Missing Data Imputation in Healthcare Datasets Using GAIN and KNN

Kadriye Nur Erman[1], *Ahmet Cevahir* Çınar[2*]

[1]Department of Computer Engineering, Faculty of Engineering and Natural Sciences, Kirikkale University, Kirikkale, Turkiye
[2]Department of Computer Engineering, Faculty of Technology, Selcuk University, Konya, Turkiye

Abstract. Missing data in health-related datasets poses a significant challenge to data analysis processes and adversely affects the accuracy of decision support systems. In this study, two imputation methods— the statistically grounded K-Nearest Neighbors (KNN) algorithm and a deep learning-based approach known as Generative Adversarial Imputation Nets (GAIN)—are examined and compared. While KNN offers a simple and interpretable non-parametric solution, GAIN employs a complex artificial neural network architecture composed of generator and discriminator networks to estimate missing values. These two methods, differing in structural complexity, were evaluated on 14 publicly available health datasets characterized by small sample sizes and limited dimensionality. The missing rate was set at 10%, and missing values were simulated under the Missing Completely at Random (MCAR) mechanism. Performance was assessed using Mean Absolute Error (MAE), Mean Squared Error (MSE), Root Mean Squared Error (RMSE), and computational time as evaluation metrics. The results indicate that the KNN method yields more consistent and accurate results on datasets with small sample sizes and low dimensionality. On the other hand, GAIN demonstrated promising potential, particularly in handling larger and more complex datasets.

1 Introduction

Missing data represent a prevalent and critical issue, particularly in domains such as healthcare. Health-related data are often incomplete due to factors such as sensor malfunctions, patient non-compliance, or errors during data transmission. However, methods such as machine learning and statistical modeling require complete data to produce reliable and valid outcomes. Missing data compromise model accuracy, lead to biased decisions, and undermine the reliability of decision support systems [1].

A variety of imputation techniques have been proposed in the literature to address this challenge. While simple statistical imputation methods offer the advantage of rapid implementation, they often fail to adequately reflect the underlying data distribution, resulting in low accuracy [2]. Regression-based methods, on the other hand, are limited in their capacity to model only linear relationships and thus may be insufficient for complex

* Corresponding author: accinar@selcuk.edu.tr

healthcare datasets characterized by non-linearity and heterogeneity. Many of these traditional approaches exhibit performance limitations, particularly when applied to high-dimensional and heterogeneous datasets [3].

In this context, machine learning-based imputation methods have garnered increasing attention. The KNN algorithm, despite its simplicity, demonstrates applicability across various data types due to its non-parametric nature. Nevertheless, it is sensitive to outliers and incurs high computational costs when dealing with large-scale datasets. As an alternative, the Random Forest-based MissForest [4] method offers improved modeling of inter-variable relationships and can capture non-linear dependencies more effectively. However, its high computational burden limits its practicality for large and dynamic datasets.

In recent years, the GAIN method, built upon the Generative Adversarial Network (GAN) architecture, has emerged as a prominent approach in the field of missing data imputation. GAIN has demonstrated the ability to more realistically impute missing values, particularly in complex and highly multivariate data structures [5].

In this study, 10% Missing Completely at Random (MCAR) values were artificially introduced into 14 publicly available healthcare datasets, and the imputation performances of the GAIN and KNN algorithms were comparatively evaluated. The heterogeneity of these datasets in terms of sample size, and the diversity of categorical and continuous variables, allowed for an objective assessment of algorithmic performance under various conditions. While the literature also includes other widely used imputation techniques such as Multiple Imputation by Chained Equations (MICE) [6], MissForest, and Expectation-Maximization (EM) [7], this study specifically aims to compare the advanced neural network-based architecture of GAIN with the non-parametric structure of KNN.

Although experimental results generally indicate that GAIN performs well on complex datasets, the KNN algorithm outperformed GAIN in many cases by achieving lower MAE, MSE, and RMSE values. The performance of KNN was particularly notable in datasets such as “Ecoli”, “Cardiotocography” and “Breast Cancer Wisconsin”. These findings suggest that, under certain conditions, traditional methods may still be competitive with more recent and sophisticated approaches.

The remainder of this paper is structured as follows: Section I presents the aim and scope of the study. Section II reviews the related literature. Section III describes the experimental setup, datasets, preprocessing steps, and obtained results. Section IV provides a general discussion and outlines the main conclusions drawn from the study.

2 Related Works

Missing data is categorized into three main types based on the mechanism by which they occur [8-10]. MCAR represents completely random missing data, where the absence of data is unrelated to any observed or unobserved factors, and the missingness occurs randomly. Missing at Random (MAR) refers to the scenario where the missing data is dependent on observed data. In this case, the missing values are related only to observed variables. Missing Not at Random (MNAR) refers to a situation where the missing data is related to the missing values themselves. In this case, the absence of data is directly associated with a specific feature or observation. The missing data mechanism plays a critical role in determining the strategies used in data processing and modeling.

In the literature, approaches to missing data are often classified into various categories by different studies. Some studies [11,12] classify these approaches into two main categories: statistical methods and machine learning-based methods, while others propose [13,14] a threefold classification, including traditional statistical methods, machine learning-based methods, and deep learning-based methods.

Deletion methods can be preferred when the dataset is sufficiently large, the missing data rate is very low (below 5%), and the MCAR mechanism is met. However, these methods may result in the loss of important information. They can reduce the amount of data available for analysis, lowering statistical power, and can lead to biased results. For this reason, appropriate imputation of missing data is generally considered a more preferred approach.

Some of the most used methods for missing data imputation include mean imputation, regression-based imputation, and multiple imputation techniques. Mean imputation is frequently used due to its simplicity; however, it has significant drawbacks, such as reducing the variance in the data distribution and distorting relationships between variables. Regression-based approaches allow for the prediction of the dependent variable based on other available variables but are limited by the assumption of linearity. MICE and similar methods generate multiple possible value sets for missing data, thereby incorporating uncertainty into the model. However, these methods can be computationally expensive when applied to high-dimensional and complex datasets.

Among non-parametric methods, the KNN algorithm is a widely used, simple yet effective approach for missing data imputation. This method makes predictions by utilizing the nearest k neighbors surrounding each data point [15]. In the imputation of missing data, the missing value of each data point is predicted based on the similarities of its surrounding data points. This prediction is determined as a function of the neighbors' characteristics, typically by calculating the average of the neighbors' values.

One of the key advantages of the KNN algorithm is the simplicity of its hyperparameters. The number of neighbors, K, is the most important parameter, as it defines the number of neighbors contributing to the prediction. This parameter can directly affect the model's accuracy and performance. The selection of the number of neighbors is typically optimized using methods such as cross-validation.

KNN is an extremely simple method, easy to apply, and makes the interpretation of the data straightforward. Another significant advantage of KNN is that it does not require complex model structures and does not assume any distributional properties of the data. KNN can work with both continuous and categorical variables, making it applicable to a wide variety of data types. These features make KNN a flexible and widely used method, especially for solving missing data problems. However, in large datasets, the computational cost can increase.

In recent years, deep learning methods have achieved significant success in the field of missing data imputation. Variational Autoencoders (VAEs) [16], a popular unsupervised learning algorithm, have been widely utilized [17]. VAEs combine the strengths of deep learning with the Variational Bayesian framework, allowing for more flexible modeling of complex non-linear relationships while learning data representations. VAEs are probabilistic and generative [18,19] models capable of generating new data samples by learning from the data. Approaches adapting VAEs to datasets with missing data have some notable disadvantages. For instance, they cannot handle MNAR data. They replace missing entries with zero, which causes the model to be unable to distinguish between observed data with a value of zero and missing data. Additionally, they cannot scale with high-dimensional inputs.

As an alternative, another deep learning-based approach that emerged around the same time and has been gaining increasing attention is the GAN. Originally introduced by [5] in 2014, GANs are structured around the competition between two neural networks: a generator and a discriminator. The GAN architecture has demonstrated remarkable success in various domains such as image generation, text synthesis, and data augmentation.

An adaptation of the GAN architecture specifically designed for missing data imputation is the GAIN, developed by [20] in 2018. GAIN aims to fill in missing values using statistically similar samples while leveraging the generator-discriminator framework of GANs. Unlike classical GANs, however, GAIN incorporates two key innovations tailored for the missing data problem: the mask matrix and the hint matrix.

GAIN utilizes a binary mask matrix (M) to indicate which observations in the dataset are missing. This matrix is defined for each observation as shown in Equation (1) and is used to guide the generator in determining which components of the data need to be imputed.

$$M_{ij}=\begin{cases}1, \text{ if the } X_{ij} \text{ observation is present}\\ 0, \text{ if the } X_{ij} \text{ observation is missing}\end{cases} \quad (1)$$

The hint matrix (H), which represents the main difference between GAIN and traditional GANs, allows the discriminator not only to distinguish between imputed and real values but also to incorporate partial information from observed values. This enables the discriminator to more reliably learn the distinction between real and generated data. The hint matrix is constructed based on the mask information and provides the discriminator with partial knowledge regarding the observed/missing status of each value.

In GAIN, the generator (G) imputes the missing values and produces a completed dataset. The discriminator (D), guided by the hint matrix, attempts to determine whether each value is observed or imputed. The total loss function of GAIN consists of both the adversarial loss and the reconstruction loss (measured via Mean Squared Error, MSE).

$$L_G=L_{adv}+a.L_{rec} \quad (2)$$

• L_G : Represents the generator loss, which is the overall loss function for GAIN.

• L_{adv} : Denotes the adversarial loss, which reflects the generator's objective to deceive the discriminator. Since the discriminator attempts to classify each value as real or fake, binary cross-entropy (BCE) loss is used to quantify the difference between the predicted and actual class probabilities.

• L_{rec} : Refers to the reconstruction loss, calculated using MSE to minimize the difference between the observed and imputed values.

• $a.$: A hyperparameter that controls the weight of the reconstruction loss within the total generator loss, balancing the contributions of the adversarial and reconstruction components.

With this architecture, GAIN can learn a more realistic distribution of missing values and can generate statistically consistent imputations.

In this study, the performance of the traditional non-parametric KNN method is compared with that of the deep learning-based GAIN algorithm for missing data imputation. Considering the strengths and limitations of both approaches, the study evaluates which method yields more reliable results, particularly in sensitive domains such as healthcare data.

3 Experimental setup and results

In this study, the performance of the GAIN and KNN methods was evaluated using 14 different real-world datasets obtained from the University of California Irvine (UCI) Machine Learning Repository [21]. The selected datasets possess relatively small sample sizes, ranging from 116 to 2,126 instances, thereby enabling a focused analysis of imputation performance on small-scale datasets. Additionally, these datasets include varying numbers of features, consisting of 1 to 14 categorical variables and 1 to 21 continuous variables. The information regarding the datasets used is presented in Table 1. This experimental setting provides an appropriate framework to assess the effectiveness of deep learning-based methods such as GAIN in low-sample scenarios, and to compare them against traditional techniques such as KNNImputer.

The impact of missing data imputation methods was evaluated based on the experimental procedure summarized in Figure 1.

In the initial phase, rows and columns containing missing values or values marked as "?" were excluded from the analysis. Categorical variables in the datasets were transformed into

numerical format through label encoding for binary classes and one-hot encoding for multi-class variables. All data were normalized to a [0,1] range.

Table 1. Short description of datasets

Datasets	Number of Samples	Number of Categorical Features	Number of Continuous Features
Acute Inflammations Nephritis	120	5	1
Blood Transfusion	748	1	4
Breast Cancer Coimbra	116	1	9
Breast Cancer Wisconsin	683	1	9
Cardiotocography	2126	1	21
Contraceptive Method Choice (CMC)	1473	8	1
Diabetic Retinopathy	1151	4	16
Ecoli	336	1	7
Haberman's Survival	306	1	3
Liver Disorders (BUPA)	345	2	5
Pima Indians Diabetes	768	1	8
South African Heart Disease (SAheart)	462	2	8
Statlog Heart Disease	270	7	7
Thoracic Surgery	470	14	3

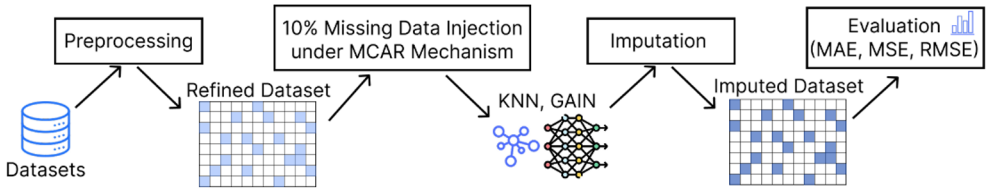

Fig. 1. Overall experimental procedure

Following the preprocessing steps, missing data were imputed under the assumption of continuous variables using the KNN and GAIN algorithms. The missing data scenario applied in this study was based on the MCAR mechanism. For each dataset, missing values corresponding to 10% of the total data were randomly selected and injected. These missing values were then imputed using the GAIN and KNN algorithms.

The hyperparameters employed in the GAIN model were as follows: missing rate = 0.1, batch size = 8, hint rate = 0.9, alpha = 100, and the number of training iterations = 10,000. The GAIN algorithm was obtained from its original implementation on GitHub [22].

The KNN algorithm, one of the widely used imputation methods, was implemented in this study based on the standard approach commonly found in the literature. The algorithm estimates each missing observation based on the values of its k nearest neighbors (k = 5), determined according to a selected distance metric. In this implementation, a tool-independent (from scratch) version of the standard KNN algorithm was developed in Python, allowing for

greater control over the method and eliminating reliance on external libraries. Specifically, the Minkowski distance (p = 3) was used, and each missing value was predicted based on the five nearest neighbors (n_neighbors = 5).

To evaluate the results, the MAE, MSE, RMSE, and the standard deviations of these metrics were calculated. All experimental results are reported in Table 1 as mean ± standard deviation, while Figure 2 provides a visual representation of the RMSE performance. Additionally, the execution time of each method was measured to assess the computational efficiency. To enhance statistical reliability, each experiment was repeated 10 times for each dataset.

Table 2. MAE, MSE and RMSE performance and response times in seconds

Datasets	Methods	MAE	MSE	RMSE	Time
Acute Inflammations Nephritis	GAIN	0.2507 ± 0.0499	0.2276 ± 0.0542	0.4737 ± 0.0564	205.02
	KNN	0.1559 ± 0.0258	0.0737 ± 0.0198	0.2688 ± 0.0384	1.61
Blood Transfusion	GAIN	0.1274 ± 0.0027	0.0660 ± 0.0051	0.2566 ± 0.0100	229.34
	KNN	0.1317 ± 0.0098	0.0547 ± 0.0068	0.2335 ± 0.0144	50.01
Breast Cancer Coimbra	GAIN	0.1398 ± 0.0158	0.0399 ± 0.0088	0.1987 ± 0.0213	203.12
	KNN	0.1322 ± 0.0093	0.0361 ± 0.0043	0.1897 ± 0.0112	2.33
Breast Cancer Wisconsin	GAIN	0.1592 ± 0.0122	0.0619 ± 0.0086	0.2482 ± 0.0175	226.85
	KNN	0.1144 ± 0.0049	0.0368 ± 0.0025	0.1917 ± 0.0065	79.34
Cardiotocography	GAIN	0.0946 ± 0.0028	0.0251 ± 0.0015	0.1583 ± 0.0048	249.68
	KNN	0.0547 ± 0.0017	0.0099 ± 0.0008	0.0993 ± 0.0039	1646.43
Contraceptive Method Choice (CMC)	GAIN	0.2434 ± 0.0268	0.1573 ± 0.0250	0.3954 ± 0.0315	247.26
	KNN	0.2097 ± 0.0033	0.0907 ± 0.0031	0.3012 ± 0.0051	345.01
Diabetic Retinopathy	GAIN	0.0852 ± 0.0064	0.0530 ± 0.0061	0.2299 ± 0.0134	252.27
	KNN	0.0726 ± 0.0019	0.0294 ± 0.0017	0.1714 ± 0.0051	460.49
Ecoli	GAIN	0.2029 ± 0.0226	0.1588 ± 0.0211	0.3975 ± 0.0275	247.46
	KNN	0.0397 ± 0.0043	0.0132 ± 0.0045	0.1131 ± 0.0195	29.49
Haberman's Survival	GAIN	0.2159 ± 0.0125	0.1186 ± 0.0173	0.3433 ± 0.0261	172.03
	KNN	0.2115 ± 0.0088	0.0902 ± 0.0083	0.3001 ± 0.0139	6.78
Liver Disorders (BUPA)	GAIN	0.1570 ± 0.0211	0.0834 ± 0.0208	0.2865 ± 0.0361	277.94
	KNN	0.1399 ± 0.0096	0.0483 ± 0.0068	0.2193 ± 0.0156	14.42
Pima Indians Diabetes	GAIN	0.1367 ± 0.0071	0.0566 ± 0.0058	0.2375 ± 0.0122	251.64
	KNN	0.1338 ± 0.0035	0.0414 ± 0.0023	0.2034 ± 0.0057	82.05
South African Heart Disease (SAheart)	GAIN	0.1853 ± 0.0170	0.1008 ± 0.0117	0.3170 ± 0.0185	310.97
	KNN	0.1753 ± 0.0062	0.0677 ± 0.0033	0.2601 ± 0.0065	36.10
Statlog Heart Disease	GAIN	0.2625 ± 0.0138	0.1918 ± 0.0148	0.4377 ± 0.0170	257.09
	KNN	0.2455 ± 0.0058	0.1213 ± 0.0060	0.3482 ± 0.0086	17.05
Thoracic Surgery	GAIN	0.0697 ± 0.0086	0.0584 ± 0.0083	0.2409 ± 0.0175	317.57
	KNN	0.0763 ± 0.0084	0.0493 ± 0.0077	0.2213 ± 0.0179	111.36

All experiments were conducted on the Google Colaboratory platform using an Intel(R) Xeon(R) CPU @ 2.20GHz processor and 12 GB RAM.

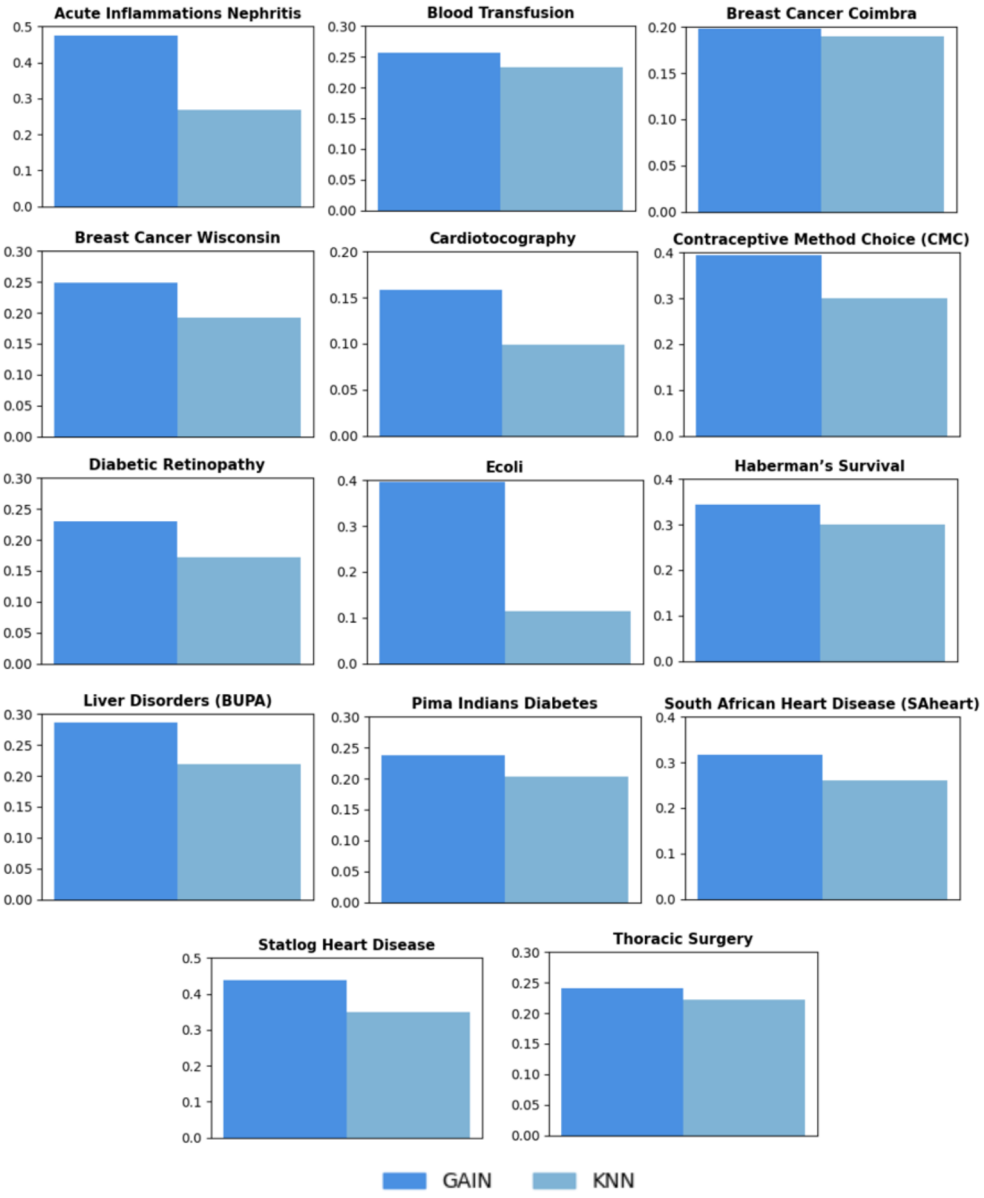

Fig. 2. RMSE performance

4 Conclusion

Due to the relatively small sample sizes of the datasets used in this study, the KNN algorithm demonstrated satisfactory performance on such small-scale data structures. This finding indicates that KNN remains effective, particularly in tabular data structures commonly

encountered in healthcare domains where sample sizes are typically limited. In datasets with fewer samples, KNN can outperform GAIN, suggesting that KNN may offer a more stable performance under scenarios with restricted data availability.

Table 2 presents the comparative performance results of the GAIN and KNNImputer algorithms under a 10% missingness rate generated using the MCAR (Missing Completely at Random) mechanism. Each experiment was repeated ten times, and results are reported as mean ± standard deviation.

Overall, KNNImputer consistently outperformed GAIN across nearly all datasets in terms of reconstruction accuracy. Quantitatively, the average MAE of KNNImputer was approximately 15–20% lower, while MSE and RMSE values were 25–30% lower than those of GAIN. This consistent reduction in error indicates that KNNImputer yields more stable and reliable imputations in small and medium-sized healthcare datasets.

In terms of computational cost, GAIN required between 170 and 320 seconds on average per dataset, whereas KNN completed imputations within under 100 seconds in most cases. This corresponds to a 4–6x speed advantage for KNNImputer, confirming that GAIN's adversarial training structure is significantly more time-consuming and less efficient for tabular healthcare data.

Dataset-specific analysis revealed that KNNImputer's advantage was most pronounced in the Ecoli, Cardiotocography, and Breast Cancer Wisconsin datasets. For instance, in Ecoli, the MAE of GAIN (0.2029) was five times higher than that of KNN (0.0397), while the RMSE difference reached over 0.28. Similarly, in Cardiotocography, KNN achieved an RMSE of 0.0993, outperforming GAIN's 0.1583 by nearly 37%. These results suggest that KNNImputer's distance-based mechanism performs well when the data structure is dense and continuous, efficiently leveraging neighborhood similarity to restore missing values.

Conversely, GAIN achieved relatively competitive or slightly superior performance in the Thoracic Surgery and Haberman's Survival datasets, where the differences in MAE were marginal (GAIN: 0.0697 vs. KNN: 0.0763 and GAIN: 0.2159 vs. KNN: 0.2115, respectively). These datasets are characterized by low dimensionality and moderate sample sizes, which may enable GAIN's generative model to effectively capture global dependencies among attributes. However, even in these cases, the computational overhead of GAIN remains substantially higher, limiting its practical applicability.

Interestingly, both algorithms demonstrated relatively close performance in the Pima Indians Diabetes and Blood Transfusion datasets, where the MAE difference was less than 0.005, indicating that both models can effectively handle moderate-dimensional data with mixed numeric attributes.

Overall, KNNImputer's stability and computational efficiency make it particularly suitable for healthcare datasets with small to medium sample sizes and structured numeric attributes. GAIN, while conceptually powerful, exhibited unstable behavior in several datasets, likely due to sensitivity to hyperparameter selection and limited sample availability, which restricts its ability to accurately learn the data distribution.

The relatively poor performance of GAIN in small-scale datasets can be attributed to its adversarial architecture, which requires a sufficiently large number of samples to learn the underlying data distribution effectively. When the data size is limited, the discriminator and generator networks may fail to converge properly, leading to unstable imputations. In datasets with categorical or highly imbalanced attributes (e.g., Ecoli and Cardiotocography), GAIN struggles to capture discrete feature dependencies, as its neural network structure is primarily optimized for continuous variables. In contrast, KNN directly leverages instance-level similarity, making it more robust in such cases. GAIN is also sensitive to hyperparameter configurations and random initialization. Without dataset-specific tuning, the generator can overfit or underfit, which might explain the higher variance observed in the results. On the other hand, KNNImputer performs well on small datasets because it relies on local

neighborhood information rather than global data distribution, which makes it more stable under limited sample conditions.

This study provides a systematic comparison of GAIN and KNNImputer across multiple healthcare datasets under the MCAR mechanism, highlighting the practical boundaries of deep learning-based imputation in small-sample medical contexts. The findings reveal that while GAIN has strong potential in large-scale, high-dimensional data, its performance may degrade in small or heterogeneous healthcare datasets, where classical approaches like KNNImputer remain more reliable. These results suggest that model selection for missing data imputation in healthcare should consider data size, feature type, and clinical interpretability, rather than relying solely on modern deep learning methods. Future research should focus on (i) developing hybrid imputation frameworks that combine the interpretability of instance-based methods with the representational power of neural networks, (ii) optimizing GAIN architectures for categorical and imbalanced medical data, and (iii) evaluating the downstream impact of imputation on clinical prediction models.

5 Acknowledgements

This study was supported by the Turkish Council of Scientific and Technological Research and Scientific Research Projects Coordinatorship of Selcuk University (Project No: 25701054). We also gratefully acknowledge the project support provided by the Health Institutes of Turkey (TUSEB) under Project No: 44824. The conference presentation is one of the outputs of these projects.

References

1 K. Alnowaiser, IEEE Access **12**, 16783 (2024).
2 S. I. Khan and A. S. M. L. Hoque, Journal of big Data **7** (1), 37 (2020).
3 M. H. Chowdhury, M. K. Islam, and S. I. Khan, in *2017 20th International Conference of Computer and Information Technology (ICCIT)* (IEEE, 2017), pp. 1.
4 D. J. Stekhoven and P. Bühlmann, Bioinformatics **28** (1), 112 (2012).
5 I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, Communications of the ACM **63** (11), 139 (2020).
6 H. Hegde, N. Shimpi, A. Panny, I. Glurich, P. Christie, and A. Acharya, Informatics in Medicine Unlocked **17**, 100275 (2019).
7 J. W. Graham, Annual review of psychology **60** (1), 549 (2009).
8 R. J. Little and D. B. Rubin, *Statistical analysis with missing data*. (John Wiley & Sons, 2019).
9 D. B. Rubin, Biometrika **63** (3), 581 (1976).
10 P. J. García-Laencina, J.-L. Sancho-Gómez, and A. R. Figueiras-Vidal, Neural Computing and Applications **19**, 263 (2010).
11 D. T. Neves, J. Alves, M. G. Naik, A. J. Proença, and F. Prasser, Journal of Computational Science **61**, 101640 (2022).
12 R. Shahbazian and S. Greco, IEEE Access **11**, 88908 (2023).
13 Y. Sun, J. Li, Y. Xu, T. Zhang, and X. Wang, Expert Systems with Applications **227**, 120201 (2023).
14 X. Miao, Y. Wu, L. Chen, Y. Gao, and J. Yin, IEEE Transactions on Knowledge and Data Engineering **35** (7), 6630 (2022).
15 O. Troyanskaya, M. Cantor, G. Sherlock, P. Brown, T. Hastie, R. Tibshirani, D. Botstein, and R. B. Altman, Bioinformatics **17** (6), 520 (2001).
16 D. J. Rezende, S. Mohamed, and D. Wierstra, in *International conference on machine learning* (PMLR, 2014), pp. 1278.

17 M. Tschannen, O. Bachem, and M. Lucic, arXiv preprint arXiv:1812.05069 (2018).
18 R. D. Camino, C. A. Hammerschmidt, and R. State, arXiv preprint arXiv:1902.10666 (2019).
19 J. T. McCoy, S. Kroon, and L. Auret, IFAC-PapersOnLine **51** (21), 141 (2018).
20 J. Yoon, J. Jordon, and M. Schaar, in *International conference on machine learning* (PMLR, 2018), pp. 5689.
21 D. Dua, Graff, C., (UCI Machine Learning Repository, 2017), Vol. 2025.
22 J. J. Jinsung Yoon, Mihaela Schaar, (2021), Vol. 2025.