Soft Computing
https://doi.org/10.1007/s00500-023-07899-2

SOFT COMPUTING IN DECISION MAKING AND IN MODELING IN ECONOMICS

Complex nonlinear neural network prediction with IOWA layer

Walayat Hussain[1] · Jóse M. Merigó[2] · Jaime Gil-Lafuente[3] · Honghao Gao[4]

Accepted: 30 January 2023

Abstract
Neural network methods are widely used in business problems for prediction, clustering, and risk management to improving customer satisfaction and business outcome. The ability of a neural network to learn complex nonlinear relationship is due to its architecture that uses weight parameters to transform input data within the hidden layers. Such methods perform well in many situations where the ordering of inputs is simple. However, for a complex reordering of a decision-maker, the process is not enough to get an optimal prediction result. Moreover, existing machine learning algorithms cannot reduce computational complexity by reducing data size without losing any information. This paper proposes an induced ordered weighted averaging (IOWA) operator for the artificial neural network IOWA-ANN. The operator reorders the data according to the order-inducing variable. The proposed sorting mechanism in the neural network can handle a complex nonlinear relationship of a dataset, which results in reduced computational complexities. The proposed approach deals with the complexity of the neuron, collects the data and allows a degree of customisation of the structure. The application further extended to IGOWA and Quasi-IOWA operators. We present a numerical example in a financial decision-making process to demonstrate the approach's effectiveness in handling complex situations. This paper opens a new research area for various complex nonlinear predictions where the dataset is big enough, such as cloud QoS and IoT sensors data. The approach can be used with different machine learning, neural networks or hybrid fuzzy neural methods with other extensions of the OWA operator.

Keywords Complex prediction · Financial forecasting · IOWA operator · Financial decision-making · Optimisation · Aggregation operator · Neural network · Computational complexity

1 Introduction

The opportunities to utilise neural network have significantly increased across different industries and business sectors over the last few decades. The neural network architecture comprises layers of neurons—an input layer, one or multiple hidden layers and an output layer connected across different layers. Input neurons have a set of features that a neural network use for prediction. The number of output neurons depends on the number of forecasts required. A single prediction value such as a regression or binary classification needs a single output neuron. In multivariate regression or multi-class classification, the number of neurons depends on per predicted value. The number of hidden layers depends on the complexity of a problem domain. The more complex the problem is, the higher the number of hidden layers. Each neuron in hidden layers takes input data and multiplies it with a certain weight, adding bias and sending it to the next

✉ Walayat Hussain
walayat.hussain@acu.edu.au

Jóse M. Merigó
jose.merigo@uts.edu.au

Jaime Gil-Lafuente
j.gil@ub.edu

Honghao Gao
gaohonghao@shu.edu.cn

[1] Peter Faber Business School, Australian Catholic University, North Sydney 2060, Australia

[2] University of Technology Sydney, Sydney 2007, Australia

[3] Business School, University of Barcelona, 08007 Barcelona, Spain

[4] School of Computer Engineering and Science, Shanghai University, Shanghai 200444, China

Published online: 20 February 2023

layer. Weights represent a strength of connections that decide the amount of influence the input will have on the output. The efficiency of a neural network depends on its learning process. Gradient-based and stochastic methods are commonly used in training methods (Wang et al. 2015). However, these methods have their issues. One of the main disadvantages of the gradient-based method is its slow performance with complex code. Other disadvantages of the gradient-based methods are inaccurate gradient, intolerance of difficulties, and high dependency on initial parameters and categorical variables (Zingg et al. 2008). Many methods (Gao et al. 2020; Abdel-Basset et al. 2018) tried to optimise the connection weights (Gao et al. 2020). Metaheuristic algorithms such as genetic algorithm (GA), particle swarm optimization (PSO), water waves optimization (WWO) and others are commonly used to solve various sophisticated optimisation problems (Abdel-Basset et al. 2018). Multiple research (Wu et al. 2020; Zhang et al. 2020; Panahi et al. 2020) have used a hybrid approach to combine various metaheuristic algorithms with machine learning and neural network algorithms. Panahi et al. (Panahi et al. 2020) used SVR, ANFIS, Bee algorithm and grey wolf optimiser (GWO) to predict landslide susceptibility. In another approach, Zhang, Huang, Wang and Ma (Zhang et al. 2020) used machine learning and multi-objective PSO to predict concrete mixture properties. Although discussed methods attempted to address the problem, most have ignored handling the complex reordering of a decision-making process. Moreover, for complex nonlinear prediction such as cloud SLA Quality of Service (QoS) prediction, the data have multiple dimensions (Hussain et al. 2017, 2018). The computational complexity of the system increases with the increase of a dataset (Cheng et al. 2013). Discussed complexity handling methods do not devise a mechanism to improve computational complexity by reducing the data without losing meaningful information. One way to deal with the problem is reordering inputs based on the OWA weights.

The ordered weighted average (OWA) introduced by Yager (Yager 1988) is an aggregation operator that can aggregate uncertain information with specific weights. The OWA operator has been used in a wide range of applications to solve different problem domains such as environmental applications (Bordogna et al. 2011), resource management (Zarghami et al. 2008) and various business applications (Merigó and Gil-Lafuente 2010, 2011). The OWA has been used extensively to solve different fuzzy application (Kacprzyk et al. 2019; Cho 1995; Merigó et al. 2018). Yager (Yager and Filev 1999) introduced a more general type of OWA operator-induced OWA (IOWA) operator in which the ordering of arguments is based on the arrangement of the induced variable. The weights in the IOWA operator, rather than associated with a specific argument, are linked with the position in the ordering. It enables the approach to deal with the complex characteristics of the argument. Motivated by these, Merigó and Gil-Lafuente (2009) proposed induced generalised OWA (IGOWA) and Quasi-IOWA (QIOWA) operators that combine the characteristics of IOWA and GOWA operator. Moreover, the operator has received significant growing interest among the scientific community (Flores-Sosa et al. 2020; Jin et al. 2020; Yi and Li 2019) and applied to address number of real-life issues (Maldonado et al. 2019; Blanco-Mesa et al. 2020).

To address the complex issue of a large dataset for nonlinear prediction, Yager (1993) introduced the OWA layer in the neural network. The input data are arranged so that the largest input goes to the first branch, the second largest to the next branch and so on. Using the same concept Kaur et al. (2016, 2014) used the OWA layer in ANFIS to predict future stock price. Similarly, Cheng et al. (2013) used the OWA layer in ANFIS to predict TAIEX stock data. These approaches work well for different simple reordering and decision-making processes. However, to deal with other complex reordering processes, the approaches are not enough to deal efficiently. Hence, to address the problem, this work aims to introduce an induced OWA (IOWA) operator as an additional layer in ANN to reorder the inputs based on the ordered inducing variable. Bo et al. (Li et al. 2021) have demonstrated the application of the IOWA layer with the fruit fly algorithm to predict vegetable price prediction. Hussain et al. (2022a, 2022b, 2022c) have used OWA layer in several prediction algorithms to predict complex stock market and cloud QoS data (Hussain et al. 2022d, 2021). This paper theoretically contributes the use of the IOWA layer to any neural network prediction methods. The IOWA operator reorders inputs not only based on the value of the argument, but it reorders inputs based on the associated order-inducing variables (Merigó and Gil-Lafuente 2009). The proposed approach makes the decision-making process more efficient and enables the system to handle different complex reordering of the decision-maker. It can be further generalised to IGOWA and Quasi-IOWA operators (Merigó and Gil-Lafuente 2009). The performance of the proposed approach is presented through a financial case study. The evaluative results demonstrate the effectiveness of the approach in dealing with complex behaviour and optimal prediction results.

The rest of the paper is organised as follows. Section 2 briefly discusses some basic concept such as OWA, IOWA, IGOWA, Quasi-IOWA and neural network. Section 3 presents a theoretical discussion of the IOWA layer in ANN. Section 4 presents the extension of IOWA-ANN towards IGOWA and Quasi-IOWA operators. Section 5

evaluates the approach, and Section 6 concludes the paper with future research direction.

2 Conceptual background

This section discusses some preliminary definitions such as OWA, IOWA, IGOWA, Quasi-IOWA operators, and OWA in ANN.

2.1 Ordered weighted averaging (OWA) operator

Yager (1988) introduced the OWA operator as a family of aggregation operators such as arithmetic mean, median, minimum and maximum. The operator can obtain optimum weights based on aggregated arguments. Definition 1 presents the OWA operator.

Definition 1 An OWA operator of dimension n is a mapping OWA: $R^n \rightarrow R$ that has an associated weighting vector W = [w1,w2,…… wn] such that $w_i \in [0,1]$ and $\sum_{i=1}^{n} w_i = 1$. Equation 1 presents the OWA operator.

$$OWA(x_1, x_2, x_3, \ldots, x_n) = \sum_{i=1}^{n} w_i y_i \tag{1}$$

where $(y_1, y_2, y_3, \ldots, y_n)$ is the reordered set of inputs $(x_1, x_2, x_3, \ldots, x_n)$ from largest to smallest inputs.

One can generalise the direction of reordering between ascending OWA (AOWA) and descending OWA (DOWA) operators.

2.2 Induced ordered weighted averaging (IOWA) operator

The extension of the OWA operator, induced OWA (IOWA) operator introduced by Yager and Filev (1999). The distinct feature of IOWA is that instead of depending on the argument's value, the reordering is performed using order induced variable. Definition 2 presents the IOWA operator.

Definition 2 The IOWA operator of dimension n is a mapping IOWA: $R^n \rightarrow R$ that has an associated set of weighting vectors of dimension n, such that $w_i \in [0,1]$ and $\sum_{i=1}^{n} w_i = 1$, and the set of order-inducing variables u_i, presented in Eq. 2:

$$IOWA(\langle u_1, a_1\rangle, \langle u_2, a_2\rangle, \ldots, \langle u_n, a_n\rangle) = \sum_{j=1}^{n} w_j c_j \tag{2}$$

where $(c_1, c_2, c_3, \ldots, c_n)$ is the reorder input arguments $(a_1, a_2, a_3, \ldots, a_n)$ in decreasing order of the value of u_i.

2.3 Induced generalised ordered weighted averaging (IGOWA) operator

Using the characteristics of IOWA and generalised OWA (GOWA), Merigó and Gil-Lafuente (2009) introduced IGOWA. IGOWA operator uses inducing variables to reorder the position of arguments and generalised means. Definition 3 presents the IGOWA operator.

Definition 3 The IGOWA operator of dimension n is a mapping IGOWA: $R^n \rightarrow R$ defined by the association of weights of dimension n such that $w_i \in [0,1]$ and $\sum_{i=1}^{n} w_i = 1_i$, and the set of ordered inducing variables u_i, and the parameters $\gamma \in (-\infty, \infty)$. Equation 3 presents the IGOWA operator:

$$IGOWA(\langle u_1, x_1\rangle, \langle u_2, x_2\rangle, \ldots, \langle u_n, x_n\rangle) = \left(\sum_{j=1}^{n} w_j y_j^{\gamma}\right)^{1/\gamma} \tag{3}$$

where $(y_1, y_2, y_3, \ldots, y_n)$ is the reorder of argument variables $(x_1, x_2, x_3, \ldots, x_n)$ in decreasing order of the order-inducing variables u_i.

2.4 Quasi- IOWA operator

The Quasi-IOWA is an extension of IGOWA introduced by Merigo and Gil-Lafuente (Merigó and Gil-Lafuente 2009). The operator is the generalisation of IGOWA with quasi-arithmetic means. The operator gives complete generalisation, by including a large number of cases that are not included in the IGOWA operators. Definition 4 presents the Quasi-IOWA operator.

Definition 4 The Quasi-IOWA operator of dimension n is a mapping QIOWA: $R^n \rightarrow R$ defined by the association of weights of dimension n such that $w_i \in [0,1]$ and $\sum_{i=1}^{n} w_i = 1$, and for strictly monotonic continuous function $q(y)$. Equation 4 presents the Quasi-IOWA operator:

$$QIOWA(\langle u_1, x_1\rangle, \langle u_2, x_2\rangle, \ldots, \langle u_n, x_n\rangle) = q^{-1}\left(\sum_{j=1}^{n} w_j q(y_j)\right) \tag{4}$$

where y_j are the argument value of the Quasi-IOWA pairs $\langle u_i, x_i\rangle$, are arranged in decreasing order of the order-inducing variables u_i.

One can generalise the direction of the reordering and distinguish between the descending Quasi-IOWA operator (Quasi-DIOWA) and the ascending operator (Quasi-AIOWA).

2.5 Artificial neural network (ANN)

ANN comprised of a number of interconnected neurons that process the input x to give desired output y. Each neuron in one layer is connected to the other neurons in the next layer through connection weights w, and the bias b is added to increase or decrease input weights. It is presented in Eq. 5:

$$Y = w(b) + \sum_{j=1}^{n} x_j w_j \tag{5}$$

Hebb (Hebb 1949) proposed a Hebbian learning algorithm in which the weights get increase proportionally to the product of input x and output y, as presented in Eq. 6:

$$w(new) = w(old) + x * y \tag{6}$$

Stephen (Stephen 1990) proposed perceptron networks for single and multiple perceptron networks. The input layer connected with the output layer with weights -1, 0 or 1. The weights are updated between the hidden layer and the output layer to reduce the loss and get the target output, as presented in Eq. 7:

$$w(new) = w(old) + \alpha * t * x \tag{7}$$

where α is the learning rate, and t is the target output.

Similarly, Widrow and Lehr (Widrow and Lehr 1993) proposed Widrow Hoff learning algorithm or least mean square (LMS) rule or delta rule that follows gradient descent rule for linear regression. This algorithm follows multiple adaptive linear neural networks (MADALINE) that update the connection weight between the current and targeted output value, as presented in Eq. 8:

$$w = \alpha * x(t - y) \tag{8}$$

where α is the learning rate, x is the input value, y is the output value, and t is the target output.

2.6 OWA operator in ANN

The addition of the OWA layer in ANN was proposed by Yager (Yager 1993) to reorder arguments based on their respective weights (Yager 1994). Definition 5 presents the OWA operator in ANN:

Definition 5 The OWA operator in ANN of a single dimension is a mapping OWA: $R^n \rightarrow R$ defined by the associated weights of dimension n such that $w_i \in [0,1]$ and $\sum_{i=1}^{n} w_i = 1$. Assuming that we have m parts of a data each have $n + 1$ tuple of values (a_{i1}, a_{i2}, $a_{i3,\ldots\ldots\ldots}$, a_{in}, y_i), where a_{ij} are input values (aggregation) of the ith sample, and y_i is the aggregated value for this i^{th} sample.

3 IOWA layer in artificial neural network

The use of the IOWA operator in ANN is the extension of OWA in ANN. The primary difference between the approach and OWA-ANN is the inclusion of an inducing variable that is used to reorder arguments for better decision-making in a different complex situation. It is defined as:

Definition 6 The IOWA operator in ANN of n dimension input is a mapping IOWA: $R^n \rightarrow R$ defined by the associated weights w of dimension n such that $w_i \in [0,1]$ and $\sum_{i=1}^{n} w_i = 1$ the set of inducing variables of order ui, as presented in ?tic=?>Fig. 1 and Eqs. 9–12.

$$IOWA - NN(\langle u_1, a_1\rangle, \langle u_2, a_2\rangle, \ldots, \langle u_n, a_n\rangle) = p_i \tag{9}$$

p_i is the activation function which is the sum of the product of w_i and b_i that is

$$pi = \sum_{i=1}^{n} w_i b_i \tag{10}$$

where $\langle u_i, a_i\rangle$ is a set of two tuple input, where u_i is inducing variable associated with the input a_i, b_i is the reordered input a_i in descending order of the u_i, w_i is the associated a_i weight, and y_i is the actual output of the output neuron.

As in the IOWA operator, if the weights are not normalised, that is $\sum_{i=1}^{n} w_i \neq 1$, then the activation function p_i is presented as

$$pi = \frac{1}{W} \sum_{j=1}^{n} w_j b_j \tag{11}$$

When the calculated value is greater than or equal to the threshold value, then it is transmitted to the neurons of the next layer, otherwise not, as presented in Eq. 12:

$$pi \geq \theta_i, yi > 0 \vee pi < \theta_i, yi = 0 \tag{12}$$

The IOWA operator has the properties of—commutative, monotonic, idempotent and bounded. These properties can be proved from Theorem 1–4.

Theorem 1 (*Monotonicity*): *The monotonicity of the IOWA-ANN is presented in Eqs.* 13–15:

Let f be the IOWA operator in ANN that calculates the value of i. If $a_i \geq x_i$ for all $a_{i,}$ then

$$f(\langle u_1, a_1\rangle, \langle u_2, a_2\rangle, \ldots, \langle u_n, a_n\rangle) \geq f(\langle u_1, x_1\rangle, \langle u_2, x_2\rangle, \ldots, \langle u_n, x_n\rangle) \tag{13}$$

Proof. Let

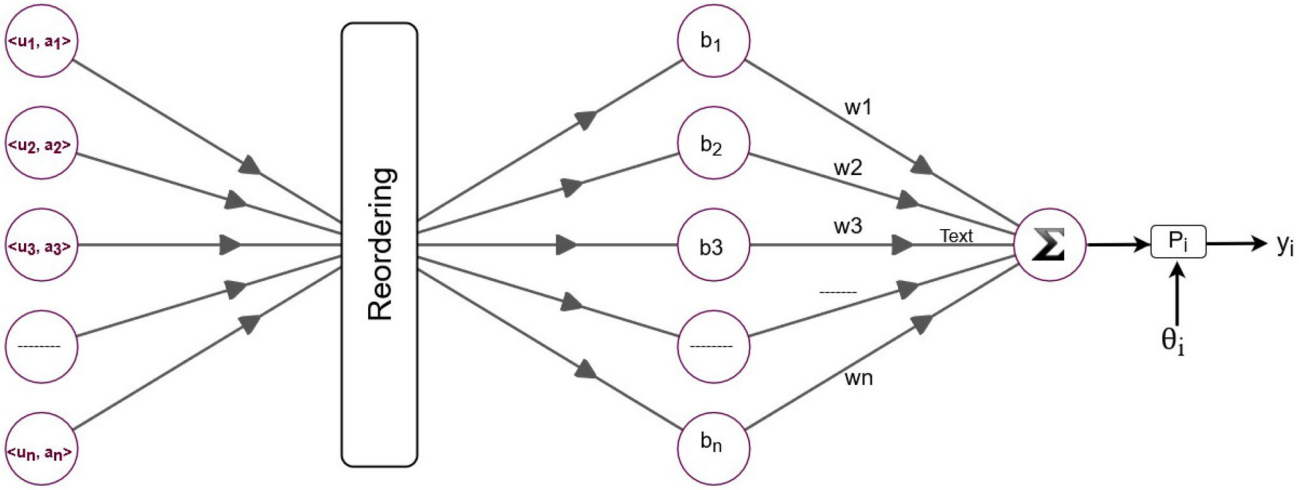

Fig. 1 IOWA in ANN

$$f(\langle u_1, a_1\rangle, \langle u_2, a_2\rangle, \ldots, \langle u_n, a_n\rangle) = \sum_{i=1}^{n} w_i b_i \tag{14}$$

and

$$f(\langle u_1, x_1\rangle, \langle u_2, x_2\rangle, \ldots, \langle u_n, x_n\rangle) = \sum_{i=1}^{n} w_i d_i \tag{15}$$

since $a_i \geq x_i$ for all a_i it follows that $a_i \geq x_i$, so

$$f(\langle u_1, a_1\rangle, \langle u_2, a_2\rangle, \ldots, \langle u_n, a_n\rangle) \geq f(\langle u_1, x_1\rangle, \langle u_2, x_2\rangle, \ldots, \langle u_n, x_n\rangle)$$

Theorem 2 (*Idempotency*): *The idempotency of the IOWA-ANN is presented in Eqs.* 16–18:

Let f be the IOWA operator in ANN that calculates the value of pi. Then

$$\begin{aligned} &f(\langle u_1, a_1\rangle, \langle u_2, a_2\rangle, \ldots, \langle u_n, a_n\rangle) \\ &\quad = f(\langle u_1, x_1, u_2\rangle, x_2, \ldots, \langle u_n, x_n\rangle) \end{aligned} \tag{16}$$

where $(\langle u_1, a_1\rangle, \langle u_2, a_2\rangle, \ldots, \langle u_n, a_n\rangle)$ is any variation of the arguments $\langle u_1, x_1\rangle, \langle u_2, x_2\rangle, \ldots, \langle u_n, x_n\rangle$.

Proof. Let

$$f(\langle u_1, a_1\rangle, \langle u_2, a_2\rangle, \ldots, \langle u_m, a_m\rangle) = \sum_{k=1}^{m} w_k b_k \tag{17}$$

and

$$f(\langle u_1, x_1\rangle, \langle u_2, x_2\rangle, \ldots, \langle u_m, x_m\rangle) = \sum_{k=1}^{m} w_k d_k \tag{18}$$

since $(\langle u_1, a_1\rangle, \langle u_2, a_2\rangle, \ldots, \langle u_m, a_m\rangle)$ is a variation of the arguments $\langle u_1, x_1\rangle, \langle u_2, x_2\rangle, \ldots, \langle u_m, x_m\rangle$, we obtain $a_k = x_k$ for all k, so

$$\begin{aligned} &f(\langle u_1, a_1\rangle, \langle u_2, a_2\rangle, \ldots, \langle u_m, a_m\rangle) \\ &\quad = f(\langle u_1, x_1\rangle, \langle u_2, x_2\rangle, \ldots, \langle u_m, x_m\rangle) \end{aligned}$$

Theorem 3 (*Commutativity*): *The commutativity of the IOWA-ANN is presented in Eqs.* 19–20:

Let f be the IOWA operator in ANN that calculates the value of pi. If $a_i = a$, then

$$f(\langle u_1, a_1\rangle, \langle u_2, a_2\rangle, \ldots, \langle u_m, a_m\rangle) = a \tag{19}$$

Proof If $a_i = a$, for all a_i, we obtain

$$\begin{aligned} f(\langle u_1, a_1\rangle, \langle u_2, a_2\rangle, \ldots, \langle u_n, a_m\rangle) &= \sum_{k=1}^{m} w_k b_k = \sum_{k=1}^{m} w_k a \\ &= a \sum_{k=1}^{m} w_k \end{aligned} \tag{20}$$

Since $\sum_{k=1}^{n} w_k = 1$, we get

$$f(\langle u_1, a_1\rangle, \langle u_2, a_2\rangle, \ldots, \langle u_m, a_m\rangle) = a$$

Theorem 4 (*Bounded*): *The bounded property of the IOWA-ANN is presented in Eqs.* 21–25:

Let f be the IOWA operator in ANN that calculates the value of pi.

Then

$$\min[a_i] \leq f(\langle u_1, a_1\rangle, \langle u_2, a_2\rangle, \ldots, \langle u_m, a_m\rangle) \leq \max[a_i] \tag{21}$$

Proof Let $min[a_i] = y$ and $max[a_i] = z$. Then

$$f(\langle u_1, a_1\rangle, \langle u_2, a_2\rangle, \ldots, \langle u_n, a_m\rangle) = \sum_{k=1}^{m} w_k b_k \leq \sum_{k=1}^{m} w_k z = z \sum_{k=1}^{m} w_k \tag{22}$$

and

$$f(\langle u_1, a_1\rangle, \langle u_2, a_2\rangle, \ldots, \langle u_n, a_m\rangle) = \sum_{k=1}^{m} w_k b_k \geq \sum_{k=1}^{m} w_k y = y \sum_{k=1}^{m} w_k \tag{23}$$

Since $\sum_{k=1}^{n} w_k = 1$, we get

$$f(\langle u_1, a_1\rangle, \langle u_2, a_2\rangle, \ldots, \langle u_n, a_m\rangle) \leq z \tag{24}$$

and

$$f(\langle u_1, a_1\rangle, \langle u_2, a_2\rangle, \ldots, \langle u_n, a_m\rangle) \geq y \tag{25}$$

Therefore,

$$\min[a_i] \leq f(\langle u_1, a_1\rangle, \langle u_2, a_2\rangle, \ldots, \langle u_m, a_m\rangle) \leq \max[a_i]$$

4 The generalisation of the application in IGOWA and Quasi-IOWA

In this section, we generalise the application to IGOWA and Quasi-IOWA operators. As discussed in the above sections, the IGOWA is the OWA operator's extension that has features of the IOWA and GOWA operator. Quasi-IOWA operator is a generalised OWA operator that uses a quasi-arithmetic mean instead of a normal mean that provides a more comprehensive generalisation. Below sections discuss the conception of IOWA-ANN in IGOWA and Quasi-IOWA.

4.1 IGOWA in ANN

The IGOWA operator in ANN is defined as follows:

Definition 7 The IGOWA operator in ANN of n dimension input is a mapping IGOWA: $R^n \rightarrow R$ defined by the associated weights w of dimension n such that $w_i \in [0,1]$ and $\sum_{i=1}^{n} w_i = 1$, the set of inducing variables of order u_i and the parameter $\lambda \in (-\infty, \infty)$ as presented in Eqs. 26–28 and ?tic=?>Fig. 2.

$$IGOWA - ANN(\langle u_1, a_1\rangle, \langle u_2, a_2\rangle, \ldots, \langle u_n, a_n\rangle) = pi \tag{26}$$

p_i is the activation function which is described as follows:

$$pi = \left(\sum_{k=1}^{n} w_k b_k^{\lambda}\right)^{1/\lambda} \tag{27}$$

where $\langle u_i, a_i\rangle$ is a set of two tuple input, where u_i is inducing variable associated with the input a_i,

b_j is the reordered input a_i in descending order of the u_i, w_j is the associated a_i weight, y_i is the actual output of the output neuron, λ is the parameter that adds a variation of the complexity.

In the IGOWA operator, if the weights are not normalised, that is $\sum_{i=1}^{n} w_i \neq 1$, then the activation function p_i is presented as in Eq. 28:

$$pi = \frac{1}{W}\left(\sum_{k=1}^{n} w_k b_k^{\lambda}\right)^{1/\lambda} \tag{28}$$

Like the IOWA operator, the IGOWA has commutative, monotonic, idempotent, and bounded properties.

4.2 Quasi-IOWA in ANN

The Quasi-IOWA operator in ANN is defined as follows:

Definition 8 The Quasi-IOWA operator in ANN of n dimension input is a mapping QIOWA: $R^n \rightarrow R$ defined by the associated weights w of dimension n such that $w_i \in [0,1]$ and $\sum_{i=1}^{n} w_i = 1$, the set of inducing variables of order u_i and a strictly monotonic continuous function $g(b)$. The Quasi-IOWA-ANN is presented in Eqs. 29–30 and ?tic=?>Fig. 3.

$$QIOWA - ANN(\langle u_1, a_1\rangle, \langle u_2, a_2\rangle, \ldots, \langle u_n, a_n\rangle) = pi = g^{-1}\left(\sum_{j=1}^{n} w_j g(b_j)\right) \tag{29}$$

where $\langle u_i, a_i\rangle$ is a set of two tuple input, where u_i is inducing variable associated with the input a_i, b_j is the reordered input a_i in descending order of the u_i, w_j is the associated a_i weight, y_j is the actual output of the output neuron, λ is the parameter that adds a variation of the complexity, $g(b_i)$ strictly monotonic continuous function, p_i is the activation function.

In the Quasi-IOWA operator, if the weights are not normalised, that is $\sum_{i=1}^{n} w_i \neq 1$, then the activation function p_i is presented as in Eq. 30:

$$pi = \frac{1}{W}\left\{g^{-1}\left(\sum_{i=1}^{n} w_i g(b_i)\right)\right\} \tag{30}$$

The Quasi-IOWA has the same properties of—commutative, monotonic, idempotent and bounded.

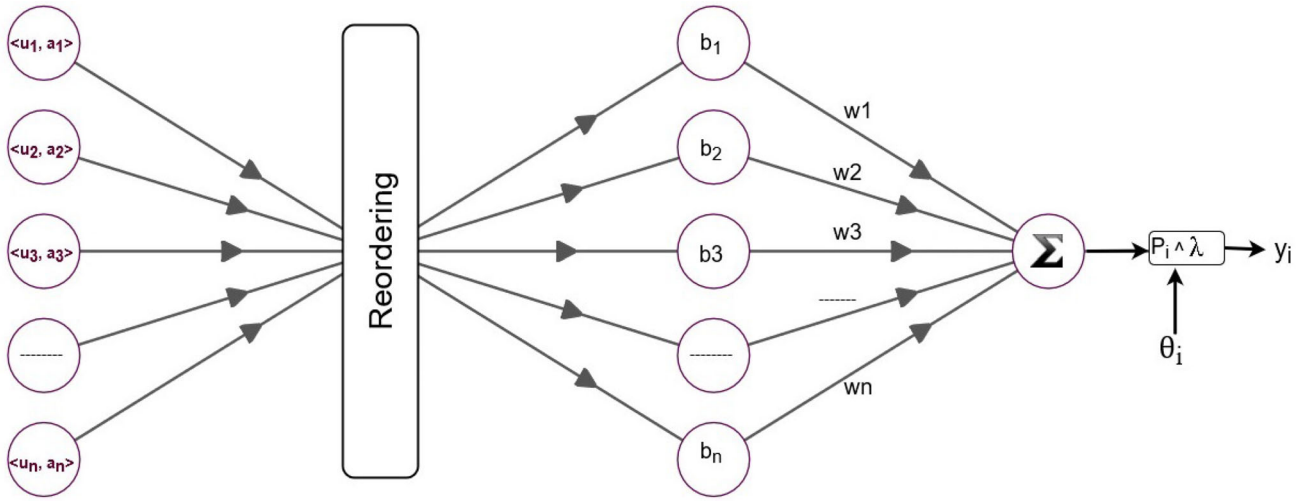

Fig. 2 IGOWA in ANN

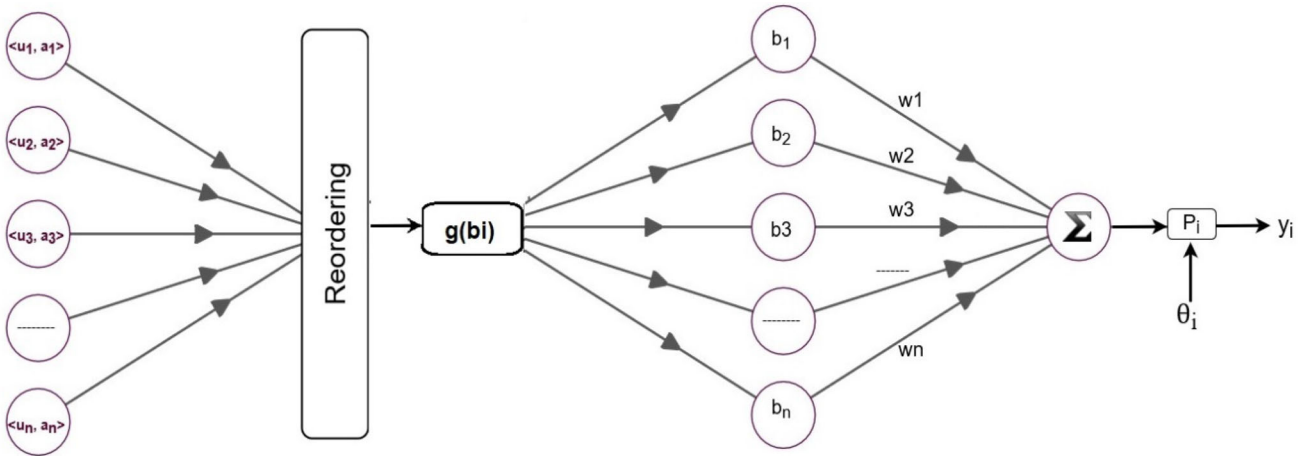

Fig. 3 Quasi-IOWA in ANN

5 Evaluation

This section evaluates the effectiveness of the proposed approach by considering a financial case study and demonstrating how the approach takes an optimum decision in a complex situation. For this case study, we considered a business that seeks total funding ($10,000) from 5 financial sources, let say—A, B, C, D and E, for a certain time interval n (4 intervals). A business can use a combination of financial sources. We assume a business request an equal amount ($2,500) for each interval to simplify the calculation. The interest rate x_i for each financial sources is as follows:

An interest rate of financial source A, $x_a = 0.058$.
An interest rate of financial source B, $x_b = 0.062$.
An interest rate of financial source C, $x_c = 0.070$.
An interest rate of financial source D, $x_d = 0.043$.
An interest rate of financial source E, $x_e = 0.049$.

The reliability value (10 most reliable, 01 least reliable) or inducing variable u_i of each financial sources is presented as follows:

Reliability value of financial source A, $u_a = 4$.
Reliability value of financial source B, $u_b = 6$.
Reliability value of financial source C, $u_c = 5$.

Table 1 Weights for each financial sources for different intervals

Weights	Time intervals			
	1st interval	2nd interval	3rd interval	4th interval
w_1	0.63			
w_2	0.18	0.42		
w_3	0.11	0.19	0.52	
w_4	0.02	0.20	0.12	0.38
w_5	0.06	0.19	0.36	0.62

Table 2 Customised threshold value for each time interval

Amount	θ Limit
$ 2,500	0.0030
$ 5,000	0.0160
$ 7,500	0.0331
$ 10,000	0.0532

Reliability value of financial source D, $u_d = 10$.

Reliability value of financial source E, $u_e = 9$.

Weights w for each financial sources and for each interval are given in such a way that $w_i \in [0,1]$ and $\sum_{i=1}^{n} w_i = 1$, as presented in Table 1.

The maximum acceptable threshold θ limit for a given amount in each time interval is presented in Table 2.

We use synthetic data for the approach, with 80% data used for training and 20% data for testing. The approach is implemented in MATLAB R2020a using a nonlinear autoregressive with exogenous (NARX) neural network. The process uses two layers of ten neurons with the Levenberg–Marquardt backpropagation function to train input data. We use the tangent sigmoid (TANSIG) transfer function to calculate the layer output from its net input. All input arguments are reordered in descending order based on the inducing variable, as presented in Table 3.

Based on updated ordering using inducing variables, neurons are rearranged as follows (Fig. 4).

Now we calculate the costing for different financial sources based on their respective weights to find the best possible source or a combination of sources. The choice of sources depends on the value, which is less than the customised threshold defined by a business for different intervals. Therefore, we select the minimum value of sources generated for each interval.

Using the proposed approach, we calculate the most suitable source for the first interval, as presented below.

$$p_{1n} = Min\{(w_{11} * x_d), (w_{21} * x_e), (w_{31} * x_b), (w_{41} * x_c), (w_{51} * x_a)\} \leq \theta_1$$

$$p_{1n} = Min\{(0.63 * 0.043), (0.18 * 0.049), (0.11 * 0.062), (0.02 * 0.070), (0.06 * 0.058)\} \leq 0.0030$$

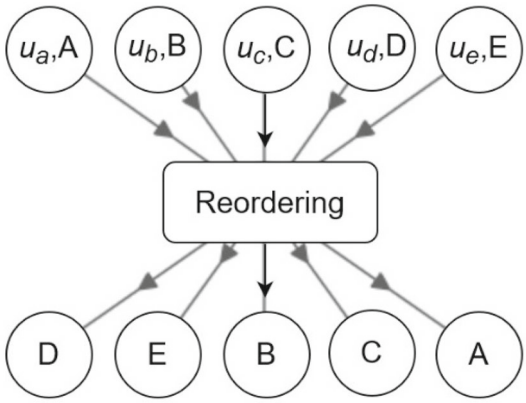

Fig. 4 Rearrange of input neurons based on inducing variable

$$p_{1n} = Min\{(0.0271), (0.0089), (0.0068), (0.0014), (0.0035)\} \leq 0.0030$$

$$p_{1n} = \{(0.0014)\} \leq 0.0030$$

From the calculation, we see that financial source C has the minimum value, which is 0.0014. Moreover, the only financial source C satisfies the customised threshold below the threshold value of 0.0030. Therefore, for the first interval for \$2,500/ = , the optimal financial source is provider C.

$$p_{1c} = 0.0014$$

For the second time interval, the business request another \$2,500. Now, the business will need a total of \$5,000. We use the proposed approach to determine the most suitable financer for the second term.

$$p_{2n} = Min\{(w_{22} * x_e + p_{1c}), (w_{32} * x_b + p_{1c}), (w_{42} * x_c + p_{1c}), (w_{52} * x_a + p_{1c})\} \leq \theta_2$$

$$p_{2n} = Min\{(0.38 * 0.049 + 0.0014), (0.19 * 0.062 + 0.0014), (0.20 * 0.070 + 0.0014), (0.19 * 0.058 + 0.0014)\} \leq 0.0160$$

$$p_{2n} = Min\{(0.0220), (0.0132), (0.0154), (0.0124)\} \leq 0.0160$$

$$p_{2n} = \{(0.0124)\} \leq 0.0160$$

We see that financer B, C and A satisfy a requestor criterion for a second interval because all of them have the value below than the threshold value 0.0160. However, option A is the most optimal financer because it has a value

Table 3 Reordering inputs based on ordered inducing variables

Financial sources	Interest rate	Inducing variable (reliability)	Weights
D	0.043	10	w_1
E	0.049	9	w_2
B	0.062	6	w_3
C	0.070	5	w_4
A	0.058	4	w_5

Table 4 Summary of obtained value using IOWA-ANN method

Financer	Obtained value for different time Intervals for each financer			
	1st interval θ = 0.0055	2nd interval θ = 0.0160	3rd interval θ = 0.0331	4th interval θ = 0.0559
D	0.0271			
E	0.0089	0.0220		
B	0.0068	0.0132	0.0446	
C	0.0014	0.0154	0.0208	0.0474
A	0.0035	0.0124	0.0333	0.0568
Optimal financer	C	A	C	C

of 0.0124, which is minimum than all other financers. Therefore, for the second interval, financer A is the best option.

$$p_{2a} = 0.0124$$

For the third interval, we apply the formula to determine the most suitable financer.

$$p_{3n} = Min\{(w_{33} * x_b + p_{2a}), (w_{43} * x_c + p_{2a}), (w_{53} * x_a + p_{2a})\} \leq \theta_3$$

$$p_{3n} = Min\{(0.52 * 0.062 + 0.0124), (0.12 * 0.070 + 0.0124), (0.36 * 0.058 + 0.0124)\} \leq 0.0331$$

$$p_{3n} = Min\{(0.0446), (0.0208), (0.0333)\} \leq 0.0331$$

$$p_{3n} = \{(0.0208)\} \leq 0.0331$$

Based on the above calculation, we see that financial source C is the most suitable financer for a third interval because it has a value of 0.02430, which is a minimum than all other financers and satisfies requestor requirements below the threshold value 0.0331.

$$p_{3c} = 0.0208$$

To determine the optimal financer for the last interval, we use the formula as presented below.

$$p_{4n} = Min\{(w_{44} * x_c + p_{3c}), (w_{54} * x_a + p_{3c})\} \leq \theta_4$$

$$p_{4n} = Min\{(0.38 * 0.070 + 0.0208), (0.62 * 0.058 + 0.0208)\} \leq 0.0532$$

$$p_{4n} = Min\{(0.0474), (0.0568)\} \leq 0.0532$$

$$p_{4n} = \{(0.0474)\} \leq 0.0532$$

The financer C is the most suitable for the last interval with the value of 0.0474, which is less than financer A and below the threshold value of 0.0532.

From Table 4, we can see that only for the second interval financer A is the best, while for the rest of the three intervals, financer C is the optimal option. Hence, we can recommend that a combination of financers C and A is the best solution to finance.

6 Conclusion

From the last few decades, many industries, businesses and different financial firms adopt NN that offers a step-change in the power of AI. The use of ANN assists firms to improve their efficiency, accuracy, timeliness, customer satisfaction and optimal decision-making that enable the firm to gain and sustain their competitive advantage. The efficiency of a neural network depends on its learning process. The existing literature discussed reordering inputs based on the weight to improve efficiency and accuracy. However, we observed that the current literature could not address complex reordering of the decision-making processes. In this paper, we propose the IOWA layer in ANN, in which the ordering of arguments is based on an ordered inducing variable. The new reordering method assists in handling a complex decision-making process. The approach decreases the computational complexity by reducing the data size without losing any information. We further generalised the approach with the IGOWA and Quasi-IOWA operator. To demonstrate the effectiveness of our approach, we presented a financial case study where our approach performed well in a complex and sequential decision-making process. We will analyse applying the method for deep learning in our future work and examine the optimisation in the autonomous learning process.

Funding Open Access funding enabled and organized by CAUL and its Member Institutions. The authors declare that no funds, grants, or other support were received during the preparation of this manuscript.

Data Availability Enquiries about data availability should be directed to the authors.

Declarations

Ethical approval Not applicable. There is no human or animal involved in this research.

Conflict of interest The authors have no relevant financial or non-financial interests to disclose.

Informed consent Not applicable. In this research, no human or animal is involved.

References

Abdel-Basset M, Abdel-Fatah L, Sangaiah AK (2018) Metaheuristic algorithms: a comprehensive review. In: Computational intelligence for multimedia big data on the cloud with engineering applications, pp 185–231

Blanco-Mesa F, León-Castro E, Merigó JM (2020) Covariances with OWA operators and Bonferroni means. Soft Comput pp 1–16

Bordogna G, Boschetti M, Brivio A, Carrara P, Pagani M, Stroppiana D (2011) Fusion strategies based on the OWA operator in environmental applications. In: recent developments in the ordered weighted averaging operators: theory and practice. Springer, pp 189–207

Cheng C-H, Wei L-Y, Liu J-W, Chen T-L (2013) OWA-based ANFIS model for TAIEX forecasting. Econ Model 30:442–448

Cho S-B (1995) Fuzzy aggregation of modular neural networks with ordered weighted averaging operators. Int J Approx Reason 13(4):359–375

Flores-Sosa M, Avilés-Ochoa E, Merigó JM (2020) Induced OWA operators in linear regression. J Intell Fuzzy Syst 38(5):5509–5520

Gao Z, Chen Y, Yi Z (2020) A novel method to compute the weights of neural networks. Neurocomputing 407:409–427

Hebb DO (1949) The organization of behavior: a neuropsychological theory. Wiley, Chapman and Hall

Hussain W, Hussain FK, Hussain OK, Damiani E, Chang E (2017) Formulating and managing viable SLAs in cloud computing from a small to medium service provider's viewpoint: a state-of-the-art review. Inf Syst 71:240–259

Hussain W, Hussain FK, Saberi M, Hussain OK, Chang E (2018) Comparing time series with machine learning-based prediction approaches for violation management in cloud SLAs. Futur Gener Comput Syst 89:464–477

Hussain W, Raza MR, Jan MA, Merigó JM, Gao H (2022) Cloud risk management with OWA-LSTM and fuzzy linguistic decision making. IEEE Trans Fuzzy Syst 30(11):4657–4666. https://doi.org/10.1109/TFUZZ.2022.3157951

Hussain W, Merigó JM, Raza MR (2022d) Predictive intelligence using ANFIS-induced OWAWA for complex stock market prediction. Int J Intell Syst 37(8):4586–4611

Hussain W, Merigo JM, Gao H, Alkalbani AM, Rabhi FA (2021) Integrated AHP-IOWA, POWA framework for ideal cloud provider selection and optimum resource management. IEEE Trans Serv Comput

Hussain W, Merigó JM, Gao H, Alkalbani AM, Rabhi FA (2023) Integrated AHP-IOWA, POWA framework for ideal cloud provider selection and optimum resource management. IEEE Trans Serv Comput 16(1):370–382. https://doi.org/10.1109/TSC.2021.3124885

Hussain W, Gao H, Raza MR et al (2022) Assessing cloud QoS predictions using OWA in neural network methods. Neural Comput Applic 34:14895–14912. https://doi.org/10.1007/s00521-022-07297-z

Jin L, Mesiar R, Yager RR (2021) On WA expressions of induced OWA operators and inducing function based orness with application in evaluation. IEEE Trans Fuzzy Syste 29(6):1695-1700. https://doi.org/10.1109/TFUZZ.2020.2979387

Kacprzyk J, Yager RR, Merigo JM (2019) Towards human-centric aggregation via ordered weighted aggregation operators and linguistic data summaries: A new perspective on zadeh's inspirations. IEEE Comput Intell Mag 14(1):16–30

Kaur G, Dhar J, Guha RK (2014) Stock market forecasting using ANFIS with OWA operator. Int J Artific Intell 12(2):102–114

Kaur G, Dhar J, Guha RK (2016) Minimal variability OWA operator combining ANFIS and fuzzy c-means for forecasting BSE index. Math Comput Simul 122:69–80

Li B, Ding J, Yin Z, Li K, Zhao X, Zhang L (2021) Optimized neural network combined model based on the induced ordered weighted averaging operator for vegetable price forecasting. Expert Syst Appl 168:114232

Maldonado S, Merigó J, Miranda J (2020) IOWA-SVM: a density-based weighting strategy for SVM classification via OWA operators. IEEE Trans Fuzzy Syst 28(9):2143–2150. https://doi.org/10.1109/TFUZZ.2019.2930942

Merigó JM, Gil-Lafuente AM (2009) The induced generalized OWA operator. Inf Sci 179(6):729–741

Merigó JM, Gil-Lafuente AM (2010) New decision-making techniques and their application in the selection of financial products. Inf Sci 180(11):2085–2094

Merigó JM, Gil-Lafuente AM (2011) Fuzzy induced generalized aggregation operators and its application in multi-person decision making. Expert Syst Appl 38(8):9761–9772

Merigó JM, Gil-Lafuente AM, Yu D, Llopis-Albert C (2018) Fuzzy decision making in complex frameworks with generalized aggregation operators. Appl Soft Comput 68:314–321

Panahi M, Gayen A, Pourghasemi HR, Rezaie F, Lee S (2020) Spatial prediction of landslide susceptibility using hybrid support vector regression (SVR) and the adaptive neuro-fuzzy inference system (ANFIS) with various metaheuristic algorithms. Sci Total Environ 741:139937

Stephen I (1990) Perceptron-based learning algorithms. IEEE Trans Neural Netw 50(2):179

Wang L, Zeng Y, Chen T (2015) Back propagation neural network with adaptive differential evolution algorithm for time series forecasting. Expert Syst Appl 42(2):855–863

Widrow B, Lehr MA (1993) Adaptive neural networks and their applications. Int J Intell Syst 8(4):453–507

Wu L, Huang G, Fan J, Ma X, Zhou H, Zeng W (2020) Hybrid extreme learning machine with meta-heuristic algorithms for monthly pan evaporation prediction. Comput Electron Agric 168:105115

Yager RR (1988) On ordered weighted averaging aggregation operators in multicriteria decisionmaking. IEEE Trans Syst Man Cybern 18(1):183–190

Yager RR (1993) Families of OWA operators. Fuzzy Sets Syst 59(2):125–148

Yager RR (1994) On the RAGE aggregation method with applications to neural networks and decision making. Int J Approx Reason 11(3):175–204

Yager RR, Filev DP (1999) Induced ordered weighted averaging operators. IEEE Trans Syst Man Cybern B (cybernetics) 29(2):141–150

Yi P, Li W (2019) Induced cluster-based OWA operators with reliability measures and the application in group decision-making. Int J Intell Syst 34(4):527–540

Zarghami M, Ardakanian R, Memariani A, Szidarovszky F (2008) Extended OWA operator for group decision making on water resources projects. J Water Resour Plan Manag 134(3):266–275

Zhang J, Huang Y, Wang Y, Ma G (2020) Multi-objective optimization of concrete mixture proportions using machine learning and metaheuristic algorithms. Constr Build Mater 253:119208

Zingg DW, Nemec M, Pulliam TH (2008) A comparative evaluation of genetic and gradient-based algorithms applied to aerodynamic optimization. Eur J Computl Mech 17(1–2):103–126