Self-Supervision Improves Diffusion Models for Tabular Data Imputation

Yixin Liu*
yixin.liu@monash.edu
Monash University
Melbourne, Australia

Thalaiyasingam Ajanthan†
ajthal@amazon.com
Amazon
Canberra, Australia

Hisham Husain†
hushisha@amazon.com
Amazon
Melbourne, Australia

Vu Nguyen†
vutngn@amazon.com
Amazon
Adelaide, Australia

Abstract

Incomplete tabular datasets are ubiquitous in many applications for a number of reasons such as human error in data collection or privacy considerations. One would expect a natural solution for this is to utilize powerful generative models such as diffusion models, which have demonstrated great potential across image and continuous domains. However, vanilla diffusion models often exhibit sensitivity to initialized noises. This, along with the natural sparsity inherent in the tabular domain, poses challenges for diffusion models, thereby impacting the robustness of these models for data imputation. In this work, we propose an advanced diffusion model named **S**elf-supervised **imp**utation **D**iffusion **M**odel (SimpDM for brevity), specifically tailored for tabular data imputation tasks. To mitigate sensitivity to noise, we introduce a self-supervised alignment mechanism that aims to regularize the model, ensuring consistent and stable imputation predictions. Furthermore, we introduce a carefully devised state-dependent data augmentation strategy within SimpDM, enhancing the robustness of the diffusion model when dealing with limited data. Extensive experiments demonstrate that SimpDM matches or outperforms state-of-the-art imputation methods across various scenarios.

CCS Concepts

• **Information systems** → **Incomplete data**; • **Computer systems organization** → **Neural networks**.

Keywords

Tabular Data, Data Imputation, Incomplete Data, Diffusion Model, Self-Supervised Learning

ACM Reference Format:
Yixin Liu, Thalaiyasingam Ajanthan, Hisham Husain, and Vu Nguyen. 2024. Self-Supervision Improves Diffusion Models for Tabular Data Imputation. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management (CIKM '24), October 21–25, 2024, Boise, ID, USA.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3627673.3679829

*This work was completed while the author was an intern at Amazon.
†These authors made equal contributions to this work.

CIKM '24, October 21–25, 2024, Boise, ID, USA
© 2024 Copyright held by the owner/author(s).
ACM ISBN 979-8-4007-0436-9/24/10
https://doi.org/10.1145/3627673.3679829

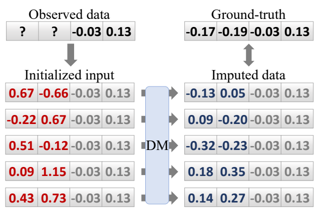
1 Introduction

Tabular data are ubiquitous across industries and disciplines, including but not limited to finance [2], healthcare [9], and environmental sciences [15]. In real-world scenarios, tabular data often contain missing values for many reasons, such as human error, privacy considerations, and the inherent challenges associated with data collection processes [27]. For instance, some characteristics of a patient might not be accurately documented during their visit [1, 13]. The presence of missing data significantly impacts the quality of tabular datasets, introducing bias and rendering a majority of machine learning methods inapplicable.

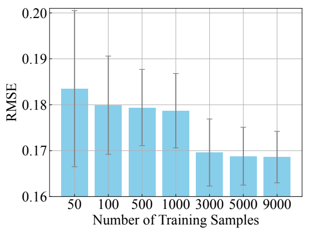
To handle the missing data problem, data imputation is a promising solution that aims to estimate missing values based on the observed data [28]. Existing data imputation methods usually use statistical algorithms [8], shallow machine learning algorithms [40, 41], and deep neural networks [27, 20] to complete missing data. Among them, deep generative model-based imputation models exhibit competitive performance owing to their capability in modeling the data manifold, which helps complete the missing elements [46, 26]. Presently, diffusion models [10, 45], the leading deep generation models, have shown remarkable capabilities in generating data across diverse types, including images [6], audio [17], and time-series data [32]. Building on these achievements, recent studies have used diffusion models for tabular data generation [18, 21, 16], as well as data imputation [48, 14].

In this research, we identify that vanilla diffusion models are suboptimal for tabular data imputation. Specifically, we pinpoint two key mismatches, namely **objective mismatch** and **data scale mismatch**, between tabular data imputation tasks and other scenarios where diffusion models excel. These mismatches potentially contribute to the degradation in data imputation performance of diffusion models.

Firstly, the inherent mismatch in *learning objective* between generation and imputation problems can severely impede the imputation performance of diffusion models. Specifically, *diversity* is a key objective of the generation problem, which requires the generated data to vary significantly while maintaining relevance to the given context. Diffusion models being sensitive to the initial noise (x_T)



(a) With different Gaussian noises (red) as initialization, the diffusion model outputs diverse and inaccurate imputation (blue).



(b) The limited data samples lead to poor imputation results (higher RMSE) and unstable performance (higher variance).

Figure 1: Motivating experiments on UCI Power dataset. (a) Given a data sample, the imputation results by diffusion model with different Gaussian initialization at the first diffusion step. (b) The imputation performance under different numbers of training samples.

at the generation stage helps generate diverse samples – different noise usually leads to different generated samples. Conversely, the objective of the imputation task is *accuracy* rather than diversity, requiring the imputed data to closely resemble the singular ground-truth values. In this case, the sensitivity to the initial noise results in a large variance in imputed results, hurting accuracy. As shown in the motivating example in Fig. 1(a), given identical observation (−0.03 and 0.13), the vanilla diffusion model generates diverse imputed values, leading to substantial gaps between imputed values and the ground-truth.

Moreover, the much smaller *data scale* of tabular data compared to other domains (*e.g.*, image) also hinders diffusion models from comprehending the data manifold, yielding subpar data imputation models. For example, CIFAR-10 is a relatively small image dataset, but it still has 60k samples, supporting the diffusion models to capture data patterns [19]. In contrast, tabular data usually have only a few thousand or even hundreds of samples, making it much more difficult for diffusion models to capture the true data distribution, resulting in overfitting. An example in Fig. 1(b) shows the negative impact of data samples on imputation effectiveness.

To improve the performance of diffusion models on tabular data imputation tasks, we introduce an advanced diffusion model termed **S**elf-supervised **imp**utation **D**iffusion **M**odel (SimpDM for short). By integrating self-supervised learning techniques into tabular diffusion model, we address the aforementioned mismatch issues. Specifically, to tackle objective mismatch, we introduce a self-supervised alignment mechanism to regularize the output of the diffusion model. The key idea is to encourage the diffusion model to provide consistent and accurate imputation for the same observed data, enhancing the stability of imputation results. Furthermore, to handle data scale mismatch, we introduce state-dependent augmentation, a perturbation-based data augmentation strategy that is carefully designed for tabular data imputation tasks. With data augmentation, we extend the training set, improving the robustness of the diffusion model. Meanwhile, an effective augmentation can also ensure the effectiveness of the self-supervised learning framework. To verify the imputation capability of our SimpDM model, we conducted extensive experiments on real-world benchmark datasets across multiple missing data scenarios. The empirical results highlight the strong imputation performance of SimpDM compared to state-of-the-art methods. To sum up, the contributions of this paper are three-fold:

- **Finding.** We identify two key challenges that hinder diffusion models in solving tabular data imputation tasks: target mismatch and data scale mismatch.
- **Solution.** We propose a novel diffusion model termed SimpDM. The proposed method integrates two effective techniques, i.e., self-supervised alignment and state-dependent augmentation, to address the aforementioned challenges.
- **Experiments.** We conduct extensive experiments on 17 benchmark datasets in multiple missing scenarios, and the results illustrate the effectiveness of SimpDM.

2 Related Works

In this section, we briefly review two related research directions, i.e., tabular missing data imputation and diffusion models.

2.1 Tabular Missing Data Imputation

Tabular data imputation is an essential research topic to handle the missing data problem. Early solutions use statistical algorithms to estimate missing entries according to the mean, median, or mode estimation of observed data [8]. Besides statistical methods, shallow machine learning methods such as kNN imputation [40], MICE [41], and missForest [37] are also effective to complete the missing data.

To further exploit the potential of deep learning techniques, recent studies use deep neural networks for data imputation [13]. For instance, Muzellec et al. [27] propose training deep imputation model by minimizing the optimal transport distance between two groups of data, while Kyono et al. [20] use deep models to discover the causal structure underlying data for data imputation. Graph neural networks [23, 24, 50, 22, 49] are also adapted to data imputation since they are capable of modeling inter-sample correlation [47, 39, 51]. Within deep methodologies, generative model-based approaches stand out for their remarkable performance. Specifically, GAIN [46] harnesses GAN while MIWAE [26] utilizes VAE as their backbone, enhancing their imputation capabilities. Several recent studies also attempted to apply diffusion models to tabular data imputation tasks [48, 14, 29]. Nevertheless, these approaches tend to underestimate the inherent disparity between data imputation and diffusion models, resulting in suboptimal imputation performance.

2.2 Diffusion Models

Diffusion models [35] are a generative paradigm that strives to approximate the target distribution through the endpoint of a Markov chain. This chain initiates from a specified parametric distribution, usually a standard Gaussian. Each step of this Markov process is executed by a deep neural network that learns the inversion of the diffusion process. DDPM [10] bridges the gap between diffusion models and score matching approaches [36], showcasing the powerful capability of diffusion models in image generation. So far, diffusion models have made great success in the generation tasks on various domains, such as image [7], text [11], audio [17], time-series [32], and graph [44, 30].

Given the formidable capabilities of diffusion models, recent studies attempt to leverage diffusion models to handle learning tasks on tabular data. For instance, TabDDPM [18] is a representative method that combines Gaussian and Multinomial diffusion models together to generate mixed-type tabular data. CoDi [21] and STaSy [16] also show impressive capability in tabular data synthesis. Inspired by the success of diffusion models in image inpainting [25] and time-series imputation [38], researchers start to discover the potential of diffusion models in tabular data imputation [48, 14, 29]. This paper further delves into this research direction, enhancing diffusion models for imputation from a novel perspective.

3 Preliminary

3.1 Problem Definition

Let $\overline{\mathbf{X}} = \{\overline{\mathbf{x}}_{[i]}\}_{i=1}^{n}$ denotes the feature matrix of a complete tabular dataset, where the i-th row $\overline{\mathbf{x}}_{[i]}$ is a d-dimensional feature vector of the i-th sample. Each feature can be numerical or categorical. The missing data problem can be modeled by a binary mask matrix $\mathbf{M} \in \{0,1\}^{n\times d}$, where $\mathbf{M}_{[i,j]} = 1$ indicates the entry $\overline{\mathbf{X}}_{[i,j]}$ is missing, otherwise $\mathbf{M}_{[i,j]} = 0$. The observed incomplete data matrix can be represented by

$$\mathbf{X} = \mathbf{X}^{(obs)} \odot (\mathbb{1}_{n\times d} - \mathbf{M}) + \varnothing_{n\times d} \odot \mathbf{M}, \tag{1}$$

where $\varnothing_{n\times d}$ is an $n \times d$ matrix of null (missing) values, $\mathbf{X}^{(obs)}$ contains the observed entries that are from $\overline{\mathbf{X}}$, $\odot$ is the element-wise product and $\mathbb{1}_{n\times d}$ is an $n \times d$ matrix containing 1 for every entry. Given $\mathbf{X}$, the goal of **tabular data imputation** is to estimate an imputed data matrix $\hat{\mathbf{X}}$ where the missing entries of $\mathbf{X}$ are filled, which can be written by

$$\hat{\mathbf{X}} = \mathbf{X}^{(obs)} \odot (\mathbb{1}_{n\times d} - \mathbf{M}) + \hat{\mathbf{X}}^{(imp)} \odot \mathbf{M}, \tag{2}$$

where $\hat{\mathbf{X}}^{(imp)}$ contains the imputed entries. The objective is to learn $\hat{\mathbf{X}}$ that is as close as possible to $\overline{\mathbf{X}}$.

3.2 Diffusion Models

Diffusion models [35] are deep generative models derived from a forward and reverse Markov process. The forward process q is to gradually disturb a sample $\mathbf{x}_0$ into a noisy sample $\mathbf{x}_T$, while the reverse process p is to denoise and generate the sample $\hat{\mathbf{x}}_0$ from $\mathbf{x}_T$:

$$\begin{aligned} q\left(\mathbf{x}_{1:T} \mid \mathbf{x}_0\right) &:= \prod_{t=1}^{T} q\left(\mathbf{x}_t \mid \mathbf{x}_{t-1}\right), \\ p_\theta\left(\mathbf{x}_{0:T}\right) &:= p\left(\mathbf{x}_T\right) \prod_{t=1}^{T} p_\theta\left(\mathbf{x}_{t-1} \mid \mathbf{x}_t\right), \end{aligned} \tag{3}$$

where $p_\theta\left(\mathbf{x}_{t-1} \mid \mathbf{x}_t\right)$ is parametrized by a neural network whose parameter is θ. Here θ can be optimized by minimizing the variational upper bound on the negative log-likelihood:

$$\begin{aligned} \mathcal{L}_{\text{vb}} = \mathbb{E}_q[&\underbrace{D_{\text{KL}}\left[q\left(\mathbf{x}_T \mid \mathbf{x}_0\right) || p\left(\mathbf{x}_T\right)\right]}_{L_T} \underbrace{-\log p_\theta\left(\mathbf{x}_0 \mid \mathbf{x}_1\right)}_{L_0} \\ &+ \sum_{t=2}^{T} \underbrace{D_{\text{KL}}\left(q\left(\mathbf{x}_{t-1} \mid \mathbf{x}_t, \mathbf{x}_0\right) || p_\theta\left(\mathbf{x}_{t-1} \mid \mathbf{x}_t\right)\right)}_{L_{t-1}}]. \end{aligned} \tag{4}$$

We can describe the evolution of the distributions with the following forward and reverse process equations

$$d\mathbf{x}_t = \mathbf{x}_t dt + \sqrt{2} d\mathbf{w}, \tag{5}$$

$$d\mathbf{x}_t = \left(\mathbf{x}_t - 2\nabla_x \log p_t\right) + \sqrt{2} d\mathbf{w}, \tag{6}$$

where $\mathbf{w}$ corresponds to Brownian motion and p_t is the law of the random variable $\mathbf{x}_t$. Gaussian diffusion models [10] have forward and reverse processes characterized by Gaussian distributions:

$$\begin{aligned} q\left(\mathbf{x}_t \mid \mathbf{x}_{t-1}\right) &= \mathcal{N}\left(\mathbf{x}_t; \sqrt{1-\beta_t}\mathbf{x}_{t-1}, \beta_t I\right), \\ q\left(\mathbf{x}_T\right) &= \mathcal{N}\left(\mathbf{x}_T; 0, I\right), \\ p_\theta\left(\mathbf{x}_{t-1} \mid \mathbf{x}_t\right) &= \mathcal{N}\left(x_{t-1}; \mu_\theta\left(\mathbf{x}_t, t\right), \Sigma_\theta\left(\mathbf{x}_t, t\right)\right), \end{aligned} \tag{7}$$

where Gaussian noise is added to the sample following a variance schedule $\beta_t \in (0,1)$. Using equivalences between score matching and error matching in [12], [10] further propose a simplified objective function for model optimization:

$$\mathcal{L}_{\text{ddpm}} = \mathbb{E}_{t,\mathbf{x}_0,\boldsymbol{\epsilon}}\left[\|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta\left(\mathbf{x}_t, t\right)\|^2\right], \tag{8}$$

where $\boldsymbol{\epsilon}_\theta$ is a neural network aiming to predict the Gaussian noise $\boldsymbol{\epsilon}$, which can be used to generate $\hat{\mathbf{x}}_0$ during inference.

To deal with categorical data, multinomial diffusion models [11] define a categorical distribution that perturbs the data with noise over K classes:

$$\begin{aligned} q\left(\mathbf{x}_t \mid \mathbf{x}_{t-1}\right) &= \text{Cat}\left(\mathbf{x}_t; \left(1-\beta_t\right)\mathbf{x}_{t-1} + \beta_t/K\right), \\ p_\theta\left(\mathbf{x}_{t-1} \mid \mathbf{x}_t\right) &= \sum_{\hat{\mathbf{x}}_0=1}^{K} q\left(\mathbf{x}_{t-1} \mid \mathbf{x}_t, \hat{\mathbf{x}}_0\right) p_\theta\left(\hat{\mathbf{x}}_0 \mid \mathbf{x}_t\right), \end{aligned} \tag{9}$$

where $\text{Cat}(\cdot)$ is a categorical distribution.

4 Methodology

In this section, we present SimpDM, a self-supervision improved diffusion model for tabular data imputation. The overall training pipeline of SimpDM is demonstrated in Fig. 2. In Sec. 4.1, we first introduce the base diffusion model specially constructed for tabular data imputation. We then extend the base model to SimpDM by introducing two pivot designs, *i.e.*, state-dependent self-supervised alignment (Sec. 4.2) and data augmentation (Sec. 4.3). Finally, we discuss how to apply SimpDM to mixed-type tabular data in Sec. 4.4.

4.1 Diffusion Model for Data Imputation

We first introduce a Gaussian diffusion model for imputing tabular data with numerical features. Note that the vanilla diffusion models [10] are mainly designed for data generation, making it difficult to apply them directly to imputation tasks [25]. Therefore, we implement a series of modifications to adapt the diffusion model effectively for tabular data imputation.

Hybrid input data. A major characteristic of the data imputation problem is that the data is partly accessible. In other words, we have the observed entries to guide the estimation of the missing ones. To involve the observed entries in model input, we use a "hybrid input data" design in SimpDM. More precisely, the model input consists of both the ground-truth values from observed entries and the initialized/estimated values for the missing entries, combined together. Such a design, similar to RePaint [25] for image inpainting, allows the diffusion model to effectively leverage known information to

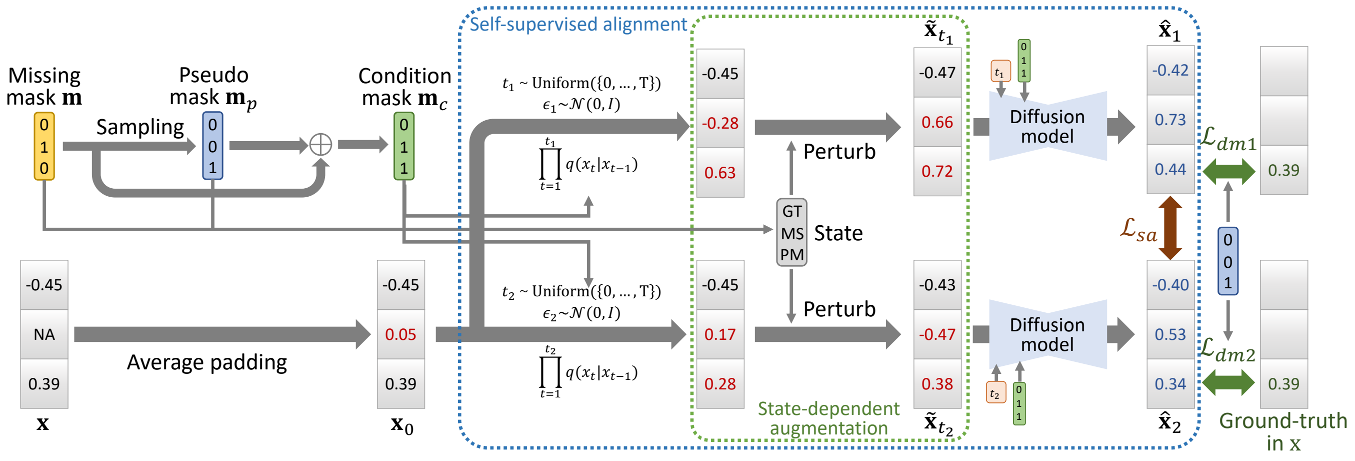

Missing mask $\mathbf{m}$
Sampling
Pseudo mask $\mathbf{m}_p$
Condition mask $\mathbf{m}_c$
Self-supervised alignment
$t_1 \sim \text{Uniform}(\{0,\dots,T\})$ $\epsilon_1 \sim \mathcal{N}(0, I)$
$\prod_{t=1}^{t_1} q(x_t|x_{t-1})$
$t_2 \sim \text{Uniform}(\{0,\dots,T\})$ $\epsilon_2 \sim \mathcal{N}(0, I)$
$\prod_{t=1}^{t_2} q(x_t|x_{t-1})$
Perturb
State
GT MS PM
State-dependent augmentation
Diffusion model
$\tilde{\mathbf{x}}_{t_1}$
$\tilde{\mathbf{x}}_{t_2}$
$\hat{\mathbf{x}}_1$
$\hat{\mathbf{x}}_2$
$\mathcal{L}_{dm1}$
$\mathcal{L}_{dm2}$
$\mathcal{L}_{sa}$
Average padding
$\mathbf{x}$
$\mathbf{x}_0$
Ground-truth in x

Figure 2: The overall pipeline of the training procedure of SimpDM. Given a training sample x and its missing mask m, the first step is to apply average padding for the missing entries and sample the pseudo mask $\mathbf{m}_p$ and condition mask $\mathbf{m}_c$. In self-supervised alignment, we sample different t and ϵ at two channels, and then run the diffusion model at each channel. Apart from the diffusion model loss $\mathcal{L}_{dm}$, we use a self-supervised alignment loss $\mathcal{L}_{sa}$ to minimize the distance of the predictions at two channels. We further use a state-dependent augmentation strategy to perturb the model's input according to the states (GT, MS, or PM) of each entry.

infer missing data while preserving consistency across various samples in the input space.

MLP-based diffusion model. As tabular data is usually low-dimensional and with few data samples, complex network model architectures are prone to overfitting and rather unnecessary. Since previous study [18] shows that a shallow MLP-based diffusion model is adequate for tabular data generation, in SimpDM, we use a plain MLP as our model, without U-net architecture and cross-layer shortcut. The simple model architecture not only substantially reduces the computational cost of our method but also guards against overfitting the limited data. The learnable embedding of diffusion step t is added to the latent representation at the first layer. Notably, we further introduce the *missing mask as an extra input condition* of the diffusion model. In particular, we use a learnable projection layer to map the missing mask vector to a mask embedding, and then add the mask embedding into the first-layer latent representation. The mask embedding enables the diffusion model to discern the status of the hybrid input, aiding in making accurate imputations.

Pseudo missing training strategy. In the setting of data imputation problem, a key challenge is that the ground-truth values of missing entries are unknown during training procedure [20]. That is to say, we cannot acquire supervision signals from these entries for model training. To address this challenge, we develop a "pseudo missing" training strategy inspired by CSDI [38]. Firstly, during the training phase, we pad the values of missing entries to their respective column-wise average values, which can be regarded as an estimation via mean imputation. Then, in each training iteration, we sample a pseudo missing mask to remask a partition of observed entries. Subsequently, we merge the original missing mask and pseudo mask to create the condition mask of the current training iteration. For the entries where the condition mask equals 1, we add Gaussian noise ϵ to them based on the sampled diffusion time step t. As for the remaining entries, which act as "observed data" in the current iteration, we maintain their original values. In this case, we can train the model by denoising the pseudo missing entries since their ground-truth values are accessible.

Value prediction and loss calculation. In vanilla Gaussian diffusion models [10], predicting the Gaussian noises of diffusion (Eq. (8)) has been shown to be effective for data generation. However, in the context of tabular data imputation, our empirical findings suggest that predicting the missing values can result in improved imputation performance compared to predicting noises. Hence, we directly let the model predict the ground-truth values of the missing entries. When computing the loss function, we solely calculate the loss on the pseudo missing terms, directing the model's attention toward completing the missing data.

To sum up, the training process of Gaussian diffusion model for tabular data imputation can be expressed by:

$$\mathcal{L}_{\text{dm}} = \mathbb{E}_{t,\mathbf{x}_0,\epsilon,\mathbf{m}_p}\left[\left\|\left(\mathbf{x}_0 - \mathbf{x}_\theta\left(\tilde{\mathbf{x}}_t, t, \mathbf{m}_c\right)\right) \odot \mathbf{m}_p\right\|^2\right], \tag{10}$$

$$\tilde{\mathbf{x}}_t = \mathbf{x}_0 \odot (\mathbb{1}_d - \mathbf{m}_c) + \mathbf{x}_t \odot \mathbf{m}_c, \tag{11}$$

where $\mathbf{m}_p$ and $\mathbf{m}_c = \mathbf{m} + \mathbf{m}_p$ are the pseudo mask and condition mask respectively, $\tilde{\mathbf{x}}_t$ is the hybrid model input, and $\mathbf{x}_\theta$ is the MLP-based model. During inference time, we can initialize the missing entries by $q(\mathbf{x}_T)$ and set condition mask $\mathbf{m}_c = \mathbf{m}$ directly. Before each inference denoising step, we execute Eq. (11) to ensure the observed data is embedded in the model input.

4.2 Self-Supervised Alignment

With the diffusion model introduced in Sec. 4.1 (we denote it as "base model" in the rest of this paper), we can complete missing entries with recurrent denoising processes. Nevertheless, the base model struggles to achieve optimal imputation performance due to the inherent target mismatch between imputation and generation tasks. Specifically, to achieve the diversity target in generation tasks, diffusion models are sensitive to the initial noise x_T. With varying initial noises, the generated data should exhibit diversity. However, this sensitivity contrasts with the goals of data imputation tasks, which require precise prediction of missing values rather than diversity. To further improve the imputation performance, we design a self-supervised alignment mechanism to suppress this sensitivity.

Concretely, in the training procedure of SimpDM, we construct two parallel channels to run the diffusion model for each sample. The two channels share the same pseudo mask and condition mask. For each channel, we sample its diffusion step (denoted by t_1 and t_2) and diffusion noise (denoted by ϵ_1 and ϵ_2), respectively. With different t and ϵ, we can execute the forward diffusion process to generate x_{t_1} and x_{t_2}, and finally acquire the corresponding hybrid inputs $\tilde{x}_{t_1}$ and $\tilde{x}_{t_2}$ via Eq. (11). Since they have a shared condition mask, the observed data in $\tilde{x}_{t_1}$ and $\tilde{x}_{t_2}$ are identical. As the example shown in Fig. 2, both of the two channels have an observed value -0.45. Differently, due to the disparity between x_{t_1} and x_{t_2}, the pseudo missing and real missing entries in $\tilde{x}_{t_1}$ and $\tilde{x}_{t_2}$ exhibit notable differences.

Recalling that our goal is to suppress the sensitivity to diverse noisy inputs. In particular, given two input data with the same observed entries (i.e., $\tilde{x}_{t_1}$ and $\tilde{x}_{t_2}$), the outputs of the diffusion model should be close to each other. Motivated by this, we acquire their corresponding outputs $\hat{\mathbf{x}}_1 = \mathbf{x}_\theta\left(\tilde{\mathbf{x}}_{t_1}, t_1, \mathbf{m}_c\right)$ and $\hat{\mathbf{x}}_2 = \mathbf{x}_\theta\left(\tilde{\mathbf{x}}_{t_2}, t_2, \mathbf{m}_c\right)$, and try to minimize the difference between them with a self-supervised alignment loss $\mathcal{L}_{sa}(\hat{\mathbf{x}}_1, \hat{\mathbf{x}}_2)$. At the same time, we calculate the basic loss of the diffusion model (Eq. (10)) at two channels, writing the final loss function of SimpDM as:

$$\mathcal{L} = \mathcal{L}_{dm1}(\hat{\mathbf{x}}_1) + \mathcal{L}_{dm2}(\hat{\mathbf{x}}_2) + \gamma \mathcal{L}_{sa}(\hat{\mathbf{x}}_1, \hat{\mathbf{x}}_2), \tag{12}$$

where γ is a tunable trade-off hyper-parameter for $\mathcal{L}_{sa}$. In practice, we have several options for $\mathcal{L}_{sa}$, such as MSE loss, contrastive loss [5], and Sinkhorn divergence [27]. Considering its empirical performance (see Sec. 5.4) and high efficiency, we employ MSE loss for self-supervised alignment in SimpDM.

4.3 State-Dependent Data Augmentation

Another significant challenge in tabular data imputation is the limited size of the dataset, which might not offer adequate information for diffusion models to learn the data manifold effectively. Consequently, the model can lean towards overfitting due to data scarcity, diminishing the overall robustness of diffusion models. To address this problem, data augmentation emerges as a promising solution, creating extra synthetic samples from the original ones [34, 19]. Despite various data augmentation methods designed for image data, most of them cannot be applied to tabular data, which motivates us to produce a well-crafted augmentation strategy for our target scenario.

To augment tabular data, data perturbation with random noises (*e.g.*, uniform, Gaussian, or others) can be a simple yet effective strategy [33, 4, 43]. Nevertheless, this simple strategy may not fully adapt to the diffusion model for SimpDM where input entries are missing or already noisy. Specifically, weak perturbations might minimally affect the missing or uncertain input entries since they are already noisy. However, increasing the perturbation strength significantly shifts the augmented data away from the original data distribution.

On the basis of Gaussian noise-based data perturbation, we propose a state-dependent data augmentation. The core idea is to perturb entries in different states with different strengths (i.e., Gaussian variance). Concretely, given an input vector of SimpDM, each entry can be in three "states": ground-truth state (GT) where $\mathbf{m}_c = 0$, pseudo missing state (PM) where $\mathbf{m}_p = 0$, and missing state (MS) where $\mathbf{m} = 1$. For entries in different states, their certainties can be quite different: for a GT entry, the data is fully reliable; for a PM entry, its certainty is moderate, since this entry is generated by adding ground-truth data with t-related noise; for a MS entry, it has the lowest certainty because we know nothing about the truth value of this entry. In our state-dependent data augmentation strategy, we propose to define the perturbation strength p according to states. Concretely, for entries with lower certainty, we will allocate a higher perturbation strength to it. Formally, in training phase, Eq. (11) can be rewritten by:

$$\begin{aligned} \tilde{\mathbf{x}}_t &= \mathbf{x}_0 \odot (\mathbb{1}_d - \mathbf{m}_c) + \mathbf{x}_t \odot \mathbf{m}_c + \boldsymbol{\xi} \odot \mathbf{p}, \\ \mathbf{p} &= (\mathbb{1}_d - \mathbf{m}) \times p_{GT} + \mathbf{m}_p \times p_{PM} + \mathbf{m} \times p_{MS}, \end{aligned} \tag{13}$$

where $\boldsymbol{\xi}$ is a zero-centered perturbation noise vector, $\mathbf{p}$ is the perturbation strength vector, p_{GT}, p_{PM}, and p_{MS} are the perturbation strengths for the corresponding states that satisfy $p_{GT} < p_{PM} < p_{MS}$. With state-dependent augmentation, we can generate more reliable training samples for model training without disrupting the original data distribution. As a result, the model robustness can be boosted; Also, data augmentation can further facilitate self-supervised learning by amplifying the diversity within the data across two channels. The overall algorithm and complexity analysis of SimpDM is demonstrated in Appendix A and B, respectively.

4.4 Extending to Mixed-Type Data

Previous subsections introduce the Gaussian diffusion version of SimpDM for pure numerical tabular data. In practice, we can easily extend SimpDM to mixed-type data via the following simple modifications for categorical features.

At the diffusion model stage, we introduce multinomial diffusion process for categorical features, while keeping Gaussian diffusion process for numerical features. Similar to TabDDPM [18], each categorical feature can be assigned a K-dimensional prediction head (where K is the number of categories), followed by a Softmax function. The cross-entropy loss is used to optimize the categorical output. For self-supervised alignment, we minimize the output scores of each prediction head. For state-dependent data augmentation, we use random state transformation [11] as the perturbation for categorical data.

5 Experiments

5.1 Experimental Settings

5.1.1 Dataset. We evaluate the imputation performance on 17 real-world datasets across various domains from the UCI Machine Learning repository [3] and Kaggle, including Iris, Yacht, Housing, Diabetes, Blood, Energy, German, Concrete, Yeast, Airfoil, Wine-red, Abalone, Wine-white, Phoneme, Power, Ecommerce, and California. We use the first two letters of each dataset name to indicate it

Table 1: Imputation performance comparison in terms of RMSE. The best and runner-up performances are highlighted by bold and underline, respectively. "OOM" indicates Out-Of-Memory on a 16GB GPU.

Method	Iris	Yacht	Housing	Diabetes	Blood	Energy	German	Concrete	Yeast
mean	.2634±.0054	.2970±.0072	.2471±.0140	.4595±.0134	.1654±.0036	.3535±.0019	.3163±.0185	.2255±.0017	.1185±.0009
KNN	.1428±.0057	.2630±.0057	.1352±.0123	.3339±.0084	.1287±.0033	.2408±.0022	.2968±.0217	.1746±.0032	.1164±.0036
MF	.1428±.0139	.2472±.0086	.1458±.0150	.4579±.0134	.1201±.0047	.2291±.0039	.3106±.0261	.1776±.0032	.1122±.0011
MICE	.1454±.0142	.2805±.0080	.1852±.0162	.4470±.0144	.1337±.0045	.2606±.0024	.3374±.0207	.2004±.0018	.1286±.0003
OT	.1393±.0108	.2742±.0051	.1657±.0174	.3255±.0096	.1442±.0033	.2466±.0025	.3078±.0197	.1747±.0043	.1164±.0008
MIRACLE	.1364±.0183	.2560±.0084	.1633±.0161	.3573±.0488	.1157±.0028	.2294±.0023	.3027±.0214	.1719±.0028	.1118±.0013
GRAPE	.1343±.0206	.2450±.0104	.1382±.0172	.3187±.0097	.1194±.0067	.2365±.0071	.2981±.0191	.1380±.0064	.1144±.0012
IGRM	.1197±.0123	.2391±.0077	.1363±.0175	.3235±.0127	.1182±.0047	**.1863±.0049**	.3032±.0206	**.1240±.0038**	.1161±.0013
MIWAE	.1323±.0160	.2669±.0027	.1572±.0111	.3750±.0223	.1280±.0047	.2528±.0012	.3515±.0185	.1894±.0068	.1236±.0012
GAIN	.1353±.0131	.2526±.0082	.1601±.0167	.3942±.0166	.1537±.0099	.2702±.0097	.3242±.0199	.2203±.0021	.1241±.0033
TabCSDI	.1365±.0058	.2588±.0062	.1577±.0153	.3636±.0132	.1374±.0058	.2611±.0047	.2997±.0223	.2477±.0036	.1198±.0009
FD	.1392±.0058	.2509±.0081	.1592±.0106	.3789±.0044	.1346±.0040	.2592±.0053	.3024±.0309	.1927±.0016	.1192±.0017
SimpDM	**.1083±.0087**	**.2324±.0141**	**.1338±.0119**	**.2937±.0031**	**.1088±.0046**	.2194±.0053	**.2889±.0194**	.1366±.0008	**.1107±.0010**

Method	Airfoil	Wine-red	Abalone	Wine-white	Phoneme	Power	Ecommerce	California	Average Rank
mean	.2898±.0024	.1298±.0060	.1952±.0011	.1051±.0087	.1622±.0007	.1966±.0007	.3193±.0066	.1463±.0003	11.7
KNN	.2513±.0018	.0931±.0013	.1305±.0006	.0811±.0042	.1344±.0009	.1571±.0015	**.2959±.0113**	.1419±.0004	5.2
MF	.2379±.0023	.0980±.0025	.1461±.0021	.0858±.0066	.1369±.0010	.1512±.0012	.3654±.0105	.1186±.0005	6.1
MICE	.2929±.0015	.1071±.0028	.1420±.0010	.0920±.0076	.1780±.0006	.1631±.0014	.3606±.0079	.1292±.0007	10.2
OT	.2726±.0039	.0998±.0040	.1499±.0018	.0865±.0070	.1573±.0008	.1871±.0011	.3177±.0067	.1426±.0004	8.2
MIRACLE	.2577±.0024	.0942±.0027	.1293±.0012	.0865±.0070	.1559±.0006	.1442±.0010	.3142±.0056	.1175±.0005	4.9
GRAPE	.2289±.0045	.0897±.0033	.1314±.0061	.0982±.0198	.1237±.0012	.1338±.0016	.3071±.0082	**.0997±.0006**	3.5
IGRM	**.1797±.0033**	**.0860±.0027**	.1358±.0063	.0981±.0206	.1228±.0011	OOM	OOM	OOM	3.2
MIWAE	.2517±.0020	.1138±.0000	.1302±.0023	.0864±.0074	.1633±.0015	.1578±.0028	.3405±.0104	.1285±.0005	7.6
GAIN	.2581±.0019	.1140±.0044	.1493±.0113	.1024±.0076	.1685±.0027	.1626±.0050	.3450±.0119	.1311±.0007	9.9
TabCSDI	.2473±.0026	.0994±.0078	.1487±.0035	.0952±.0079	.1600±.0015	.1526±.0020	.3250±.0112	.1287±.0010	7.8
FD	.2458±.0020	.1026±.0026	.1468±.0013	.0976±.0118	.1596±.0014	.1633±.0011	.3357±.0087	.1289±.0004	7.7
SimpDM	.2024±.0046	.0873±.0016	**.1204±.0012**	**.0727±.0056**	**.1165±.0004**	**.1320±.0009**	.3051±.0111	.1056±.0006	1.4

for simplification purposes. The statistics of datasets are given in Appendix C.

5.1.2 Baselines. We compare SimpDM with three groups of methods: 1) shallow methods, including mean imputation, kNN imputation [40], miss forest (MF) [37], and MICE [41]; 2) deep methods, including OT [27], MIRACLE [20], GRAPE [47], and IGRM [51]; 3) deep generative methods, including MIWAE [26], GAIN [46], TabCSDI [48], and ForestDiffusion (FD) [14]. Among them, TabCSDI and FD are also based on diffusion models.

5.1.3 Implementation Details. In our major experiments, we simulate the default data missing setting as missing with complete random (MCAR) scenario with 30% missing ratio. We use Root Mean Squared Error (RMSE) as our evaluation metric, which is commonly used in previous works [20, 27]. We report the averaged test accuracy over 5 runs of experiments. We reproduce baselines based on HyperImpute package [13] or their corresponding official source codes. We select some important hyper-parameters through grid search, and keep the rest insensitive hyper-parameters to be fixed values. Concretely, the grid search is carried out on the following search space:

- Diffusion steps: {10, 50, 100}
- Training epochs: {10000, 20000, 30000}
- Learning rate: {0.001, 0.0001}
- The number of layers: {3,4,5}
- Hidden dimensions: {256, 512, 1024}
- Trade-off parameter of self-supervised alignment loss γ: {0.2, 0.5, 1, 3, 5, 10}
- Perturbation strengths [p_{GT},p_{PM},p_{MS}]: { [0.0001,0.08,0.1], [0.0001,0.4,0.5], [0.001,0.8,1], [0.001,2,3] }

5.1.4 Computing Infrastructures. We run all experiments with an Amazon Web Service (AWS) EC2 instance with the instance size `g4dn.xlarge`, which features a 4-core CPU, 16 GB Memory, and a Nvidia T4 GPU with 16 GB GPU Memory. We implement the proposed SimpDM with Python 3.10 and PyTorch 1.11.0 [31].

5.1.5 Source Code. The source code of SimpDM is available at: https://github.com/yixinliu233/SimpDM.

5.2 Performance Comparison

The performance comparison in the default setting is illustrated in Table 1. From the results, we have the following observations.

(1) SimpDM outperforms all the baselines on 11 datasets while achieving runner-up results on the rest 6 datasets. The consistent superiority of a majority of datasets highlights the

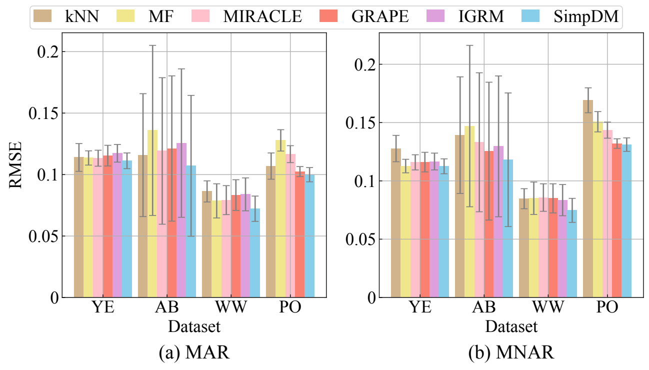


Figure 3: Performance on MAR and MNAR scenarios.

adaptability and robustness of our method in addressing missing data imputation challenges.

(2) Compared to diffusion model-based methods (i.e., TabCSDI and FD), SimpDM has significantly better performance, indicating the effectiveness of self-supervised alignment and state-dependent augmentation.

(3) Among generative methods, two diffusion model-based approaches perform slightly better than the GAN/VAE-based methods. This observation demonstrates the potential of diffusion models in addressing imputation problems.

(4) Deep learning-based methods, such as MIRACLE, GRAPE, and IGRM, show competitive performance, indicating the superior capability of deep neural networks in data completion.

(5) In the realm of shallow methods, kNN and MF exhibit strong performance, underscoring their superior capability in tabular data imputation.

(6) Compared to the strongest baseline IGRM, SimpDM requires less memory, indicating the efficiency and scalability of our approach.

5.3 Different Missing Scenarios and Ratios

To investigate the generalization ability in various missing situations, we conduct experiments on different missing scenarios (i.e., missing at random (MAR) and missing not at random (MNAR)) and missing ratios. For the sake of space, we compare SimpDM with five highly competitive baselines, namely kNN, MF, MIRACLE, GRAPE, and IGRM. The experiments are conducted on four datasets, namely, Yeast (YE), Abalone (AB), Wine-white (WW), and Power (PO).

5.3.1 Different missing scenarios. We conduct experiments to assess the effectiveness of SimpDM in both MAR and MNAR scenarios, with results detailed in Fig. 3. The figures demonstrate that SimpDM consistently surpasses all representative baselines in both scenarios, highlighting its robust generalization capabilities. Conversely, certain baselines exhibit suboptimal and unstable performance in certain cases.

5.3.2 Different missing ratios. To verify the robustness of SimpDM across varying degrees of data missingness, we alter the missing ratios from 0.1 to 0.7, assessing its imputation performance

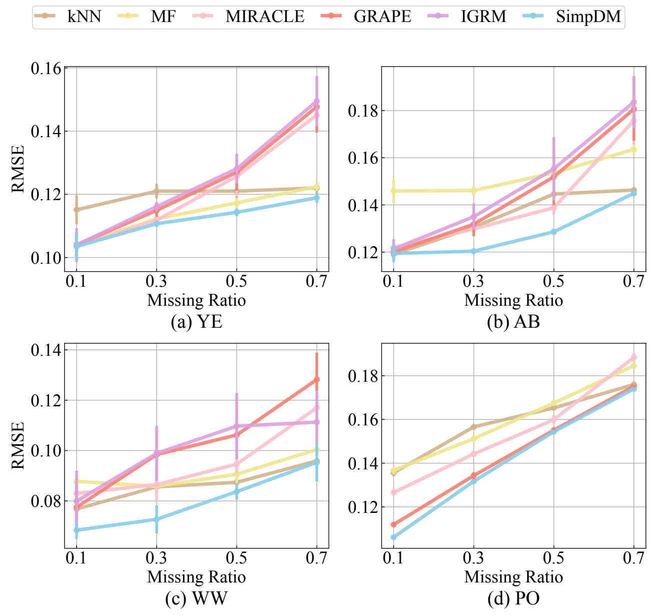


Figure 4: Performance under different missing ratios.

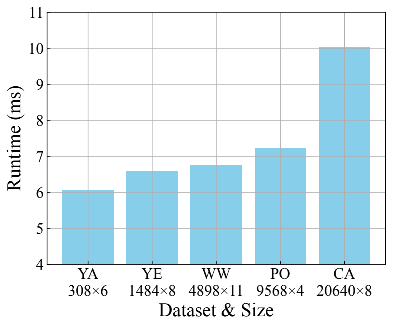
across diverse scenarios. As depicted in Fig. 4, SimpDM consistently demonstrates optimal or competitive performance across various missing ratios. Notably, in situations of severe data absence, SimpDM exhibits more substantial performance gains compared to the deep baselines.

5.4 Ablation Studies

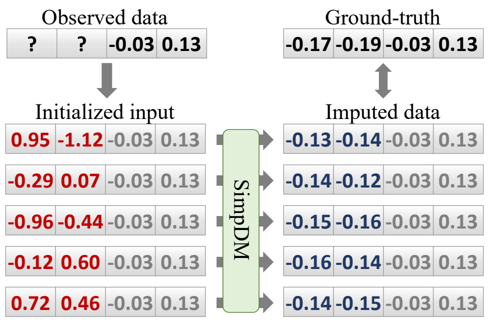
5.4.1 Effect of key components. To examine the contributions of two key components in SimpDM, namely self-supervised alignment (SA) and state-dependent augmentation (AUG), we conduct ablation experiments on our base model and its two variants, each incorporating one of these components. From the results in Table 2, we can find that both SA and AUG yield substantial performance enhancements for the base model. Furthermore, SA demonstrates a more pronounced impact on performance across the majority of datasets. Finally, we can witness that SimpDM, which incorporates both of these techniques, exhibits the best performance.

5.4.2 Effect of different self-supervised losses. In SimpDM, self-supervised alignment loss is a critical component to eliminate the unexpected diversity of imputation results. We explore three self-supervised alignment loss types: MSE loss, contrastive loss (CL) [5], and Sinkhorn divergence (SK) [27] (refer to Appendix xx for definitions), evaluating their respective effectiveness. As illustrated in Table 3, MSE loss attains the optimal results on 7 out of 8 datasets and exhibits competitive performance on the remaining one. We attribute the superior performance of MSE loss to its consistency with the objective of aligning the imputation results of two channels. Simultaneously, CL and SK losses also yield improvements, underscoring the effectiveness of self-supervised alignment.

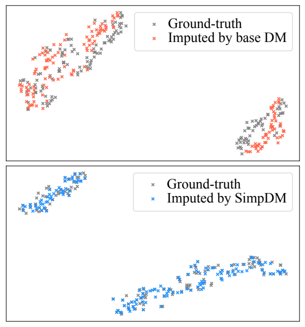
5.4.3 Effect of different augmentation strategies. To verify the effectiveness of state-dependent (SD) augmentation, we compare it



(a) Runtime Analysis



(b) Case study



(c) Visualization

Figure 5: (a) Runtime per training epoch on different datasets (with size $n \times d$). (b) The imputation results from different initialization. (c) t-SNE visualization of ground-truth data and imputed data.

Table 2: Ablation study for the key components in SimpDM.

Variant	YA	DI	EN	YE	WR	AB	WW	PO
Base model	.294	.307	.286	.149	.100	.163	.089	.169
+SA	.243	.298	.232	.118	.091	.130	.076	.140
+AUG	.241	.297	.241	.123	.092	.132	.077	.149
SimpDM	**.232**	**.294**	**.219**	**.111**	**.087**	**.120**	**.073**	**.132**

Table 3: Ablation study for self-supervised losses.

Variant	YA	DI	EN	YE	WR	AB	WW	PO
Base model	.294	.307	.286	.149	.100	.163	.089	.169
+SA (CL)	.261	.307	.251	.121	.095	.138	.081	.151
+SA (SK)	.259	**.295**	.259	.134	.092	.148	.080	.154
+SA (MSE)	**.243**	.298	**.232**	**.118**	**.091**	**.130**	**.076**	**.140**

Table 4: Ablation study for augmentation strategies.

Variant	YA	DI	EN	YE	WR	AB	WW	PO
Base model	.294	.307	.286	.149	.100	.163	.089	.169
+AUG (strong)	.298	.384	.280	.139	.127	.172	.099	.184
+AUG (weak)	.296	.306	.290	.149	.099	.165	.090	.171
+AUG (SD)	**.241**	**.297**	**.241**	**.123**	**.092**	**.132**	**.077**	**.149**

with two plain data augmentation strategies that perturb all entries with strong or weak strengths. From Table 4, we can see that weak data augmentation does not deeply affect the performance, whereas strong data augmentation tends to have a negative impact since it distorts the distribution of the original data. Conversely, state-dependent augmentation brings improvement by applying customized perturbation strengths to each entry according to its state and certainty.

5.5 Runtime Analysis

We present the runtime of each training epoch of SimpDM on different datasets. All the experiments are conducted on an Amazon EC2 server (see Sec. 5.1) and with a fixed-size MLP diffusion model (with 3 layers and 256 hidden units). The experimental results are illustrated in Fig. 5(a). The figure clearly illustrates that the training time of SimpDM exhibits a linear relationship with both the number of samples n and the number of dimensions d. This observation aligns seamlessly with our analysis detailed in Appendix B. Also, the training cost of SimpDM is quite small (with few milliseconds for each epoch), indicating the high running efficiency and scalability of our method.

5.6 Qualitative Analysis

5.6.1 Case study. We perform a case study experiment on a sample from the Power dataset (the same one as Fig. 1(a)) to investigate whether SimpDM can deliver stable imputation results from diverse initial noises. From Fig. 5(b), it is evident that SimpDM yields both increased stability and enhanced accuracy. This improvement can be attributed to the integration of the self-supervised alignment mechanism.

5.6.2 Visualization. Using t-SNE algorithm [42], we visualize the distribution of the original Iris dataset and the data imputed by the base model and SimpDM, respectively. In Fig. 5(c), we observe a significant overlap between the distribution of data imputed by SimpDM and the original data distribution, which suggests that SimpDM adeptly captures the data manifold. In contrast, the base diffusion model struggles to match the data distribution.

6 Conclusion

In this paper, we introduce a novel variant of diffusion models, termed SimpDM, designed for tabular data imputation. To enhance the imputation capabilities of the diffusion model, we propose a self-supervised alignment mechanism aimed at reducing its sensitivity to noise, thereby improving the stability of imputation results. Simultaneously, to address discrepancies in data scale, we present a state-dependent augmentation strategy that generates synthetic data during model training, which aims to bolster the robustness of SimpDM. Extensive experiments showcase the superior performance of SimpDM across various scenarios.

References

[1] Ahmed M Alaa, Jinsung Yoon, Scott Hu, and Mihaela Van der Schaar. 2017. Personalized risk scoring for critical care prognosis using mixtures of gaussian processes. *IEEE Transactions on Biomedical Engineering*, 65, 1, 207–218.

[2] Samuel A Assefa, Danial Dervovic, Mahmoud Mahfouz, Robert E Tillman, Prashant Reddy, and Manuela Veloso. 2020. Generating synthetic data in finance: opportunities, challenges and pitfalls. In *Proceedings of the First ACM International Conference on AI in Finance*, 1–8.

[3] Arthur Asuncion and David Newman. 2007. Uci machine learning repository. (2007).

[4] Jinyu Cai and Jicong Fan. 2022. Perturbation learning based anomaly detection. In *Advances in Neural Information Processing Systems*, 14317–14330.

[5] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*. PMLR, 1597–1607.

[6] Florinel-Alin Croitoru, Vlad Hondru, Radu Tudor Ionescu, and Mubarak Shah. 2023. Diffusion models in vision: a survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

[7] Prafulla Dhariwal and Alexander Nichol. 2021. Diffusion models beat gans on image synthesis. *Advances in neural information processing systems*, 34, 8780–8794.

[8] Alireza Farhangfar, Lukasz A Kurgan, and Witold Pedrycz. 2007. A novel framework for imputation of missing values in databases. *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, 37, 5, 692–709.

[9] Mikel Hernandez, Gorka Epelde, Ane Alberdi, Rodrigo Cilla, and Debbie Rankin. 2022. Synthetic data generation for tabular health records: a systematic review. *Neurocomputing*, 493, 28–45.

[10] Jonathan Ho, Ajay Jain, and Pieter Abbeel. 2020. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33, 6840–6851.

[11] Emiel Hoogeboom, Didrik Nielsen, Priyank Jaini, Patrick Forré, and Max Welling. 2021. Argmax flows and multinomial diffusion: learning categorical distributions. *Advances in Neural Information Processing Systems*, 34, 12454–12465.

[12] Aapo Hyvärinen and Peter Dayan. 2005. Estimation of non-normalized statistical models by score matching. *Journal of Machine Learning Research*, 6, 4.

[13] Daniel Jarrett, Bogdan C Cebere, Tennison Liu, Alicia Curth, and Mihaela van der Schaar. 2022. Hyperimpute: generalized iterative imputation with automatic model selection. In *International Conference on Machine Learning*, 9916–9937.

[14] Alexia Jolicoeur-Martineau, Kilian Fatras, and Tal Kachman. 2023. Generating and imputing tabular data via diffusion and flow-based gradient-boosted trees. *arXiv preprint arXiv:2309.09968*.

[15] Richard Judson et al. 2009. The toxicity data landscape for environmental chemicals. *Environmental health perspectives*, 117, 5, 685–695.

[16] Jayoung Kim, Chaejeong Lee, and Noseong Park. 2023. Stasy: score-based tabular data synthesis. In *The Eleventh International Conference on Learning Representations*.

[17] Zhifeng Kong, Wei Ping, Jiaji Huang, Kexin Zhao, and Bryan Catanzaro. 2021. Diffwave: a versatile diffusion model for audio synthesis. In *International Conference on Learning Representations*.

[18] Akim Kotelnikov, Dmitry Baranchuk, Ivan Rubachev, and Artem Babenko. 2023. Tabddpm: modelling tabular data with diffusion models. In *International Conference on Machine Learning*. PMLR, 17564–17579.

[19] Vladimir Kulikov, Shahar Yadin, Matan Kleiner, and Tomer Michaeli. 2023. Sinddm: a single image denoising diffusion model. In *International Conference on Machine Learning*. PMLR, 17920–17930.

[20] Trent Kyono, Yao Zhang, Alexis Bellot, and Mihaela van der Schaar. 2021. Miracle: causally-aware imputation via learning missing data mechanisms. *Advances in Neural Information Processing Systems*, 34, 23806–23817.

[21] Chaejeong Lee, Jayoung Kim, and Noseong Park. 2023. Codi: co-evolving contrastive diffusion models for mixed-type tabular synthesis. In *International Conference on Machine Learning*. PMLR.

[22] Ke Liang et al. 2024. Mines: message intercommunication for inductive relation reasoning over neighbor-enhanced subgraphs. In *Proceedings of the AAAI Conference on Artificial Intelligence* number 9. Vol. 38, 10645–10653.

[23] Yixin Liu, Shiyuan Li, Yu Zheng, Qingfeng Chen, Chengqi Zhang, and Shirui Pan. 2024. Arc: a generalist graph anomaly detector with in-context learning. *arXiv preprint arXiv:2405.16771*.

[24] Yue Liu, Ke Liang, Jun Xia, Sihang Zhou, Xihong Yang, Xinwang Liu, and Stan Z Li. 2023. Dink-net: neural clustering on large graphs. In *International Conference on Machine Learning*. PMLR, 21794–21812.

[25] Andreas Lugmayr, Martin Danelljan, Andres Romero, Fisher Yu, Radu Timofte, and Luc Van Gool. 2022. Repaint: inpainting using denoising diffusion probabilistic models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 11461–11471.

[26] Pierre-Alexandre Mattei and Jes Frellsen. 2019. Miwae: deep generative modelling and imputation of incomplete data sets. In *International conference on machine learning*. PMLR, 4413–4423.

[27] Boris Muzellec, Julie Josse, Claire Boyer, and Marco Cuturi. 2020. Missing data imputation using optimal transport. In *International Conference on Machine Learning*. PMLR, 7130–7140.

[28] Muhammad S Osman, Adnan M Abu-Mahfouz, and Philip R Page. 2018. A survey on data imputation techniques: water distribution system as a use case. *IEEE Access*, 6, 63279–63291.

[29] Yidong Ouyang, Liyan Xie, Chongxuan Li, and Guang Cheng. 2023. Missdiff: training diffusion models on tabular data with missing values. *arXiv preprint arXiv:2307.00467*.

[30] Shirui Pan, Yizhen Zheng, and Yixin Liu. 2024. Integrating graphs with large language models: methods and prospects. *IEEE Intelligent Systems*, 39, 1, 64–68.

[31] Adam Paszke et al. 2019. Pytorch: an imperative style, high-performance deep learning library. *Advances in Neural Information Processing Systems*, 32, 8026–8037.

[32] Kashif Rasul, Calvin Seward, Ingmar Schuster, and Roland Vollgraf. 2021. Autoregressive denoising diffusion models for multivariate probabilistic time series forecasting. In *International Conference on Machine Learning*. PMLR, 8857–8868.

[33] Balachander Sathianarayanan, Yogesh Chandra Singh Samant, Prahalad S Conjeepuram Guruprasad, Varshin B Hariharan, and Nirmala Devi Manickam. 2022. Feature-based augmentation and classification for tabular data. *CAAI Transactions on Intelligence Technology*, 7, 3, 481–491.

[34] Connor Shorten and Taghi M Khoshgoftaar. 2019. A survey on image data augmentation for deep learning. *Journal of big data*, 6, 1, 1–48.

[35] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. 2015. Deep unsupervised learning using nonequilibrium thermodynamics. In *International conference on machine learning*. PMLR, 2256–2265.

[36] Yang Song and Stefano Ermon. 2020. Improved techniques for training score-based generative models. *Advances in neural information processing systems*, 33, 12438–12448.

[37] Daniel J Stekhoven and Peter Bühlmann. 2012. Missforest—non-parametric missing value imputation for mixed-type data. *Bioinformatics*, 28, 1.

[38] Yusuke Tashiro, Jiaming Song, Yang Song, and Stefano Ermon. 2021. Csdi: conditional score-based diffusion models for probabilistic time series imputation. *Advances in Neural Information Processing Systems*, 34, 24804–24816.

[39] Lev Telyatnikov and Simone Scardapane. 2023. Egg-gae: scalable graph neural networks for tabular data imputation. In *International Conference on Artificial Intelligence and Statistics*. PMLR, 2661–2676.

[40] Olga Troyanskaya, Michael Cantor, Gavin Sherlock, Pat Brown, Trevor Hastie, Robert Tibshirani, David Botstein, and Russ B Altman. 2001. Missing value estimation methods for dna microarrays. *Bioinformatics*, 17, 6, 520–525.

[41] Stef Van Buuren and Karin Groothuis-Oudshoorn. 2011. Mice: multivariate imputation by chained equations in r. *Journal of statistical software*, 45, 1–67.

[42] Laurens Van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-sne. *Journal of machine learning research*, 9, 11.

[43] Yili Wang, Yixin Liu, Xu Shen, Chenyu Li, Kaize Ding, Rui Miao, Ying Wang, Shirui Pan, and Xin Wang. 2024. Unifying unsupervised graph-level anomaly detection and out-of-distribution detection: a benchmark. *arXiv preprint arXiv:2406.15523*.

[44] Minkai Xu, Lantao Yu, Yang Song, Chence Shi, Stefano Ermon, and Jian Tang. 2022. Geodiff: a geometric diffusion model for molecular conformation generation. *arXiv preprint arXiv:2203.02923*.

[45] Ling Yang, Zhilong Zhang, Yang Song, Shenda Hong, Runsheng Xu, Yue Zhao, Wentao Zhang, Bin Cui, and Ming-Hsuan Yang. 2022. Diffusion models: a comprehensive survey of methods and applications. *ACM Computing Surveys*.

[46] Jinsung Yoon, James Jordon, and Mihaela Schaar. 2018. Gain: missing data imputation using generative adversarial nets. In *International conference on machine learning*. PMLR, 5689–5698.

[47] Jiaxuan You, Xiaobai Ma, Yi Ding, Mykel J Kochenderfer, and Jure Leskovec. 2020. Handling missing data with graph representation learning. *Advances in Neural Information Processing Systems*, 33, 19075–19087.

[48] Shuhan Zheng and Nontawat Charoenphakdee. 2022. Diffusion models for missing value imputation in tabular data. *arXiv preprint arXiv:2210.17128*.

[49] Xin Zheng, Yixin Liu, Zhifeng Bao, Meng Fang, Xia Hu, Alan Wee-Chung Liew, and Shirui Pan. 2023. Towards data-centric graph machine learning: review and outlook. *arXiv preprint arXiv:2309.10979*.

[50] Yizhen Zheng, He Zhang, Vincent Lee, Yu Zheng, Xiao Wang, and Shirui Pan. 2023. Finding the missing-half: graph complementary learning for homophily-prone and heterophily-prone graphs. In *International Conference on Machine Learning*. PMLR, 42492–42505.

[51] Jiajun Zhong, Ning Gui, and Weiwei Ye. 2023. Data imputation with iterative graph reconstruction. In *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 37, 11399–11407.

Algorithm 1 SimpDM Training

```
Input: Sample $\mathbf{x}$, missing mask $\mathbf{m}$
1:  $\mathbf{m}_p \sim \text{Bernoulli}((\mathbb{1}_n - \mathbf{m}) \times r_m)$
2:  $\mathbf{m}_c = \mathbf{m}_p + \mathbf{m}$
3:  for $k = 1, 2$ do
4:      $t_k \sim \text{Uniform}(\{0, \dots, T\})$
5:      $\epsilon_k \sim \mathcal{N}(0, \boldsymbol{I})$
6:      $\xi_k \sim \mathcal{N}(0, \boldsymbol{I})$
7:      Calculate $\tilde{\mathbf{x}}_{t_k}$ via Eq. (13)
8:      $\hat{\mathbf{x}}_k = \mathbf{x}_\theta\left(\tilde{\mathbf{x}}_{t_k}, t_k, \mathbf{m}_c\right)$
9:      Calculate $\mathcal{L}_{dm_k}$ via Eq. (10)
10: end for
11: Calculate $\mathcal{L}_{sa} = \text{MSE}(\hat{\mathbf{x}}_1, \hat{\mathbf{x}}_2)$
12: Calculate overall loss $\mathcal{L}$ via Eq. (12)
13: Update model $\mathbf{x}_\theta$ by taking gradient descent step on $\mathcal{L}$
```

Algorithm 2 SimpDM Imputing

```
Input: Sample $\mathbf{x}$, missing mask $\mathbf{m}$
1:  $\mathbf{x}_T \sim \mathcal{N}(0, \boldsymbol{I})$
2:  for $t = T, \cdots, 1$ do
3:      $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ if $t > 1$, else $\mathbf{z} = \mathbf{0}$
4:      Calculate $\tilde{\mathbf{x}}_t$ via Eq. (11)
5:      $\mathbf{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_\theta\left(\tilde{\mathbf{x}}_t, t, \mathbf{m}\right)\right) + \sigma_t \mathbf{z}$
6:  end for
7:  $\hat{\mathbf{x}} = \mathbf{x} \odot (\mathbb{1}_{n\times d} - \mathbf{m}) + \mathbf{x}_0 \odot \mathbf{m}$
8:  return $\hat{\mathbf{x}}$
```

Table 5: The statistics of datasets, including the number of samples, numerical features, and categorical features.

Dataset	#Samples	#Numerical Feat	#Categorical Feat
Iris	150	4	0
Yacht	308	6	0
Housing	506	12	1
Diabetes	520	1	15
Blood	748	4	0
Energy	767	8	0
German	1,000	7	13
Concrete	1,030	8	0
Yeast	1,484	8	0
Airfoil	1,503	5	0
Wine-red	1,599	11	0
Abalone	4,177	7	1
Wine-white	4,898	11	0
Phoneme	5,404	5	0
Power	9,568	4	0
Ecommerce	10,999	3	7
California	20,640	8	0

A Algorithm of SimpDM

In this section, we provide the training and inference algorithms of SimpDM with the example of a single data. In practice, we employ mini-batch-based training. Here we use Gaussian noise as the example of perturbation and employ MSE loss as the self-supervised alignment loss.

The training algorithm of SimpDM is summarized in Algorithm 1. In the first step, we sample the pseudo mask $\mathbf{m}_p$ for the missing training strategy. The condition mask can be computed by adding $\mathbf{m}_p$ and the original missing mask $\mathbf{m}$. After that, we run the diffusion model on two channels ($k = 1, 2$) respectively. In each channel, we sample the diffusion time step t_k and noise ϵ_k first. Then, we conduct the state-dependent augmentation with an extra sampled perturbation ξ_k (Line 6-7). With the perturbed input, we obtain the imputed data with the diffusion model (Line 8) and calculate the diffusion model loss $\mathcal{L}_{dm_k}$ accordingly (Line 9). Once we obtain the imputed data by two channels ($\hat{\mathbf{x}}_1$ and $\hat{\mathbf{x}}_2$), we calculate the self-supervised alignment loss $\mathcal{L}_{sa}$. Finally, we can add the losses together and train the diffusion model $\mathbf{x}_\theta$ via gradient descent.

The inference (imputation) of SimpDM is demonstrated in Algorithm 2. Similar to the vanilla diffusion model, we first initialize $\mathbf{x}_T$ with random Gaussian noise, and then recursively run the denoising iteration with the diffusion model. Differently, in each iteration, we set the observed entries to the ground-truth values (Line 4) before each denoising step, which ensures the observed values can well guide the imputation process. Finally, the imputed data $\hat{\mathbf{x}}$ can be obtain by combining $\mathbf{x}$ and the final estimation $\mathbf{x}_0$ (Line 7).

B Complexity Analysis

We study the time complexity of a single training epoch of SimpDM, given a tabular dataset with n samples and d dimensions. For the sampling of pseudo mask, Gaussian noise, and augmented perturbation, their complexities are all $O(nd)$. For the diffusion model, the time complexity is $O(nd_h(d + d_hL))$, where d_h and L are the latent dimensions and the layer number of MLP, respectively. For the diffusion model loss and the MSE self-supervised alignment loss, the complexities are also $O(nd)$. Note that if we use more complex self-supervised alignment loss (e.g., contrastive loss or Sinkhorn loss), the complexity can be higher to $O(n^2d)$, which severely reduces the running efficiency. After eliminating the smaller terms, the training time complexity of SimpDM becomes $O(nd_h(d+d_hL))$. For the testing phase of SimpDM, the complexity is also similar to vanilla diffusion models. In specific, the complexity is $O(ndT)$ for the whole process, since we require T diffusion time steps for data refinement. To sum up, this complexity of SimpDM scales linearly with both n and d, resembling the computational costs of vanilla diffusion models.

C Dataset

We conduct the experiments on 17 datasets from the UCI Machine Learning repository [3] and Kaggle, including Iris, Yacht, Housing, Diabetes, Blood, Energy, German, Concrete, Yeast, Airfoil, Wine-red, Abalone, Wine-white, Phoneme, Power, Ecommerce, and California. The statistics of datasets are provided in Table 5.