



An integrated eigenvalue based neural network approach for MAGDM with intuitionistic Fuzzy sets

Gopal Sumithra ^{a,*}, John Robinson P ^a, Wilson Arul Prakash S ^b

^aDepartment of Mathematics, Bishop Heber College, Affiliated to Bharathidasan University, Tiruchirappalli-620017.

^bDepartment of Mathematics, Jeppiaar College of Arts & Science, Affiliated to University of Madras, Chennai-603103.

Abstract

This paper addresses Multi-Criteria Group Decision Making (MCGDM), also known as Multiple Attribute Group Decision Making (MAGDM), under the framework of intuitionistic fuzzy sets. To solve fuzzy linear algebraic equations, linear space techniques involving real eigenvalues are employed. These solutions are then used to determine decision-maker weights in MAGDM problems. During the weight determination process, multiple criteria are explicitly incorporated, and several results obtained through the proposed methods are normalized. Additionally, decision-maker weights for attributes, along with corresponding decision-making approaches, are introduced. Furthermore, Artificial Neural Network (ANN) techniques are applied to enhance the determination of decision-maker weights. The feasibility and effectiveness of the proposed approach are demonstrated through numerical examples. The convergence curve shows stable error reduction without underfitting or overfitting, validating the robustness of the proposed ANN framework for reliable application in intuitionistic fuzzy setbased MAGDM.

Keywords: MAGDM, Eigen value, Intuitionistic fuzzy sets, Weight vector, ANN.

2020 MSC: 05C50

©2026 All rights reserved.

1. Introduction

Artificial Neural Networks (ANNs) have become one of the most widely used tools for modeling nonlinear systems because of their ability to learn directly from data. Early contributions such as the backpropagation method [15], later consolidated through systematic treatments [4, 7], provided the theoretical and algorithmic foundation for ANN training. Since then, these models have been applied across domains ranging from classification and forecasting to large-scale decision support [16, 19]. Alongside this progress, the study of fuzzy sets and their extensions has provided a complementary direction for representing uncertainty in decision environments. Atanassovs intuitionistic fuzzy sets (IFS) [2] extended classical fuzzy sets by explicitly accounting for membership, non-membership, and hesitation, thereby offering a richer structure for handling vagueness. Numerous works have further developed measures such as distance [1], similarity [8], and correlation [3, 18, 6], showing their usefulness in multi-criteria decision-making (MCDM).

*Corresponding author

Email addresses: sumithraphd14@gmail.com (Gopal Sumithra ) , johnrobinson.ma@bhc.edu.in (John Robinson P ) , wilsonapmathematics@gmail.com (Wilson Arul Prakash S )

doi: [10.30511/mcs.2025.2075323.1541](https://doi.org/10.30511/mcs.2025.2075323.1541)

Received: 20 October 2025 Accepted: 04 December 2025

More recently, researchers have investigated the integration of ANN and intuitionistic fuzzy approaches. Examples include intuitionistic fuzzy neural networks with Gaussian membership [10], deep neural designs [2], and applications to time series prediction [9]. In parallel, linguistic and orthogonalized fuzzy neural systems have been proposed for decision support [11, 12, 13, 14], reflecting the growing interest in hybrid intelligent systems. Decision-making problems involving multiple attributes and stakeholders (MAGDM) particularly benefit from such hybrid models. In these contexts, where uncertainty and conflicting evaluations are common, ANN-based approaches combined with IFS have been reported to improve both predictive reliability and aggregation accuracy [17, 20, 5]. These methods allow decision-makers to incorporate vague or incomplete information without losing mathematical rigor. In this study, we extend this line of research by presenting an ANN-based framework for MAGDM under intuitionistic fuzzy environments. The proposed approach combines the adaptive learning capability of neural networks with the descriptive richness of IFS, aiming to ensure convergence stability, reduce the risk of overfitting, and provide reliable rankings of alternatives in group decision-making contexts.

2. Decision Maker Weight Determining Methods with real eigen values

The MAGDM problem solving involves effective involvement of the decision maker who is responsible in providing a consensus for the process. At one end, the weight vector provided by the decision maker plays a crucial role and, in such situations, where the problem-solving techniques solely depends on the decision maker weighting vector, a proper method to elicit the same from the decision maker becomes crucial at the other end. In this chapter and in particular when dealing with a decision maker profound in providing weights in the form of some Linear space problems, we need means and methods to solve those information and retrieve the weight information in a proper sense which will appropriately suit the decision problem involved. Here is a model of eliciting weight vector from the decision maker.

Step 1: Let T_i be linear operator on H which is a Hilbert space and $e_{i1}, e_{i2}, \dots, e_{in}$ is basis for H . Let $T_i(e_{i1}), T_i(e_{i2}), \dots, T_i(e_{in})$ be values in H .

Let $e_{11}, e_{12}, \dots, e_{1n}$ basis for H , where $A_1 = \begin{pmatrix} T_1(e_{11}) \\ T_1(e_{12}) \\ \vdots \\ T_1(e_{1n}) \end{pmatrix}$. Let $e_{21}, e_{22}, \dots, e_{2n}$ be another basis for H , where $A_2 = \begin{pmatrix} T_1(e_{21}) \\ T_1(e_{22}) \\ \vdots \\ T_1(e_{2n}) \end{pmatrix}$. Similarly, we can find A_n , where $A_n = \begin{pmatrix} T_1(e_{n1}) \\ T_1(e_{n2}) \\ \vdots \\ T_1(e_{nn}) \end{pmatrix}$.

Step 2: Let $\lambda_i = (\lambda_{i1}, \lambda_{i2}, \dots, \lambda_{in})$ be the real eigen values for a matrix A_i , where the eigenvalues are in spectrum of the space, and form the matrix $B_j = (\lambda_1 \ \lambda_2 \ \dots \ \lambda_n)$.

Step 3: Find $B_j * B_j^T$ and then normalize the matrix to elicit the weights w_1, w_2, \dots, w_n such that $\sum_{i=1}^n w_i = 1$.

Problem proposed by Decision maker-1: Assume that the stakeholder must unlock the following linear space issue, which is the weight information provided by decision maker-1.

Step 1: Let the Hilbert space \mathfrak{R}^3 have the linear operative function T_1 and let $\beta_1 = \{e_1 = (1, 0, 0), e_2 = (0, 1, 0), e_3 = (0, 0, 1)\}$ be a basis set. Let T_1 be proposed as:

$$\begin{aligned} T_1(e_1) &= (x_1, x_1 - y_1, z_1) = (1, 1, 0); T_1(e_2) = (x_2, 2y_2, z_2) = (0, 2, 0); \\ T_1(e_3) &= (x_3, -z_3, 4z_3) = (0, -1, 4). \end{aligned}$$

The corresponding matrix A_1 is $A_1 = \begin{pmatrix} 1 & 1 & 0 \\ 0 & 2 & 0 \\ 0 & -1 & 4 \end{pmatrix}$. The collection of eigenvalues are $\lambda_1 = \{1, 2, 4\}$. Let T_2 be defined by,

$$T_2(e_1) = (x_1, x_1 + y_1, z_1) = (1, 1, 0); T_2(e_2) = (x_2, y_2, z_2) = (0, 2, 0);$$

$$T_2(e_3) = (x_3, -z_3, 2z_3) = (0, -1, 2).$$

The corresponding matrix A_2 is $A_2 = \begin{pmatrix} 1 & 1 & 0 \\ 0 & 2 & 0 \\ 0 & -1 & 2 \end{pmatrix}$. The collection of eigenvalues are $\lambda_2 = \{1, 2, 2\}$. Let T_3 be defined by,

$$T_3(e_1) = (z_1, y_1, 2x_1) = (0, 0, 2); T_3(e_2) = (-3y_2 + x_2, y_2, 6y_2 + z_2) = (-3, 1, 6);$$

$$T_3(e_3) = (x_3, y_3, z_3) = (0, 0, 1).$$

The corresponding matrix A_3 is $A_3 = \begin{pmatrix} 0 & 0 & 2 \\ -3 & 1 & 6 \\ 0 & 0 & 1 \end{pmatrix}$. The collection of eigenvalues are $\lambda_3 = \{0, 1, 1\}$.

Step 2: The matrix B_1 is, $B_1 = (\lambda_1 \ \lambda_2 \ \lambda_3)$, where $B_1 = \begin{pmatrix} 1 & 1 & 0 \\ 2 & 2 & 1 \\ 4 & 2 & 1 \end{pmatrix}$.

Step 3: Finding $B_1 * B_1^T$ we have the following computation:

$$B_1 * B_1^T = \begin{pmatrix} 1 & 1 & 0 \\ 2 & 2 & 1 \\ 4 & 2 & 1 \end{pmatrix} \begin{pmatrix} 1 & 2 & 4 \\ 1 & 2 & 2 \\ 0 & 1 & 1 \end{pmatrix} = \begin{pmatrix} 2 & 4 & 6 \\ 4 & 9 & 13 \\ 6 & 13 & 21 \end{pmatrix}.$$

Normalizing $B_1 * B_1^T$, we get the weight vector of Decision Maker-1 as: $\lambda = (0.156837, 0.334829, 0.508333)$.

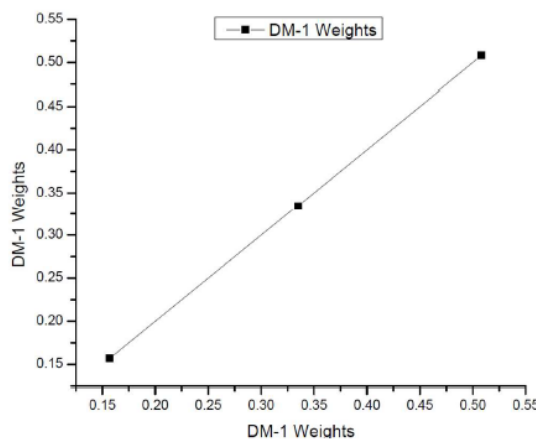


Figure 1: Normalized Weights of Decision Maker-1 (Real Eigen Values)

Problem proposed by Decision maker-2: Assume that the stakeholder must unlock the following linear space issue, which is the weight information provided by decision maker-2.

Step 1: Let the Hilbert space \mathfrak{R}^3 have the linear operative function T_4 and let

$$\beta_2 = \{b_1 = (4, 2, 0), b_2 = (2, 1, 5), b_3 = (1, 0, 0)\}$$

be a basis set. The operative function T_4 is defined by,

$$T_4(b_1) = \left(z_1, \frac{y_1}{2}, \frac{x_1}{2} \right) = (0, 1, 2); T_4(b_2) = \left(-2x_2, y_2, \frac{x_2 + y_2 + z_2}{2} \right) = (-4, 1, 4);$$

$$T_4(b_3) = (-5x_3, x_3, 7x_3 + z_3) = (-5, 1, 7).$$

The corresponding matrix A_4 is, $A_4 = \begin{pmatrix} 0 & 1 & 2 \\ -4 & 1 & 4 \\ -5 & 1 & 7 \end{pmatrix}$. The collection of eigenvalues is $\lambda_4 = \{1, 2, 5\}$. Let the operator T_5 is defined by,

$$T_5(b_1) = \left(x_1, \frac{y_1}{2}, x_1 + y_1 \right) = (4, 1, 6); T_5(b_2) = (z_2 - 2x_2 - y_2, x_2, z_2 - x_2) = (0, 2, 3);$$

$$T_5(b_3) = (z_3, y_3, 9x_3) = (0, 0, 9).$$

The corresponding matrix A_5 is, $A_5 = \begin{pmatrix} 4 & 1 & 6 \\ 0 & 2 & 3 \\ 0 & 0 & 9 \end{pmatrix}$. The collection of eigenvalues is $\lambda_5 = \{2, 4, 9\}$. Let the operator T_6 be defined by,

$$T_6(b_1) = \left(z_1, \frac{y_1}{2}, \frac{x_1}{4} \right) = (0, 1, 1); T_6(b_2) = (-y_2, x_2, z_2 - 2x_2) = (-1, 2, 1);$$

$$T_6(b_3) = (-x_3, -x_3, 4x_3) = (-1, -1, 4).$$

The corresponding matrix A_6 is, $A_6 = \begin{pmatrix} 0 & 1 & 1 \\ -1 & 2 & 1 \\ -1 & -1 & 4 \end{pmatrix}$. The collection of eigenvalues is $\lambda_6 = \{1, 2, 3\}$.

Step 2: The matrix B_2 is formed as: $B_2 = (\lambda_4 \ \lambda_5 \ \lambda_6)$, where $B_2 = \begin{pmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 5 & 9 & 3 \end{pmatrix}$.

Step 3: Finding $B_2 * B_2^T$ we have the following computation:

$$B_2 * B_2^T = \begin{pmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 5 & 9 & 3 \end{pmatrix} \begin{pmatrix} 1 & 2 & 5 \\ 2 & 4 & 9 \\ 1 & 2 & 3 \end{pmatrix} = \begin{pmatrix} 6 & 12 & 26 \\ 12 & 24 & 152 \\ 26 & 52 & 115 \end{pmatrix}.$$

Normalizing $B_j * B_j^T$, we get the weight vector of Decision Maker-2 as: $w = (0.135814, 0.271628, 0.592557)$.

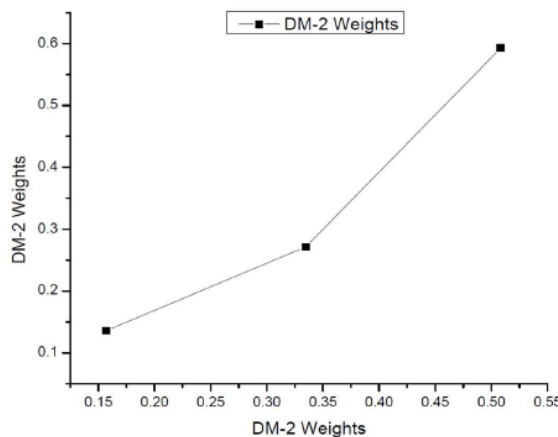


Figure 2: Normalized Weights of Decision Maker-2 (Real Eigen Values)

Problem proposed by Decision Maker-3: Assume that the stakeholder must unlock the following linear space issue, which is the weight information provided by decision maker-3.

Step 1: Let the Hilbert space \mathfrak{R}^4 have the linear operative function T_7 and

$$\beta_3 = \{c_1 = (1, 0, 0, 0), c_2 = (0, 1, 0, 0), c_3 = (0, 0, 1, 0), c_4 = (0, 0, 0, 1)\}$$

be the basis. The operative function T_7 is proposed as:

$$\begin{aligned} T_7(c_1) &= (2x_1, x_1, x_1, r_1) = (2, 1, 1, 0); T_7(c_2) = (y_2, y_2, z_2, y_2 + x_2) = (1, 2, 0, 1); \\ T_7(c_3) &= (z_3, x_3, 2z_3, z_3) = (1, 0, 2, 1); T_7(c_4) = (z_3, r_4, r_4, 2r_4) = (0, 1, 1, 2). \end{aligned}$$

The corresponding matrix A_7 is, $A_7 = \begin{pmatrix} 2 & 1 & 1 & 0 \\ 1 & 2 & 0 & 1 \\ 1 & 0 & 2 & 1 \\ 0 & 1 & 1 & 2 \end{pmatrix}$. The collection of eigenvalues is $\lambda_7 = \{4, 2, 0, 2\}$. Let

the operator T_8 is defined by,

$$\begin{aligned} T_8(c_1) &= (x_1, 4x_1, y_1, r_1) = (1, 4, 0, 0); T_8(c_2) = (x_2, y_2, z_2, r_2) = (0, 2, 0, 0); \\ T_8(c_3) &= (5z_3, 3z_3, z_3 + x_3, z_3) = (5, 3, 1, 1); T_8(c_4) = (4r_4, 7r_4, 2r_4, 2r_4) = (4, 7, 2, 2). \end{aligned}$$

The corresponding matrix A_8 is, $A_8 = \begin{pmatrix} 1 & 4 & 0 & 0 \\ 0 & 2 & 0 & 0 \\ 5 & 3 & 1 & 1 \\ 4 & 7 & 2 & 2 \end{pmatrix}$. The collection of eigenvalues is $\lambda_8 = \{0, 1, 2, 3\}$. Let

the operator T_9 be defined by,

$$\begin{aligned} T_9(c_1) &= (2x_1, x_1, 3x_1, 4x_1 + r_1) = (2, 1, 3, 4); T_9(c_2) = (x_2, 2y_2, y_2, 3y_2) = (0, 2, 1, 3); \\ T_9(c_3) &= (2z_3, z_3, 6z_3 + x_3, 5z_3) = (2, 1, 6, 5); T_9(c_4) = (r_4, 2r_4, 4r_4, 8r_4) = (1, 2, 4, 8). \end{aligned}$$

The corresponding matrix A_9 is, $A_9 = \begin{pmatrix} 2 & 1 & 3 & 4 \\ 0 & 2 & 1 & 3 \\ 2 & 1 & 6 & 5 \\ 1 & 2 & 4 & 8 \end{pmatrix}$. The collection of eigenvalues is

$$\lambda_9 = \{13.0990, 1, 2.9010, 1\}.$$

Let the operator T_{10} be defined by,

$$\begin{aligned} T_{10}(c_1) &= (4x_1, -x_1, -x_1, -x_1 + r_1) = (4, -1, -1, -1); \\ T_{10}(c_2) &= (-y_2, 4y_2, -y_2, -y_2) = (-1, 4, -1, -1); \\ T_{10}(c_3) &= (-z_3, -z_3, 4z_3, -z_3) = (-1, -1, 4, -1); \\ T_{10}(c_4) &= (-r_4, -r_4, -r_4, 4r_4) = (-1, -1, -1, 4). \end{aligned}$$

The corresponding matrix A_{10} is, $A_{10} = \begin{pmatrix} 4 & -1 & -1 & -1 \\ -1 & 4 & -1 & -1 \\ -1 & -1 & 4 & -1 \\ -1 & -1 & -1 & 4 \end{pmatrix}$. The collection of eigenvalues is $\lambda_{10} = \{1, 5, 5, 5\}$.

Step 2: The matrix B_3 is formed as: $B_3 = (\lambda_7 \ \lambda_8 \ \lambda_9 \ \lambda_{10})$, and hence $B_3 = \begin{pmatrix} 4 & 0 & 13.0990 & 1 \\ 2 & 1 & 1 & 5 \\ 0 & 2 & 2.9010 & 5 \\ 2 & 3 & 1 & 5 \end{pmatrix}$.

Step 3: Finding $B_3 * B_3^T$ we have the following computation:

$$B_3 * B_3^T = \begin{pmatrix} 4 & 0 & 13.0990 & 1 \\ 2 & 1 & 1 & 5 \\ 0 & 2 & 2.9010 & 5 \\ 2 & 3 & 1 & 5 \end{pmatrix} \begin{pmatrix} 4 & 2 & 0 & 2 \\ 0 & 1 & 2 & 3 \\ 13.0990 & 1 & 2.9010 & 1 \\ 1 & 5 & 5 & 5 \end{pmatrix}.$$

Normalizing $B_j * B_j^T$, we get the weight vector of Decision Maker-3 as:

$$w = (0.343122, 0.191853, 0.226570, 0.238453)$$

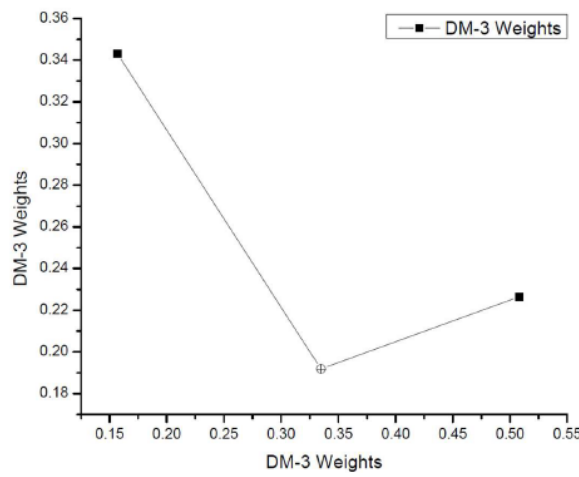


Figure 3: Normalized Weights of Decision Maker-3 (Real Eigen Values)

Now, as we have elicited the weighting vectors from the 1st, 2nd and 3rd decision makers, $\gamma = (0.156837, 0.334829, 0.508333)$, $w = (0.135814, 0.271658, 0.592557)$ and $\omega = (0.343122, 0.191853, 0.226570, 0.238453)$ respectively, we can make use of these vectors in the aggregation computation of the decision matrices of the given MAGDM problem to select the best alternatives.

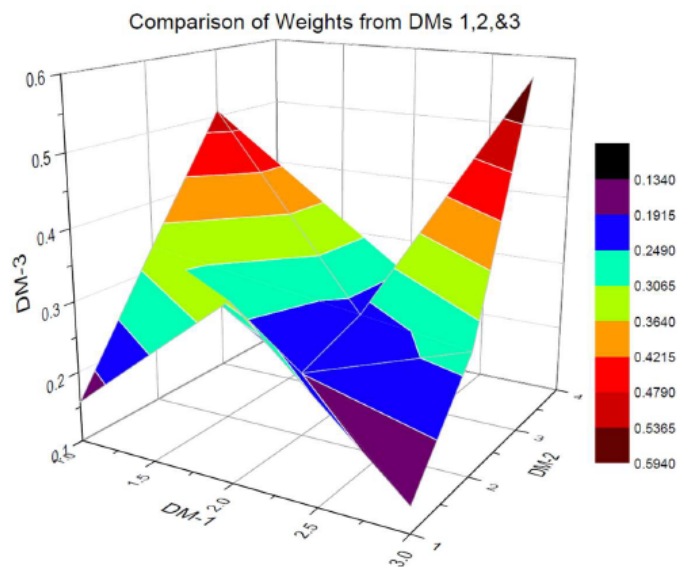


Figure 4: Comparison of Weights of Decision Makers 1,2 & 3 (Real Eigen Values)

The 3D surface plot presented in figure 4 illustrates the comparative analysis of weights assigned by three distinct Decision Makers (DM-1, DM-2, and DM-3) within a multi-criteria decision-making (MCDM) context. The surface gradient, mapped through a color-coded scale, reveals significant insights into the level of agreement and disparity among the decision makers. Higher weight consensus is observed in the regions colored in purple and blue, indicating that these alternatives or criteria are consistently rated as highly significant by the DMs. In contrast, the presence of valleys shaded in darker tones such as black and deep blue signifies areas of disagreement or lower importance attributed by at least one of the DMs. The sharp transitions in surface elevation further highlight the heterogeneity in expert judgments. This visualization proves to be a powerful diagnostic tool for identifying conflicting preferences in group decision-making scenarios and facilitates a better understanding of the consensus structure among evaluators.

3. Algorithm for MAGDM with intuitionistic fuzzy information

Now the MAGDM problem will be resolved with the help of the weights determined from the decision makers as in the above sections, and the decision matrices will be combined using these weight vectors. The algorithm for the same is presented below:

Step: 1 With the help of IFOWA/IFWAA operator, one can ascertain the alternative's intuitionistic fuzzy values to determine one's ultimate preferences. Assume that a collection of intuitionistic fuzzy numbers follows the operation of the IFOWA/ IFWAA operation.

Step: 2 To figure out the alternative's cumulative consensus of intuitionistic fuzzy values, apply the IFHA operator. Assume that a collection of intuitionistic fuzzy numbers follows the operation of the IFHA along with the balancing vector.

Step: 3 Rank the alternatives based on the methods proposed as in [13, 14].

Step: 4 Arrange all the choices A_i , i being a natural number, and select the top ranked option.

4. Numerical Illustration: MAGDM with weights derived from linear space methods with real eigen values

Evaluating Investment Alternatives Using Intuitionistic Fuzzy Decision-Making: In this scenario, a risk investment company is tasked with allocating funds to the most promising investment option among five potential alternatives. The decision-making process is structured around a comprehensive evaluation of four key attributes:

1. **G1: Assessing Potential Financial Risks** – This criterion assesses the potential financial risks associated with each investment option. It focuses on identifying and quantifying risks to minimize potential losses.
2. **G2: Evaluating Potential for Future Growth** – This involves evaluating the expected growth trajectory of each alternative, considering factors such as market trends, financial performance, and future potential.
3. **G3: Considering the Socio-Political Ramifications** – This criterion examines the socio-political implications of the investment, including regulatory considerations, political stability, and societal benefits or concerns.
4. **G4: Analyzing the Environmental Consequences** – This evaluates the environmental consequences of each investment option, considering sustainability, ecological footprint, and compliance with environmental regulations.

Decision-Making Process: To ensure a comprehensive evaluation, the company engages three decision-makers, each contributing their expertise and judgment. Their influence on the final decision is quantified through specific weighting vectors:

- **Decision Maker 1 (Weighting Vector w):** $w = (0.156837, 0.334829, 0.508333)$
- **Decision Maker 2 (Weighting Vector γ):** $\gamma = (0.135814, 0.271628, 0.592557)$

The attributes themselves are also weighted to reflect their relative importance in the decision-making process. The balancing vector for these criteria is:

- **Balancing Vector ω :** $\omega = (0.343122, 0.191853, 0.226570, 0.238453)$.

Evaluation using Intuitionistic Fuzzy Numbers:

The alternatives A_i (where $i=1,2,3,4,5$) are evaluated using intuitionistic fuzzy numbers. This approach captures the inherent uncertainty and subjectivity in decision-making by accounting for degrees of membership, non-membership, and hesitation. Each decision-maker provides an evaluation of the alternatives based on the four criteria.

Constructing the Decision Matrices: The decision matrices are constructed by aggregating the evaluations from all three decision-makers, weighted by their respective vectors. This process ensures that the final decision reflects a balanced consideration of all criteria and the input of each decision-maker. The intuitionistic fuzzy approach facilitates a robust and nuanced aggregation of the data, ultimately guiding the company to the optimal investment decision.

$$R_1 = \begin{pmatrix} (0.5, 0.4) & (0.6, 0.3) & (0.3, 0.6) & (0.2, 0.7) \\ (0.7, 0.3) & (0.7, 0.2) & (0.7, 0.2) & (0.4, 0.5) \\ (0.6, 0.4) & (0.5, 0.4) & (0.5, 0.3) & (0.2, 0.3) \\ (0.8, 0.1) & (0.6, 0.3) & (0.3, 0.4) & (0.2, 0.6) \\ (0.6, 0.2) & (0.4, 0.3) & (0.7, 0.1) & (0.1, 0.3) \end{pmatrix}$$

$$R_2 = \begin{pmatrix} (0.4, 0.3) & (0.5, 0.2) & (0.2, 0.5) & (0.1, 0.6) \\ (0.6, 0.2) & (0.6, 0.1) & (0.6, 0.1) & (0.3, 0.4) \\ (0.5, 0.3) & (0.4, 0.3) & (0.4, 0.2) & (0.5, 0.2) \\ (0.7, 0.1) & (0.5, 0.2) & (0.2, 0.3) & (0.1, 0.5) \\ (0.5, 0.1) & (0.3, 0.2) & (0.6, 0.2) & (0.4, 0.2) \end{pmatrix}$$

$$R_3 = \begin{pmatrix} (0.4, 0.5) & (0.5, 0.4) & (0.2, 0.7) & (0.1, 0.8) \\ (0.6, 0.4) & (0.6, 0.3) & (0.6, 0.3) & (0.3, 0.6) \\ (0.5, 0.5) & (0.4, 0.5) & (0.4, 0.4) & (0.5, 0.4) \\ (0.7, 0.2) & (0.5, 0.4) & (0.2, 0.5) & (0.1, 0.7) \\ (0.5, 0.3) & (0.3, 0.4) & (0.6, 0.2) & (0.4, 0.4) \end{pmatrix}$$

Step -1: With the help of IFWAA operator, one can ascertain the alternative's intuitionistic fuzzy values to determine one's ultimate preferences. Assume that a collection of intuitionistic fuzzy numbers follows the operation of the IFWAA operation. Let,

$$\text{IFWAA}(\tilde{a}_1, \tilde{a}_2, \dots, \tilde{a}_n) = \sum_{j=1}^n \tilde{a}_j \omega_j = \left(1 - \prod_{j=1}^n (1 - \mu_{a_j})^{\omega_j}, \prod_{j=1}^n (\gamma_{a_j})^{\omega_j} \right).$$

$$\tilde{r}_1^{(1)} = (1 - [(1 - \mu_{a_1})^{\omega_1} (1 - \mu_{a_2})^{\omega_2} (1 - \mu_{a_3})^{\omega_3} (1 - \mu_{a_4})^{\omega_4}], (\gamma_{a_1})^{\omega_1} (\gamma_{a_2})^{\omega_2} (\gamma_{a_3})^{\omega_3} (\gamma_{a_4})^{\omega_4})$$

$$\begin{aligned}
r_1^{(1)} &= \left(\left(1 - \left[\frac{(1-0.5)^{0.343122} * (1-0.6)^{0.191853}}{(1-0.3)^{0.226570} * (1-0.2)^{0.238453}} \right] \right), \left(\frac{(0.4)^{0.343122} * (0.3)^{0.191853}}{(0.6)^{0.226570} * (0.7)^{0.238453}} \right) \right) = (0.4217, 0.4742) \\
r_2^{(1)} &= \left(\left(1 - \left[\frac{(1-0.7)^{0.343122} * (1-0.7)^{0.191833}}{(1-0.7)^{0.226570} * (1-0.4)^{0.238453}} \right] \right), \left(\frac{(0.3)^{0.343122} * (0.2)^{0.191833}}{(0.2)^{0.226570} * (0.5)^{0.238453}} \right) \right) = (0.6461, 0.2860) \\
r_3^{(1)} &= \left(\left(1 - \left[\frac{(1-0.6)^{0.343122} * (1-0.5)^{0.191833}}{(1-0.5)^{0.226570} * (1-0.2)^{0.238453}} \right] \right), \left(\frac{(0.4)^{0.343122} * (0.4)^{0.191833}}{(0.3)^{0.226570} * (0.3)^{0.238453}} \right) \right) = (0.4819, 0.3499) \\
r_4^{(1)} &= \left(\left(1 - \left[\frac{(1-0.8)^{0.343122} * (1-0.6)^{0.191853}}{(1-0.3)^{0.226570} * (1-0.2)^{0.238453}} \right] \right), \left(\frac{(0.1)^{0.343122} * (0.3)^{0.191853}}{(0.4)^{0.226570} * (0.6)^{0.238453}} \right) \right) = (0.5777, 0.2591) \\
r_5^{(1)} &= \left(\left(1 - \left[\frac{(1-0.6)^{0.343122} * (1-0.4)^{0.191853}}{(1-0.7)^{0.226570} * (1-0.1)^{0.238453}} \right] \right), \left(\frac{(0.2)^{0.343122} * (0.3)^{0.191853}}{(0.1)^{0.226570} * (0.3)^{0.238453}} \right) \right) = (0.5085, 0.2035)
\end{aligned}$$

$$\begin{aligned}
r_1^{(2)} &= \left(\left(1 - \left[\frac{(1-0.4)^{0.343122} * (1-0.5)^{0.191853}}{(1-0.2)^{0.226570} * (1-0.1)^{0.238453}} \right] \right), \left(\frac{(0.3)^{0.343122} * (0.2)^{0.191853}}{(0.5)^{0.226570} * (0.6)^{0.238453}} \right) \right) = (0.3180, 0.3676) \\
r_2^{(2)} &= \left(\left(1 - \left[\frac{(1-0.6)^{0.343122} * (1-0.6)^{0.191853}}{(1-0.6)^{0.226570} * (1-0.3)^{0.238453}} \right] \right), \left(\frac{(0.2)^{0.343122} * (0.1)^{0.191853}}{(0.1)^{0.226570} * (0.4)^{0.238453}} \right) \right) = (0.5429, 0.1765) \\
r_3^{(2)} &= \left(\left(1 - \left[\frac{(1-0.5)^{0.343122} * (1-0.4)^{0.191853}}{(1-0.4)^{0.226570} * (1-0.5)^{0.238453}} \right] \right), \left(\frac{(0.3)^{0.343122} * (0.3)^{0.191853}}{(0.2)^{0.226570} * (0.2)^{0.238453}} \right) \right) = (0.4604, 0.2485) \\
r_4^{(2)} &= \left(\left(1 - \left[\frac{(1-0.7)^{0.343122} * (1-0.5)^{0.191853}}{(1-0.2)^{0.226570} * (1-0.1)^{0.238453}} \right] \right), \left(\frac{(0.1)^{0.343122} * (0.2)^{0.191853}}{(0.3)^{0.226570} * (0.5)^{0.238453}} \right) \right) = (0.4630, 0.2150) \\
r_5^{(2)} &= \left(\left(1 - \left[\frac{(1-0.5)^{0.343122} * (1-0.3)^{0.191853}}{(1-0.6)^{0.226570} * (1-0.4)^{0.238453}} \right] \right), \left(\frac{(0.1)^{0.343122} * (0.2)^{0.191853}}{(0.2)^{0.226570} * (0.2)^{0.238453}} \right) \right) = (0.4704, 0.1577)
\end{aligned}$$

$$\begin{aligned}
r_1^{(3)} &= \left(\left(1 - \left[\frac{(1-0.4)^{0.343122} * (1-0.5)^{0.191853}}{(1-0.2)^{0.226570} * (1-0.1)^{0.238453}} \right] \right), \left(\frac{(0.5)^{0.343122} * (0.4)^{0.191853}}{(0.7)^{0.226570} * (0.8)^{0.238453}} \right) \right) = (0.3188, 0.5783) \\
r_2^{(3)} &= \left(\left(1 - \left[\frac{(1-0.6)^{0.343122} * (1-0.6)^{0.191853}}{(1-0.6)^{0.226570} * (1-0.3)^{0.238453}} \right] \right), \left(\frac{(0.4)^{0.343122} * (0.3)^{0.191853}}{(0.3)^{0.226570} * (0.6)^{0.238453}} \right) \right) = (0.5429, 0.3906) \\
r_3^{(3)} &= \left(\left(1 - \left[\frac{(1-0.5)^{0.343122} * (1-0.4)^{0.1}}{(1-0.4)^{0.226570} * (1-0.5)^{0.238453}} \right] \right), \left(\frac{(0.5)^{0.343122} * (0.5)^{0.1}}{(0.4)^{0.226570} * (0.4)^{0.238453}} \right) \right) = (0.4604, 0.4507) \\
r_4^{(3)} &= \left(\left(1 - \left[\frac{(1-0.7)^{0.343122} * (1-0.5)^{0.191853}}{(1-0.2)^{0.226570} * (1-0.1)^{0.238453}} \right] \right), \left(\frac{(0.2)^{0.343122} * (0.4)^{0.191853}}{(0.5)^{0.226570} * (0.7)^{0.238453}} \right) \right) = (0.4630, 0.3790) \\
r_5^{(3)} &= \left(\left(1 - \left[\frac{(1-0.5)^{0.343122} * (1-0.3)^{0.191853}}{(1-0.6)^{0.226570} * (1-0.4)^{0.238453}} \right] \right), \left(\frac{(0.3)^{0.343122} * (0.4)^{0.191853}}{(0.2)^{0.226570} * (0.4)^{0.238453}} \right) \right) = (0.4704, 0.3097)
\end{aligned}$$

Step - 2: To figure out the alternative's cumulative consensus of intuitionistic fuzzy values, apply the IFHA operator. Assume that a collection of intuitionistic fuzzy numbers follows the operation of the IFHA along with the balancing vector. Then we have, $r_1^{(1)} = (0.4217, 0.4742)$; $r_1^{(2)} = (0.3180, 0.3676)$; $r_1^{(3)} =$

(0.3188, 0.5783). Where $\omega = (0.156837, 0.334829, 0.508333)$ and $\gamma^T = (0.592557, 0.271628, 0.135814)$

$$\begin{aligned}\tilde{\mu}_1 &= \mathbf{b}^{(n \times \gamma_1)} = (0.4217)^{(3 * 0.135814)} = 0.7034 \\ \tilde{\mu}_2 &= \mathbf{b}^{(n \times \gamma_2)} = (0.3180)^{(3 * 0.271628)} = 0.3931 \\ \tilde{\mu}_3 &= \mathbf{b}^{(n \times \gamma_3)} = (0.3188)^{(3 * 0.592557)} = 0.1310 \\ \tilde{\gamma}_1 &= \mathbf{b}^{(n \times \gamma_1)} = (0.4742)^{(3 * 0.135814)} = 0.7379 \\ \tilde{\gamma}_2 &= \mathbf{b}^{(n \times \gamma_2)} = (0.3676)^{(3 * 0.271628)} = 0.4424 \\ \tilde{\gamma}_3 &= \mathbf{b}^{(n \times \gamma_3)} = (0.5783)^{(3 * 0.592557)} = 0.3777\end{aligned}$$

Utilizing IFHA operator we get,

$$\begin{aligned}\text{IFHA}_{\omega, w}(\tilde{\mathbf{a}}_1, \tilde{\mathbf{a}}_2, \dots, \tilde{\mathbf{a}}_n) &= \left[1 - \prod_{j=1}^n (1 - \tilde{\mu}_{\tilde{\mathbf{a}}_{\sigma(j)}})^{w_j}, \prod_{j=1}^n (\tilde{\gamma}_{\tilde{\mathbf{a}}_{\sigma(j)}})^{w_j} \right] \\ r_1 &= (1 - [(1 - \mu_{a_1})^{w_1} (1 - \mu_{a_2})^{w_2} (1 - \mu_{a_3})^{w_3}], (\gamma_{a_1})^{w_1} (\gamma_{a_2})^{w_2} (\gamma_{a_3})^{w_3}) \\ r_1 &= \left(\left(1 - \left[\begin{array}{l} (1 - 0.7034)^{0.508333 *} \\ (1 - 0.3931)^{0.334829 *} \\ (1 - 0.1310)^{0.156837} \end{array} \right] \right), \left(\begin{array}{l} (0.7379)^{0.508333 *} \\ (0.4424)^{0.334829 *} \\ (0.3777)^{0.156837} \end{array} \right) \right) = (0.5538, 0.5598)\end{aligned}$$

Similarly, $r_2^{(1)} = (0.6461, 0.2860); r_2^{(2)} = (0.5429, 0.1765); r_2^{(3)} = (0.5429, 0.3906)$

$$\begin{aligned}\tilde{\mu}_1 &= \mathbf{b}^{(n \times \gamma_1)} = (0.6461)^{(3 * 0.135814)} = 0.8370 \\ \tilde{\mu}_2 &= \mathbf{b}^{(n \times \gamma_2)} = (0.5429)^{(3 * 0.271628)} = 0.6079 \\ \tilde{\mu}_3 &= \mathbf{b}^{(n \times \gamma_3)} = (0.5429)^{(3 * 0.592557)} = 0.3376 \\ \tilde{\gamma}_1 &= \mathbf{b}^{(n \times \gamma_1)} = (0.2860)^{(3 * 0.135814)} = 0.6005 \\ \tilde{\gamma}_2 &= \mathbf{b}^{(n \times \gamma_2)} = (0.1765)^{(3 * 0.271628)} = 0.2433 \\ \tilde{\gamma}_3 &= \mathbf{b}^{(n \times \gamma_3)} = (0.3906)^{(3 * 0.592557)} = 0.1880\end{aligned}$$

$$r_2 = \left(\left(1 - \left[\begin{array}{l} (1 - 0.8370)^{0.508333 *} \\ (1 - 0.6079)^{0.334829 *} \\ (1 - 0.3376)^{0.156837} \end{array} \right] \right), \left(\begin{array}{l} (0.6005)^{0.508333 *} \\ (0.2433)^{0.334829 *} \\ (0.1880)^{0.156837} \end{array} \right) \right) = (0.7275, 0.3699)$$

$r_3^{(1)} = (0.4819, 0.3499); r_3^{(2)} = (0.4604, 0.2485); r_3^{(3)} = (0.4604, 0.4507)$

$$\begin{aligned}\tilde{\mu}_1 &= \mathbf{b}^{(n \times \gamma_1)} = (0.4819)^{(3 * 0.135814)} = 0.7427 \\ \tilde{\mu}_2 &= \mathbf{b}^{(n \times \gamma_2)} = (0.4604)^{(3 * 0.271628)} = 0.5315 \\ \tilde{\mu}_3 &= \mathbf{b}^{(n \times \gamma_3)} = (0.4604)^{(3 * 0.592557)} = 0.2519 \\ \tilde{\gamma}_1 &= \mathbf{b}^{(n \times \gamma_1)} = (0.3499)^{(3 * 0.135814)} = 0.6519 \\ \tilde{\gamma}_2 &= \mathbf{b}^{(n \times \gamma_2)} = (0.2485)^{(3 * 0.271628)} = 0.3216 \\ \tilde{\gamma}_3 &= \mathbf{b}^{(n \times \gamma_3)} = (0.4507)^{(3 * 0.592557)} = 0.2425\end{aligned}$$

$$r_3 = \left(\left(1 - \begin{bmatrix} (1 - 0.7427)^{0.508333*} \\ (1 - 0.5315)^{0.334829*} \\ (1 - 0.2519)^{0.156837} \end{bmatrix} \right), \left(\begin{bmatrix} (0.6519)^{0.508333*} \\ (0.3216)^{0.334829*} \\ (0.2425)^{0.156837} \end{bmatrix} \right) \right) = (0.6282, 0.4406)$$

$$r_4^{(1)} = (0.5777, 0.2591); r_4^{(2)} = (0.4630, 0.2150); r_4^{(3)} = (0.4630, 0.3790)$$

$$\tilde{\mu}_1 = \mathbf{b}^{(n \times \gamma_1)} = (0.5777)^{(3*0.135814)} = 0.7997$$

$$\tilde{\mu}_2 = \mathbf{b}^{(n \times \gamma_2)} = (0.4630)^{(3*0.271628)} = 0.5339$$

$$\tilde{\mu}_3 = \mathbf{b}^{(n \times \gamma_3)} = (0.4630)^{(3*0.592557)} = 0.2544$$

$$\tilde{\gamma}_1 = \mathbf{b}^{(n \times \gamma_1)} = (0.2591)^{(3*0.135814)} = 0.5768$$

$$\tilde{\gamma}_2 = \mathbf{b}^{(n \times \gamma_2)} = (0.2150)^{(3*0.271628)} = 0.2858$$

$$\tilde{\gamma}_3 = \mathbf{b}^{(n \times \gamma_3)} = (0.3790)^{(3*0.592557)} = 0.1782$$

$$r_4 = \left(\left(1 - \begin{bmatrix} (1 - 0.7997)^{0.508333*} \\ (1 - 0.5339)^{0.334829*} \\ (1 - 0.2544)^{0.156837} \end{bmatrix} \right), \left(\begin{bmatrix} (0.5768)^{0.508333*} \\ (0.2858)^{0.334829*} \\ (0.1782)^{0.156837} \end{bmatrix} \right) \right) = (0.6734, 0.3792)$$

$$r_5^{(1)} = (0.5085, 0.2035); r_5^{(2)} = (0.4704, 0.1577); r_5^{(3)} = (0.4704, 0.3097)$$

$$\tilde{\mu}_1 = \mathbf{b}^{(n \times \gamma_1)} = (0.5085)^{(3*0.135814)} = 0.7592$$

$$\tilde{\mu}_2 = \mathbf{b}^{(n \times \gamma_2)} = (0.4704)^{(3*0.271628)} = 0.5409$$

$$\tilde{\mu}_3 = \mathbf{b}^{(n \times \gamma_3)} = (0.4704)^{(3*0.592557)} = 0.2617$$

$$\tilde{\gamma}_1 = \mathbf{b}^{(n \times \gamma_1)} = (0.2035)^{(3*0.135814)} = 0.5227$$

$$\tilde{\gamma}_2 = \mathbf{b}^{(n \times \gamma_2)} = (0.1577)^{(3*0.271628)} = 0.2220$$

$$\tilde{\gamma}_3 = \mathbf{b}^{(n \times \gamma_3)} = (0.3097)^{(3*0.592557)} = 0.1245$$

$$r_5 = \left(\left(1 - \begin{bmatrix} (1 - 0.7592)^{0.508333*} \\ (1 - 0.5409)^{0.334829*} \\ (1 - 0.2617)^{0.156837} \end{bmatrix} \right), \left(\begin{bmatrix} (0.5227)^{0.508333*} \\ (0.2220)^{0.334829*} \\ (0.1245)^{0.156837} \end{bmatrix} \right) \right) = (0.6437, 0.3133)$$

Hence,

$$r_1 = (0.5538, 0.5598); r_2 = (0.7275, 0.3699); r_3 = (0.6282, 0.4406); r_4 = (0.6734, 0.3792); r_5 = (0.6437, 0.3133).$$

Step - 3: Ranking the alternatives using the methods of [13] and [14], we have:

$$\text{Cr}(r_1, \tilde{r}_1) = 0.7036; \text{Cr}(r_2, \tilde{r}_2) = 0.3183; \text{Cr}(r_3, \tilde{r}_3) = 0.3303;$$

$$\text{Cr}(r_4, \tilde{r}_4) = 0.2448; \text{Cr}(r_5, \tilde{r}_5) = 0.1955.$$

Step - 4: The following is the outcome of ranking each option according to the highest closeness (correlation coefficient) determined in step 3: $A_1 > A_3 > A_2 > A_4 > A_5$, and hence, the best choice is A_1 .

5. Rationale for Combining Eigenvalue Analysis with ANN in the MAGDM Framework

In the proposed MAGDM model, the eigenvalue method is used in the first stage to derive the objective weights of the decision criteria. The eigenvalue approach provides a mathematically consistent way of capturing the relative importance of criteria by analysing the spectral properties of the decision matrix. This ensures that the weighting scheme is grounded in linear algebraic principles and is free from subjectivity. Once the weights are derived, they are used to construct the collective decision matrix, which represents an aggregated and weighted evaluation of all alternatives. This matrix captures the essential information required for ranking. However, traditional eigenvalue-based ranking methods operate in a fully deterministic linear framework. They do not account for nonlinear interactions, hidden patterns, or subtle dependencies that may exist among criteria or alternatives. To address this, an Artificial Neural Network (ANN) is incorporated in the second stage of analysis. The ANN takes the collective matrix as input and learns a data-driven scoring function. Since the network minimizes error during training, the alternative corresponding to the minimum error convergence point (or highest predicted score) can be interpreted as the most stable and consistent alternative under the learned representation. Thus, the ANN serves as a nonlinear refinement tool that validates and enhances the decision outcome obtained from eigenvalue weighting. By integrating eigenvalue analysis and ANN, the framework benefits from both:

1. Mathematical rigor and objectivity (via eigenvalue-based weighting), and
2. Adaptive, nonlinear pattern discovery (via ANN learning).

This hybrid mechanism increases robustness and provides a richer interpretation of the decision-making process than either method alone.

6. The Markov Chain Modeled ANN for Solving MAGDM Problem

The decision matrices involved in the above numerical illustration are given as follows, where they are transformed into Symmetric matrices obtained from the Stochastic matrices as discussed by Robinson & Prakash, in [14].

$$\widetilde{RSM}_1 = \begin{bmatrix} 0.239 & 0.2525 & 0.1795 & 0.2025 & 0.2235 \\ 0.2525 & 0.238 & 0.214 & 0.192 & 0.1695 \\ 0.1795 & 0.214 & 0.207 & 0.1955 & 0.201 \\ 0.2025 & 0.192 & 0.1955 & 0.111 & 0.164 \\ 0.2235 & 0.1695 & 0.201 & 0.164 & 0.22 \end{bmatrix}$$

$$\widetilde{RSM}_2 = \begin{bmatrix} 0.239 & 0.2525 & 0.1795 & 0.16 & 0.22 \\ 0.2525 & 0.238 & 0.214 & 0.175 & 0.1665 \\ 0.1795 & 0.214 & 0.207 & 0.194 & 0.213 \\ 0.16 & 0.175 & 0.194 & 0.2365 & 0.186 \\ 0.22 & 0.1665 & 0.213 & 0.186 & 0.1585 \end{bmatrix}$$

$$\widetilde{RSM}_3 = \begin{bmatrix} 0.237 & 0.254 & 0.166 & 0.199 & 0.2405 \\ 0.254 & 0.2365 & 0.208 & 0.1805 & 0.1725 \\ 0.166 & 0.208 & 0.2 & 0.1845 & 0.2275 \\ 0.199 & 0.1805 & 0.1845 & 0.085 & 0.1975 \\ 0.2405 & 0.1725 & 0.2275 & 0.1975 & 0.182 \end{bmatrix}$$

Using these corresponding symmetric matrices $\widetilde{RSM}_1, \widetilde{RSM}_2, \widetilde{RSM}_3$ find the Eigen values and Eigen vectors separately as follows: Eigen Values and vectors for \widetilde{RSM}_1 :

$$\begin{aligned} \lambda_1 = -0.065; \quad v_1 &= \begin{pmatrix} -0.335 \\ 0.534 \\ -0.075 \\ -1.200 \\ 1 \end{pmatrix} & \lambda_2 = -0.060; \quad v_2 &= \begin{pmatrix} -1.865 \\ 1.002 \\ -1.697 \\ 1.906 \\ 1 \end{pmatrix} \\ \lambda_3 = 0.029; \quad v_3 &= \begin{pmatrix} 0.675 \\ -1.142 \\ -0.629 \\ 0.176 \\ 1 \end{pmatrix} & \lambda_4 = 0.050; \quad v_4 &= \begin{pmatrix} -1.988 \\ -1.567 \\ 2.452 \\ 0.537 \\ 1 \end{pmatrix} \\ \lambda_5 = 1.006; \quad v_5 &= \begin{pmatrix} 1.159 \\ 1.130 \\ 1.047 \\ 0.947 \\ 1 \end{pmatrix} \end{aligned}$$

Eigen Values for \widetilde{RSM}_2 :

$$\begin{aligned} \lambda_1 = -0.071; \quad v_1 &= \begin{pmatrix} 0.754 \\ 0.607 \\ -0.638 \\ -0.156 \\ 1 \end{pmatrix} & \lambda_2 = 0.021; \quad v_2 &= \begin{pmatrix} 2.011 \\ -1.548 \\ -2.583 \\ 1.227 \\ 1 \end{pmatrix} \\ \lambda_3 = 0.024; \quad v_3 &= \begin{pmatrix} -5.043 \\ -3.972 \\ 2.088 \\ 6.831 \\ 1 \end{pmatrix}; \lambda_4 = 0.103; \quad v_4 &= \begin{pmatrix} -5.043 \\ -3.972 \\ 2.088 \\ 6.831 \\ 1 \end{pmatrix}; \\ \lambda_5 = 1.002; \quad v_5 &= \begin{pmatrix} 1.117 \\ 1.114 \\ 1.064 \\ 1.000 \\ 1 \end{pmatrix} \end{aligned}$$

Eigen Values for \widetilde{RSM}_3 :

$$\begin{aligned} \lambda_1 = -0.086; \quad v_1 &= \begin{pmatrix} -0.774 \\ 0.600 \\ -0.691 \\ -0.142 \\ 1 \end{pmatrix}; \lambda_2 = -0.075; \quad v_2 &= \begin{pmatrix} 1.138 \\ 0.152 \\ 1.156 \\ -4.152 \\ 1 \end{pmatrix}; \\ \lambda_3 = 0.029; \quad v_3 &= \begin{pmatrix} 1.001 \\ -1.352 \\ -0.893 \\ 0.217 \\ 1 \end{pmatrix}; \lambda_4 = 0.065; \quad v_4 &= \begin{pmatrix} -1.378 \\ -1.378 \\ 1.732 \\ 0.295 \\ 1 \end{pmatrix} \end{aligned}$$

$$\lambda_5 = 1.007; \quad v_5 = \begin{pmatrix} 1.081 \\ 1.038 \\ 0.965 \\ 0.844 \\ 1 \end{pmatrix}$$

Consider the input values as X_1, X_2, X_3 corresponding to the eigen values of $\widetilde{RSM}_1, \widetilde{RSM}_2, \widetilde{RSM}_3$ respectively:

$$X_1 = \begin{pmatrix} -0.065 \\ -0.060 \\ 0.029 \\ 0.050 \\ 1.006 \end{pmatrix}, \quad X_2 = \begin{pmatrix} -0.071 \\ 0.021 \\ 0.024 \\ 0.103 \\ 1.002 \end{pmatrix}, \quad X_3 = \begin{pmatrix} -0.086 \\ -0.075 \\ 0.029 \\ 0.065 \\ 1.007 \end{pmatrix}.$$

These above three column matrices will represent the input vectors respectively for the artificial neural network.

$$X_1 = [-0.065, -0.060, 0.029, 0.050, 1.006]$$

$$X_2 = [-0.071, 0.021, 0.024, 0.103, 1.002]$$

$$X_3 = [-0.086, -0.075, 0.029, 0.065, 1.007]$$

7. Operational Workflow of the ANN-Based MAGDM Model

A single-layer perceptron model was implemented to evaluate five decision alternatives based on three input features per alternative. The input data consisted of three column vectors, each containing five values, which were combined to form a data matrix of dimension 5×3 . A target vector was then specified to guide supervised training. The perceptron architecture comprised one dense layer with a single linear output neuron, corresponding mathematically to $\omega^T x_i + b$, where ω and b are the trainable weights and bias, respectively. Training was performed using stochastic gradient descent (SGD) with a learning rate of 0.01, optimized against the mean squared error (MSE) loss function. The model was trained for 200 epochs, with all five alternatives, processed in a single batch due to the small dataset size. During training, the MSE was recorded at each epoch to monitor learning dynamics. After training, the perceptron generated predicted scores for each alternative, from which a ranking was derived by sorting in descending order. Finally, the error convergence curve, plotting MSE against epochs, was used to visualize and confirm the stability of the training process.

8. Pseudo-code for the functioning of the ANN

1. Initialize weight $\omega \in \mathbb{R}^3$ and bias $b \in \mathbb{R}^3$ with small random values.
2. Input data: construct matrix $X = \{X_1, X_2, X_3\}$ of size 5×3 , where each row corresponds to an alternative.
3. Set targets: Define target vector $y \in \mathbb{R}^5$.
4. For each epoch $t = 1, \dots, T$:
 - (a) Compute predicted outputs: $\hat{y}_i = \omega^T x_i + b$, for all $i = 1, \dots, 5$.
 - (b) Calculate error: $e_i = y_i - \hat{y}_i$.
 - (c) Compute loss: $L = \frac{1}{N} \sum_{i=1}^N e_i^2$.

(d) Update weights: $\omega \leftarrow \omega + \eta \cdot \frac{2}{N} \sum_{i=1}^N e_i x_i$. Where η = learning rate (step size in gradient descent updates).

(e) Update bias: $b \leftarrow b + \eta \cdot \frac{2}{N} \sum_{i=1}^N e_i$.

5. After training: Generate final prediction \hat{y} .

6. Ranking: Sort alternatives by \hat{y} in descending order to obtain ranking from best to worst.

7. Convergence check: Plot L versus epochs to verify decreasing error trend.

9. The Sequence of Events in the Algorithm

(i) Loss Function (Mean Squared Error, MSE)

$$L = \frac{1}{N} \sum_{i=1}^N (y_i - \hat{y}_i)^2$$

(a) y_i = target value

(b) $\hat{y}_i = \omega^T x_i + b$ = predicted value

(c) N = number of training samples

(ii) Gradient with respect to weights ω

Taking the derivative of L w.r.t ω , we get:

$$\frac{\partial L}{\partial \omega} = \frac{1}{N} \sum_{i=1}^N 2(y_i - \hat{y}_i)(-x_i)$$

Simplifying, we get:

$$\frac{\partial L}{\partial \omega} = \frac{2}{N} \sum_{i=1}^N e_i x_i,$$

where $e_i = y_i - \hat{y}_i$.

(iii) Gradient with respect to bias b

$$\frac{\partial L}{\partial b} = \frac{1}{N} \sum_{i=1}^N 2(y_i - \hat{y}_i)(-1); \quad \frac{\partial L}{\partial b} = \frac{2}{N} \sum_{i=1}^N e_i.$$

Gradient Descent Update Rule:

General update: $\theta \leftarrow \theta - \eta \frac{\partial L}{\partial \theta}$. So, for ω we have:

$$\omega \leftarrow \omega - \eta \left(\frac{2}{N} \sum_{i=1}^N e_i x_i \right) = \omega + \eta \cdot \frac{2}{N} \sum_{i=1}^N e_i x_i.$$

For b we have:

$$b \leftarrow b + \eta \cdot \frac{2}{N} \sum_{i=1}^N e_i.$$

Remark: The factor $\frac{2}{N}$ arises directly from the gradient of the mean squared error loss with respect to ω and b. The 2 comes from differentiating the square $(y_i - \hat{y}_i)^2$ and the $\frac{1}{N}$ comes from averaging over N samples.

Neural Network Architecture A single-layer perceptron (SLP) model was employed for supervised ranking of the alternatives. The network architecture consists of:

1. **Input layer:** 3 neurons (corresponding to the three feature variables X_1, X_2, X_3)
2. **Output layer:** 1 neuron (linear activation, producing a continuous score for ranking)

The complete architecture therefore, contains one fully connected (Dense) layer with 1 output unit and 3 input connections, giving a total of:

$$\text{Trainable parameters} = (3 \times 1) + 1 = 4$$

The model employs a pure linear activation, rendering it mathematically equivalent to a linear regression classifier, yet optimised through neural network gradient-based learning.

Training Parameters: The network was compiled with:

Table 1: Details of the Training Parameters

Parameter	Value
Optimizer	Stochastic Gradient Descent (SGD)
Learning rate	0.01
Loss function	Mean Squared Error (MSE)
Batch size	Full batch (5 samples)
Epochs	200
Activation (output)	Linear

These choices ensure stable gradient updates suitable for very small datasets.

Data Preparation: The input matrix $X \in \mathbb{R}^{5 \times 3}$ contains five alternatives with three numerical attributes each. The target vector y assigns identical labels (value 1) to all alternatives, allowing the network to infer subtle differences in feature patterns rather than absolute class differences. No additional normalization or preprocessing was required due to the small numerical range of the inputs.

Training Strategy: The model was trained for 200 epochs using full-batch gradient descent. Loss convergence was monitored through the Mean Squared Error curve. Convergence occurred smoothly without oscillation, confirming suitability of the learning rate. The trained perceptron produced a continuous score for each alternative, which was subsequently sorted in descending order to generate their final ranking.

Prediction and Ranking: For each alternative i , the model predicts a score s_i . The ranking is obtained by sorting these scores in descending order: $\text{Rank}(i) = \text{argsort}(-s_i)$. Higher scores correspond to higher rankings. ANN Output: Predicted scores: [1.0028679 0.97178876 1.0182507 1.0075092 0.96625006]
Ranking (best to worst): [3, 4, 1, 2, 5]

Hence, the order of ranking the alternatives is given as: $A_3 > A_4 > A_1 > A_2 > A_5$, where A_3 is the optimal alternative.

Error Convergence from the ANN

The error convergence profile of the artificial neural network (ANN) is presented in Figure X. The curve demonstrates the typical learning dynamics of a well-trained network, where the mean squared error (MSE) decreases monotonically with training epochs. During the early training phase (up to ~ 50 epochs), the error drops sharply, indicating rapid weight adaptation as the network captures the dominant structure of the input-output relationship. This steep descent corresponds to the learning of global patterns through significant parameter adjustments. In the subsequent phase (~ 50 – 100 epochs), the reduction in error becomes more gradual, representing the fine-tuning of network parameters as the ANN focuses on minimizing smaller residual discrepancies. Beyond ~ 100 epochs, the curve asymptotically approaches near-zero error, signifying convergence. This plateau reflects training stability and suggests that the chosen network architecture, learning rate, and optimization strategy were effective for the given problem domain. A critical observation is that the curve does not exhibit premature flattening at a high error level, which would have indicated underfitting (i.e., insufficient model complexity or inadequate training).

Similarly, the absence of oscillations or divergence at later epochs suggests that overfitting has not occurred, as overfitting typically manifests as a decline in training error accompanied by a rise in validation error. The smooth, monotonic convergence confirms that the learning process was numerically stable

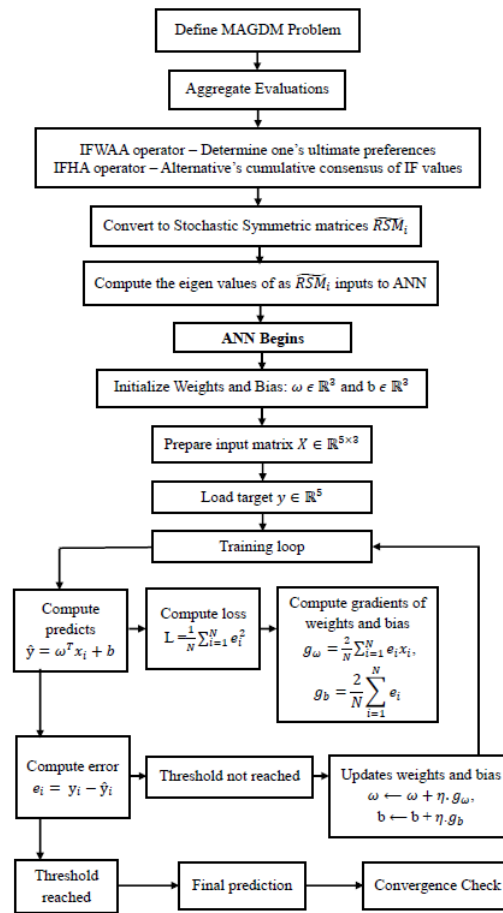


Figure 5: Workflow of the ANN-Based MAGDM Model.

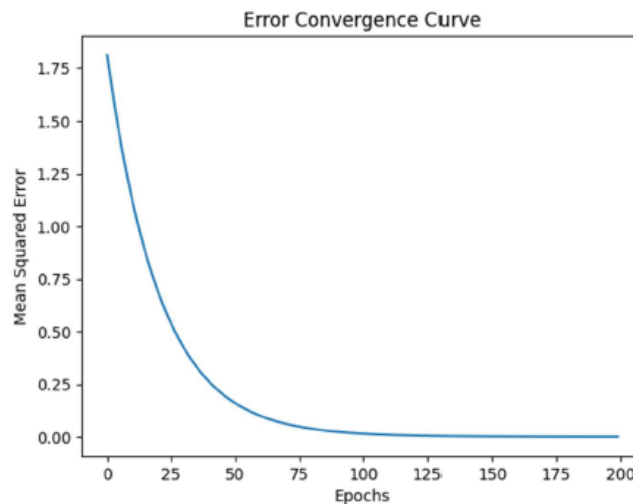


Figure 6: Mean Squared Error Convergence across Epochs for the Proposed Fuzzy-ANN Framework.

and that the model has achieved a balance between bias and variance. If, hypothetically, the convergence curve had plateaued too early at a high MSE value, it would imply that the ANN failed to adequately capture the underlying data distribution, necessitating either more training epochs, an increased number of hidden units, or alternative activation/learning rate strategies. Conversely, if the error had decreased initially but then fluctuated or increased after convergence, it would indicate overfitting, where the model

memorizes training data rather than generalizing effectively. The observed convergence behavior avoids these pitfalls, aligning with best practices in ANN training reported in the literature [13]. Overall, the convergence profile demonstrates both stability and efficiency of learning, confirming the robustness of the proposed ANN model and its suitability for generalization tasks. This convergence behavior further validates the applicability of the proposed ANN-based framework within the intuitionistic fuzzy/MAGDM setting, as the stable error reduction ensures reliable aggregation and ranking of alternatives without the risk of model instability or overfitting.

10. Comparison of the Proposed Methods of MCGDM and ANN based decision making

Since there were two different approaches proposed in this work, the comparison of the proposed approaches and some of the existing methods are tabulated below.

Table 2: Comparison of the proposed methods with existing methods

Sl. No.	Different MAGDM/MCGDM Methods	Ranking of Optimal Alternatives
1	Proposed Method of MCGDM with Real Eigen Valus for weight determination	$A_1 > A_3 > A_2 > A_4 > A_5$
2	Proposed Method of MCGDM with Artificial Neural Network & Error Convergence	$A_3 > A_4 > A_1 > A_2 > A_5$
3	Robinson & Saranraj [13] ANN inputs as Orthogonal vectors & P-IFWG operator	A_4 and A_5
4	Robinson & Saranraj [13] ANN inputs as Orthogonal vectors & IFWG operator	A_1, A_3, A_2
5	Robinson & Saranraj [13] ANN inputs as Orthogonal vectors & IFWA operator	A_1 and A_4
6	Robinson & Saranraj [13] ANN inputs as Orthogonal vectors & OWA operator	A_1, A_3, A_2
7	Robinson & Saranraj [13] ANN inputs as Orthogonal vectors & G-OWA operator	A_1, A_3, A_2
8	Robinson & Saranraj [13] ANN inputs as Orthogonal vectors & OWG operator	A_2 and A_3
9	Robinson & Saranraj [13] ANN inputs as Orthogonal vectors & G-OWG operator	A_2, A_3, A_5
10	Robinson & Prakash [14]	$A_5 > A_4 > A_3 > A_2 > A_1$

11. Discussion

The comparative analysis between the classical MAGDM approach using aggregation operators and the Artificial Neural Network (ANN)-based decision model as depicted in Table-1 revealed a striking variation in the final ranking pattern of alternatives. Interestingly, the best-ranked alternative in the classical method emerged as the least preferred option in the ANN framework. This divergence, although seemingly contradictory, can be logically explained by the fundamental difference in the underlying mechanisms of the two approaches. The classical aggregation-based MAGDM model relies on deterministic mathematical operators such as weighted averaging, OWA, or hybrid aggregations that preserve linear relationships among criteria. These methods are static, rule-driven, and operate under the assumption that attribute interactions are additive and consistent across all alternatives. Consequently, they produce rankings that are stable and transparent but may overlook nonlinear dependencies or implicit correlations among decision factors. In contrast, the ANN model represents a dynamic, data-driven paradigm. Neural networks capture nonlinear mappings between inputs and outputs through iterative learning and weight adaptation. During training, the model automatically adjusts its internal structure based on the patterns hidden within the data, which can amplify or suppress certain criteria interactions. Therefore, the ANN does not merely aggregate values and it learns complex relationships and internal decision boundaries that may not be evident in the classical model. As a result, the learned feature space can completely reshape the ranking pattern, especially if some criteria exhibit interdependence, redundancy, or conflicting influence. This reversal in ranking order thus reflects the adaptive nature of the ANN, which tends to prioritize criteria combinations that maximize predictive consistency or reduce error, rather than following pre-defined weight distributions. The observed inconsistency should not be viewed as a flaw, but rather as evidence that the ANN uncovers alternative structures of importance embedded in the decision matrix structures that classical models, limited by linearity, fail to expose. Such results highlight an essential insight: different computational paradigms lead to different interpretations of optimality. The classical aggregation method emphasizes transparency and interpretability, whereas the ANN model emphasizes adaptability and hidden pattern extraction. The contrast underscores the importance of hybrid frameworks that integrate the explainability of fuzzy aggregation with the learning capability of ANN, ensuring both interpretive clarity and analytical robustness.

12. Conclusion

This study presented a comprehensive framework for tackling Multi-Criteria Group Decision Making (MCGDM/MAGDM) within the setting of intuitionistic fuzzy sets. By employing linear space methods with both real and complex eigenvalue solutions, we established an effective strategy for deriving decision-maker weights, while normalization ensured consistency across multiple criteria. The integration of artificial neural network techniques further strengthened the weight determination process, capturing complex relationships among attributes and decision-makers. The error convergence curve confirmed the stability of the proposed ANN model, showing smooth and reliable reduction of training error without signs of overfitting or underfitting. Together, these results highlight the capability of the proposed approach to blend fuzzy mathematical theory with intelligent learning methods, offering a robust and adaptable tool for group decision making under uncertainty.

References

- [1] Adak, A. K., Gunjan, M. K., & Haghghi, A. M. (2024). Distance Measure of Picture Fuzzy sets and its application in Multi-Criteria Decision Making Problem. *Journal of Fuzzy Extension and Applications*, 5(4), 594–604. [1](#)
- [2] Atanassov, K., Sotirov, S., Pencheva, T. (2023). Intuitionistic Fuzzy Deep Neural Network. *Mathematics*, 11, 716, 1–14. [1](#)
- [3] Augustine, E. P. (2021). Novel correlation coefficient for intuitionistic fuzzy sets and its application to multi-criteria decision-making problems. *International Journal of Fuzzy System Applications (IJFSA)*, 10(2), 39–58. [1](#)
- [4] Bishop, C. M. (2006). *Pattern Recognition and Machine Learning*. Springer. [1](#)

- [5] Dargam, F. C., Perz, E., Bergmann, S., Rodionova, E., Sousa, P., Souza, F. A., Matias, T., Ortiz, J. M., Esteve-Nuñez, A., Rodenas, P., & Bonachela, P. Z. (2023). Operational Decision-Making on Desalination Plants: From Process Modelling and Simulation to Monitoring and Automated Control With Machine Learning. *International Journal of Decision Support System Technology*, 15(2), 1–20. [1](#)
- [6] Ejegwa, P. A., & Onyeke, I. C. (2022). A novel intuitionistic fuzzy correlation algorithm and its applications in pattern recognition and student admission process. *International Journal of Fuzzy System Applications (IJFSA)*, 11(1), 1–20. [1](#)
- [7] Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep Learning*. MIT Press. [1](#)
- [8] Gora, P. (2023). Intuitionistic Fuzzy Modulus Similarity Measure. *International Journal of Decision Support System Technology*, 15(1), 1–22. [1](#)
- [9] Hajek, P., Olej, V., Froelich, W., & Novotny, J. (2021). Intuitionistic Fuzzy Neural Network for Time Series Forecasting The Case of Metal Prices. In: Maglogiannis, I., Macintyre, J., Iliadis, L. (eds) *Artificial Intelligence Applications and Innovations*. IFIP Advances in Information and Communication Technology, vol 627. Springer, Cham. [1](#)
- [10] Kuo, R. J., & Cheng, W. C. (2019). An intuitionistic fuzzy neural network with gaussian membership function. *Journal of Intelligent & Fuzzy Systems*, 36(6), 6731–6741. [1](#)
- [11] Leonishiya, A., Robinsiya, A., & Robinson, J. P. (2023). Varieties of Linguistic Intuitionistic Fuzzy Distance Measures for Linguistic Intuitionistic Fuzzy TOPSIS Method. *Indian Journal of Science and Technology*, 16(33). [1](#)
- [12] Leonishiya, A., & Robinson, J. P. (2023). A Fully Linguistic Intuitionistic Fuzzy Artificial Neural Network Model for Decision Support Systems. *Indian Journal of Science and Technology*, 16(SP4), 29-36. [1](#)
- [13] Robinson, J. P., & Saranraj, A. (2024). Intuitionistic Fuzzy Gram-Schmidt Orthogonalized Artificial Neural Network for solving MAGDM Problems. *Indian Journal of Science and Technology*, 17(24), 2529-2537. [1](#), [3](#), [4](#), [2](#)
- [14] Robinson, P. J., & Prakash, S. W. A. (2024). Stochastic artificial neural network for MAGDM problem solving in intuitionistic fuzzy environment. *The Scientific Temper*, 15(03), 2481-2488. [1](#), [3](#), [4](#), [6](#), [2](#)
- [15] Rumelhart, D., Hinton, G. E., & Williams, R. (1986). Learning representations by back-propagating errors. *Nature*, 323, 533-536. [1](#)
- [16] Samet, S., Laouar, M. R., Bendib, I., & Eom, S. (2022). Analysis and Prediction of Diabetes Disease Using Machine Learning Methods. *International Journal of Decision Support System Technology*, 14(1), 119. [1](#)
- [17] Sarmiento dos Santos-Neto, J. B., & Costa, A. P. (2023). A Multi-Criteria Decision-Making Model for Selecting a Maturity Model. *International Journal of Decision Support System Technology*, 15(1), 115. [1](#)
- [18] Singh, S., Sharma, S., & Lalotra, S. (2020). Generalized correlation coefficients of intuitionistic fuzzy sets with application to MAGDM and clustering analysis. *International Journal of Fuzzy Systems*, 22, 1582-1595. [1](#)
- [19] Taherdoost, H. (2023). *Deep Learning and Neural Networks: Decision-Making Implications*. *Symmetry*, 15(9), 1723. [1](#)
- [20] Zhang, Q., Li, H., Lu, L., & Wu, C. (2022). A Neural Network-Based Approach to Multi-Attribute Group Decision-Making with Heterogeneous Preference Information. *Scientific Programming*, vol. 2022 (Article ID 9033237). [1](#)