \tag{16}$$

Since the network's output is an interval, two performance indices are defined for evaluation purposes. The first is coverage, an index that indicates how many actual output intervals can 'cover' the expected output. The second is specificity, an index reflecting the average length of the output interval.

For any data (x_k, $target_k$), where k = 1, 2, ⋯, n, in a dataset, when the network's input is x_k, the network produces an interval result $Y_{\mathrm{k}} = \left[y_k^-, y_k^+\right]$. The quality of the results is evaluated in terms of coverage and specificity. The measures are defined as follows:

Coverage: For any data (x_k, $target_k$) and the output of the network Y_k, if $target_k \in Y_k$, then Y_k 'covers' $target_k$. Coverage quantifies the ratio of the number of data points covered to the total data amount.

$$Q1\left(\varepsilon\right) = \frac{1}{N}\sum_{k=1}^{N} incl\left(target_k, Y_k\right) \tag{17}$$

where

$$incl\left(target_k, Y_k\right) = \begin{cases} 1 & \text{if } target_k \in \left[y_k^-, y_k^+\right] \\ 0, & \text{otherwise} \end{cases} \tag{18}$$

Specificity: It quantifies the average length of output intervals:

$$Q2 = \frac{1}{N}\sum_{k=1}^{N}\max\left(0, 1 - \frac{\left|y_k^+ - y_k^-\right|}{range}\right) \tag{19}$$

where the range = $|\mathrm{target}_{\max}\text{-}\mathrm{target}_{\min}|$.

To summarize the discussion, the process can be delineated as follows:

Phase 1: Utilize the provided data to derive a new neural network employing numeric data and the backpropagation method.

Phase 2: Assign an information granularity level ε to the NNN constructed in the first phase, converting the numerical connection into an information granularity connection (interval value).

Phase 3: Optimize the objective function to identify the best parameters using PSO.

Phase 4: Construct the IVNN using the weights from the original NNN and the optimized information granularity level ε.

4 Single-Phase Design of GNN

Specificity (Q2) calculates the average length of output intervals, and coverage (Q1) assesses how many actual output intervals can 'cover' the expected output. These performance indices articulate and quantify information granularity in distinct manners. Based on the definitions of these performance indices, superior values indicate enhanced network performance. To thoroughly compare two methodologies for constructing a GNN, the product of the two performance indices of coverage and specificity is considered the objective function for the network. This novel performance index aptly balances the two indices. Consequently, the optimal values for the network parameters are determined by maximizing this new objective function, enhancing the network's adaptability.

$$V(\varepsilon) = Q1(\varepsilon) * Q2(\varepsilon) \tag{20}$$

where ε is a non-negative value, with ε_{max} denoting the maximum allowable value of ε.

Considering the range of values of ε, a global performance index is formulated as:

$$S(W) = \int_0^1 V(\varepsilon)\, d\varepsilon \tag{21}$$

For ε within the range of (0,1], where the level of information granules is within the unit interval, W represents the weights and biases of the network. In practical applications, integration is substituted with summation.

Given the nature of the performance index, it is clear that optimization cannot be limited to gradient-based methods. Therefore, PSO is considered for the optimization challenges presented [9,10].

As previously mentioned, the weights derived from the numeric neural network in phase 1, along with the ε that provides the best coverage and specificity for these weights, are not necessarily the optimal parameters for achieving high coverage and specificity. Consequently, a different strategy is adopted to develop a GNN. This approach involves combining the first and second phases of the earlier algorithm. In essence, instead of calculating the weights beforehand, the objective function V is optimized directly to simultaneously determine all parameters.

With this approach, the construction of the connections is also modified:

$$
\begin{aligned}
w^{-} &= \min\left(w\frac{1}{1+\varepsilon}, w\left(1+\varepsilon\right)\right) \\
w^{+} &= \max\left(w\frac{1}{1+\varepsilon}, w\left(1+\varepsilon\right)\right) \\
\mathrm{b}_{j}^{-} &= \min\left(b_{j}\frac{1}{1+\varepsilon}, b_{j}\left(1+\varepsilon\right)\right) \\
b_{j}^{+} &= \max\left(b_{j}\frac{1}{1+\varepsilon}, b_{j}\left(1+\varepsilon\right)\right)
\end{aligned} \tag{22}
$$

Triangular fuzzy numbers, which express uncertainty effectively, are frequently utilized as information granules in granular computing. Their cut sets can be converted into a series of intervals, allowing the use of interval values in the construction and performance evaluation of a GNN.

Regardless of the type of information granule—whether interval values or triangular fuzzy numbers—the steps for constructing these two variations of GNNs remain consistent. The GNN derived from an NNN is designated as Version #1, whereas the GNN developed independently of an NNN is referred to as Version #2. The methods for constructing a GNN are concisely outlined. Figs. 2 and 3 illustrate the algorithmic flow of each version.

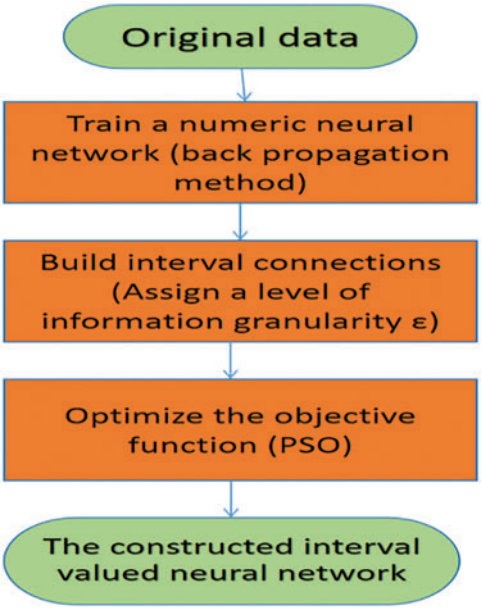


Figure 2: Flow of the construction of Version #1

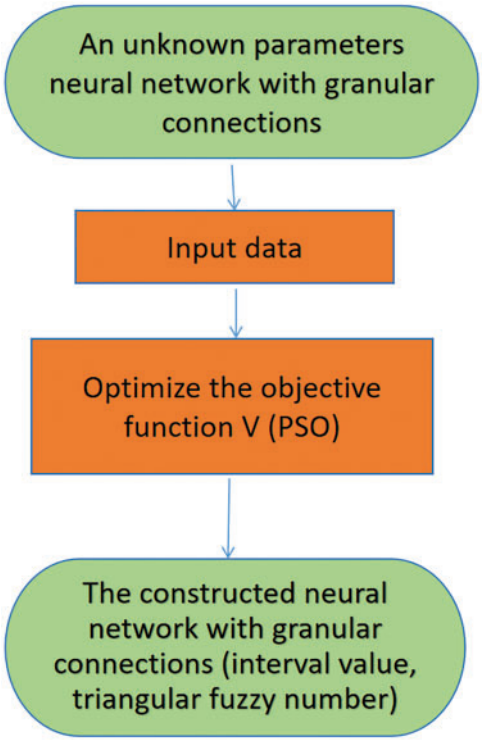


Figure 3: Flow of the construction of Version #2

Procedure for the construction of Version #1.

Procedure for the construction of Version #2.

Phase 1: For a neural network with undetermined parameters, establish an information granularity level ε, converting the network's connections into information granularity connections (interval values or triangular fuzzy numbers).

Phase 2: Input the provided data into the network and derive the objective function V, which depends on the network parameters. Optimize this objective function to identify the optimal network parameters.

Phase 3: Utilize the optimized parameters to establish the information granular connections of the network (using interval values or triangular fuzzy numbers), thereby constructing the GNN. Consequently, a granular network is developed.

5 PSO

In the optimization of neural networks, PSO was employed as an optimizer. A concise summary of its algorithm flow is provided below. Recognizing the distinct advantages inherent in various algorithms, integrating these methodologies has emerged as a significant focus of research aimed at overcoming the limitations of individual approaches. Prominent algorithms for hybrid computation have included the ant colony algorithm, genetic algorithm, and simulated annealing, among others. PSO emerged as a robust optimization algorithm capable of working with nonconvex models, logical models, complex simulation models, black box models, and even real experiments for optimization calculations. In PSO, each particle represented a potential solution consisting of a vector of optimized

levels of information granularity, weights, and biases. The problem-solving intelligence was realized through simple behaviors and information exchanges among particles and particle swarms. The procedure was as follows:

Phase 1: Initialization involved setting the maximum number of iterations, the number of independent variables in the objective function, the maximum velocity of particles, and their location information across the entire search space. The speed and location were randomly initialized within the speed interval and search space, respectively. The population size of the particle swarm was also established.

Phase 2: Individual extremum and global optimum. A fitness function was defined. The individual extremum referred to the optimal solution found by each particle, while the global optimum was determined from the best solutions identified by all particles. This global optimum was then compared with historical optima, and the best was selected as the current historical optimal solution.

Phase 3: The velocity and position were updated according to:

$$\begin{aligned} v_i^d &= w * v_i^d + c_1 * r_1 \left(pbest_i^d - x_i^d\right) + c_2 * r_2 \left(gbest_i^d - x_i^d\right) \\ x_i^d &= x_i^d + v_i^d \end{aligned} \tag{23}$$

where w denotes the inertia factor influencing the convergence of the swarm, c_1 and c_2 are the learning factors for adjusting the maximum learning phase size, i denotes the first particle, and d denotes the dimensionality of the particle. r_1 and r_2 are two random numbers in the range [0,1] (for different particle sizes, the values of $r_1$and $r_2$vary), $pbest_i^d$ denotes the position where the particle achieves its highest (lowest) fitness, and $gbest_i^d$ denotes the position where the entire system achieves its highest (lowest) fitness.

6 Experimental Studies

Multiple datasets were utilized, and a series of experiments were conducted to compare the performance of the newly proposed GNN in this study with existing GNNs. All datasets, including synthetic and benchmark datasets, were sourced from the UCI Machine Learning Repository (http://archive.ics.uci.edu/datasets). To evaluate the performance of numeric neural networks, the root mean squared error (RMSE) was employed as the indicator. A 10-fold cross-validation was introduced to generate the training and testing datasets.

In this study, both IVNNs and triangular fuzzy number neural networks shared the same network structure; the only difference lay in the information granules of the network nodes. These neural networks had only one hidden layer and a single output node. A series of experiments were also conducted to examine the relationship between network performance and the number of nodes in the hidden layer. The NNN was trained using the standard Levenberg-Marquardt minimization method, with the number of generations set to 1000. Before initiating the network, data preprocessing was performed to ensure that different dimensions of multidimensional data played the same role. The initial weights for all networks were set between [−1, 1]. The activation function for the input layer was sigmoid, while that for the hidden layer was linear.

A method described in previous sections was employed to construct a new network to test its performance, utilizing 10-fold cross-validation for data processing. Following the methodology in [42], the parameters of PSO were initialized with the number of generations set to 100, w to 0.7, c_1 to 2, and c_2 to 2. Adjusting different initial parameters did not yield any significant change in network performance, leading to the adoption of these parameters for all networks. The training time was also

documented due to significant differences in training costs when using different information granules to construct the network.

Table 3 presents the datasets used in this study, all of which are regression.

Table 3: Datasets

Data sets	Number of instances	Number of attributes (features)
$y = 0.8. * \sin(x_1/4). * \sin(x_2/2)$	600	2
Servo	167	4
AirfoilSelfNoise	1503	5
Housing	506	13
Fertility	100	9
Bodyfat	252	14
Automobile	205	25
Auto MPG	398	7

Note: The dataset '$y = 0.8 * \sin(x_1/4) * \sin(x_2/2)$', used as regression benchmark data [33], features x_1 in the range [0,10] and x_2 in the range [−5, 5].

6.1 Synthetic Data

These two-dimensional synthetic data, derived from the regression benchmark dataset [43], are modeled by the following nonlinear function $f(x_1, x_2)$:

$$f(x_1, x_2) = 0.8 * \sin(x_1/4) . * \sin(x_2/2) + z \tag{24}$$

where z is obtained from the Gaussian normal distribution $N(0, 0.2)$. A total of 600 sets of random data, adhering to a uniform distribution, were generated. The visualization of these 600 datasets is depicted in Fig. 4.

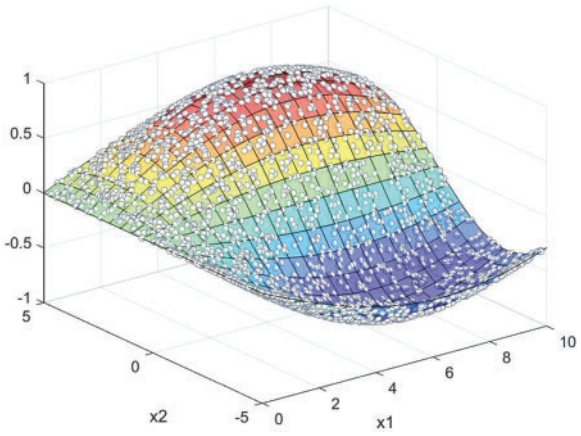


Figure 4: Two dimensional nonlinear function 'y = $0.8 * \sin(x_1/4) * \sin(x_2/2)$' used in the experiment

We consider an MLP with only one hidden layer of size h. Training is conducted using a standard BP algorithm. The performance of the NNN, as a function of h, is illustrated in Fig. 5; the size of h varies from 2 to 10. An optimal value of h is determined when the performance index reaches

its minimum value. According to Fig. 5, the optimal h is 8. Subsequently, a new network featuring granular connections is constructed by selecting its ε value within the range (0, 1] for Version #1 and in the range (0, 10] for Version #2.

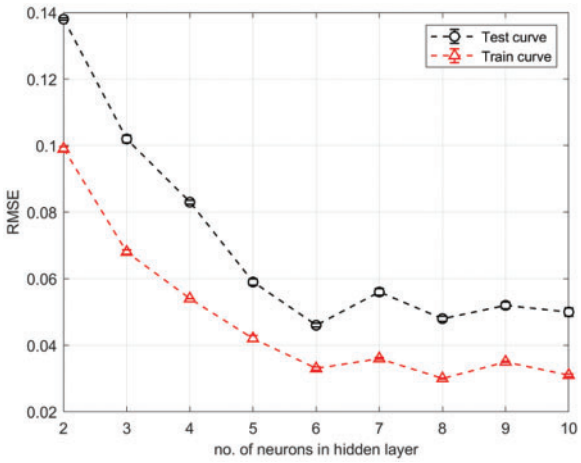
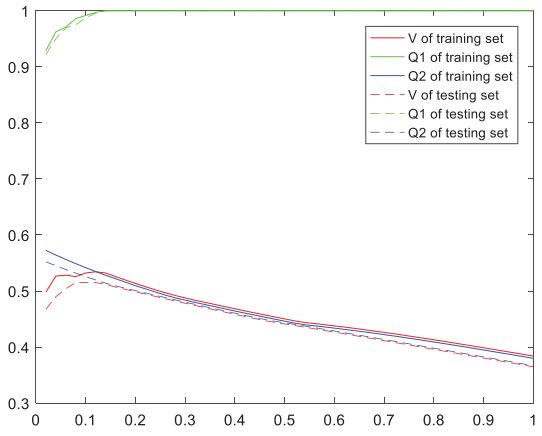
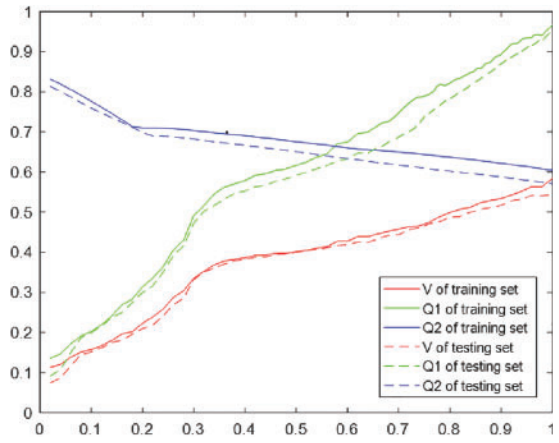


Figure 5: RMSE *vs.* h (number of neurons in hidden layer): Training data (red line) and testing data (black line)

Fig. 6 illustrates the variations in the objective functions V, $Q1$, and $Q2$ when ε is within the range (0,1]. Notably, irrespective of whether the information granule is an interval value or a triangular fuzzy number, the change tendency of these objective functions remains the same when the construction approach (Version #1 or Version #2) of the network is consistent.

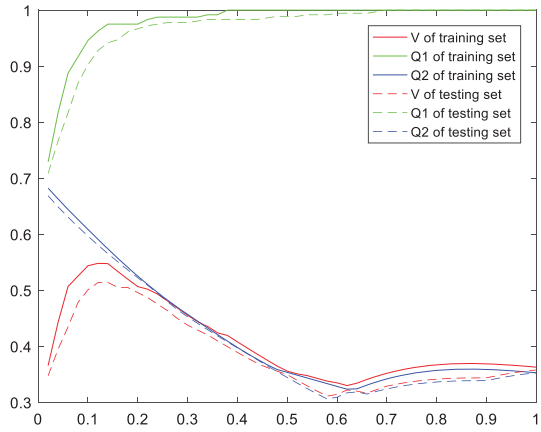


(a) IVNN for Version #1

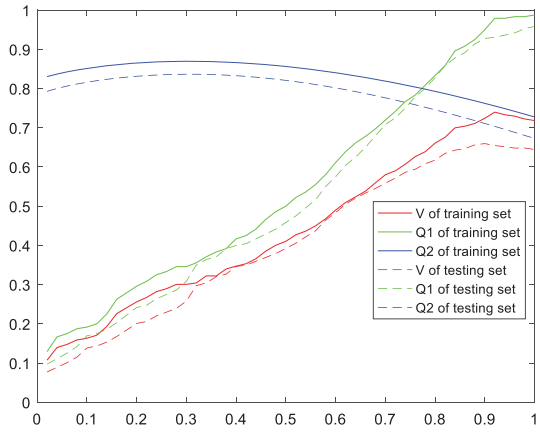


(b) IVNN for Version #2

Figure 6: (Continued)



(c) Triangular fuzzy number-valued neural network for Version #1



(d) Triangular fuzzy number-valued neural network for Version #2

Figure 6: $Q1$, $Q2$, and V *vs.* ε (ε is in (0, 1]); solid line—training data; dotted line—testing data

For a GNN of Version #1, the optimal number of neurons in the hidden layer is determined to be 8. However, as shown in Table 4, for the same dataset in the Version #2 neural network, whether interval or triangular fuzzy number-valued, h is found to be 4. According to Table 5, the performance indices V and S for Version #2, both for interval-valued and triangular fuzzy number-valued neural networks, are significantly better than those for the IVNN of Version #1. This aligns with prior analyses. Additionally, the IVNN of Version #2 demonstrates superior performance across all indices.

Table 4: Results of Version #2 using the dataset '$y = 0.8 * \sin(x_1/4) * \sin(x_2/2)$'; it compares V, $Q1$, $Q2$, and S *vs.* h, with the information granularity being either an interval value or a triangular fuzzy number

Data sets	h	2	3	4	5	6	7	8	9	10
V	Interval value	0.7338	0.7252	0.7345	0.6871	0.6996	0.6740	0.6470	0.6375	0.6373
	Triangle fuzzy number	0.7076	0.7128	0.7156	0.7084	0.6646	0.6750	0.6736	0.6394	0.6557
$Q1$	Interval value	0.9590	0.9631	0.9504	0.9376	0.9557	0.9499	0.9431	0.9567	0.9601
	Triangle fuzzy number	0.9453	0.9472	0.9491	0.9563	0.9645	0.9492	0.9663	0.9388	0.9452
$Q2$	Interval value	0.7665	0.7547	0.7723	0.7339	0.7323	0.7102	0.6883	0.6676	0.6646
	Triangle fuzzy number	0.7498	0.7536	0.7538	0.7389	0.6897	0.7106	0.6971	0.6825	0.6953
V of training set	Interval value	0.7628	0.7587	0.7570	0.7112	0.7161	0.7002	0.6744	0.6653	0.6645
	Triangle fuzzy number	0.7212	0.7379	0.7289	0.7242	0.6849	0.6936	0.6870	0.6589	0.6723
S	Interval value	0.1867	0.3933	0.3542	0.4620	0.3366	0.3580	0.3465	0.4132	0.3736
	Triangle fuzzy number	0.2372	0.2106	0.2483	0.3076	0.3349	0.3139	0.4017	0.3147	0.3263

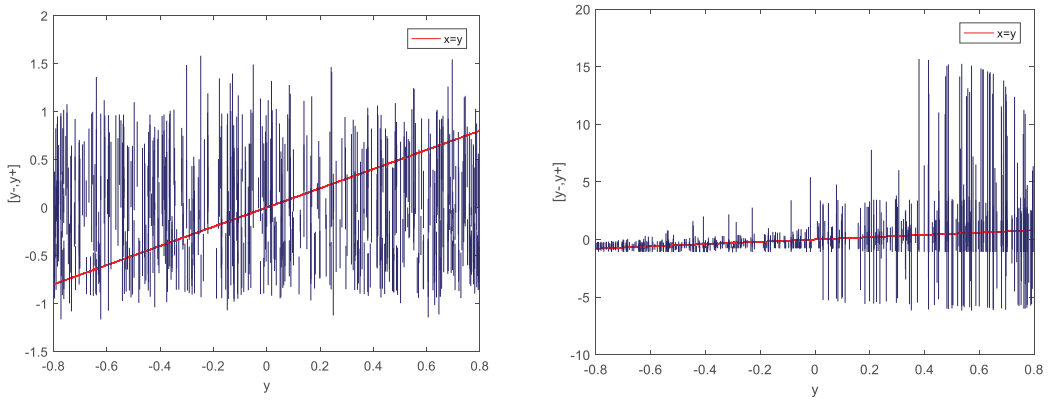
For Version #1, the V (for both training and testing sets) and S indices obtained by the triangular fuzzy number-valued network are slightly better than those from the interval-valued network. Conversely, for Version #2, the IVNNs outperform the triangular fuzzy number-valued networks in terms of V and S. The experimental results from this dataset do not conclusively determine whether the performance of IVNNs surpasses that of triangular fuzzy number neural networks with identical network structures, whether Version #1 or Version #2, or the reverse.

Table 5: Results of for the dataset '$y = 0.8 * \sin(x_1/4) * \sin(x_2/2)$'; V, $Q1$, $Q2$, and S for training and testing data, with the granularity again described in terms of interval values or triangular fuzzy numbers

Data	Information granule	Version	V	$Q1$	$Q2$	S
$y = 0.8. * \sin$ $(x_1/4). * \sin(x_2/2)$	Interval value	Version #1	0.5311/0.5188	0.9252/0.9161	0.5740/0.5663	0.2981/0.2878
		Version #2	0.7570/0.7345	0.9807/0.9504	0.7719/0.7723	0.3879/0.3524
	Triangular fuzzy number	Version #1	0.5425/0.5197	0.9691/0.9570	0.5598/0.5430	0.3756/0.3645
		Version #2	0.7289/0.7156	0.9735/0.9491	0.7487/0.7538	2658/0.2483

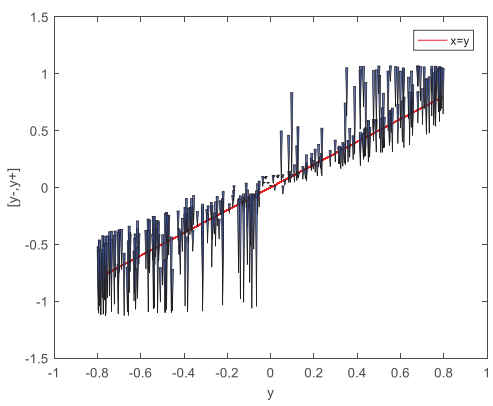
Furthermore, no regular pattern was observed in how the performance indices V and S change with variations in the number of nodes in the hidden layer. This suggests that the new GNNs proposed in this paper are feasible and perform better. To further validate these findings, experiments will be conducted on the seven datasets listed on the left side of Table 3 in the following section.

Fig. 7 provides an intuitive comparison of the performance of the four GNNs. The vertical lines in the figures indicate the network outputs (intervals, triangular fuzzy numbers), with a shorter average length correlating with a better Specificity index. Outputs that cover the theoretical outputs are marked with a slash across the vertical lines, indicating that more lines crossed by a slash signify a better Coverage index.

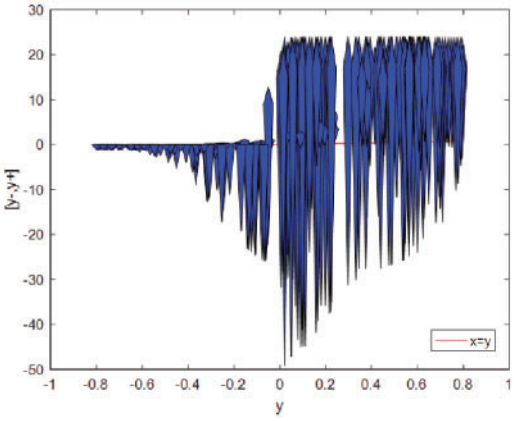


(a) IVNN for Version #1 (b) IVNN for Version #2

Figure 7: (Continued)



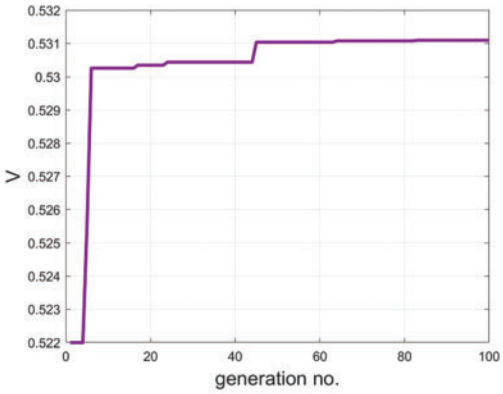
(c) Triangular fuzzy number-valued neural network for Version #1



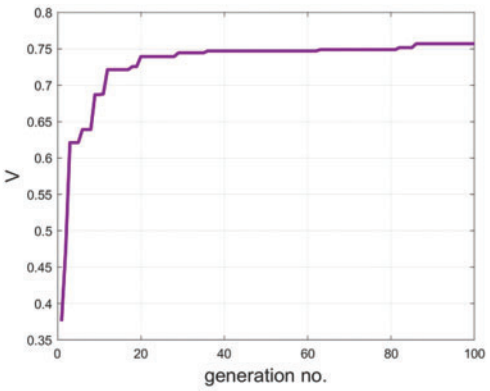
(d) Triangular fuzzy number-valued neural network for Version #2

Figure 7: Dataset '$y = 0.8 * \sin(x_1/4) * \sin(x_2/2)$'; x-axis—target output; y-axis—actual output (interval value/triangular fuzzy number value)

Fig. 8 presents the evolution of the fitness functions across 100 generations. Notably, the triangular fuzzy number-valued neural network for Version #2 exhibits the quickest convergence, achieving this within the initial twenty generations.

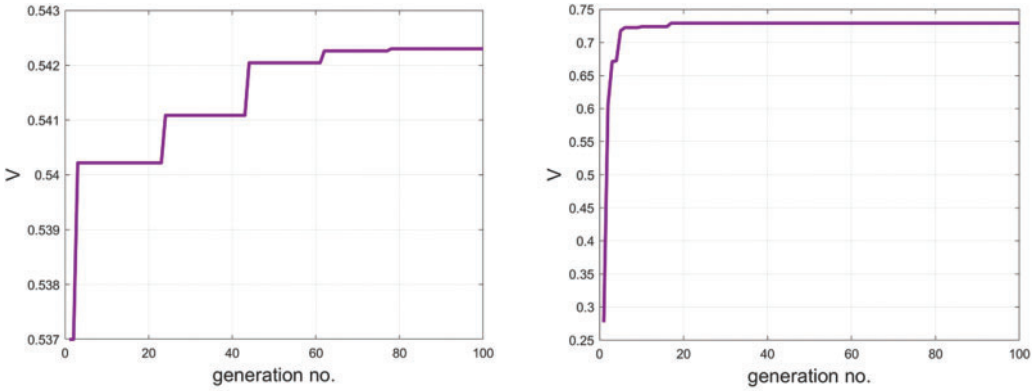


(a) IVNN for Version #1



(b) IVNN for Version #2

Figure 8: (Continued)



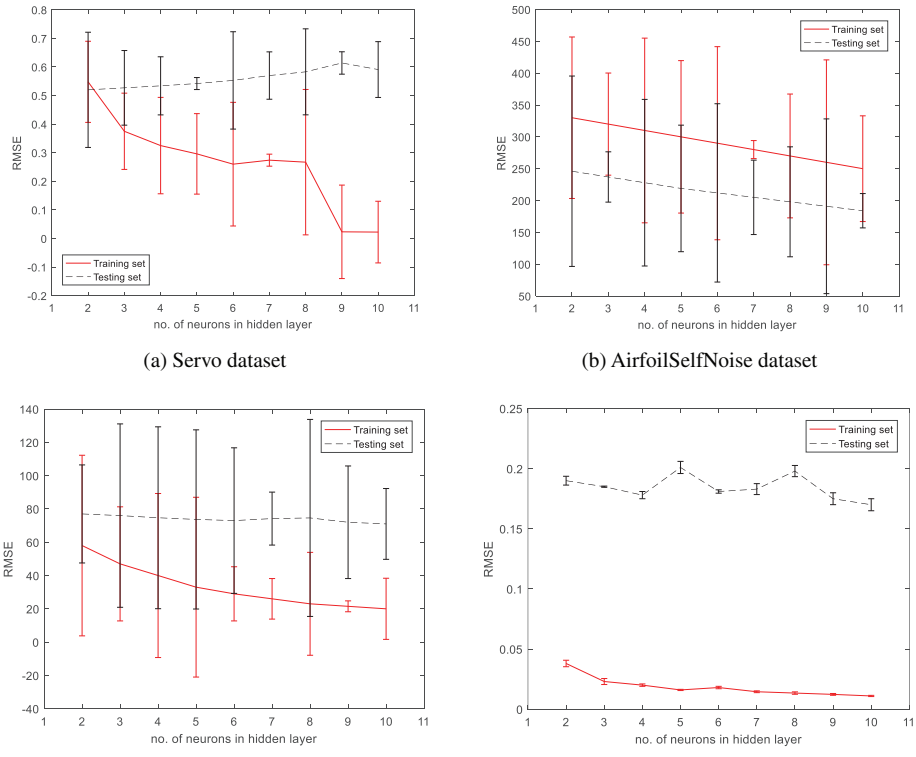
(c) Triangular fuzzy number-valued neural network for Version #1

(d) Triangular fuzzy number-valued neural network for Version #2

Figure 8: Fitness function of PSO in 100 generations

6.2 Publicly Available Data from the Machine Learning Repository

The performances of the constructed networks, functioning as variables of h, are illustrated in Fig. 9, where the size of h varies from 2 to 10. In version #1, irrespective of the GNN's utilization of interval values or triangular fuzzy numbers, h is assigned values of 2, 10, 10, 10, 2, 2, and 3 for the datasets Servo, Airfoil Self Noise, Housing, Fertility, Bodyfat, Automobile, and Auto MPG, respectively.



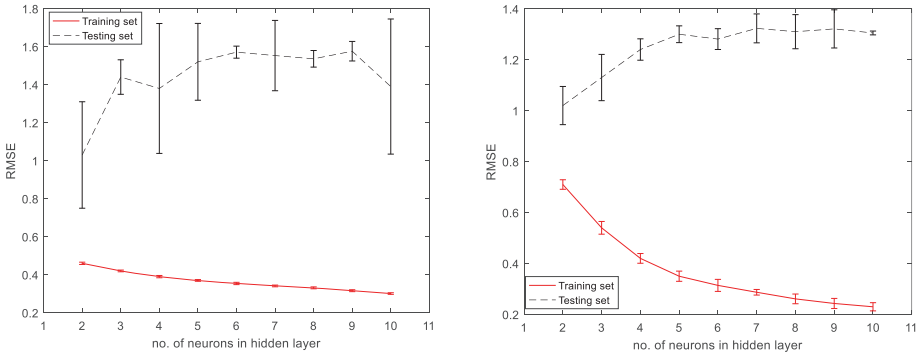
(a) Servo dataset

(b) AirfoilSelfNoise dataset

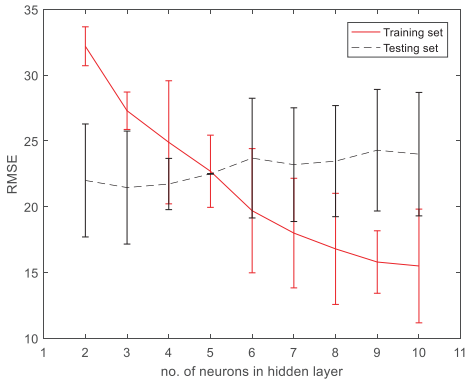
(c) Housing dataset

(d) Fertility dataset

Figure 9: (Continued)



(e) Bodyfat dataset (f) Automobile dataset

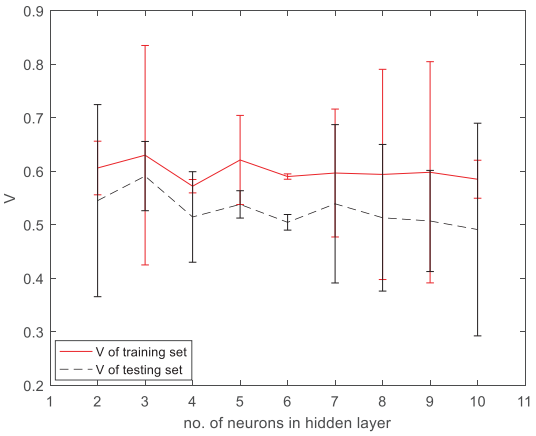


(g) Auto MPG dataset

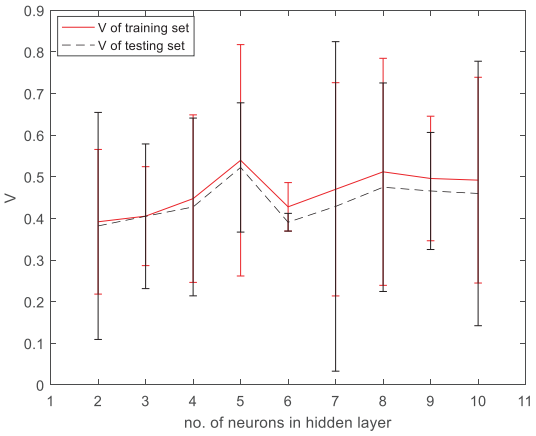
Figure 9: RMSE *vs. h* (number of neurons in hidden layer): Interval-valued/triangular fuzzy number-valued neural network (Version #1); training data (red line) and testing data (black line)

The index V, obtained by the constructed neural networks for Version #2 and quantified with respect to the number of neurons in the hidden layer, is displayed in Figs. 10 and 11. For the datasets Servo, Airfoil Self Noise, Housing, Fertility, Bodyfat, Automobile, and Auto MPG, the optimal h values for the interval-valued network are 3, 5, 8, 2, 5, 3, and 2, respectively, and for the triangular fuzzy number network, they are 3, 6, 2, 2, 10, 2, and 3, respectively. Consistent with the preceding section, there is no evidence to suggest that the performance index systematically varies as the number of neurons in the hidden layer changes.

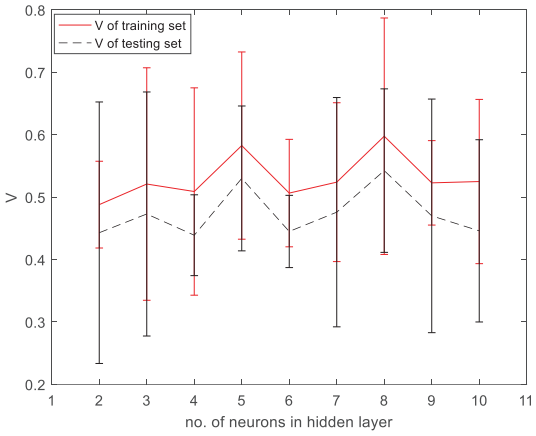
The determination of the value of h is achieved by minimizing the performance index. Subsequently, with the determined value of h, a network is constructed by selecting the value of ε within the range (0,1] for Version #1 and (0,10] for Version #2.



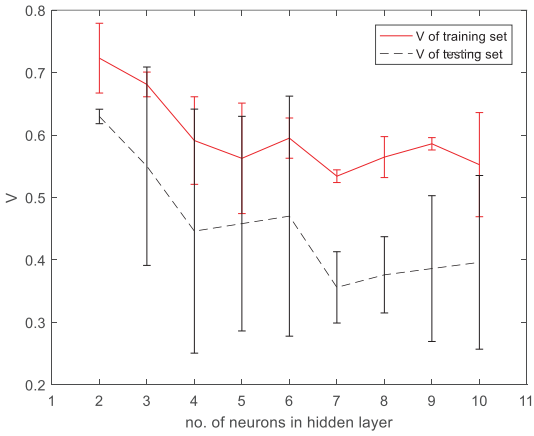
(a) Servo dataset



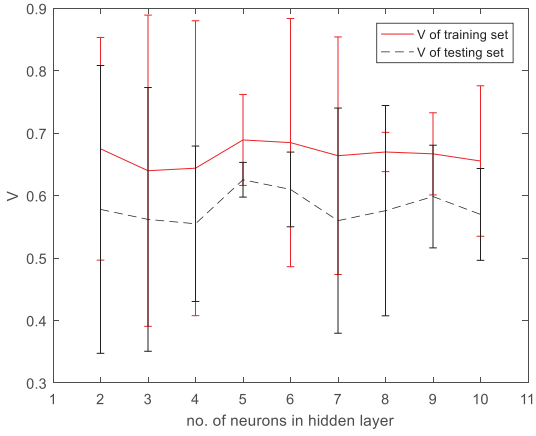
(b) AirfoilSelfNoise dataset



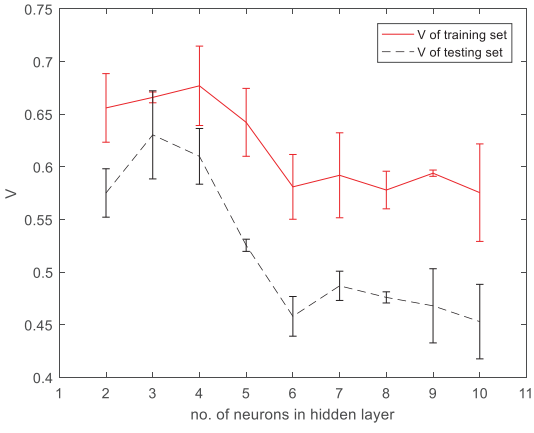
(c) Housing dataset



(d) Fertility dataset

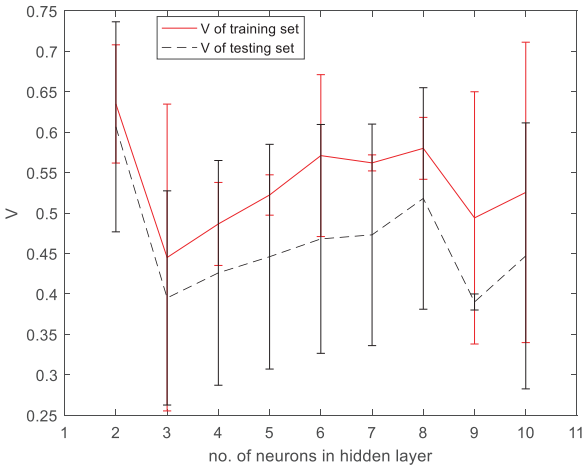


(e) Bodyfat dataset



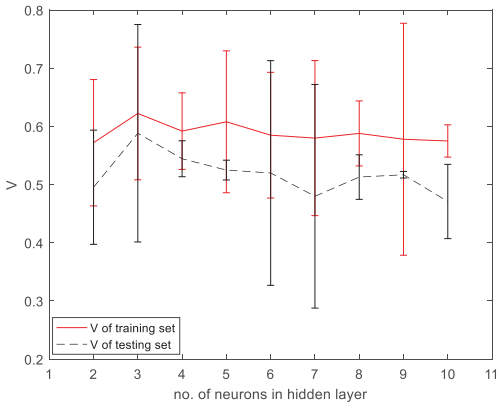
(f) Automobile dataset

Figure 10: (Continued)

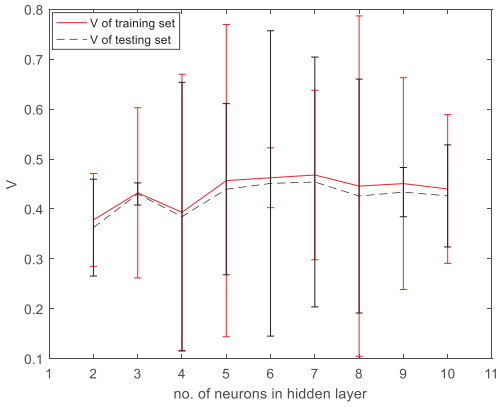


(g) Auto MPG dataset

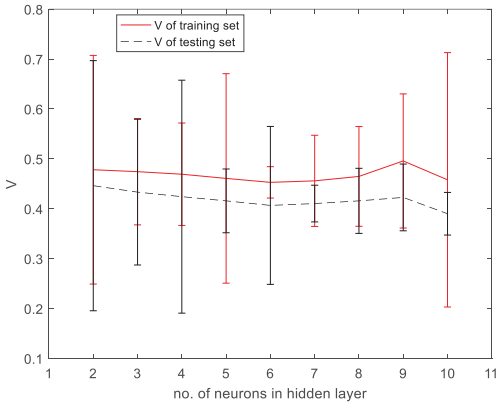
Figure 10: *V vs. h* (number of neurons in hidden layer): IVNN (Version #2)



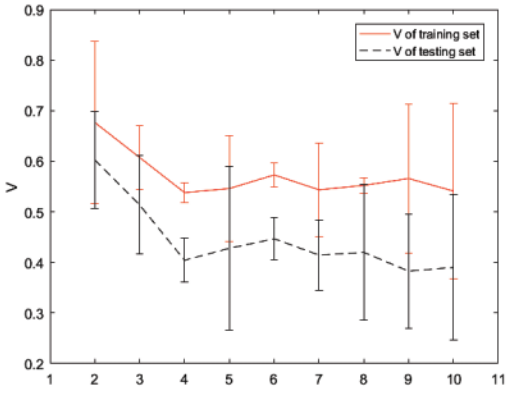
(a) Servo dataset



(b) AirfoilSelfNoise dataset

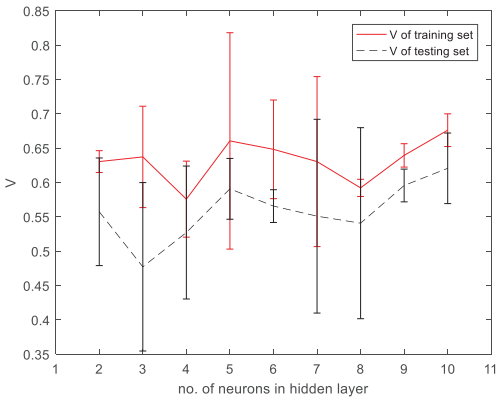


(c) Housing dataset

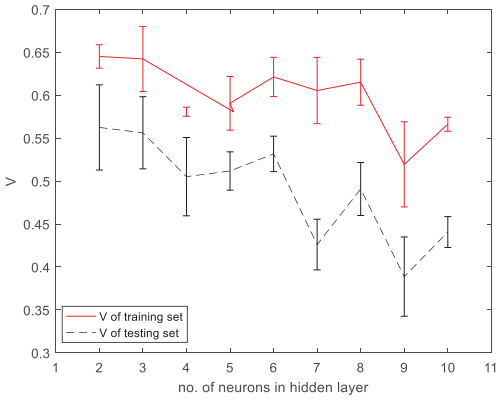


(d) Fertility dataset

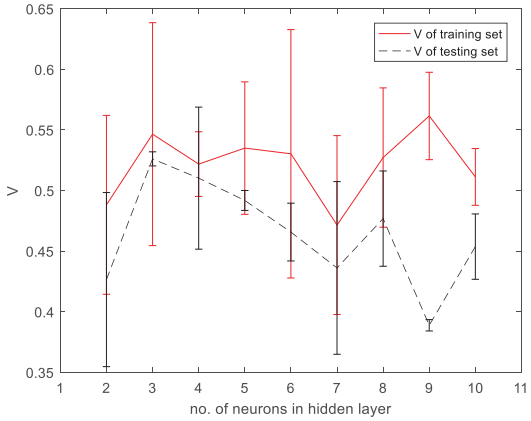
Figure 11: (Continued)



(e) Bodyfat dataset



(f) Automobile dataset



(g) Auto MPG dataset

Figure 11: *V vs. h* (number of neurons in hidden layer): Triangular fuzzy number-valued neural network (Version #2)

The dataset '$y = 0.8 * \sin(x_1/4) * \sin(x_2/2)$' has its experimental results previously discussed and visually represented in Fig. 12. The results of other data sets are all in Tables 6–12. Like earlier experiments, it can be concluded that for both training and testing sets, the indices V and S achieved by both interval-valued and triangular fuzzy number-valued neural networks for Version #2 surpass those obtained by the IVNN for Version #1. There is, however, no clear evidence to suggest a consistent superiority in performance between the interval-valued and triangular fuzzy number-valued neural networks when the network structure remains unchanged (whether for Version #1 or Version #2).

Nevertheless, other results indicate variable performances for indexes S and V. The IVNN for Version #2 shows the best performance for index V, while the triangular fuzzy number-valued neural network for Version #2 presents the second-best outcomes, except for the dataset 'Servo'. Notably, the best values of S are predominantly achieved by the IVNNs for Version #2 and the triangular fuzzy number-valued neural networks for Version #1. For all datasets, the IVNNs for Version #1 did not attain the optimal S values.

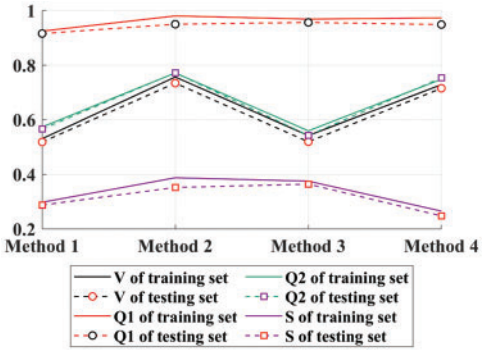


Figure 12: Plot of coverage, specificity, V, and S; solid line indicates training data; dotted line indicates testing data. Method 1 employs an IVNN for Version #1, Method 2 utilizes an IVNN for Version #2, Method 3 uses a triangular fuzzy number-valued neural network for Version #1, and Method 4 applies a triangular fuzzy number-valued neural network for Version #2

Table 6: Results for the servo dataset, including V, $Q1$, $Q2$, and S for both training and testing data. Information granularity is presented as interval values/triangular fuzzy numbers

Information granule	Version	V	$Q1$	$Q2$	S
Interval value	Version #1	0.5604/0.4957	0.8739/0.8246	0.6413/0.5990	0.4152/0.3572
	Version #2	0.6299/0.5909	0.8953/0.8701	0.7036/0.6818	0.4316/0.3972
Triangular fuzzy number	Version #1	0.4203/0.3427	0.6045/0.5599	0.7000/0.6191	0.4203/0.3427
	Version #2	0.6223/0.5883	0.8814/0.8590	0.7060/0.6832	0.3567/0.2830

Table 7: Results for the AirfoilSelfNoise dataset, including V, $Q1$, $Q2$, and S for training and testing data. Information granularity is provided as an interval

Information granule	Version	V	$Q1$	$Q2$	S
Interval value	Version #1	0.3367/0.3055	0.9693/0.9213	0.3474/0.3320	0.3127/0.2785
	Version #2	0.5397/0.5225	0.8518/0.8421	0.6336/0.6256	0.2451/0.2000
Triangular fuzzy number	Version #1	0.3712/0.3545	0.6652/0.6547	0.5580/0.5458	0.3372/0.2840
	Version #2	0.4625/0.4511	0.8254/0.8118	0.56030.5558	0.2758/0.2249

Table 8: Results for the housing dataset, including V, $Q1$, $Q2$, and S for training and testing data. The granularity of information is expressed as interval values/triangular fuzzy numbers

Information granule	Version	V	$Q1$	$Q2$	S
Interval value	Version #1	0.3726/0.3282	0.5431/0.4973	0.6861/0.6600	0.2384/0.2038
	Version #2	0.5976/0.5425	0.9036/0.8772	0.6614/0.6167	0.3582/0.2841
Triangular fuzzy number	Version #1	0.4029/0.3451	0.5825/0.5215	0.6917/0.6617	0.3178/0.2650
	Version #2	0.4782/0.4462	0.7748/0.7311	0.6172/0.5948	0.2752/0.2192

Table 9: Results for the fertility dataset, including V, $Q1$, $Q2$, and S for training and testing data. Information granularity is described as interval values/triangular fuzzy numbers

Information granule	Version	V	$Q1$	$Q2$	S
Interval value	Version #1	0.5247/0.2613	0.6815/0.4550	0.7699/0.5800	0.2983/0.2402
	Version #2	0.7231/0.6297	0.8162/0.7488	0.8859/0.8414	0.3530/0.3200
Triangular fuzzy number	Version #1	0.5563/0.3206	0.8451/0.7852	0.6583/0.4598	0.3947/0.3423
	Version #2	0.6765/0.6024	0.8352/0.7875	0.8100/0.7666	0.3167/0.2631

Table 10: Results for the bodyfat dataset, including V, $Q1$, $Q2$, and S for training and testing data. The information granularity is indicated as interval values/triangular fuzzy numbers

Information granule	Version	V	$Q1$	$Q2$	S
Interval value	Version #1	0.4778/0.3375	0.8152/0.6523	0.5861/0.5201	0.1624/0.1427
	Version #2	0.6893/0.6255	0.8551/0.8015	0.8061/0.7780	0.2764/0.2306
Triangular fuzzy number	Version #1	0.5001/0.3969	0.7663/0.6700	0.6526/0.5924	0.2149/0.1728
	Version #2	0.6762/0.6207	0.9132/0.8625	0.7405/0.7204	0.3049/0.2532

Table 11: Results for the automobile dataset, including V, $Q1$, $Q2$, and S for training and testing data. Information granularity is noted as interval values/triangular fuzzy numbers

Information granule	Version	V	$Q1$	$Q2$	S
Interval value	Version #1	0.2857/0.1957	0.5119/0.3902	0.5581/0.5014	0.2114/0.1740
	Version #2	0.6660/0.6304	0.9291/0.9059	0.7168/0.7345	0.3346/0.2965
Triangular fuzzy number	Version #1	0.3327/0.2776	0.6260/0.5854	0.5315/0.4742	0.1694/0.1460
	Version #2	0.6452/0.5625	0.9137/0.8728	0.7061/0.6505	0.2671/0.2335

Table 12: Results for the auto MPG dataset, including V, $Q1$, $Q2$, and S for training and testing data. Information granularity is specified as interval values/triangular fuzzy numbers

Information granule	Version	V	$Q1$	$Q2$	S
Interval value	Version #1	0.4509/0.3283	0.8548/0.7911	0.5275/0.4149	0.2814/0.2579
	Version #2	0.6349/0.6066	0.9073/0.8725	0.6998/0.6891	0.3489/0.3355
Triangular fuzzy number	Version #1	0.4757/0.3443	0.7151/0.6076	0.6652/0.5666	0.3415/0.3021
	Version #2	0.5465/0.5261	0.8713/0.8590	0.6272/0.6184	0.2868/0.2669

7 Conclusions and Further Work

The broad application of deep learning technology has sparked significant interest in the structural design of neural networks. GNNs offer a novel perspective for network structure design and can handle various data types. These networks, characterized by their granular connections, output granular data, thus providing a novel method for achieving higher-level abstract results. The role of information granularity is crucial in the design process of neural networks.

This paper introduces several new granular networks, building upon an established GNN. The work is summarized as follows: Firstly, it integrates two performance indices into a new one to serve as the network's performance index. Secondly, it optimizes the network directly to construct GNNs, moving away from NNNs. Thirdly, it explores the construction of connections using triangular fuzzy numbers. A series of experiments with synthetic and real data demonstrate that the new design scheme is viable and attains favorable performance indices.

Due to their simplicity, interval values and triangular fuzzy numbers are employed as the information granularity for granular connections. Exploring other forms of information granularity could offer alternatives for the network's granularity structure. Moreover, GNNs are not restricted to a single network architecture; multilayer perceptrons could also be considered. However, substituting numerical connections in the original neural network with granular connections increases the number of network parameters, complicating network optimization. Gradient-based methods may offer a promising solution.

Acknowledgement: The authors would like to express their gratitude for the valuable feedback and suggestions provided by all the anonymous reviewers and the editorial team.

Funding Statement: This work is partially supported by the National Key R&D Program of China under Grant 2018YFB1700104.

Author Contributions: Conceptualization, Witold Pedrycz, Yushan Yin and Zhiwu Li; Data curation, Yushan Yin; Formal analysis, Yushan Yin and Witold Pedrycz; Investigation, Yushan Yin; Methodology, Yushan Yin, Witold Pedrycz and Zhiwu Li; Software, Yushan Yin; Validation, Witold Pedrycz and Yushan Yin; Visualization, Yushan Yin; Writing–original draft, Yushan Yin and Witold Pedrycz. All authors reviewed the results and approved the final version of the manuscript.

Availability of Data and Materials: All the data sets (synthetic data sets and benchmark data sets) are from the UCI Machine Learning Repository (http://archive.ics.uci.edu/datasets).

Conflicts of Interest: The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] M. Song and Y. Wang, "A study of granular computing in the agenda of growth of artificial neural networks," *Granul. Comput.*, vol. 1, no. 4, pp. 1–11, 2016. doi: 10.1007/s41066-016-0020-7.

[2] K. Liu, L. Chen, J. Huang, S. Liu, and J. Yu, "Revisiting RFID missing tag identification," in *IEEE Conf. Comput. Commun.*, London, UK, 2022, pp. 710–719. doi: 10.1109/INFOCOM48880.2022.9796971.

[3] E. Olmez, E. Eriolu, and E. Ba, "Bootstrapped dendritic neuron model artificial neural network for forecasting," *Granul. Comput.*, vol. 8, no. 6, pp. 1689–1699, 2023. doi: 10.1007/s41066-023-00390-1.

[4] W. Pedrycz and G. Vukovich, "Granular neural networks," *Neurocomput.*, vol. 36, no. 1–4, pp. 205–224, 2001. doi: 10.1016/S0925-2312(00)00342-8.

[5] M. Beheshti, A. Berrached, A. de Korvin, C. Hu, and O. Sirisaengtaksin, "On interval weighted three-layer neural networks," in *Proc. 31st Ann. Simul. Symp.*, Boston, MA, USA, 1998, pp. 188–194. doi: 10.1109/SIMSYM.1998.668487.

[6] H. Ishibuchi, H. Tanaka, and H. Okada, "An architecture of neural networks with interval weights and its application to fuzzy regression analysis," *Fuzzy Sets Syst.*, vol. 57, no. 1, pp. 27–39, 1993. doi: 10.1016/0165-0114(93)90118-2.

[7] S. Ridella, S. Rovetta, and R. Zunino, "IAVQ-interval-arithmetic vector quantization for image compression," *IEEE Trans. Circ. Syst. II: Analog Digit. Signal Process.*, vol. 47, no. 12, pp. 1378–1390, 2000. doi: 10.1109/82.899630.

[8] M. Song and W. Pedrycz, "Granular neural networks: Concepts and development schemes," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 24, no. 4, pp. 542–553, 2013. doi: 10.1109/TNNLS.2013.2237787.

[9] S. Nabavi-Kerizi, M. Abadi, and E. Kabir, "A PSO-based weighting method for linear combination of neural networks," *Comput. Electric. Eng.*, vol. 36, no. 5, pp. 886–894, 2012. doi: 10.1016/j.compeleceng.2008.04.006.

[10] T. M. Shami, A. A. El-Saleh, M. Alswaitti, Q. Al-Tashi, M. A. Summakieh and S. Mirjalili, "Particle swarm optimization: A comprehensive survey," *IEEE Access*, vol. 10, pp. 10031–10061, 2022. doi: 10.1109/ACCESS.2022.3142859.

[11] Z. Anari, A. Hatamlou, and B. Anari, "Automatic finding trapezoidal membership functions in mining fuzzy association rules based on learning automata," *Int. J. Interact. Multimed. Artif. Intell.*, vol. 7, no. 4, pp. 27–43, 2022. doi: 10.9781/ijimai.2022.01.001.

[12] Y. N. Pawan, K. B. Prakash, S. Chowdhury, and Y. Hu, "Particle swarm optimization performance improvement using deep learning techniques," *Multimed. Tool Appl.*, vol. 81, no. 19, pp. 27949–27968, 2022. doi: 10.1007/s11042-022-12966-1.

[13] C. Fonseca and P. Fleming, "Multiobjective optimization and multiple constraint handling with evolutionary algorithms. I. A unified formulation," *IEEE Trans. Syst., Man, Cybernet.—Part A: Syst. Human.*, vol. 28, no. 1, pp. 26–37, 1998. doi: 10.1109/3468.650319.

[14] H. Rezk, J. Arfaoui, and M. R. Gomaa, "Optimal parameter estimation of solar PV panel based on hybrid particle swarm and grey wolf optimization algorithms," *Int. J. Interact. Multimed. Artif. Intell.*, vol. 6, no. 6, pp. 145–155, 2021. doi: 10.9781/ijimai.2020.12.001.

[15] L. A. Zadeh and J. Kacprzyk, "Words about uncertainty: Analogies and contexts," in *Computing with Words in Information Intelligent Systems*, Heidelberg, Germany: Physica-Verlag, 1999, pp. 119–135.

[16] W. Pedrycz and M. Song, "Analytic hierarchy process (AHP) in group decision making and its optimization with an allocation of information granularity," *IEEE Trans. Fuzzy Syst.*, vol. 19, no. 3, pp. 527–539, 2011. doi: 10.1109/TFUZZ.2011.2116029.

[17] Z. Kuang, X. Zhang, J. Yu, Z. Li, and J. Fan, "Deep embedding of concept ontology for hierarchical fashion recognition," *Neurocomput.*, vol. 425, no. 4, pp. 191–206, 2020. doi: 10.1016/j.neucom.2020.04.085.

[18] Y. Wan, G. Zou, C. Yan, and B. Zhang, "Dual attention composition network for fashion image retrieval with attribute manipulation," *Neural Comput. Appl.*, vol. 35, no. 8, pp. 5889–5902, 2023. doi: 10.1007/s00521-022-07994-9.

[19] S. Lee and E. Lee, "Fuzzy sets and neural networks," *J. Cybernet.*, vol. 4, no. 2, pp. 83–103, 1974. doi: 10.1080/01969727408546068.

[20] A. M. Abadi, D. U. Wutsqa, and N. Ningsih, "Construction of fuzzy radial basis function neural network model for diagnosing prostate cancer," *Telkomnika (Telecommun. Comput. Electron. Control)*, vol. 19, no. 4, pp. 1273, 2021. doi: 10.12928/telkomnika.v19i4.20398.

[21] S. A. Shanthi and G. Sathiyapriya, "Universal approximation theorem for a radial basis function fuzzy neural network," *Mater. Today: Proc.*, vol. 51, no. 4, pp. 2355–2358, 2022. doi: 10.1016/j.matpr.2021.11.576.

[22] K. H. Chen, L. Yang, J. P. Wang, S. I. Lin, and C. L. Chen, "Automatic manpower allocation for public construction projects using a rough set enhanced neural network," *Canadian J. Civil Eng.*, vol. 1, no. 8, pp. 1020–1025, 2020. doi: 10.1139/cjce-2019-0561.

[23] M. Song, L. Hu, S. Feng, and Y. Wang, "Feature ranking based on an improved granular neural network," *Gran. Comput.*, vol. 8, no. 1, pp. 209–222, 2023. doi: 10.1007/s41066-022-00324-3.

[24] M. Wang *et al.*, "Data-driven strain-stress modelling of granular materials via temporal convolution neural network," *Comput. Geotechn.*, vol. 152, no. 5, pp. 105049, 2022. doi: 10.1016/j.compgeo.2022.105049.

[25] N. M. Lanbaran and E. Elik, "Prediction of breast cancer through tolerance-based intuitionistic fuzzy-rough set feature selection and artificial neural network," *Gazi Univ. J. Sci.*, 2021. doi: 10.35378/GUJS.857099.

[26] A. Hakan, C. Resul, T. Erkan, D. Besir, and K. Korhan, "Advanced control of three-phase PWM rectifier using interval type-2 fuzzy neural network optimized by modified golden sine algorithm," *IEEE Trans. Cybernet.*, vol. 52, no. 5, pp. 2994–3005, 2023. doi: 10.1109/TCYB.2020.3022527.

[27] H. J. He, X. Meng, J. Tang, and J. F. Qiao, "Event-triggered-based self-organizing fuzzy neural network control for the municipal solid waste incineration process," *Sci. China Technol. Sc.*, vol. 66, pp. 1096–1109, 2023.

[28] J. Cao, X. Xia, L. Wang, and Z. Zhang, "A novel back propagation neural network optimized by rough set and particle swarm algorithm for remanufacturing service provider classification and selection," *J. Phys.: Conf. Series*, vol. 2083, no. 4, pp. 042058, 2021. doi: 10.1088/1742-6596/2083/4/042058.

[29] A. Sheikhoushaghi, N. Y. Gharaei, and A. Nikoofard, "Application of rough neural network to forecast oil production rate of an oil field in a comparative study," *J. Pet. Sci. Eng.*, vol. 209, no. 1, pp. 109935, 2022. doi: 10.1016/j.petrol.2021.109935.

[30] A. Kumar and P. S. V. S. S. Prasad, "Incremental fuzzy rough sets based feature subset selection using fuzzy min-max neural network preprocessing," *Int. J. Approx. Reason.*, vol. 139, no. 3, pp. 69–87, 2021. doi: 10.1016/j.ijar.2021.09.006.

[31] A. Vasilakos and D. Stathakis, "Granular neural networks for land use classification," *Soft Comput.*, vol. 9, no. 5, pp. 332–340, 2005. doi: 10.1007/s00500-004-0412-5.

[32] Z. Yu *et al.*, "Refined fault tolerant tracking control of fixed-wing UAVs via fractional calculus and interval type-2 fuzzy neural network under event-triggered communication," *Inf. Sci.*, vol. 644, no. 5, pp. 119276, 2023. doi: 10.1016/j.ins.2023.119276.

[33] Y. Liu, J. Zhao, W. Wang, and W. Pedrycz, "Prediction intervals for granular data streams based on evolving type-2 fuzzy granular neural network dynamic ensemble," *IEEE Trans. Fuzzy Syst.*, vol. 29, no. 4, pp. 874–888, 2021. doi: 10.1109/TFUZZ.2020.2966172.

[34] W. Ding, M. Abdel-Basset, H. Hawash, and W. Pedrycz, "Multimodal infant brain segmentation by fuzzy-informed deep learning," *IEEE Trans. Fuzzy Syst.*, vol. 30, no. 4, pp. 1088–1101, 2022. doi: 10.1109/TFUZZ.2021.3052461.

[35] L. Zhang, M. Zhong, and H. Han, "Detection of sludge bulking using adaptive fuzzy neural network and mechanism model," *Neurocomputing*, vol. 481, pp. 193–201, 2022. doi: 10.3389/978-2-88974-540-1.

[36] D. Sánchez, P. Melin, and O. Castillo, "Optimization of modular granular neural networks using a hierarchical genetic algorithm based on the database complexity applied to human recognition," *Inf. Sci.*, vol. 309, no. 2, pp. 73–101, 2015. doi: 10.1016/j.ins.2015.02.020.

[37] D. Sánchez and P. Melin, "Optimization of modular granular neural networks using hierarchical genetic algorithms for human recognition using the ear biometric measure," *Eng. Appl. Artif. Intell.*, vol. 27, no. 11, pp. 41–56, 2014. doi: 10.1016/j.engappai.2013.09.014.

[38] E. Weerdt, Q. Chu, and J. Mulder, "Neural network output optimization using interval analysis," *IEEE Trans. Neural Netw.*, vol. 20, no. 4, pp. 638–653, 2009. doi: 10.1109/TNN.2008.2011267.

[39] M. Mizumoto and K. Tanaka, "The four operations of arithmetic on fuzzy numbers, systems computing," *Controls*, vol. 7, no. 5, pp. 73–81, 1976.

[40] M. Rocha, P. Cortez, and J. Neves, "Evolutionary design of neural networks for classification and regression," in *Adaptive Natural Computing Algorithms*, Coimbra, Portugal, 2005, pp. 304–307. doi: 10.1007/3-211-27389-1_73.

[41] M. T. Hagan and M. B. Menhaj, "Training feedforward networks with the Marquardt algorithm," *IEEE Trans. Neural Netw.*, vol. 5, no. 6, pp. 989–993, 1994. doi: 10.1109/72.329697.

[42] E. Zitzler, K. Deb, and L. Thiele, "Comparison of multiobjective evolutionary algorithms: Empirical results," *Evol. Comput.*, vol. 8, no. 2, pp. 173–195, 2000. doi: 10.1162/106365600568202.

[43] D. Wedge, D. Ingram, D. McLean, C. Mingham, and Z. Bandar, "On global-local artificial neural networks for function approximation," *IEEE Trans. Neural Netw.*, vol. 17, no. 4, pp. 942–952, 2006. doi: 10.1109/TNN.2006.875972.