International Journal of Approximate Reasoning 187 (2025) 109553

Contents lists available at ScienceDirect

International Journal of Approximate Reasoning

journal homepage: www.elsevier.com/locate/ijar

Self-supervised multi-level generative adversarial network data imputation algorithm

Yi Xu [a,b,*], Shujuan Fang [a], Xuhui Xing [a]

[a] School of Computer Science and Technology, Anhui University, Hefei 230601, China
[b] Key Laboratory of Intelligent Computing and Signal Processing, Ministry of Education, Anhui University, Hefei 230601, China

ARTICLE INFO

Keywords:
Granular computing
Missing data
Imputation
Generative adversarial network

ABSTRACT

Data missing has always been a challenging problem in machine learning. The Generative Adversarial Imputation Networks (GAIN) have been shown to outperform many existing solutions. However, in GAIN, because missing values lack ground truth as supervision, it is unable to construct reconstruction loss for missing values and can only judge the reasonableness of imputed values based on reconstruction loss of non-missing values and adversarial loss. From the perspective of granular computing, data has levels, and data at different levels of granularity encapsulates different knowledge. Therefore, based on granular computing, this paper proposes a self-supervised multi-level generative adversarial network data imputation algorithm (MGAIN). Firstly, multiple levels of data are constructed using nested feature set sequences. Then, GAIN is used to impute missing values at the coarsest granularity level, and the imputation results of missing values at the coarse granularity level are used as supervision for imputing missing values at the fine granularity level, constructing reconstruction loss for missing values at the fine granularity level. Finally, based on reconstruction loss of missing values, reconstruction loss of non-missing values, and adversarial loss, data at the finer granularity level is imputed. MGAIN imputes missing values level by level from the coarse granularity level to the fine granularity level to obtain more accurate imputation results. Experimental results validate the effectiveness of the proposed method.

1. Introduction

With the rapid development and popularization of information technology, the global volume of data is growing rapidly, and these data contain valuable information [1]. However, data often goes missing due to various reasons, such as temporary unavailability or loss of records [2,3]. Fig. 1 presents a numerical data sample with six missing values. The dataset comprises three numerical features: height, weight, and health risk. The data includes records for five users, some of whom have missing values. For instance, User 2 in the figure has missing values for both weight and health risk features. Methods for handling missing data can be broadly categorized into deletion and imputation approaches [4]. Deletion methods can be used when the amount of missing data is small. However, when a significant portion of the data is missing, deletion methods result in a loss of data utility, so imputation methods are commonly employed to handle missing data. The Generative Adversarial Imputation Network (GAIN), as a mainstream model for missing data imputation, has been proven to outperform many existing methods [5].

* Corresponding author at: School of Computer Science and Technology, Anhui University, Hefei 230601, China.
E-mail addresses: xuyi1023@126.com (Y. Xu), e22301222@stu.ahu.edu.cn (S. Fang), xxhstudy2021@163.com (X. Xing).

https://doi.org/10.1016/j.ijar.2025.109553
Received 5 February 2025; Received in revised form 28 July 2025; Accepted 18 August 2025

Available online 27 August 2025

Feature	Height (cm)	Weight (kg)	Health risk(%)
User1	165	50	NaN
User2	170	NaN	NaN
User3	190	90	0.22
User4	NaN	80	0.56
User5	168	NaN	NaN

non-missing value missing value

Fig. 1. A numerical data sample containing six missing values.

When using the Generative Adversarial Imputation Network (GAIN) to impute numeric data with missing values, the generator generates a value for each component, regardless of whether its value is missing, and after replacing the generated value on the non-missing component with a non-missing value, the discriminator determines whether the value of that component is generated by the generator or a non-missing value. The optimal imputation results are achieved through adversarial training between the generator and the discriminator. However, in the process of using the generator of GAIN to impute missing values, there is no ground truth available for the missing values as supervisory information. As a result, it is not possible to construct a reconstruction loss specifically for the missing values, and the imputation results can only be evaluated based on the reconstruction loss of non-missing values and the discriminator loss. Without the supervisory information on the missing values, the full potential of the information contained in the data cannot be fully exploited, thereby limiting the improvement of data imputation quality.

From the perspective of granular computing, data has levels of granularity [6], and different levels of data contain different knowledge. If we can divide the data into multiple levels of granularity, then GAN can not only learn different knowledge across different granularity levels but also leverage the relationships between these levels to improve the model, thereby enhancing the quality of data imputation. For the convenience of expression, the level of granularity is simplified as level in the following text. Therefore, this paper proposes a self-supervised multi-level generative adversarial network data imputation algorithm (MGAIN). Firstly, multiple levels of data are constructed based on a nested sequence of feature sets. Then, GAIN is used to impute the missing values at the coarsest granulation level, and the imputation results of missing values at the coarse granulation level are utilized as supervisory information for imputing missing values at the fine granulation level, leading to the construction of reconstruction loss for missing values at the fine granulation level. Finally, the missing values at the fine granulation level are imputed based on the reconstruction loss of missing values, the reconstruction loss of non-missing values, and the adversarial loss. The imputation results at the finest granularity level are considered the final results. The self-supervised multi-level generative adversarial network (MGAIN), which imputes missing values progressively from coarse to fine granularity, achieves more accurate results in imputing missing values in numerical data and provides high-quality data for downstream tasks. For example, in the field of intelligent healthcare, patients are typically subjected to various types of examinations based on their medical conditions and diagnostic needs. However, due to reasons such as incomplete examinations, equipment failures, or data transmission loss, data may be missing. In such cases, MGAIN can be applied to impute the missing values in patient data, not only enhancing the completeness of the dataset but also effectively facilitating downstream tasks such as disease prediction and the formulation of personalized treatment plans. Experimental results demonstrate that the MGAIN outperforms mainstream algorithms, including GAIN, especially when the missing rate is high, the superiority becomes more evident.

The main contributions of this work are as follows:

(i) A multilevel construction method for data is proposed for the GAIN network based on nested feature set sequences.
(ii) The missing value imputation results from the coarse granulation level are utilized as supervised information for imputing missing values in the fine granulation level, constructing a reconstruction loss for missing values.
(iii) The proposed self-supervised multi-level generative adversarial network imputes data in a level-by-level manner, starting from the coarse granulation level and progressing to the fine granulation level. By utilizing the self-supervised information between multiple levels, the quality of data imputation is improved.

2. Related work

In this section, a brief overview is provided on three main themes related to this work, namely three types of imputation methods, granularity calculation, and Generative Adversarial Imputation Networks.

2.1. Imputation methods

The existing traditional data imputation methods can generally be classified into three categories [7]. The first category is based on statistical analysis, such as zero imputation, mean imputation, and constant imputation [8]. However, these methods can easily

cause data distortion, reduce data robustness, and disrupt the original distribution of the data. The second category of methods is based on traditional machine learning techniques for data imputation. Multivariate Imputation by Chained Equations (MICE) initially assigns a random value to each missing value and then iteratively updates the values of specific variables using the values of other variables until the algorithm converges and completes the imputation [9,10]. However, the MICE algorithm is less efficient for datasets with a large number of missing values, and it is based on simple models like linear regression or logistic regression for prediction, which may not capture the complex relationships in the data. The MissForest algorithm treats variables with missing data as the target variable and other variables as features [11]. It trains a random forest model using the non-missing data as the training set, then uses this model to predict the missing values. However, the MissForest algorithm relies on random forest for prediction, and such complex models may lead to overfitting and inaccurate imputation results. The Matrix Completion algorithm decomposes an incomplete dataset into two low-rank matrices, and the product of these two matrices approximates the original matrix [12]. The imputation of missing values is done using this approximate matrix. However, this method has a high computational cost and requires a substantial amount of known data as a basis for imputation. The EM algorithm (Expectation Maximization algorithm) estimates the parameter values of the model based on the available data and then predicts the missing data based on these parameter values [13,14]. These two steps are iteratively repeated until convergence is achieved for data imputation. However, the EM algorithm can have a slow convergence speed when dealing with high-dimensional data. The KNN (k-Nearest Neighbor) algorithm imputes missing data by taking the average of the values from the K nearest neighbors [15]. However, the KNN algorithm relies on distance metrics for prediction, making it sensitive to outliers and noisy data, which can lead to inaccurate imputation results.

The third category of methods is based on deep learning for data imputation. In recent years, due to the powerful nonlinear fitting capabilities of neural networks, deep generative models have been used to impute missing data [16]. Gondara and Wang proposed a multiple imputation model based on deep denoising auto-encoders for data imputation [17]. Nazabal A. Olmos, Ghahramani, and others introduced a general framework for variational autoencoders that effectively fill in incomplete heterogeneous data [18]. Spinelli et al. designed an imputation method for missing data based on graph-denoising autoencoders, where each edge in the graph encodes the similarity between two patterns [19]. Lai et al. proposed an architecture called Tracking Removal Autoencoder (TRAE) that dynamically redesigns the input structure of hidden neurons based on a traditional autoencoder to impute missing values [20]. Pathak, Krahenbuhl, and others introduced a context encoder using pixel-wise reconstruction loss and adversarial loss to generate content for missing regions in images based on the surrounding context [21]. With the continuous development of deep learning techniques, generative adversarial networks have become a recent research hotspot [22], known for their powerful generation capabilities and commonly used in image generation, image denoising [23], image restoration [24], and other scenarios. Recently, generative adversarial networks (GANs) have also been applied to the field of data imputation. Li and Jiang proposed MisGAN to learn and complete images from complex high-dimensional incomplete data [25]. They introduced an auxiliary GAN to learn the distribution of masks to simulate the missing situation. Zhang et al. proposed CPM-GAN, which imputes missing data by exploiting the correlation between different modalities [26]. Wang et al. proposed PC-GAIN, which learns the latent class information among samples to help the generator produce higher-quality imputation results [27]. Yoon et al. introduced the Generative Adversarial Imputation Network (GAIN) [5], which utilizes the adversarial loss and reconstruction loss of non-missing values to optimize the generator and improve the quality of imputations. However, when using the generator of GAIN to impute missing values, the reconstruction loss for missing values cannot be constructed due to the lack of ground truth as supervised information, the reasonableness of the imputed values can only be assessed based on the reconstruction loss of non-missing values and the adversarial loss. Indeed, while these methods have shown some effectiveness in imputing missing data, they still suffer from various limitations in predicting the distribution of the original data due to the lack of sufficient supervised information. As a result, the imputation accuracy of these methods is not very high.

2.2. Granular computing

Granular computing is a computing paradigm that simulates human thinking processes for problem-solving. It achieves a deeper understanding and description of problems by constructing granular structures, allowing for more effective problem-solving with reduced complexity and improved efficiency [28]. In granular computing, the granule is the fundamental concept in granular computing that represents the basic unit for describing a problem. Typically, if we consider the problem as a whole, granule would be a part of that whole. When granules that describe the problem are put together, they form a level of granularity that represents the problem. Different levels of granularity reflect different levels of abstraction in characterizing the problem referred to as granularity. Organizing the levels from coarse to fine granulation forms a multilevel structure [29]. By utilizing this multilevel granular structure, different levels of understanding and description of the problem can be achieved, leading to multilevel problem-solving approaches [30]. Then, give an example of multiple levels of granularity.

Example 2.1. Let's consider a sample dataset $\widetilde{\mathbf{X}}$ related to heart disease, where the feature set F includes blood pressure f_1, age f_2, and blood sugar f_3. Here, we use G^l to denote the l-th level of granularity, and F^l $(1 \leq l \leq 3)$ represents the features used to describe patients in the l-th level of granularity. The sample data described by F^l is denoted as $\widetilde{\mathbf{X}}^l$.

From the perspective of granular computing, if we only use the blood pressure f_1 as the feature to describe patients, then $\mathrm{F}^1 = \{\mathrm{f}_1\}$, and the data described by F^1 is denoted as $\widetilde{\mathbf{X}}^1$. In this case, the level of describing patients is relatively coarse. If we use both blood pressure f_1 and age f_2 as the features to describe patients, then $\mathrm{F}^2 = \{f_1, f_2\}$, and the data described by F^2 is denoted as $\widetilde{\mathbf{X}}^2$. Here,

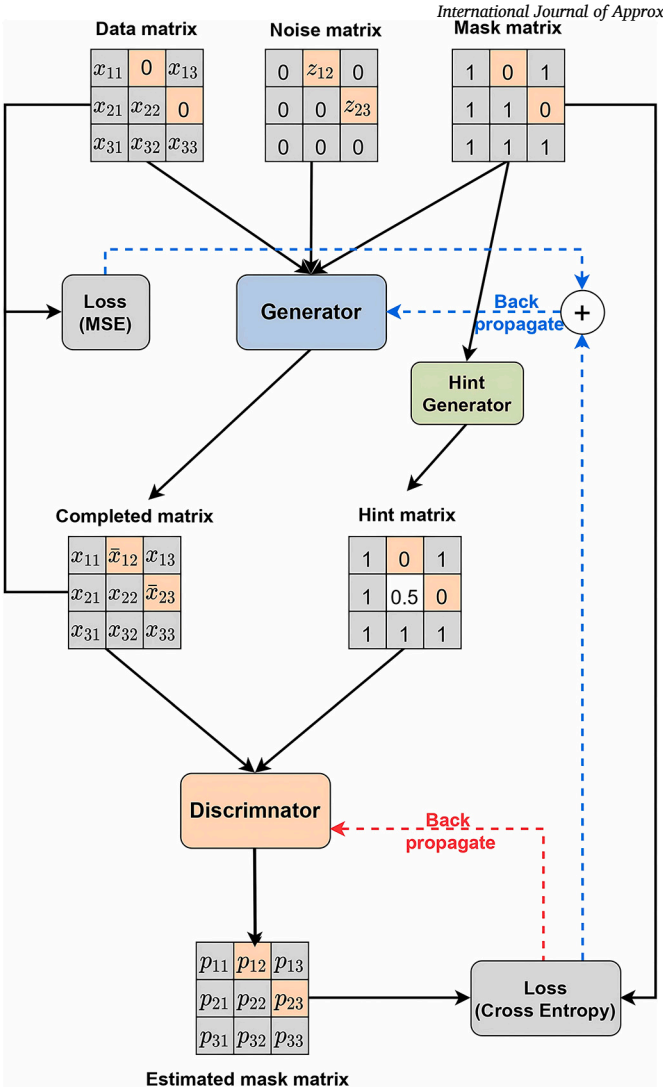

Fig. 2. The architecture of GAIN.

the level of describing patients is slightly finer. When we use blood pressure f_1, age f_2, and blood sugar f_3 as the features to describe patients, then $\mathrm{F}^3 = \{f_1, f_2, f_3\}$, and the data described by F^3 is denoted as $\widetilde{\mathbf{X}}^3$. In this case, this level of describing patients is the finest. The more features and data we use to describe patients, the finer the level of granularity we obtain. Thus, we can have a multilevel structure G consisting of three levels: $\mathrm{G} = \{\mathrm{G}^1, \mathrm{G}^2, \mathrm{G}^3\}$. This allows us to characterize and describe the problem at different levels of granularity, analyze and solve the problem based on different levels, and explore knowledge at different levels.

2.3. Generative adversarial imputation networks

In this subsection, we introduce the traditional generative adversarial imputation networks (GAIN) network. The architecture of the traditional network is shown in Fig. 2. In the imputation process of the model, it can be divided into two parts: the generator G and the discriminator D.

Let $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_N\} \in \mathbb{R}^{N\times d}$ denote the original data set, where i-th data vector $\mathbf{x}_i = \{x_{i1}, x_{i2}, \cdots, x_{id}\} \in \mathbb{R}^{1\times d}$, and x_{ij} represents the j-th feature component of the i-th data vector in $\mathbf{X}$. Given data set $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_N\} \in \mathbb{R}^{N\times d}$, $\mathbf{x}_i = \{x_{i1}, x_{i2}, \cdots, x_{id}\} \in \mathbb{R}^{1\times d}$.

(1) For each $\mathbf{x}_i \in \mathbf{X}$, define a corresponding mask vector $\mathbf{m}_i = \{m_{i1}, m_{i2}, \cdots, m_{id}\} \in \{0,1\}^d$, when x_{ij} is not missing, $m_{ij} = 1$, and when x_{ij} is missing, $m_{ij} = 0$, all mask vectors $\mathbf{m}_i$ together form a mask matrix $\mathbf{M} = \{\mathbf{m}_1, \mathbf{m}_2, \cdots, \mathbf{m}_N\}$.

(2) For each $\mathbf{x}_i \in \mathbf{X}$, define a noise vector $\mathbf{z}_i = \{z_{i1}, z_{i2}, \cdots, z_{id}\} \in \mathbb{R}^{1\times d}$, where each component $z_{ij} \sim p_z(z)$ is independently sampled from the noise prior distribution $p_z(z)$ [22], all noise vectors $\mathbf{z}_i$ form a noise matrix $\mathbf{Z} = \{\mathbf{z}_1, \mathbf{z}_2, \cdots, \mathbf{z}_N\}$.

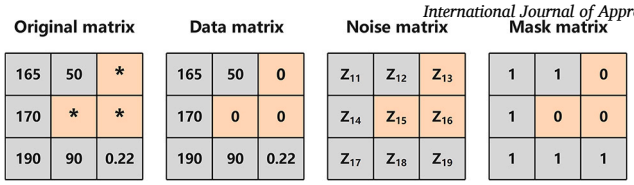

Fig. 3. Example plots of original data, data matrix, noise matrix, and mask matrix.

(3) For each $\mathbf{x}_i \in \mathbf{X}$, define a data vector $\widetilde{\mathbf{x}}_i = \left\{\widetilde{x}_{i1}, \widetilde{x}_{i2}, \cdots, \widetilde{x}_{id}\right\} \in \mathbb{R}^{1\times d}$, the elements within it,

$$\widetilde{x}_{ij} = \begin{cases} = & x_{ij}, \quad if \quad m_{ij} = 1 \\ = & 0, \quad if \quad m_{ij} = 0 \end{cases}$$

Here, the missing value $*$ is replaced with 0. All data vectors together form a data matrix $\widetilde{\mathbf{X}} = \left\{\widetilde{\mathbf{x}}_1, \widetilde{\mathbf{x}}_2 \cdots, \widetilde{\mathbf{x}}_N\right\}$.

The generator of GAIN takes three parts as input: the data matrix $\widetilde{\mathbf{X}}$, the noise matrix $\mathbf{Z}$, and the mask matrix $\mathbf{M}$. The generator G of GAIN generates data at all positions in the matrix, regardless of whether it is in the location of missing value or non-missing value. It learns the data distribution of the input data and then outputs an imputed matrix $\overline{\mathbf{X}}$, the same size as the data matrix $\widetilde{\mathbf{X}}$, with a distribution that approximates the real data, the process is:

$$\overline{\mathbf{X}} = G(\widetilde{\mathbf{X}}, \mathbf{M}, (1-\mathbf{M}) \odot \mathbf{Z}) \tag{1}$$

where $\odot$ represents the element-wise product (Hadamard product).

Fig. 3 presents an example of a dataset comprising three users and three features, encompassing the original data, data matrix, noise matrix, and mask matrix. The generator within the GAIN framework is a fully connected neural network. Its inputs consist of three components: the data matrix $\widetilde{\mathbf{X}}$, the noise matrix $\mathbf{Z}$, and the mask matrix $\mathbf{M}$. The original data $\mathbf{X}$ contains the missing value $*$, and the missing value $*$ contained in $\mathbf{X}$ is replaced with 0 to obtain the data matrix $\widetilde{\mathbf{X}}$. As illustrated in Fig. 3, a value of 0 at position (1,3) in the data matrix signifies a missing value. To generate the missing data, the generator also incorporates a noise matrix $\mathbf{Z}$, composed of multiple random noise vectors, which drives the generation of new data by the generative model. As depicted in Fig. 3, the noise at position (1,3) in the noise matrix is denoted as $\mathbf{Z_{13}}$. The mask matrix $\mathbf{M}$ is crucial for providing the generator G with information regarding the missing values' locations. Its elements can only be 0 or 1, where 0 indicates a missing element in the corresponding position in $\widetilde{\mathbf{X}}$, and 1 indicates a non-missing element. By inputting the mask matrix $\mathbf{M}$, the generator G can identify the missing and non-missing positions. As illustrated in Fig. 3, the value at position (1,3) in the matrix is missing, which is represented by 0 in the mask matrix $\mathbf{M}$.

Based on Equation (1), the data matrix $\widetilde{\mathbf{X}}$ presents the available data. The mask matrix $\mathbf{M}$ indicates which data locations are missing and which are known to the generator. By utilizing a noise matrix, we impute the missing values, with $(1-\mathbf{M}) \odot \mathbf{Z}$. The generator G then leverages these inputs to reconstruct the missing data, ultimately producing the imputed matrix $\overline{\mathbf{X}}$.

The imputed matrix $\overline{\mathbf{X}}$ is a new matrix generated by the generator G to mimic the data distribution of the data matrix $\widetilde{\mathbf{X}}$. For the non-missing values in the data matrix, by concatenating the data matrix $\widetilde{\mathbf{X}}$ with the imputed matrix $\overline{\mathbf{X}}$, replacing the non-missing value in $\widetilde{\mathbf{X}}$ with the corresponding data at the same component in $\overline{\mathbf{X}}$, a newly completed matrix $\hat{\mathbf{X}}$ is formed, which is the final imputation result. The specific process is as follows:

$$\hat{\mathbf{X}} = \mathbf{M} \odot \mathbf{X} + (1-\mathbf{M}) \odot \overline{\mathbf{X}} \tag{2}$$

The loss function of the generator G is defined as:

$$\min_G \frac{1}{N} \sum_{k=1}^{N} \left(L_G\left(\mathbf{m}_k, \hat{\mathbf{m}}_k\right) + \alpha L_R(\widetilde{\mathbf{x}}_k, \bar{\mathbf{x}}_k)\right) \tag{3}$$

$$L_G\left(\mathbf{m}_i, \hat{\mathbf{m}}_i\right) = \sum_{j=1}^{d} \left(-(1-m_{ij}) \log \hat{m}_{ij}\right) \tag{4}$$

$$L_R\left(\widetilde{\mathbf{x}}_i, \bar{\mathbf{x}}_i\right) = \sum_{j=1}^{d} m_{ij} (\widetilde{x}_{ij} - \bar{x}_{ij})^2 \tag{5}$$

where α is a weight parameter, $\hat{\mathbf{m}}_k$ is the estimated mask vector output by the discriminator D. L_G is part of the adversarial loss of GAIN, which plays a role in competing with the discriminator D. By backpropagation, the generator G is optimized to make the discriminator D unable to distinguish between imputed data and non-missing data. L_R is the reconstruction loss of non-missing value. The purpose of L_R is to minimize the distance between the values generated by generator G at the components of non-missing data and the non-missing data, i.e., to require generator G to learn the distribution of non-missing data, which is the distribution of real data.

The input to the GAIN discriminator consists of the hint matrix $\mathbf{H}$ and the completed matrix $\hat{\mathbf{X}}$. Specifically, the hint matrix $\mathbf{H}$ is generated by the hint generator. The mask matrix $\mathbf{M}$ is fed into the hint generator, which produces a hint matrix $\mathbf{H}$ of the same

size as the mask matrix. The hint matrix $\mathbf{H}$ provides the discriminator with information about the positions of missing values in the incomplete data, allowing the discriminator to focus on the data generation quality at those positions. The range of values in the hint matrix is 0, 0.5, and 1. If the element of the hint matrix $\mathbf{H}_{ij} = 0$, it indicates that the element at that position is missing. If $\mathbf{H}_{ij} = 0$, it means the element at that position is not missing. However, if $\mathbf{H}_{ij} = 0.5$, it signifies that no information can be provided to the discriminator at that position. The hint generator provides some information about $\mathbf{M}$ to the discriminator D. When discriminating between missing and non-missing values, GAIN evaluates all components of the completed matrix $\hat{\mathbf{X}}$. In GAIN, the discriminator D outputs a probability for each component in the matrix, forming an estimated mask matrix $\hat{\mathbf{M}}$. The component $\hat{m}_{ij}$ of $\hat{\mathbf{M}}$ represents the probability that D judges the component $\hat{x}_{ij}$ of the completed matrix $\hat{\mathbf{X}}$ as non-missing data. The entire input-output process can be represented as follows:

$$\hat{\mathbf{M}} = D(\hat{\mathbf{X}}, \mathbf{H}) \tag{6}$$

The mask matrix $\mathbf{M}$ is the ground truth of the estimated mask matrix $\hat{\mathbf{M}}$, meaning that the closer the output values of discriminator D are to the values of the mask matrix $\mathbf{M}$, the better the discriminative ability of discriminator D. The definition of the loss function of discriminator D is as follows:

$$\min_D \frac{1}{N} \sum_{k=1}^{N} L_D(\mathbf{m}_k, \hat{\mathbf{m}}_k) \tag{7}$$

$$L_D\left(\mathbf{m}_i, \hat{\mathbf{m}}_i\right) = \sum_{j=1}^{d} \left(-m_{ij} \log \hat{m}_{ij} - (1 - m_{ij}) \log (1 - \hat{m}_{ij})\right) \tag{8}$$

L_D is also part of the adversarial loss, which plays a role in competing with the generator G. It aims to maximize the ability of discriminator D to distinguish between non-missing data and imputed data in the input data.

3. MGAIN

Although GAIN performs well in data imputation, it still has a limitation: GAIN relies solely on adversarial loss and reconstruction loss of non-missing values for imputation, without considering the reconstruction loss of missing values. In the imputation process of GAIN, the components in the data matrix can be divided into two parts based on whether their value is missing. Although GAIN's generator generates data at every component, it mainly focuses on generating value at non-missing components when learning the distribution. This is because when generating value at non-missing components, the generator has observed data as a reference, which is real data, and retains the distribution of real data, implying that these data can be used as supervision information. However, for value at missing components, besides using adversarial theory and letting the discriminator judge whether it is reasonable, GAIN has no other means to ensure the accuracy of imputation. Because the value of missing components is already lost and does not have the condition to use their true values as supervision information, it is impossible to construct reconstruction loss for missing values, which leads to a decrease in the quality of generated data.

From the perspective of granular computing, data has levels, and different levels of data contain different knowledge. If data can be divided into multiple levels of granularity, and efforts are made to maintain the distribution of data within each granularity level unchanged, then models can learn different knowledge at different granularity levels. Moreover, leveraging the relationships between these granularity levels, the imputation values for missing values in coarser granularity levels can serve as supervisory information for imputing missing values in finer granularity levels, thus constructing a reconstruction loss for imputing missing values. Therefore, based on granular computing, we propose a self-supervised multi-level generative adversarial network called MGAIN. MGAIN consists of two parts: the construction of multiple levels and the imputation process. Construction of multiple levels involves dividing the data into multiple levels of granularity. The imputation process, built upon the completion of granularity level dividing, involves sequentially imputing multiple levels of granularity, with the imputation results on the finest granularity level serving as the final result.

The framework of MGAIN is shown in Fig. 4, it includes $\{\mathrm{G}^1, \mathrm{G}^2, \cdots, G^n\}$, representing n levels of granularity. Here, the levels are numbered from coarse to fine. First, on the first level G^1, we perform data imputation using the traditional GAIN network. The generator's loss function is constructed based on the reconstruction loss of non-missing values and the adversarial loss to obtain the completed matrix, denoted as $\hat{\mathbf{X}}^1$. Next, on the second level, the underlying network remains GAIN, but the loss function of the generator is not only based on reconstruction loss of non missing values and adversarial loss, but also includes reconstruction loss of missing values, which is constructed through the completed matrix $\hat{\mathbf{X}}^1$ from the first level. Therefore, in this level, the loss function includes the reconstruction loss of non-missing values, the adversarial loss, and the reconstruction loss for missing values, resulting in completed matrix denoted as $\hat{\mathbf{X}}^2$. Then, on the third level, data imputation is performed using $\hat{\mathbf{X}}^2$ as the supervisory information for missing values at this level. The loss function is constructed accordingly. The imputation process continues from coarse granulation level to fine granulation level, iteratively imputing missing values, and ultimately obtaining the final completed matrix on the finest granulation level, which is the final imputation result. In the following, we will introduce the two components of MGAIN separately: the construction of multiple levels and the imputation process.

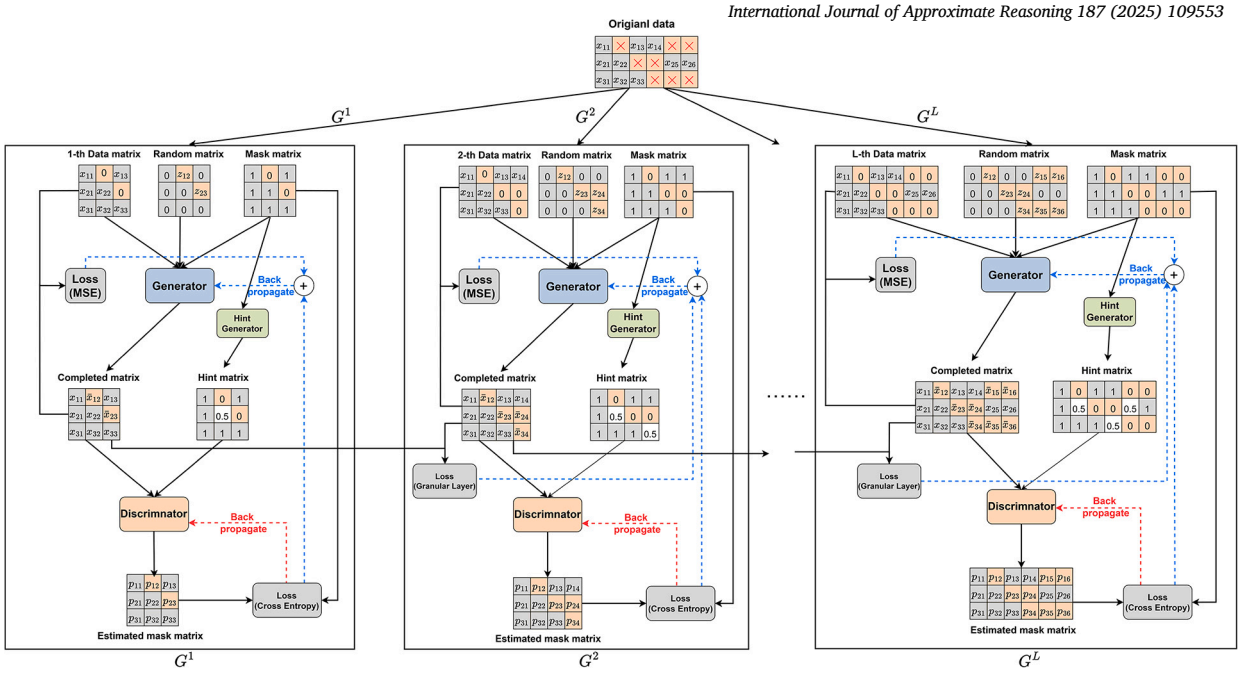

Fig. 4. The architecture of MGAIN.

3.1. Construction of multiple levels

MGAIN, based on a nested sequence of feature sets, divides the data into different levels of granularity. In this subsection, the definition of multi-granularity layers is first provided.

Given a dataset $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_N\} \in \mathbb{R}^{N\times d}$, where $\mathbf{x}_i$ $(1 \le i \le N)$ represents the i-th sample of the dataset, and f_i $(1 \le i \le d)$ represents the i-th feature in the feature set $\mathrm{F} = \{f_1, f_2, \cdots, f_d\}$ of the dataset. By constructing a nested sequence of feature sets $\mathrm{F}^1 \subseteq \mathrm{F}^2 \subseteq \cdots \subseteq \mathrm{F}^L \subseteq \mathrm{F}$, a multi-granularity structure $G = \{\mathrm{F}^1, \mathrm{F}^2, \cdots, \mathrm{F}^L\}$ consisting of L granularity layers can be obtained.

According to the definition of multi-granularity layers, different construction methods can be employed in various applications. In MGAIN, the granularity layers are constructed based on the missing rate. The next step is to introduce the construction methods for multiple granularity layers in MGAIN. Given the original data $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_N\} \in \mathbb{R}^{N\times d}$, we assume F represents the set of all features in the dataset, $\mathrm{F} = \{f_1, f_2, \cdots, f_d\}$, where f_i represents the i-th feature in the dataset, $1 \le i \le d$. In order to make full use of the available information, this paper mainly constructs multiple levels of data based on the missing rate of features. First, let's define the missing rate of a feature.

Given the dataset $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_N\} \in \mathbb{R}^{N\times d}$, $\mathbf{x}_i = \{x_{i1}, x_{i2}, \cdots, x_{id}\} \in \mathbb{R}^{1\times d}$, $\mathbf{m}_i = \{m_{i1}, m_{i2}, \cdots, m_{id}\}$ is the mask vector for $\mathbf{x}_i$, and $\mathrm{F} = \{f_1, f_2, \cdots, f_d\}$ represents the d features of the dataset. For feature f_j, $1 \le j \le d$, the missing rate $r(f_j)$ is defined as follows:

$$r(f_j) = \frac{1}{N}\sum_{i=1}^{N} m_{ij} \tag{9}$$

The features are then sorted in ascending order based on their missing rates, and the sorted features are denoted as $\mathrm{F}' = \{f_1', f_2', \cdots, f_d'\}$. Next, the sorted features are divided into L nested feature subsets: $\mathrm{F}^1, \mathrm{F}^2, \cdots, \mathrm{F}^L$, such that for F^i and F^j, $\mathrm{F}^i \subseteq \mathrm{F}^j$ when $i < j$. It is important to note that when $j = L$, $\mathrm{F}^j = \mathrm{F}$. The number of features included in F^1 is not fixed and can be set. The data representation of the l-th level described by the feature subset F^l is denoted as $\widetilde{\mathbf{X}}^l = \{\widetilde{\mathbf{x}}_1^l, \widetilde{\mathbf{x}}_2^l, \cdots, \widetilde{\mathbf{x}}_N^l\} \in \mathbb{R}^{N\times|\mathrm{F}^l|}$, where $\widetilde{\mathbf{x}}_i^l = \left\{\widetilde{x}_{i1}^l, \widetilde{x}_{i2}^l, \cdots \widetilde{x}_{i|\mathrm{F}^l|}^l\right\} \in \mathbb{R}^{1\times\left|\mathrm{F}^l\right|}$, and $\widetilde{x}_{ij}^l$ represents the j-th feature component of the i-th data vector in $\widetilde{\mathbf{X}}^l$, $\left|\mathrm{F}^l\right|$ represents the number of features included in F^l, $1 \le l \le L$.

During the process of dividing the multi-granularity layers, feature selection follows two important rules: 1) Inclusion of feature sets across hierarchies: Feature sets in finer-grained layers must encompass those in coarser-grained layers. This is essential for utilizing the imputation results from coarser levels as supervisory information for finer levels, necessitating the inclusion of coarser-grained data within finer-grained layers. The original data constitutes the finest granularity level, thus imputation at this level directly addresses the original data. 2) Prioritization in feature selection: When incorporating new features at each granularity level, features with lower missing data rates are prioritized. The missing data rate is defined as in Equation (9). Selecting features with fewer missing values helps preserve more of the original data, thereby maximizing the retention of structural information and facilitating enhanced knowledge acquisition by the model. The nested structure employed between the granularity layers not only facilitates the progressive accumulation of information across layers but also ensures consistency and complementarity among the different granularity levels.

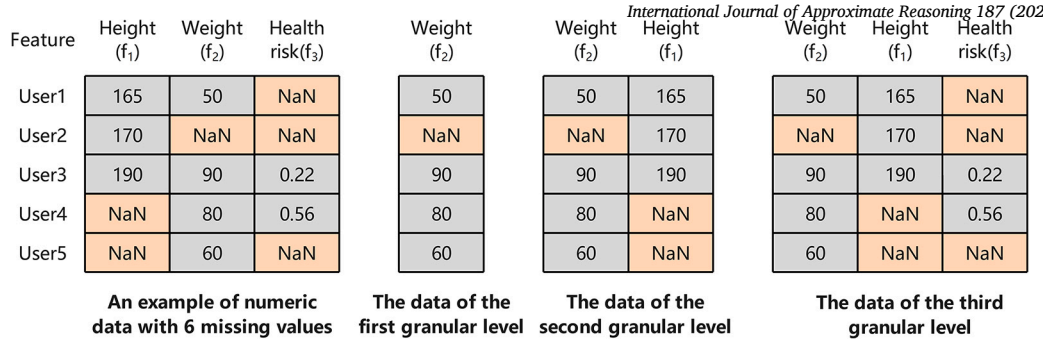

Fig. 5. An example of granular division.

Example 3.1. A numerical data example with six missing values. This dataset comprises three numerical features: height, weight, and health risk. NaN indicates missing data at these positions, as illustrated in Fig. 5. Taking the data in Fig. 5 as an example, we divide it into three granular levels according to the granularity division method proposed in this paper. Calculate the missing data rates for height (f_1), weight (f_2), and health risk (f_3), denoted as $r(f_1)=\frac{2}{5}$, $r(f_2)=\frac{1}{5}$, and $r(f_3)=\frac{3}{5}$, respectively. It is evident that $r(f_2)<r(f_1)<r(f_3)$. Therefore, at the first granular level G^1, we use the weight $\{f_2\}$, which has the lowest data missing rate, to describe the samples in this level, resulting in the first granular data $G^1=\{Weight\}$. The second granular level, G^2, incorporates the feature height (f_1), which has the second-lowest data missing rate, based on G^1. We use $\{f_2,f_1\}$ to describe the samples in this granular level, resulting in the second granular data $G^2=\{Weight, Height\}$. At the third granular level, G^3, the feature health risk (f_3) is added, and we use $\{f_2,f_1,f_3\}$ to describe the samples in this granular level, resulting in the third granular data $G^3=\{Weight, Height, Healthrisk\}$.

3.2. The imputation process of MGAIN

In this subsection, we introduce the imputation process of MGAIN. The processing steps at each level of granularity are similar to GAIN.

Given the data $\widetilde{\mathbf{X}}^l=\{\widetilde{\mathbf{x}}_1^l,\widetilde{\mathbf{x}}_2^l,\cdots,\widetilde{\mathbf{x}}_N^l\}$ at the l-th level of granularity, for each $\widetilde{\mathbf{x}}_i^l\in\widetilde{\mathbf{X}}^l$, there is a corresponding binary mask vector $\mathbf{m}_i^l$ and a noise vector $\mathbf{z}_i^l$. All the mask vectors form a mask matrix $\mathbf{M}^l=\{\mathbf{m}_1^l,\mathbf{m}_2^l,\cdots,\mathbf{m}_N^l\}$, and all the noise vectors form a noise matrix $\mathbf{Z}^l=\{\mathbf{z}_1^l,\mathbf{z}_2^l,\cdots,\mathbf{z}_N^l\}$. The imputed vector $\bar{\mathbf{x}}_i^l$ and the completed vector $\hat{\mathbf{x}}_i^l$ constitute the imputed matrix $\bar{\mathbf{X}}^l$ and the completed matrix $\hat{\mathbf{X}}^l$. Here, the levels are numbered from coarse to fine, starting from 1.

When $l=1$, we input the data $\widetilde{\mathbf{X}}^1$ from the first level and its corresponding mask matrix $\mathbf{M}^1$, noise matrix $\mathbf{Z}^1$ into GAIN, using the traditional GAIN model to obtain the imputed matrix $\bar{\mathbf{X}}^1$ and the completed matrix $\hat{\mathbf{X}}^1$.

$$\overline{\mathbf{X}}^1=G(\widetilde{\mathbf{X}}^1,\mathbf{M}^1,(1-\mathbf{M}^1)\odot\mathbf{Z}^1) \tag{10}$$

$$\hat{\mathbf{X}}^1=\mathbf{M}^1\odot\mathbf{X}^1+(1-\mathbf{M}^1)\odot\overline{\mathbf{X}}^1 \tag{11}$$

When l>1, assuming that the completed matrix obtained at the (l-1)-th level is $\hat{\mathbf{X}}^{l-1}$. By using the completed matrix $\hat{\mathbf{X}}^{l-1}$ as the supervisory information for the missing data matrix $\hat{\mathbf{X}}^l$ of the l-th level. The loss function of the generator G_M not only includes the reconstruction loss of the non-missing values and adversarial loss but also includes the reconstruction loss of the missing values. This is used to construct the reconstruction loss L_M^l for the missing values at the l-th level, given by

$$L_M^l\left(\hat{\mathbf{x}}_i^{l-1},\hat{\mathbf{x}}_i^l\right)=\sum_{j=1}^{\left|\mathrm{F}^{l-1}\right|}(\hat{x}_{ij}^{l-1}-\hat{x}_{ij}^l)^2 \tag{12}$$

The loss function of the generator G_M is defined as:

$$\min_G\frac{1}{N}\sum_{k=1}^{N}(L_G\left(\mathbf{m}_k^l,\hat{\mathbf{m}}_k^l\right)+\alpha L_R\left(\widetilde{\mathbf{x}}_k^l,\bar{\mathbf{x}}_k^l\right)+\beta L_M^l(\hat{\mathbf{x}}_k^{l-1},\hat{\mathbf{x}}_k^l)) \tag{13}$$

where α and β are weight parameters.

Next, the data matrix $\widetilde{\mathbf{X}}^l$ at the l-th level, along with its corresponding binary mask matrix $\mathbf{M}^l$ and noise matrix $\mathbf{Z}^l$, is inputted into the optimized generator G_M, obtaining the imputed matrix $\bar{\mathbf{X}}^l$ and completed matrix $\hat{\mathbf{X}}^l$ at the l-th level.

$$\overline{\mathbf{X}}^l=G_M(\widetilde{\mathbf{X}}^l,\mathbf{M}^l,(1-\mathbf{M}^l)\odot\mathbf{Z}^l) \tag{14}$$

$$\hat{\mathbf{X}}^l=\mathbf{M}^l\odot\widetilde{\mathbf{X}}^l+(1-\mathbf{M}^l)\odot\overline{\mathbf{X}}^l \tag{15}$$

Algorithm 1 Pseudo-Code of MGAIN.

Input: Incomplete dataset $\mathbf{X}$, proportion parameter α, β
Output: Complete dataset $\hat{\mathbf{X}}$

Dividing levels of granularity:
(1) Ranking of features in $\mathbf{X}$ according to missing rates from lowest to highest;
(2) Divide the sorted data into L nested feature subsets;
(3) Divide $\mathbf{X}$ into L levels of granularity $\widetilde{\mathbf{X}}^l$ $(1 \le l \le L)$ according to the feature subsets.
Imputation $\widetilde{\mathbf{X}}^1$ with GAIN to obtain $\hat{\mathbf{X}}^1$
for $l = 2$ **to** L **do**
  Based on $\widetilde{\mathbf{X}}^l$, determine $\mathbf{M}^l$, $\mathbf{Z}^l$ and $\mathbf{H}^l$;
  while train loss has not converged **do**
    Draw k samples from the l-th level $\left\{\widetilde{\mathbf{x}}_j^l\right\}_{j=1}^k$ of $\widetilde{\mathbf{X}}^l$, along with corresponding $\left\{\mathbf{m}_j^l\right\}_{j=1}^k$, $\left\{\mathbf{z}_j^l\right\}_{j=1}^k$, $\left\{\mathbf{h}_j^l\right\}_{j=1}^k$;
    for $j = 1$ **to** k **do**
      $\bar{\mathbf{x}}_j^l \leftarrow G_M\left(\widetilde{\mathbf{x}}_j^l, \mathbf{m}_j^l, \mathbf{z}_j^l\right)$;
      $\hat{\mathbf{x}}_j^l \leftarrow \mathbf{m}_j^l \odot \widetilde{\mathbf{x}}_j^l + \left(\mathbf{1} - \mathbf{m}_j^l\right) \odot \bar{\mathbf{x}}_j^l$;
      $\hat{\mathbf{m}}_j^l \leftarrow D_M\left(\hat{\mathbf{x}}_j^l, \mathbf{h}_j^l\right)$;
    end for
    Update G_M and D_M using stochastic gradient descent:
    $\nabla G_M \leftarrow -\sum_{j=1}^k (L_G\left(\mathbf{m}_j^l, \hat{\mathbf{m}}_j^l\right) + \alpha L_R\left(\widetilde{\mathbf{x}}_j^l, \bar{\mathbf{x}}_j^l\right) + \beta L_M^l(\hat{\mathbf{x}}_j^{l-1}, \hat{\mathbf{x}}_j^l))$;
    $\nabla D_M \leftarrow -\sum_{j=1}^k L_D\left(\mathbf{m}_j^l, \hat{\mathbf{m}}_j^l\right)$.
  end while
end for

When completing the data in the l-th granularity layer, the discriminator D_M, similar to the discriminator D in traditional GAIN, takes the completed input matrix $\hat{\mathbf{X}}^l$ and the hint matrix $\mathbf{H}^l$ as inputs, and outputs the predicted mask matrix $\hat{\mathbf{M}}^l$ to predict which data in $\hat{\mathbf{X}}^l$ has been completed and which data is not missing.

$$\hat{\mathbf{M}}^l = D_M(\hat{\mathbf{X}}^l, \mathbf{H}^l) \tag{16}$$

The loss function of the discriminator D_M is:

$$\min_D \frac{1}{N} \sum_{k=1}^{N} L_D(\mathbf{m}_k^l, \hat{\mathbf{m}}_k^l) \tag{17}$$

The generator and discriminator compete with each other until they reach equilibrium. The imputation result on the finest level is the final result. Based on the self-supervised multi-level generative adversarial network, imputation is performed level by level from coarse to fine granulation levels. By utilizing self-supervision information between multiple levels, more accurate imputation results can be achieved at the finest level.

Building upon Example 3.1, we will now illustrate the core steps of the MGAIN algorithm:

Example 3.2. All data and settings are the same as those in Example 3.1, the first granular layer data $G^1 = \{Weight\}$, the second granular layer data $G^2 = \{Weight, Height\}$, and the third granular layer data $G^3 = \{Weight, Height, Healthrisk\}$. Given the data matrix of the first layer G^1, suppose $\widetilde{\mathbf{X}}^1 = [50\ 0\ 90\ 80\ 60]^T$, where the value is 0, indicates that the value of the position is missing. The element value of the mask matrix can only be 0 or 1; 0 means that the elements in the corresponding position in the data matrix are missing, and 1 means that the elements in the corresponding position in the data matrix are not missing. Therefore, the mask matrix is $\mathbf{M}^1 = [1\ 0\ 1\ 1\ 1]^T$. The noise matrix is independently sampled from the noise prior distribution, which adds random noise at the missing element, assuming the noise matrix $\mathbf{Z}^1 = [0\ z_{21}^1\ 0\ 0\ 0]^T$. The hint matrix is generated by feeding the mask matrix into the hint generator, and it provides the discriminator with some information about the mask matrix. Its range of values is 0, 0.5, and 1. If the element of the hint matrix $\mathbf{H}_{ij} = 0$, it indicates that the element at that position is missing. If $\mathbf{H}_{ij} = 0$, it means the element at that position is not missing. However, if $\mathbf{H}_{ij} = 0.5$, it signifies that no information can be provided to the discriminator at that position. Then, suppose the hint matrix is $\mathbf{H}^1 = [1\ 0\ 1\ 1\ 1]^T$. Based on the observed positions indicated by the mask matrix $\mathbf{M}^1$, the data matrix $\widetilde{\mathbf{X}}^1$ undergoes preprocessing: the observed values at positions where the mask is 1 are retained, while missing positions where the mask is 0 are filled with the corresponding random noise from the noise matrix $\mathbf{Z}^1$. Consequently, the processed matrix becomes $[50\ z_{21}^1\ 90\ 80\ 60]^T$. The generator of the GAIN model then predicts the entire sample, resulting in the imputed matrix $\bar{\mathbf{X}}^1 = [50\ 65\ 90\ 80\ 60]^T$. Next, further processing of the imputed matrix involves replacing the predicted values with the original observed values at positions where the mask is 1, and retaining the predicted values at missing positions where the mask is 0. Finally, the completed matrix for the first layer is $\hat{\mathbf{X}}^1 = [50\ 65\ 90\ 80\ 60]^T$.

In the same way, the height feature is added to obtain the second layer G^2, where $l = 2$. The data matrix for the second granular

layer G^2 is $\widetilde{\mathbf{X}}^2 = \begin{bmatrix} 50 & 165 \\ 0 & 170 \\ 90 & 190 \\ 80 & 0 \\ 60 & 0 \end{bmatrix}$, the mask matrix is $\mathbf{M}^2 = \begin{bmatrix} 1 & 1 \\ 0 & 1 \\ 1 & 1 \\ 1 & 0 \\ 1 & 0 \end{bmatrix}$, the noise matrix is $\mathbf{Z}^2 = \begin{bmatrix} 0 & 0 \\ \mathbf{z}_{21}^2 & 0 \\ 0 & 0 \\ 0 & \mathbf{z}_{42}^2 \\ 0 & \mathbf{z}_{52}^2 \end{bmatrix}$, and the hint matrix is $\mathbf{H}^2 = \begin{bmatrix} 1 & 1 \\ 0 & 1 \\ 1 & 1 \\ 1 & 0 \\ 1 & 0 \end{bmatrix}$.

Based on the observation location information indicated by the mask matrix $\mathbf{M}^2$, the data matrix $\widetilde{\mathbf{X}}^2$ is preprocessed as follows: the original observed values are retained for the positions where the mask value is 1, while the missing positions where the mask value is 0 are filled with the corresponding random noise from the noise matrix $\mathbf{Z}^2$. The processed result serves as the input for the generator G. Given the extraction of five samples, where $k = 5$, and when $j = 1$, the data matrix $\widetilde{\mathbf{x}}_1^2 = [50\ 165]$, the mask matrix $\mathbf{m}_1^2 = [1\ 1]$, and the noise matrix $\mathbf{z}_1^2 = [0\ 0]$, the real values are preserved, and missing values are filled with random noise. After processing, the input to the generator G results in the generator's output being a prediction for the entire sample, yielding the imputed matrix $\bar{\mathbf{x}}_1^2 \leftarrow G_M\left(\widetilde{\mathbf{x}}_1^2, \mathbf{m}_1^2, \mathbf{z}_1^2\right) = [50\ 160]$. We process the imputed matrix $\bar{\mathbf{x}}_1^2$ as follows: for the positions where the mask value is 1, we replace the predicted values with the ground truth values, while for the missing positions where the mask value is 0, we retain the prediction results for the missing values. Therefore, we replace the predicted value 160 with the ground truth value 165, resulting in the completed matrix $\hat{\mathbf{x}}_1^2 \leftarrow \mathbf{m}_1^2 \odot \widetilde{\mathbf{x}}_1^2 + \left(\mathbf{1} - \mathbf{m}_1^2\right) \odot \bar{\mathbf{x}}_1^2 = [50\ 165]$. Then, input the completed matrix and the hint matrix into the discriminator. The elements of the predicted mask matrix $\hat{\mathbf{m}}$ are the probability that the discriminator thinks that the elements in the corresponding position of the completed matrix $\hat{\mathbf{x}}$ are real data. Suppose the predicted mask matrix $\hat{\mathbf{m}}_1^2 \leftarrow D_M\left(\hat{\mathbf{x}}_1^2, \mathbf{h}_1^2\right) = [1\ 1]$ where $\hat{\mathbf{m}}_{ij} = 1$ means that the discriminator thinks the data is true. When $j = 2$, the data matrix $\widetilde{\mathbf{x}}_2^2 = [0\ 170]$, mask matrix $\mathbf{m}_2^2 = [0\ 1]$, and noise matrix $\mathbf{z}_2^2 = [\mathbf{z}_{21}^2\ 0]$. At this point, $\widetilde{\mathbf{x}}_{21}^2$ is missing and is filled with $\mathbf{z}_{21}^2$ from the noise matrix. Subsequently, these inputs are then fed into the generator G to obtain the prediction $\bar{\mathbf{x}}_2^2 \leftarrow G_M\left(\widetilde{\mathbf{x}}_2^2, \mathbf{m}_2^2, \mathbf{z}_2^2\right) = [70\ 170]$ for the second sample. The non-missing data in $\widetilde{\mathbf{x}}_2^2$ is concatenated with the generated data from $\bar{\mathbf{x}}_2^2$ to form $\hat{\mathbf{x}}_2^2 \leftarrow \mathbf{m}_2^2 \odot \widetilde{\mathbf{x}}_2^2 + \left(\mathbf{1} - \mathbf{m}_2^2\right) \odot \bar{\mathbf{x}}_2^2 = [70\ 170]$. The predicted mask matrix is $\hat{\mathbf{m}}_2^2 \leftarrow D_M\left(\hat{\mathbf{x}}_2^2, \mathbf{h}_2^2\right) = [1\ 1]$. This process is iterated to obtain the completed matrix $\hat{\mathbf{X}}^2 = \begin{bmatrix} 50 & 165 \\ 70 & 170 \\ 90 & 190 \\ 80 & 180 \\ 68 & 168 \end{bmatrix}$ for the second layer. Next, update the G_M and D_M for the second layer using stochastic gradient descent: $\nabla G_M \leftarrow -\sum_{j=1}^{k}(L_G\left(\mathbf{m}_j^2, \hat{\mathbf{m}}_j^2\right) + \alpha L_R\left(\widetilde{\mathbf{x}}_j^2, \bar{\mathbf{x}}_j^2\right) + \beta L_M^l(\hat{\mathbf{x}}_j^1, \hat{\mathbf{x}}_j^2))$, $\nabla D_M \leftarrow -\sum_{j=1}^{k} L_D\left(\mathbf{m}_j^2, \hat{\mathbf{m}}_j^2\right)$, where $L_M^l(\hat{\mathbf{x}}_j^1, \hat{\mathbf{x}}_j^2))$ represents the imputation loss for the second granular layer. Here, MGAIN utilizes the completion results from the preceding layer as supervisory information for the current layer, thereby guiding the model to maintain consistency in completion results across multiple granularity levels. Similarly, incorporating healthrisk features yields the third granular layer, ultimately producing the completed matrix $\hat{\mathbf{X}}$. Starting with $\{Weight\}$, then $\{Weight, Height\}$, and finally $\{Weight, Height, Healthrisk\}$, the feature completion is gradually refined to address features with higher missing rates, enabling MGAIN to achieve more precise completion.

The pseudo-code for MGAIN is shown in Algorithm 1. Analyzing the algorithm's complexity and correctness based on the MGAIN pseudocode:

(1) Algorithm Complexity Analysis

Time Complexity:

Step 1: Granular Layer Division Phase, where (1) the missing rate of features is sorted, with a time complexity of $O(mlogm)$, where m is the number of features; (2) and (3), the dataset is divided into L layers according to the feature subset, with a time complexity of $O(m)$. Therefore, the time complexity of the granular layer division phase is $O(mlogm + m)$.

Step 2: Use GAIN to fill in the first layer of data, and the time complexity of the filling operation is $O(n)$, where n is the number of samples.

Steps 3 to 16: Missing Value Filling Phase, the third step loops $L - 1$ times, and the time complexity is $O(L)$. Step 4, infer $\mathbf{M}^l$, $\mathbf{Z}^l$ and $\mathbf{H}^l$ with a time complexity of $O(1)$. In steps 5 to 15, lines 7-11 extract k samples from the l-th layer data each time, and the sampling complexity is $O(k)$. Assuming that the processing complexity of each sample is $O(f)$, the complexity of processing k samples is $O(kf)$. Lines 12-14 use stochastic gradient descent (SGD) to update the parameters of the generator and discriminator, and the time complexity is usually proportional to the number of samples k and the computational complexity f of each sample, which is $O(kf)$. Assuming that the number of training iterations is T in step 5, and the complexity of each round is $O(kf)$, then the total complexity of the filling operation is $O(Tkf)$. Therefore, the time complexity of filling missing values is $O(LTkf)$.

In summary, the total time complexity is:

$$O(mlogm + m + n + LTkf) \tag{18}$$

where n represents the number of samples. m denotes the number of features. L signifies the number of granularity layers. T is the number of iterations per layer. k is the number of samples trained each time. f represents the computational complexity of each sample.

Space Complexity:

The space complexity of MGAIN primarily encompasses data storage, model parameter storage, and the storage of intermediate variables. Data Storage: The storage of the original dataset $\mathbf{X}$, the granularity datasets $\tilde{\mathbf{X}}^l$, and the imputed results $\hat{\mathbf{X}}^l$, results in a

Table 1
The basic properties of the UCI datasets.

Dataset	Samples	Numerical variables	Categorial variables	Number of classes
Breast Cancer	569	30	0	2
Divorce	170	0	54	2
Letter	20000	16	0	26
Lung Cancer	32	0	56	3
News	39797	35	25	2
Wine	178	13	0	3

space complexity of $O(n\frac{m(1+m)}{2})$, i.e. $O(nm^2)$. Model Parameter Storage: Storing the parameters of the MGAIN model, assuming the parameter size of each layer's model is p, leads to a model parameter storage complexity of $O(p)$. Intermediate Variable Storage: This includes the imputation results and gradient information for each sample. Assuming the storage complexity of intermediate variables for each layer is $O(kf)$, the space complexity is $O(Lkf)$.

Consequently, the total space complexity is:

$$O(nm^2 + p + Lkf) \tag{19}$$

(2) Analysis of the correctness of the algorithm

Input Correctness: The input to MGAIN is an incomplete dataset X, along with two parameters, α and β. Loss Function Optimization: By optimizing the reconstruction loss L_R, generator loss L_G, and missing value reconstruction loss L_M, the generator continuously refines its imputation strategy, ensuring that the final imputed dataset $\hat{\mathbf{X}}$ aligns more closely with the true data distribution. Layer-wise Optimization Correctness: The layer-wise imputation approach ensures a gradual improvement of the imputed data, preventing global errors in scenarios with extensive data missingness. Convergence: During each layer's training, the algorithm updates the generator and discriminator parameters by computing gradients until the loss function converges. According to gradient descent theory, when the loss function reaches its minimum, the data generated by the generator G_M should approximate the real data distribution as closely as possible, leading to imputed missing values that also converge towards the true values. Output Correctness: MGAIN outputs the final completed dataset $\hat{\mathbf{X}}$. Because each layer utilizes a generative adversarial network and optimizes the loss function, the imputation gradually approaches the true data distribution. Therefore, the final imputation results are reasonable. Consequently, the MGAIN algorithm is correct and capable of producing reasonable missing value imputation results given the specified input.

4. Experiment

4.1. Dataset and experimental details

To validate the effectiveness of the proposed MGAIN model, we conducted experiments using six datasets obtained from the UCI Machine Learning Repository. The specific information about the datasets is presented in Table 1 above.

4.2. Imputation accuracy in UCI datasets

(1) We compared the RMSE (Root Mean Square Error) of data imputation methods including MGAIN, KNN, EM [13], MissForest [11], MICE [9], GAIN [5], and PC-GAIN [27] after completing the data imputation. A lower RMSE value indicates better imputation performance of the method. In this experiment, we set the missing data proportion to 50% of the total data, and repeated each experiment on every dataset 10 times. We then took the average of these multiple experiments as the final result. This setup aims to ensure robustness and repeatability of the results, reducing the impact of randomness on the outcomes and providing more reliable conclusions to accurately assess the performance of the methods used. The comparison results of the experiment are shown in Table 2. In Table 2, it is shown that MGAIN exhibits the minimum RMSE on various datasets. From the experimental data, MGAIN shows a significant reduction in error on datasets such as Divorce, Breast Cancer, Lung Cancer, and Letter. The improvement is less pronounced on datasets like Wine and News, but overall, it still outperforms other methods. This indicates that compared to other baseline methods, the method proposed in this paper has smaller errors, allowing for more precise imputation of missing data.

(2) MGAIN and PC-GAIN are both models derived from the optimization of GAIN. Under conditions where data missing rates ranged from 20% to 80%, the RMSE of data imputed by MGAIN was compared against GAIN and PC-GAIN across the aforementioned six datasets. Comparisons were made at intervals of 10% missing data rates.

As shown in Fig. 6, the blue line represents MGAIN, and it can be observed that MGAIN demonstrates lower RMSE compared to GAIN and PC-GAIN at different missing rates. Moreover, the improvement becomes more pronounced at higher missing rates, indicating that the proposed method in this study enables more accurate imputation, particularly under conditions of high missing data rates.

Table 2
Imputation performance of MGAIN and comparison methods in terms of RMSE(Average ± Std of RMSE).

Algorithm	Divorce	Letter	Breast Cancer	Wine	News	Lung Cancer
KNN	0.3753 ± 0.035	0.1732 ± 0.0146	0.1476 ± 0.0184	0.2555 ± 0.0184	0.2623 ± 0.0236	0.5390 ± 0.0155
EM	0.3612 ± 0.026	0.2150 ± 0.0102	0.2020 ± 0.0074	0.2926 ± 0.0142	0.2736 ± 0.0341	0.5492 ± 0.0373
MICE	0.3453 ± 0.015	0.1714 ± 0.0225	0.1129 ± 0.0246	0.2197 ± 0.0270	0.2523 ± 0.0357	0.5521 ± 0.0367
MissForest	0.3560 ± 0.036	0.1541 ± 0.0173	0.1185 ± 0.0154	0.2467 ± 0.0143	0.2626 ± 0.0385	0.5784 ± 0.0384
GAIN	0.3309 ± 0.021	0.1554 ± 0.0072	0.0889 ± 0.0045	0.2139 ± 0.0166	0.2361 ± 0.0226	0.5579 ± 0.0278
PC-GAIN	0.3688 ± 0.022	0.1559 ± 0.0045	0.0993 ± 0.0059	0.2182 ± 0.0167	0.2353 ± 0.0237	0.5889 ± 0.0125
MGAIN	**0.2786 ± 0.025**	**0.1430 ± 0.0066**	**0.0844 ± 0.0350**	**0.2071 ± 0.0125**	**0.2297 ± 0.0217**	**0.4482 ± 0.0289**

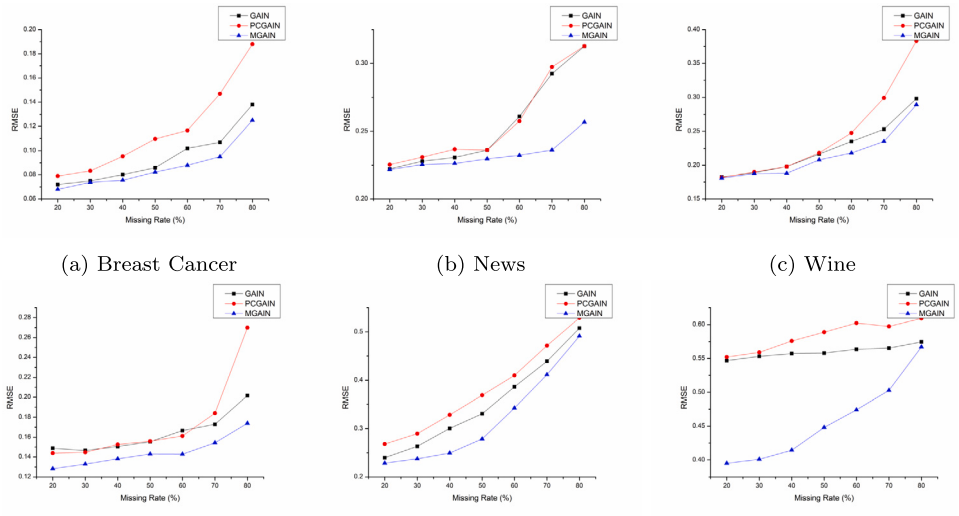

Fig. 6. Comparison of RMSE of MGAIN, GAIN, and PC-GAIN imputed data under different data missing rates. (The black line represents GAIN, the red line represents PC-GAIN, and the blue line represents MGAIN.)

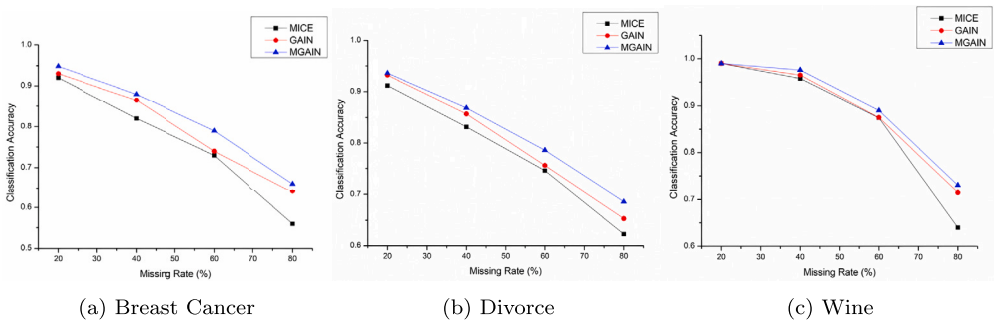

Fig. 7. Prediction performance under various missing rates. (The black line represents MICE, the red line represents GAIN, and the blue line represents MGAIN.)

4.3. Prediction performance under various missing rates

To ensure that category information is preserved during the imputation process for subsequent data exploration and usage, it's important to evaluate the post-imputation prediction accuracy of various methods. Post-imputation prediction accuracy involves imputing missing data, then predicting the data categories and calculating these predictions with the true categories to assess whether category information is maintained throughout the imputation process. This experiment selected the two best-performing methods MICE and GAIN. The research evaluated the post-imputation prediction accuracy using the XGBoost classifier on three datasets: Divorce, Wine, and Breast Cancer, at missing rates of 20%, 40%, 60%, and 80%. The experimental results of this comparison are illustrated in Fig. 7.

As shown in Fig. 7, the blue line represents the post-imputation prediction accuracy of MGAIN. It can be seen that MGAIN's post-imputation prediction accuracy at each missing rate is greater than or equal to that of the other methods, and the superiority of

Table 3
Imputation performance of GAIN and MGAIN with different numbers of levels in terms of RMSE (Average ± Std of RMSE).

Algorithm	Divorce	Letter	Breast Cancer	Wine	News	Lung Cancer
GAIN Cancer	0.3309 ± 0.02	0.1459 ± 0.0072	0.0889 ± 0.0045	0.2139 ± 0.0166	0.2361 ± 0.0026	0.5579 ± 0.0278
3L-MGAIN	0.2786 ± 0.025	0.1430 ± 0.0066	0.0844 ± 0.035	0.2071 ± 0.0125	0.2297 ± 0.0017	0.4482 ± 0.0289
4L-MGAIN	0.2732 ± 0.0021	0.1369 ± 0.0053	0.0797 ± 0.0076	0.2005 ± 0.0074	0.2284 ± 0.0017	0.4389 ± 0.0249
5L-MGAIN	0.2693 ± 0.0025	0.1352 ± 0.0048	0.0787 ± 0.003	0.1973 ± 0.0039	0.228 ± 0.0165	0.4296 ± 0.029

Table 4
Comparison of ablation experimental results RMSE for the Breast Cancer dataset.

L_G	Z	Hint	L_M	RMSE
	✓	✓	✓	0.1036
✓		✓	✓	0.0892
✓	✓		✓	0.0972
✓	✓	✓		0.0889
✓	✓	✓	✓	0.0844

MGAIN becomes more pronounced as the missing rate increases. This also demonstrates that MGAIN is better at preserving category information during the imputation process and achieves the best imputation results compared to other methods.

4.4. Imputation accuracy under various the number of levels

When the level is removed, MGAIN and GAIN show no difference. Therefore, we will increase the number of levels to compare RMSE as an alternative to ablative experiments, verifying the impact of levels on the performance of the MGAIN algorithm. In this experiment, MGAIN with four levels and five levels will be constructed, and their imputation performance will be compared with MGAIN with three levels. The data missing rate is set to 50%.

The experimental results are shown in Table 3. We can see that the performance of MGAIN on three, four, and five levels is all better than that of GAIN. Furthermore, as the number of levels increases, except for the News dataset where MGAIN remains relatively stable, MGAIN shows a significant decrease in RMSE on the other datasets. This demonstrates that adding levels of granularity is necessary and effective.

4.5. Ablation experiments

To evaluate the contribution of each component to the model's performance in the MGAIN method, ablation experiments were conducted. The experiments were conducted on the Breast Cancer dataset with a missing rate set to 50%, and the RMSE results under this condition are provided. The experimental results are shown in Table 4, where "$\surd$" indicates the component's inclusion. The ablation experiments for MGAIN included four components: adversarial loss L_G, Noise matrix Z, Hint matrix, and the reconstruction loss for missing values L_M. To assess the impact of the Noise matrix, we input a zero-noise matrix $Z = 0$ into the generator. As indicated in Table 4, the performance of MGAIN is superior when compared to the scenarios where any of the components are excluded.

4.6. Visualization of experimental results

We present visualizations of the data matrix $\widetilde{\mathbf{X}}$ and the completed matrix $\hat{\mathbf{X}}$ for both GAIN and MGAIN, as illustrated in Fig. 8. In the visualization of the data matrix, black regions denote missing values, while white regions represent observed values. For the completed matrix visualization, we assess performance by comparing the generated values with the original values. Specifically, larger discrepancies are represented by darker colors, whereas smaller discrepancies are indicated by colors closer to white. A comparison of the two completed matrix visualizations reveals that MGAIN generally outperforms GAIN in terms of completion accuracy.

4.7. Assessment of MGAIN and GAIN

To further compare GAIN and MGAIN, we employ three evaluation metrics to assess their respective performances. These three evaluation metrics include: (1) Clustering evaluation metrics: NMI and silhouette coefficient. (2) Regression evaluation metrics: In addition to the RMSE from Section 4.2, the MAE evaluation metric is also added. In the experiment, the missing data is set to 50% of the total data, and 10 experiments are repeated in each dataset.

(1) NMI(Normalized mutual information)

$$NMI(\mathbf{X}, \mathbf{Y}) = \frac{I(\mathbf{X}, \mathbf{Y})}{\sqrt{H(\mathbf{X}), H(\mathbf{Y})}} \tag{20}$$

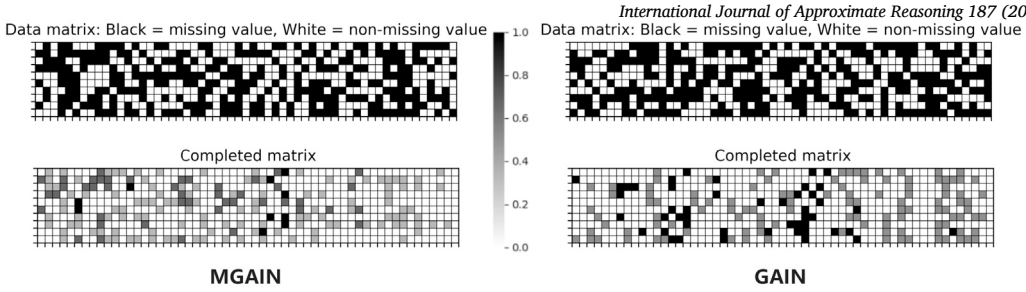

Fig. 8. At a missing rate of 50%, visualize and compare the data matrix and completed matrix of MGAIN and GAIN for the first ten rows of the Lung Cancer dataset.

Table 5
Comparison of NMI of GAIN and MGAIN on different datasets under a missing rate of 50%.

Dataset	Lung Cancer	Letter	Divorce	Breast Cancer	Wine	News
GAIN	0.2192	0.3428	0.4819	0.9947	0.462	0.8643
MGAIN	0.3853	0.3627	0.5054	0.9949	0.4889	0.869

Table 6
Comparison of the Silhouette Scores of GAIN and MGAIN (absolute value of the difference between the Silhouette Scores and the Silhouette Score of the Data matrix) on different datasets with a 50% missing rate.

Dataset	Lung Cancer	Letter	Divorce	Breast Cancer	Wine	News
Data matrix	0.096	0.0.1753	0.6216	0.6970	0.5060	0.4399
GAIN	0.0643(0.0317)	0.1904(0.0151)	0.4384(0.1832)	0.6976(0.0006)	0.4324(0.0736)	0.7144(0.2745)
MGAIN	0.0664(0.0296)	0.179(0.0037)	0.5406(0.081)	0.6871(0.0099)	0.4335(0.0725)	0.8139(0.374)

The Normalized Mutual Information (NMI) [31] serves as a standardized metric for assessing the similarity between two clustering outcomes, specifically the original matrix $\mathbf{X}$ and the completed matrix $\hat{\mathbf{X}}$. Equation (20) presents the NMI expression, and substituting the original matrix $\mathbf{X}$ and the completed matrix $\hat{\mathbf{X}}$ yields Equation (21), where k and l denote the number of clusters (or the number of clusters) in the clustering labels. NMI values consistently range from 0 to 1, reaching its maximum value of 1 only when the two sets of labels exhibit a perfect one-to-one correspondence.

$$NMI(\mathbf{X},\hat{\mathbf{X}}) = \frac{\sum_{i=1}^{k}\sum_{j=1}^{l} p(\mathbf{X}_i,\hat{\mathbf{X}}_j) log(\frac{p(\mathbf{X}_i,\hat{\mathbf{X}}_j)}{p(\mathbf{X}_i)p(\hat{\mathbf{X}}_j)})}{\sqrt{\sum_{i=1}^{k} p(\mathbf{X}_i) log p(\mathbf{X}_i) \sum_{j=1}^{l} p(\hat{\mathbf{X}}_j) log p(\hat{\mathbf{X}}_j)}} \tag{21}$$

As evidenced in Table 5, the NMI values for MGAIN surpass those of GAIN across the majority of datasets, indicating that MGAIN's completion results more closely approximate the original data.

(2) Silhouette Score

The Silhouette Score [32] serves as a metric for evaluating clustering outcomes, providing a measure of the clustering quality for a given sample. Specifically, the Silhouette Score integrates the compactness of a sample within its cluster and its separation from the nearest other cluster. The Equation (22) provides the expression for the Silhouette Score, where a(i) represents the average distance of a sample to all other samples within the same cluster, and b(i) denotes the average distance of a sample to all samples in the nearest other cluster. In Table 6, we present the Silhouette Scores for the original matrix under a 50% missing rate, along with the Silhouette Scores for the final completed matrices generated by GAIN and MGAIN, respectively. This can be interpreted as follows: when the Silhouette Score of the final completed matrix is closer to that of the original matrix, the performance of the generation model (GAIN or MGAIN) in imputation is considered superior, indicating that it preserves the clustering characteristics of the original data. In other words, after imputing the missing data, the clustering structure remains largely intact, and the essential features of the data are effectively retained. Based on the results presented in Table 6, it can be observed that, across most datasets, the Silhouette Score of MGAIN is closer to that of the original matrix compared to GAIN.

$$Silhouette\ Score = \frac{1}{N}\sum_{i=1}^{N} s(i) = \frac{1}{N}\sum_{i=1}^{N} \frac{b(i)-a(i)}{max(a(i),b(i))} \tag{22}$$

(3) MAE (Mean Absolute Error)

Mean Absolute Error (MAE) [33] assesses model accuracy by calculating the average of the absolute differences between predicted and actual values. Let $\mathbf{X}$ represent the original data, and $\hat{\mathbf{X}}$ represent the imputed data. Assuming missing values exist in $\mathbf{X}$ at positions $\mathbf{X}_1,...,\mathbf{X}_m$, and the imputed data at positions $\mathbf{X}_1,...,\mathbf{X}_m$ are imputed to $\hat{\mathbf{X}}_1,...,\hat{\mathbf{X}}_m$, the MAE is defined as follows:

Table 7
Comparison of MAE of GAIN and MGAIN on different datasets under a missing rate of 50%.

Dataset	Lung Cancer	Letter	Divorce	Breast Cancer	Wine	News
GAIN	0.0587	0.0123	0.0335	0.003	0.0143	0.0279
MGAIN	0.0308	0.0076	0.027	0.0016	0.0101	0.0402

$$MAE = \frac{1}{m}\sum_{i=1}^{m}\left\|\mathbf{X}_i - \hat{\mathbf{X}}_i\right\| \tag{23}$$

Table 7 presents the MAE values for MGAIN and GAIN across different datasets. The results indicate that MGAIN generally exhibits lower MAE values compared to GAIN across most datasets. This suggests that the discrepancies between MGAIN's predictions and the actual data are smaller than those of GAIN, indicating superior predictive accuracy for the MGAIN model.

5. Conclusion

This paper focuses on the method of imputing missing values in numerical data. Building upon the work of Yoon et al. with GAIN, we propose a novel generative model called MGAIN for missing data completion. Firstly, multiple levels of granularity are constructed using nested sequences of feature sets. Next, GAIN is employed to impute missing values at the coarse granularity level, and the imputation results for missing values at coarse level are used as supervision for imputing missing values at fine levels, thereby constructing reconstruction loss for missing values. Finally, based on reconstruction loss for missing values, reconstruction loss for non-missing values, and adversarial loss, data at the finer granularity levels is imputed. Experimental results demonstrate that our proposed method achieves more accurate data imputation, particularly in scenarios with higher missing rates, where this superiority becomes more apparent. Overall, MGAIN can be considered an improved version of GAIN, significantly enhancing the performance of the original model.

CRediT authorship contribution statement

Yi Xu: Writing – review & editing, Supervision, Methodology, Conceptualization. **Shujuan Fang:** Writing – review & editing, Writing – original draft. **Xuhui Xing:** Writing – original draft, Software.

Funding

This work was supported by the National Natural Science Foundation of China (No. 62076002, 61402005, 61972001), the Higher Education Natural Science Foundation of Anhui Province, China (No. KJ2013A015), the Natural Science Foundation of Anhui Province, China (No. 2008085MF194, 1308085QF114, 1908085MF188).

Declaration of competing interest

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

Data availability

Data will be made available on request.

References

[1] G.A. da Silva Júnior, A.M. da Silva, A simple and efficient incremental missing data imputation method for evolving neo-fuzzy network, Evolv. Syst. 13 (2) (2022) 201–220, https://doi.org/10.1007/S12530-021-09376-3.
[2] I. Ezzine, L. Benhlima, A study of handling missing data methods for big data, in: 5th IEEE International Congress on Information Science and Technology, CiSt 2018, IEEE, Marrakech, Morocco, 2018, pp. 498–501.
[3] X. Qin, H. Shi, X. Dong, S. Zhang, L. Yuan, Improved generative adversarial imputation networks for missing data, Appl. Intell. (2024) 1–15, https://doi.org/10.1007/s10489-024-05814-2.
[4] D. Adhikari, W. Jiang, J. Zhan, Z. He, D.B. Rawat, U. Aickelin, H.A. Khorshidi, A comprehensive survey on imputation of missing data in Internet of Things, ACM Comput. Surv. 55 (7) (2023) 133:1–133:38, https://doi.org/10.1145/3533381.
[5] J. Yoon, J. Jordon, M. van der Schaar, GAIN: missing data imputation using generative adversarial nets, in: Proceedings of the 35th International Conference on Machine Learning, ICML 2018, in: Proceedings of Machine Learning Research, vol. 80, PMLR, Stockholm, Sweden, 2018, pp. 5675–5684, http://proceedings.mlr.press/v80/yoon18a.html.
[6] Z. Wu, M. Cai, F. Xu, Q. Li, Pcs-granularity weighted ensemble clustering via co-association matrix, Appl. Intell. 54 (5) (2024) 3884–3901, https://doi.org/10.1007/S10489-024-05368-3.
[7] W. Wang, Y. Chai, Y. Li, GAGIN: generative adversarial guider imputation network for missing data, Neural Comput. Appl. 34 (10) (2022) 7597–7610, https://doi.org/10.1007/S00521-021-06862-2.
[8] M. Kantardzic, Data Mining: Concepts, Models, Methods, and Algorithms, John Wiley & Sons, Hoboken, NJ, 2011.

[9] I.R. White, P. Royston, A.M. Wood, Multiple imputation using chained equations: issues and guidance for practice, Stat. Med. 30 (4) (2011) 377–399, https://doi.org/10.1002/sim.4067.
[10] M.E.H. Chowdhury, T. Rahman, A. Khandakar, S. Al-Madeed, S.M. Zughaier, S.A.R. Doi, H. Hassen, M.T. Islam, An early warning tool for predicting mortality risk of COVID-19 patients using machine learning, Cogn. Comput. 16 (4) (2024) 1778–1793, https://doi.org/10.1007/S12559-020-09812-7.
[11] D.J. Stekhoven, P. Bühlmann, Missforest - non-parametric missing value imputation for mixed-type data, Bioinformatics 28 (1) (2012) 112–118, https://doi.org/10.1093/BIOINFORMATICS/BTR597.
[12] E. Acar, D.M. Dunlavy, T.G. Kolda, M. Mørup, Scalable tensor factorizations with missing data, in: Proceedings of the SIAM International Conference on Data Mining, SDM 2010, SIAM, Columbus, Ohio, USA, 2010, pp. 701–712.
[13] P.J. García-Laencina, J. Sancho-Gómez, A.R. Figueiras-Vidal, Pattern classification with missing data: a review, Neural Comput. Appl. 19 (2) (2010) 263–282, https://doi.org/10.1007/S00521-009-0295-6.
[14] X. Li, Y. Yan, J.J. Soraghan, Z. Wang, J. Ren, A music cognition-guided framework for multi-pitch estimation, Cogn. Comput. 15 (1) (2023) 23–35, https://doi.org/10.1007/S12559-022-10031-5.
[15] A.T. Hudak, N.L. Crookston, J.S. Evans, D.E. Hall, M.J. Falkowski, Nearest neighbor imputation of species-level, plot-scale forest structure attributes from lidar data, Remote Sens. Environ. 112 (5) (2008) 2232–2245, https://doi.org/10.1016/j.rse.2007.10.009.
[16] W. Wang, MAGAN: a masked autoencoder generative adversarial network for processing missing IoT sequence data, Pattern Recognit. Lett. 138 (2020) 211–216, https://doi.org/10.1016/J.PATREC.2020.07.025.
[17] L. Gondara, K. Wang, Multiple imputation using deep denoising autoencoders, CoRR, arXiv:1705.02737 [abs], 2017, arXiv:1705.02737, http://arxiv.org/abs/1705.02737.
[18] A. Nazábal, P.M. Olmos, Z. Ghahramani, I. Valera, Handling incomplete heterogeneous data using vaes, Pattern Recognit. 107 (2020) 107501, https://doi.org/10.1016/J.PATCOG.2020.107501.
[19] I. Spinelli, S. Scardapane, A. Uncini, Missing data imputation with adversarially-trained graph convolutional networks, Neural Netw. 129 (2020) 249–260, https://doi.org/10.1016/J.NEUNET.2020.06.005.
[20] X. Lai, X. Wu, L. Zhang, W. Lu, C. Zhong, Imputations of missing values using a tracking-removed autoencoder trained with incomplete data, Neurocomputing 366 (2019) 54–65, https://doi.org/10.1016/J.NEUCOM.2019.07.066.
[21] D. Pathak, P. Krähenbühl, J. Donahue, T. Darrell, A.A. Efros, Context encoders: feature learning by inpainting, in: 2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, IEEE Computer Society, Las Vegas, NV, USA, 2016, pp. 2536–2544.
[22] I.J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A.C. Courville, Y. Bengio, Generative adversarial networks, Commun. ACM 63 (11) (2020) 139–144, https://doi.org/10.1145/3422622.
[23] M. Bertalmío, G. Sapiro, V. Caselles, C. Ballester, Image inpainting, in: Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH 2000, ACM, New Orleans, LA, USA, 2000, pp. 417–424.
[24] M. Elad, M. Aharon, Image denoising via sparse and redundant representations over learned dictionaries, IEEE Trans. Image Process. 15 (12) (2006) 3736–3745, https://doi.org/10.1109/TIP.2006.881969.
[25] S.C.-X. Li, B. Jiang, B. Marlin, Misgan: learning from incomplete data with generative adversarial networks, arXiv preprint, arXiv:1902.09599, 2019.
[26] C. Zhang, Y. Cui, Z. Han, J.T. Zhou, H. Fu, Q. Hu, Deep partial multi-view learning, IEEE Trans. Pattern Anal. Mach. Intell. 44 (5) (2022) 2402–2415, https://doi.org/10.1109/TPAMI.2020.3037734.
[27] Y. Wang, D. Li, X. Li, M. Yang, PC-GAIN: pseudo-label conditional generative adversarial imputation networks for incomplete data, Neural Netw. 141 (2021) 395–403, https://doi.org/10.1016/J.NEUNET.2021.05.033.
[28] J. Yao, A.V. Vasilakos, W. Pedrycz, Granular computing: perspectives and challenges, IEEE Trans. Cybern. 43 (6) (2013) 1977–1989, https://doi.org/10.1109/TSMCC.2012.2236648.
[29] Y. Xu, B. Li, Multiview sequential three-way decisions based on partition order product space, Inf. Sci. 600 (2022) 401–430, https://doi.org/10.1016/J.INS.2022.04.007.
[30] G. Wang, J. Yang, J. Xu, Granular computing: from granularity optimization to multi-granularity joint problem solving, Granular Computing 2 (2017) 105–120, https://doi.org/10.1007/s41066-016-0032-3.
[31] D. Dinh, V. Huynh, S. Sriboonchitta, Clustering mixed numerical and categorical data with missing values, Inf. Sci. 571 (2021) 418–442, https://doi.org/10.1016/J.INS.2021.04.076.
[32] Y. Januzaj, E. Beqiri, A. Luma, Determining the optimal number of clusters using silhouette score as a data mining technique, Int. J. Biomed. Eng. 19 (4) (2023) 174–182, https://doi.org/10.3991/IJOE.V19I04.37059.
[33] N. Karmitsa, S. Taheri, A.M. Bagirov, P. Mäkinen, Missing value imputation via clusterwise linear regression, IEEE Trans. Knowl. Data Eng. 34 (4) (2022) 1889–1901, https://doi.org/10.1109/TKDE.2020.3001694.