α-GAIN: Missing data imputation using GAIN with modification on hint generator

Thiam Wai Chua
Eindhoven, Noord-Brabant
t.w.chua1985@gmail.com

Pengcheng Wang
Eindhoven, Noord-Brabant

ABSTRACT

The occurence of missing values in real-life datasets can make it hard to implement and deploy prediction models. Therefore, the techniques of imputation replaces missing values with the best possible alternative value have been created. However, building these imputation models is not straightforward. In this paper, we present an improved version of the Generative Adversarial Imputation Net (GAIN) [17] framework by modifying its hint component, which can be used in missing data imputation under the scenario of missing at complete random (MCAR). For the original GAIN framework, we see potential in its structure, more specifically the hinting mechanism. We propose a modification of the hint mechanism in the GAIN algorithm. We call our model α-Generative Adversarial Imputation Networks (α-GAIN). The generator (G) imputes the missing components according to observed components. The discriminator (D) then take the imputed dataset and determine which components were observed and which were imputed. The hint generator (H) generates the hint matrix as additional information to D. To ensure that D to learn effectively when its loss is low, in turn forces G to learn, we have improved the hint mechanism in GAIN in two way: (1) we modify the fake indication in the hint generator while generating the hint matrix (2) and we provide a modified sigmoid function in the hint generator for computing the updated hint rate based on the loss of discriminator. We tested our model on various datasets and found that α-GAIN has slightly improvement compared with GAIN between missing rate 0.5 and 0.8.

KEYWORDS

Missing data, MCAR, imputation

ACM Reference Format:
Thiam Wai Chua and Pengcheng Wang. 2022. α-GAIN: Missing data imputation using GAIN with modification on hint generator. In *Proceedings of Conference on Knowledge Discovery and Data Mining (KDD 2022).* ACM, New York, NY, USA, 11 pages. https://doi.org/0000000.0000000

1 INTRODUCTION

Over the past decades, the importance of missing data and model performance have become significant in both statistics and machine learning, while in reality acquiring the entire dataset turns out to be laborious work for many reasons. Incomplete data may not be possible to use to train certain models. However, the incomplete data may contain certain amount of information which needs further exploration. Simply discarding records with missing values may also result in data loss and bias. Hence it is desirable to handle missing data to minimize bias and maximize the precision in estimation. With lower missing rate, removing the variables or cases with missing values is feasible, but such techniques fail to success when high missing rate is present.

Instead of deleting the variables or cases with missing values, we can impute data. Imputation is the procedure of inserting values to cells of missing data that could have been observed but were not due to various reasons. Apart from various traditional single and multiple imputation methods [14], machine learning models for the task have been researched, among which the most popular methods fall under the families of the Generative Adversarial Networks (GAN) [3]. GAN comprises two neural networks that argue against each other in the form of a zero-sum game. The generator aims to produce new instances, and the discriminator criticises the instances produced by the generator whether being from the real domain. These adversarial networks are trained iteratively until the discriminator is confused. Thus, the generator starts generating plausible instances. Much research has been done to modify the original GAN framework to a certain extent by finding more suitable loss functions or network structures.

The Generative Adversarial Imputation Nets (GAIN) method is an extension of GAN, it can be used even if complete data is not available. In contrast to GAN, which uses complete data, GAIN has an incomplete, mixed-type flat-table dataset as input. In addition, a noise matrix and mask matrix is fed to the generator. Here, the mask matrix is used to tell the generator which values are missing or observed. The goal of the generator is to generate imputations. The discriminator task is to distinguish fake from real values in the imputed dataset. In GAIN, it uses a hint matrix as additional information to ensure that the discriminator forces the generator to learn.

The ideas and developments of GAIN make use of GAN, however it still has a lot of space for improvement that could potentially result in better data sets. To improve the results of missing data imputation using GAIN, in this paper, we are interested in investigating an improvement of GAIN using a different mechanism of the hint generator [17], we named it α-GAIN.

Firstly, we modify the fake indication in the hint generator while generating the hint matrix. Secondly, we provide a modified sigmoid function in the hint generator for computing the updated hint rate based on the loss of discriminator. To demonstrate the effectiveness of our proposed model, we evaluate it on three datasets with type of missingness MCAR. In this paper, we provide empirical evidence that our modification of the model improves the general performance of the GAIN algorithm.

KDD 2022, October, 2022, Eindhoven, Netherlands

ACM ISBN 978-1-4503-0000-0/18/06.
https://doi.org/0000000.0000000

2 BACKGROUND

In this section, we first introduce the preliminaries of the missing data tasks. Subsequently we look into details of the missingness mechanism and missing data techniques. After that, we describe the GAN and GAIN model.

2.1 Preliminaries

2.1.1 Data Matrix. In the tabular dataset, we define the data vector $\mathbf{X} = (x_1, ..., x_d)$ as a row of the data table that takes values in a d-dimensional space $\mathcal{X} = \mathcal{X}_1 \times \mathcal{X}_2 \times ... \mathcal{X}_d$. Furthermore, the data vector has a distribution that is denoted by $P(\mathbf{X})$. In addition, we shall also use with tilde symbol above to denote the vector or the elements from the vector with unobserved values. let $\tilde{\mathcal{X}}_i$ be a superset of $\mathcal{X}$ that contains arbitrary unobserved values in the notation of $\tilde{\mathcal{X}}_i = \mathcal{X}_i \cup \{*\}$ for each $i \in \{1, ..., d\}$, where $\{*\}$ is used to denote the set of the unobserved numerics. Thus, $\tilde{\mathbf{X}} = (\tilde{x_1}, ..., \tilde{x_d}) \in \tilde{\mathcal{X}}_i$ is used here to denote the data with unobserved components. Each dataset, in the forms of data matrix, contains a number of realizations of the data vector.

2.1.2 Mask. We define the mask vector $\mathbf{M} = \{M_1, ..., M_d\} \in \{0, 1\}^d$ as the indicator of the location of missingness in the corresponding data vector $\mathbf{X}$, in a way that it can be recovered from the $\tilde{\mathbf{X}}$ matrix. If the feature $\tilde{x_i}$ of the data vector is observed, we let $M_i = 1$, and let $M_i = 0$ otherwise.

2.1.3 Imputation. We shall use the notation $\mathcal{D} = \{(\tilde{\mathbf{x}}^i, \mathbf{m}^i)\}_{i=1}^n$ to denote the incomplete dataset for the purpose of our problem, where $\tilde{\mathbf{x}}^i$ in lower-case letter is the ith row of the dataset which can be considered as one of the i independent and identical distribution realization of the upper-case $\tilde{\mathbf{X}}$. The imputation task essentially involves filling in the missing data points in $\mathcal{D}$ by generating samples points that have the same distribution P as the population, according to the conditional distribution $P(\mathbf{X}|\tilde{\mathbf{X}} = \tilde{\mathbf{x}}^i)$

2.2 Missingness Mechanism

Missingness mechanisms describes the relationship between the missing and observed data. It assumes on the types and nature of missing data. Three missing data mechanisms are defined by Little and Rubin [11]: Missing Completely at Random (MCAR), Missing at Random (MAR), and Missing Not at Random (MNAR). In general, the missingness mechanism is concerned with whether or not the missingness is related to the study variables. This is significant as it demonstrates the difficulty to handle the missing values and meanwhile how risky it is to ignore the missing values [14].

The missingness mechanisms can be introduced as follows. As is mentioned beforehand, suppose $\{\tilde{\mathbf{x}}^i\}_{i=1}^n$ is an incomplete, mixed-type flat-table dataset matrix that includes observed. We use $\{\tilde{\mathbf{x}}^i_{obs}\}_{i=1}^n$ and $\{\tilde{\mathbf{x}}^i_{mis}\}_{i=1}^n$ to denote the observed and unobserved part of the data vector.

For the MCAR mechanism, the independent relationship is noticeable between the missing element and others. Locations of the missingness are independent of any given datasets, i.e. the missingness is unrelated to the values of any variables, whether missing or observed, hence we have $P(\mathbf{m}^i|\tilde{\mathbf{x}}^i) = P(\mathbf{m}^i)$ for all $\tilde{\mathbf{x}}^i$. Under MAR scenario, the probability of missingness hinges only on other observed values instead of missing values of the target variable. When MAR is the missing type, we have $P(\mathbf{m}^i|\tilde{\mathbf{x}}^i) = p(\mathbf{m}^i|\tilde{\mathbf{x}}^i_{obs})$. For MNAR mechanism, however, missingness are directly dependent on the variable of missingness itself. This means that $\mathbf{m}^i$ possibly depends on both $\tilde{\mathbf{x}}^i_{obs}$ and $\tilde{\mathbf{x}}^i_{mis}$.

GAIN [17] has the potential for all three missingness scenarios. However, in this research paper, we use only MCAR to test our improvements on GAIN.

2.3 Missing Data Techniques

Song and Shepperd [14] divided these missing data techniques (MDTs) into three categories, namely missing data ignoring techniques, missing data toleration techniques, missing data imputation techniques (MDITs).

The *missing data ignoring techniques* simply delete the cases that contain missing data. The *missing data toleration techniques* are the internal missing data treatment strategies, they work directly with data sets containing missing values.

Next, the MDITs refer to any approach of inserting values to the missing data in the dataset. Therefore, the dataset can be treated as complete and appropriate for standard data analysis. This approach keeps the data incomplete and assigns values of correlated variables. [14].

Single imputation is a technique of replacing the fixed value for each missing value. This technique can be used in machine learning since the lost data is handled once by implying a consequent consistency of answers from identical analyses. Whereas multiple imputation (MI) techniques [4], each missing value is substituted by various values, and several different completed data sets are generated. Due to random variation, several missing values are imputed in each generated complete data set. Multiply imputed datasets are analysed and then pooled into the final imputed dataset. In our study, we focus on single imputation. Following the [17], we focus only on single imputation in our research.

2.4 Generative Techniques and GAN

Another important family of methods for solving missing data problems are combined with state-of-the-arts deep learning algorithms, one of the most popular of which is GAN. Our proposed method α-GAIN is based on GAN, which is introduced briefly among generative models in this section.

GAN is a deep learning model that deals with the problem as a supervised learning problem, which has two sub-networks, namely the generator G and discriminator D. The G is input with random noises and generates reasonable fake instances. It aims to let the counterfeit instances close to actual instances so that D has difficulty distinguishing between real and generated cases. The D is a classification network to classify whether a sample is fake or real. The G tries to confuse the D to make the wrong decision, i.e. the G tries to decrease the accuracy of D. At the same time, D tries to distinguish the fake instances from the real ones. In this process, the differences between counterfeit and genuine instances are used to improve the generator. Hence, the G generates realistic-looking instances while the D gets better at selecting them.

Given the above discussion, it makes sense perceptibly that the GAN models would have a different loss function that other families

of deep models by performing the two-player minimax procedure. In fact, the objective function for GAN can be formulated as follow:

$$\min_G \max_D V(D,G) = \mathbb{E}_{\mathbf{x}}[\log D(\mathbf{x})] + \mathbb{E}_{\mathbf{z}}[\log \frac{1}{1 - D(G(\mathbf{z}))}]$$

where $V(D,G)$ is the value function of the minimax game, with G serving as the mapping from the pre-defined noise variables $\mathbf{z}$ to data space. $D(\mathbf{x})$ denotes the probability that sample $\mathbf{x}$ came from the real data instead of our generated fake examples. In the adversarial modeling framework, the D is trained to maximize the probability of correctly classifying real training examples and those generated by G. In the meanwhile, G is concurrently trained to minimize the loss function as a whole. By optimizing G in much smaller steps, D is mostly likely to converge close to its optimal solution.

Since its debut, GAN has been widely and extensively studied in the fields of computer vision, natural language processing and speech recognition. As it begins to play a more and more significant role, researchers has put forward various methods to integrate GAN into the missing data imputation task.

2.5 Generative Adversarial Imputation Network

GAIN [17], which is based on GAN [3], was recently developed and found to outperform other methods in terms of imputation accuracy (how true the values in comparison with being measured) in substituting MCAR data in five open-source datasets [17].

Among other prior GAN models, GAIN is unique in that its generator receives both noise and mask as an input data which is suitable for the missing data task. Also, instead of telling whether the whole sample is real or fake, the discriminator output a numerical matrix which also discerns the authenticity of the imputed dataset. Additionally, a hint mechanism is applied so that the discriminator receives auxiliary information in the form of a hint vector. The architecture of GAIN is shown in Figure 1.

In more detail, the generator performs as an imputer, while the discriminator aims to identify whether each input component is being estimated or sampled from the actual domain. The mask matrix indicates which values are missing and which are present. Randomness is added to the imputed values by the random matrix. A hint mechanism is used to adjust the discriminative difficulty of the theoretical analysis based on the MCAR case. In the following, we will briefly describe the generator, hint, discriminator and optimisation components in GAIN.

2.5.1 Generator. The generator used for GAIN is modelled with one input layer, one hidden layer and one output layer. The loss function for training of GAIN is composed by two functions. One is the mean square error of the generated data with the origin data and the other is the cross entropy loss given by the discriminator. The mask matrix consists of booleans, typically in ones and zeros, that can be recovered and drawn from the existing data matrix. By bringing the mask to the generator we are actually showing the locations of the missing element.

As what has been illustrated in figure 1, the generator, a function in the form of $\tilde{\mathcal{X}} \times \{0,1\}^d \times [0,1]^d \rightarrow \mathcal{X}$, will take the input of both realizations of $\tilde{\mathbf{X}}$ and its corresponding mask matrix $\mathbf{M}$, as well as a

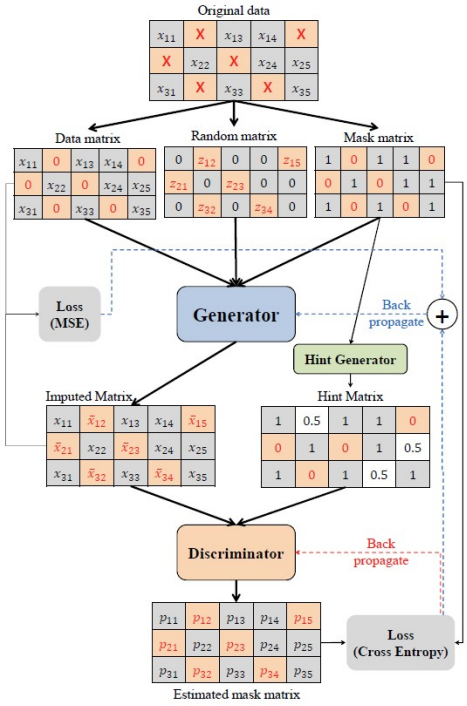


Figure 1: The architecture of GAIN [17].

random noise matrix $\mathbf{Z} = \{Z_1, ..., Z_d\}^d$. The tensors of data matrix, random matrix and mask matrix are input into the input layer.

After that, the *ReLU* activation function is applied to include the nonlinearity in the generator. In the hidden layer, the *ReLU* activation function is applied. Finally, the output layer is modeled with sigmoid activation function to generate a tensor filled with value between 0 and 1 from the real value tensor generated by hidden layer. Let $\bar{\mathbf{X}}$ with a bar denote the data vector that contains only the imputed value, we can therefore acquire $\bar{\mathbf{X}}$ by $\bar{\mathbf{X}} = G(\tilde{\mathcal{X}}, \mathbf{M}, (1-\mathbf{M}) \otimes \mathbf{Z})$, where G is the operations done by the generator network and $\otimes$ is element-wise multiplication. Let $\hat{\mathbf{X}}$ denote the completed data vector, which can be acquired by integrating original data vector and the vector with imputed values: $\hat{\mathbf{X}} = \mathbf{M} \otimes \tilde{\mathcal{X}} + \mathbf{M} \otimes (1 - \bar{\mathbf{X}})$

Since the purpose is to validate the output of generator and we cannot know exactly the truth value of the generated part, we use the conserved part of dataset as validation since the generator will output the whole map of the data.

2.5.2 Hint. The original hint mechanism in GAIN has a hint generator which processes the mask matrix with a specific hint rate and then produces a hint matrix as an input of the discriminator. The process is implemented by element-wise multiplying the mask matrix ($\mathbf{M}$) and a hint matrix ($\mathbf{H}$) comprised of elements of 1 randomly distributed by the hint rate and the rest of 0.

More specifically, the hint mechanism is a random variable, $\mathbf{H}$, sampling values in a space $\mathcal{H}$. The $\mathbf{H}$ is dependent on $\mathbf{M}$ and for each imputed sample $(\hat{x}, \mathbf{m})$, the hint matrix $\mathbf{h}$ is drew based on the distribution $\mathbf{H}|\mathbf{M} = \mathbf{m}$. Then, $\mathbf{h}$ is delivered as an extra input to the discriminator so that $D : \mathcal{X} \times \mathcal{H} \rightarrow [0,1]^d$, thus the i-th

component of $D(\hat{x}, \mathbf{h})$ corresponds to the probability that the i-th component of $\hat{x}$ was observed conditional on $\hat{X} = \hat{x}$ and $\mathbf{H} = \mathbf{h}$. However, during the element-wise multiplication, only observed data indicators represented by 1 are affected, which means only the amount of information about the observed data would be controlled, and the existing distribution of missing data might all be revealed in contrast. In other words, the indication of observed data is undoubtedly accurate though the missing components are indicated with uncertainty. The uncertainties of hint information is discussed in Section 4.1.1 and proved in Appendix A.1.

2.5.3 Discriminator. The discriminator network contains one input layer, one hidden layer and one output layer. One essential part of the inputs to the discriminator is the hint matrix in that it controls and provides a crucial fraction of information about the mask matrix. The tensor of imputed matrix and hint matrix are input into the input layer. The training goal of the discriminator is for it to be able to distinguish which components of the inputs are real or fake. This can boil down to the task of predicting the mask matrix $\mathbf{M}$ that already contains missingness information. To formulate, the discriminator is a function $D : \mathcal{X} \rightarrow [0, 1]^d$ with i-th component of $D(\hat{x})$ corresponding to the probability that the i-th component of $D(\hat{x})$ was observed.

GAIN [17] demonstrated that it works well on the low dimensional datasets with low missing rate. It also works at Modified National Institute of Standards and Technology (MNIST) database [9] of 50% missing rate, however, it converges to zero imputation or mean imputation for datasets with high missing rates [17].

2.5.4 Objective. The training goal of D is for it to predict the mask matrix. The training goal of G is for it to generate samples close to the original distribution that are difficult for D to discriminate. Similar to ordinary GAN, the objective function can be formulated as the minimax procedure $\min_G \max_D V(D, G)$, where $V(D, G)$ is given by

$$\begin{aligned} V(D,G) =& \mathbb{E}_{\hat{\mathbf{X}},\mathbf{M},\mathbf{H}}[\mathbf{M}^T \log D(\hat{\mathbf{X}}, \mathbf{H}) \\ &+ (1-\mathbf{M})^T \log(1 - D(\hat{\mathbf{X}}, \mathbf{H})), \end{aligned}$$

Since the loss function essentially asks for how much the discriminator is different from the original mask, the loss can also be formalized as

$$\begin{aligned} \min_G \max_D \mathbb{E}[\mathcal{L}(\mathbf{M}, \hat{\mathbf{M}})] =& \min_G \max_D \mathbb{E}[\sum_{i=1}^{n} (\mathbf{m}_i \log \hat{\mathbf{m}}_i \\ &+ (1-\mathbf{m}_i) \log(1 - \hat{\mathbf{m}}_i))], \end{aligned}$$

where $\hat{\mathbf{M}}$ is the outcome of the discriminator $D(\hat{\mathbf{X}}, \mathbf{H})$.

2.5.5 Optimizer. For the optimisation, Adam optimiser is used. This optimisation algorithm extends stochastic gradient descent (SGD) to update network weights during training. Furthermore, the Adam optimiser does not need a large space and requires less memory, which is very efficient. Intuitively, it combines the gradient descent with the momentum algorithm, and the RMSprop algorithm [2]. Adam optimiser is used to boost the gradient descent algorithm by considering the exponentially weighted average of the gradients. Utilising the averages leads the algorithm to converge towards the minima at a shorter pace. The learning rate is set to 0.001 to compute the gradient and update the parameters.

3 RELATED WORK

Traditionally, simple data analysis techniques, such as deletion and single imputation, can be used to tackle missing data when a small fraction of the sample is unobserved often under the assumption of MCAR. Among the main basic imputation techniques are mean imputation, regression imputation, hot-deck imputation and advanced imputation techniques such as Expectation Maximisation (EM) methods and Raw Maximum likelihood (RML) methods [4]. Regardless, more sophisticated models are required, however, when the missingness comprises a larger percent of the observation (>5%).

A number of modern missing data imputation methods belong to the family of discriminative models, including MICE [16] and Missforest [15]. Apart from that, many researchers have turned their attention to the family of generative models since the first introduction of GAN in 2014.

Early of GAN-based missing data imputation methods focuses on the task of consistent image completion. train a model with global and local discriminators to tell fake images. Completion network is a generative network based on encoder-decoder and fully convolutional structure accepting large area of input. This gives the model the ability to fill out missing areas in images. While these methods. For example, the Denoising Autoencoder Based (DAE) missing data imputation techniques [13] have been shown to work well in practice, but the downside is that it requires complete data to be available during training. In many circumstances, obtaining a complete dataset is almost impossible.

After the coming out of GAIN, a few other attempts have been made regarding the its improvement. One of such focus mainly on variable splitting, by connecting the dense layers for each variable in parallel to produce multiple outcome. They also use a gumbel-softmax activation in order to work better with the backpropagation when doing the training. Aside from VAE imputation, they also propose iterative imputation and backpropagation iterative imputation which, however, add little improvement to the model [12].

Stackelberg GAN [19] utilises multiple generators to impute the medical missing data through generating diverse imputed values. It uses fully connected neural network and its loss function is aggregated by all pairs of GAN losses. CollaGAN [7] converts the image imputation problem to a multi-domain images translation task. CollaGAN able to fill out the missing data with high quality reasonable image through multiple inputs with multiple domains. Its generator is based on U-net structure and its discriminator is composed of Leaky-ReLU and convolution layers.

MISGAN [10], which is itself based on the idea of AmbientGAN [1], is proposed learning a mask distribution to model the incompleteness. In MISGAN model, two generators are designed independent on each other for both the complete data and the mask with their own noise distributions. MISGAN also demonstrated that deep convolutional GAN architecture performs better compared with MISGAN if the data type is an image like MNIST [9].

4 α-GAIN

In this section, we explain our improvements of the GAIN hint mechanism. The algorithm of α-GAIN is shown in Algorithm 1.

Algorithm 1 Pseudo-code of Training of α -GAIN

Require: The complete dataset $\mathbf{X} \in \mathbb{R}^{d \times n}$, missing rate p, hint rate h

Ensure: Expected output Generator G which take the sample data $x \in \mathbb{R}^{d \times s}$ as input

Initial generator G and discriminator D
Initial epoch e, batch size s
generate random matrix P
update P with $P_{ij} = \{1 \text{ if } P_{ij} > p \text{ else } 0\}$
$\mathbf{X} = \mathbf{X} \otimes P$
while repeat e times **do**
  generate random matrix $\mathbf{Z} \in \mathbb{R}^{d \times s}$
  randomly pick rows X' from $\mathbf{X}$
  generate mask matrix $\mathbf{M}$
  if first iteration in each pass of the dataset **then**
    use the initial h
  else
    $\mathcal{L}_D$ = discriminator loss of previous iteration
    dynamic hint: update the hint rate $h = \frac{1}{1+e^{-(10 \times \mathcal{L}_D - 5)}}$
  end if
  generate hint matrix $\mathbf{H}$ with value 0 or 1 according to the hint rate h
  $\mathbf{Z} = \mathbf{Z} \times \mathbf{M}$
  $X_G = G(X', \mathbf{Z}, \mathbf{M})$
  $X_G = X_G \otimes (1 - \mathbf{M}) + X'$
  discriminator output $P_D = D(X_G, \mathbf{H})$
  update D with cross entropy loss with P_D
  update G with mean square error loss of $\mathbf{X}$ and X_G
end while
return G

4.1 Hint Mechanisms

We have developed two structures of the hint generator in GAIN for improvement. They are called fake indication and dynamic hint.

4.1.1 *Fake Indication.*

As mentioned in section 2.5.2, only observed data indicators in the mask matrix are affected during the procedure of generating hint matrix. The hint matrix in GAIN is given by:

$$H' = M \otimes H \tag{1}$$

where H' represents the hint matrix, $\otimes$ denotes element-wise multiplication.

To confuse the discriminator and strengthen the training outcome, we use the fake indication to control the uncertainty of hint information, which is implemented by reversing the non-hint part of the mask matrix by the following equation:

$$H' = M \otimes H + (1 - M) \otimes (1 - H) \tag{2}$$

where the notation $\mathbf{1}$ represents a 1s matrix, whose dimension is flexible to ensure the operation is feasible.

Compared with the original hint mechanism in GAIN, H' is generated by not only converting the non-hint observed indication to 0 but also converting the non-hint missing indication to 1. After the conversion, neither the 0-elements nor the 1-elements of the hint matrix can precisely indicate the missing or observed data distribution. Thus, the hint matrix is treated integrally with definite uncertainty controlled by the hint rate. The theoretical analysis of the uncertainties of hint information is presented in Appendix A.1

4.1.2 *Dynamic Hint.*

In GAIN [17], the hint rate used to produce the hint matrix is constant regardless of loss value produced by the discriminator, e.g. discriminator will receive the hint matrix with 90% of hint rate for any discriminator loss value. The downside of the constant hint rate is that the discriminator is not strengthen to learn while its loss value is low, which in turn does not force the generator to learn. Therefore, to force the discriminator to learn, we modified the hint rate from constant to dynamic, i.e. the hint matrix with high hint rate is inputted into discriminator when its loss value is high, on the other hand, the hint matrix with low hint rate should be provided to discriminator when its loss value is low. The discriminator loss value from the previous iteration is used to decide the hint rate of current iteration. The hint rate is computed with modified sigmoid function,

$$\text{hint rate} = \frac{1}{1 + e^{-(a\mathcal{L}_D - 5)}}, \tag{3}$$

where $\mathcal{L}_D$ is the loss value of the discriminator and a is the hyperparameter to control the slope of the modified sigmoid function. Figure 2 shows the plot of equation 3 with different hyperparameter a. The discriminator loss function is non-negative real number and the hint rate is the real number between 0.0 and 1.0. In this research paper, we use $a = 10$.

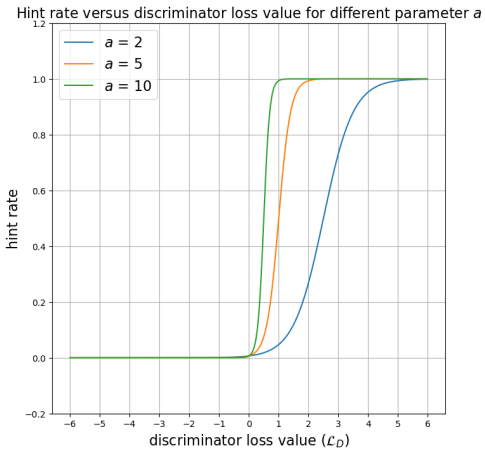


Figure 2: Plot of the hint rate versus the discriminator loss value from equation 3.

5 EXPERIMENTAL SETTING

The implementation of α-GAIN[1] is modified based on GAIN[2].

5.1 Dataset

The complete datasets spam and letter available from [18], and multi-level dataset popularity available from [5, 6], are used to test the performance of the imputed data when applying the the α-GAIN. The summary of the datasets are summarized in Table 1.

The spam and letter are the datasets tested in the GAIN algorithm [17], therefore, we have re-used these datasets for comparison between α-GAIN and GAIN. Both dataset are stored in comma-separated values (csv) format. The spam dataset contains 4601 cases, and each cases stores 57 columns with 2 variables are integer and 55 variables are numerical. Whereas, the letter dataset contains 20000 but with fewer columns which all 16 variables are integer. Dataset popularity is a multi-level dataset. It contains the pupils nested within a class, i.e. there are 2000 pupils in 100 classes, so the average class size is 20 pupils. On the pupil level, we have the outcome variable 'popularity', measured by a self-rating scale that ranges from 0 (very unpopular) to 10 (very popular). We have two explanatory variables on the pupil level: pupil gender (boy=0, girl=1) and pupil extraversion (self-rating scale ranging from 1 to 10). And on class-level explanatory variable teacher experience (ranging from 2 to 25). There are 5 categorical variables, 1 binary variable, and 9 numerical variables in the popularity dataset. However, in this research paper, we treated this multi-level dataset as flat dataset, i.e. single-level dataset. The popularity dataset is stored in sav format of IBM SPSS Statistics [8]. First, we need to open the popularity dataset with IBM SPSS Statistics and then export the dataset in csv format.

5.2 Missing data generation

In this research paper, we only consider the missing data mechanism of MCAR, which mean in the α-GAIN algorithm, the hint matrix is generated randomly according to the hint rate and hint mechanism. To simulate the incompleteness of the dataset, a mask matrix comprised of the missing coded with 0 and the observed coded with 1 is generated randomly, with the same dimension as the dataset in processing. After that, an incomplete dataset is obtained by element-wise multiplying the original dataset and the generated mask matrix.

5.3 Experiments and evaluation criteria

5.3.1 Experiments.

To validate the effects of the improved hint mechanism in the α-GAIN algorithm compared with the performance of GAIN, we set up three groups of experiments.

The potential sources of gain are the use of the dynamic hint (DH) and fake indication (FI) in the α-GAIN framework. In the first group of experiment, in order to understand how each of these affects the performance of α-GAIN, we exclude either FI or DH and compare the performance of the resulting architectures against the full α-GAIN and GAIN. The missing rate of 0.5 and initial hint rate of 0.8 are used in this first group of experiment.

In the second group of experiment, we quantitatively examine imputation performance at various missing rates from 0.1 to 0.9. Finally, in the third group of experiment, we explore the influence of the initial hint rate of 0.1, 0.3, 0.5, 0.7 and 0.9. Initial hint rate means the hint rate at the first iteration sets in the algorithm. The initial hint rate must be given to the hint generator before the calculation of the discriminator loss in the first iteration.

5.3.2 Evaluation criteria.

Root Mean Square Error (RMSE) is used as the numerical criterion to quantify the algorithm performance. The RMSE is defined as follow:

$$RMSE = \sqrt{\sum_{i=1}^{n} \frac{(\hat{y}_i - y_i)^2}{n}} \tag{4}$$

where $\hat{y}_i$ is the value on ith entry in the imputed matrix, y_i is the actual value on ith entry in the complete dataset matrix and n is the total number of entry in the dataset matrix. Additionally, in each group of experiment, we repeat the α-GAIN algorithm 100 times and then report the mean of these RMSEs.

6 RESULTS

6.1 Source of gain

Table 2 shows that the performance of α-GAIN compared with GAIN. Firstly, we compare the performance of α-GAIN when both FI and DH components are activated (fourth row in Table 2). α-GAIN reduces the imputation error by 6.1% and 24.3% for letter and popularity dataset, respectively. The high performance of α-GAIN on popularity dataset might due to the relatively low numbers of row and column in the popularity dataset. For the spam dataset, α-GAIN performs slightly worst than GAIN, with RMSE increases by 7.6%. This might due to relatively high numbers of column with 57 compared with letter and popularity with 16 and 15, respectively. This might increase the difficulty of generator to learn to impute the missing values accurately.

Secondly, we evaluate α-GAIN with only one activated component (FI at second row and DH at third row in Table 2). Here, we notice that α-GAIN does not face with loss of performance. The α-GAIN with FI only (second row in Table 2) performs better on spam dataset with increment performance by 16.2%. Whereas, the α-GAIN with DH only (third row in Table 2) performs better on letter dataset with increment performance by 11.1%.

In this experiment, in general, the α-GAIN, either full architecture or activated with only one component (FI or DH), has substantial improvement over the GAIN.

6.2 α-GAIN in different settings

In this section, we report the result of second and third groups of experiments.

6.2.1 Different missing rate.

Figure 3a and Figure 3b show the performance (RMSE) of GAIN and α-GAIN for different missing rates for two datasets, spam and popularity, respectively. The RMSEs and standard deviations are tabulated in Table 3 and Table 4 in Appendix B.

[1] https://github.com/CTW121/alphaGAIN

[2] https://github.com/dhanajitb/GAIN-Pytorch and https://github.com/jsyoon0823/GAIN

Table 1: Summary of the datasets.

Dataset	rows	columns	dataset type	numbers of column				
				integer	categorical	binary	short-floating	long-floating
spam	4601	57	single-level	2	-	-	55	-
letter	20000	16	single-level	16	-	-	-	-
popularity	2000	15	multi-level	-	5	1	4	5

Table 2: Source of gains in α-GAIN algorithm in comparison with GAIN based on the three datasets at missing rate of 0.5 (mean ± standard deviation of the RMSE (Gain(%))). Low RMSE is good. The bolded RMSE values in the table indicate that the RMSE of α-GAIN are lower than RMSE of GAIN.

	Spam	**Letter**	**Popularity**
GAIN	0.074 ± 0.008	0.190 ± 0.021	0.253 ± 0.022
α-GAIN (FI only)	**0.062** ± 0.005 (+16.2%)	**0.177** ± 0.029 (+6.8%)	**0.233** ± 0.015 (+7.9%)
α-GAIN (DH only)	**0.066** ± 0.004 (+10.8%)	**0.169** ± 0.010 (+11.1%)	**0.229** ± 0.014 (+7.9%)
α-GAIN (FI and DH)	0.079 ± 0.021 (-7.6%)	**0.178** ± 0.015 (+6.1%)	**0.192** ± 0.014 (+24.3%)

FI: fake indication; DH: dynamic hint

Figure 3a shows that there is no clear distinction in term of performance between GAIN and α-GAIN (full architecture or partial architecture) from missing rate 0.1 to 0.5. However, a clear distinction is observed at missing rate 0.5 and higher with α-GAIN (DH only) out-perform the GAIN and α-GAIN (full architecture and activated with only FI). One interesting observation is that the RMSE of the full architecture α-GAIN at missing rate 0.9 is relatively high compare with others, i.e. α-GAIN cannot as easily cope when the missing rate is considerably high. This might due to α-GAIN is over-fitted when both FI and DH are activated for the missing rate of 0.9. Figure 5 shows validation loss of the generator starts to increase after 100 epochs which indicate over-fitting occur during training for missing rate of 0.9. Another reason might be disminished gradient, i.e. the discriminator gets too successful that the generator gradient vanishes and learns nothing.

For Figure 3b, we observed that no clear distinction in term of performance between GAIN and α-GAIN (full architecture or partial architecture) from missing rate 0.1 to 0.5. However, from missing rate 0.5 to 0.8, both α-GAIN and α-GAIN (DH only) outperform the others as in Figure 3a. As We have observed the same phenomenon in Figure 3a, the performance of α-GAIN drops significantly at missing rate 0.9 compared with others, i.e. α-GAIN cannot as easily cope at missing rate 0.9 for large (spam) and small (popularity) datasets. This might due to the aforementioned reasons.

6.2.2 *Different initial hint rate.*

We now compare the performance of the α-GAIN with itself with different initial hint rate settings. Figure 4a shows the performance (RMSE) of the α-GAIN (both FI and DH are activated) for hint rate 0.1, 0.3, 0.5, 0.7 and 0.9 for dataset spam. We observe that the performance of the α-GAIN generally is considerably comparable across entire range of missing rate for these initial hint rates, although we see slightly difference of performance at missing rate 0.5, 0.6, and 0.7. In other words, the setting of initial hint rate does not affect the performance of the α-GAIN, this is because the hint rate will be adjusted accordingly based on the discriminator loss after the first iteration.

We observed the same phenomenon in Figure 4b for dataset popularity, the performance of the α-GAIN is not affected by the initial hint rate, although we can see slightly performance difference at missing rate 0.1, 0.2 and 0.3. Based on the Figure 4a (spam dataset) and Figure 4b (popularity dataset), the setting of the initial hint rate does not seem to affect to datasets of different dimensions. The RMSEs and standard deviations are tabulated in Table 5 and Table 6 in Appendix B.

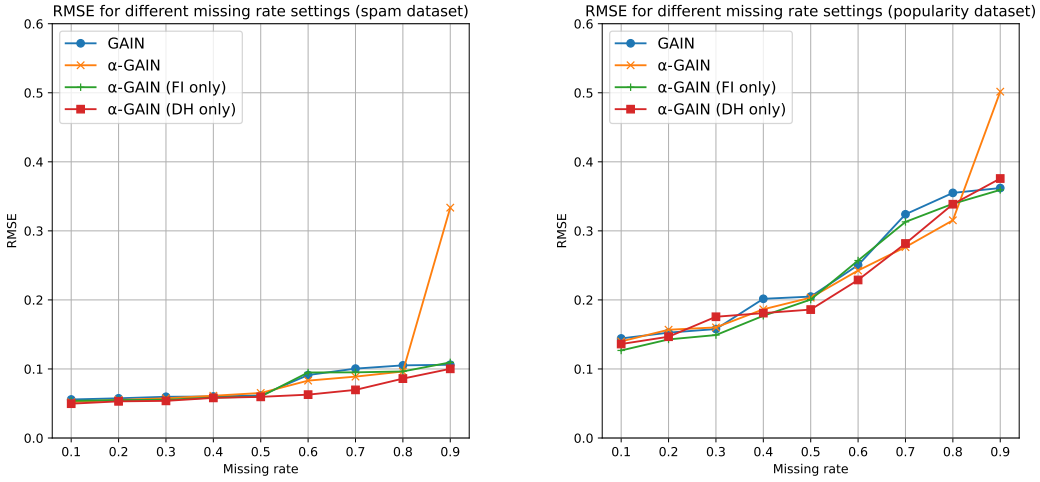
7 DISCUSSION

This section will look into the limitations of our work. The decision of either using the full α-GAIN or activated with only one component (fake indication or dynamic hint) depends on the properties of the dataset, i.e. dimensions of the dataset, integers, long floating-point and short floating-point numbers, etc. The datasets chosen for our study cover from small (popularity dataset) to large (letter dataset), whereas the dimensions of the spam dataset lies between them. From Figure 3a, we observed that the performance of α-GAIN is reduced for spam dataset. This might due to relatively high numbers of column and most of the number in cells in the spam dataset (Table 1) are floating numbers which make α-GAIN difficult to learn for the missing data imputation.

Other limitations of the α-GAIN are over-fitting and disminished gradient. In Figure 3, we observed that the performance reduced significantly for α-GAIN at missing rate 0.9. This might due to over-fitting and disminished gradient. We observed that the validation loss starts to increase from 100 epochs for missing rate of 0.9, as shown in Figure 5, this mean over-fitting starts to occur in the generator. Besides that, disminished gradient might occur if the discriminator gets too successful that the generator gradient vanished and learns nothing. This is unexpected because the performance of α-GAIN (dynamic hint only) out-perform the others, see Figure 3a. This might indicate that current full architecture α-GAIN is not capable to improve the imputation quality and accuracy for relatively high missing rate.

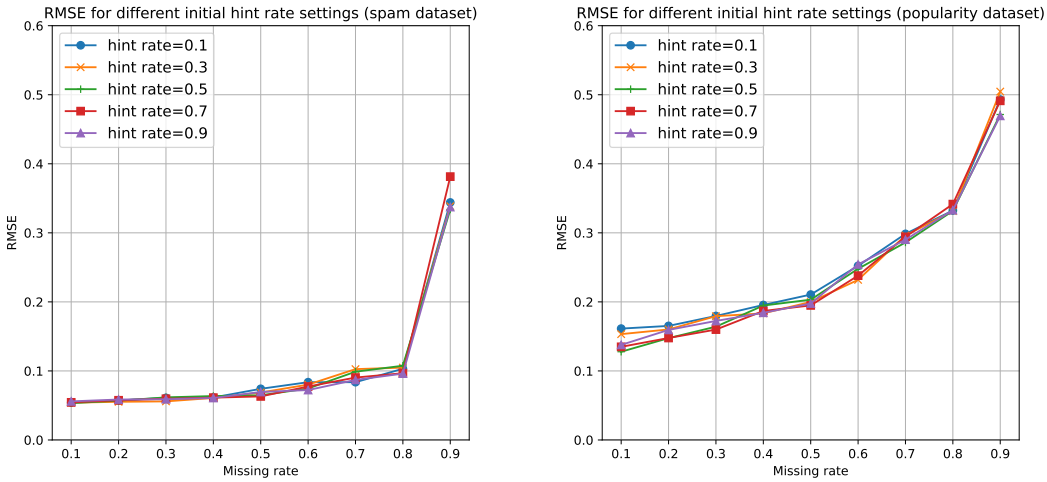
8 CONCLUSION

We purpose a generative model for missing data imputation, which is build upon the hint mechanism of well-known GAIN [17] such that it can deal with missing data imputation with better accuracy



(a) spam dataset **(b) popularity dataset**

Figure 3: Plot of RMSE versus different missing rate for GAIN and α-GAIN. FI: fake indicator; DH: dynamic hint. Low RMSE is good. The hint rate is 0.8 for GAIN and α-GAIN (FI only). The hint rate is dynamically changing for α-GAIN and α-GAIN (DH only). The RMSEs and standard deviations are tabulated in Table 3 and Table 4 in Appendix B.



(a) spam dataset **(b) popularity dataset**

Figure 4: Plot of RMSE versus different different initial hint rate for α-GAIN. Low RMSE is good. The RMSEs and standard deviations are tabulated in Table 5 and Table 6 in Appendix B.

and quality. We have improved the hint mechanism in GAIN in two way: (1) we modify the fake indication in the hint generator while generating the hint matrix (2) and we provide a modified sigmoid function in the hint generator for computing the updated hint rate based on the loss of discriminator. We named our framework as α-GAIN. Various experiments with real-world datasets show that α-GAIN has acceptable improvement over GAIN. Based on the current experiments, depends on the properties of the dataset, i.e. dimensions, integer, short or long floating-number, we recommend that either α-GAIN with activated fake indication or dynamic hint is used for the imputation across the entire range of missing rates.

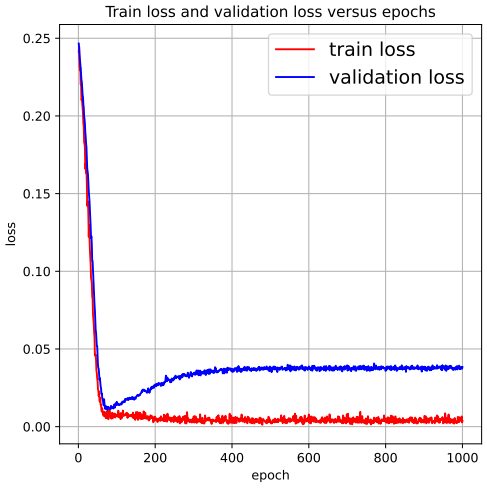


Figure 5: Training and validation losses of the generator versus the epochs for α-GAIN at missing rate of 0.9 for dataset spam.

8.1 Future Work

In this paper, we adapt two ways of improvement of the hint matrix. In the future, we want to test more methods to exploit the hint mechanism better. One of them is to make the hint generator learnable and doing the same gradient as the other part of discriminator. By such improvement, the hint matrix will no longer consists 0 or 1, instead dynamic matrix changing in every epoch. But we still need to explore the initial distribution of the hint matrix as it aims to provide the information to the mask matrix M. In this research study, popularity dataset is multi-level dataset, however, we treat this dataset as single-level dataset. Therefore, another future work is to adapt the α-GAIN for multi-level dataset missing data imputation. Furthermore, in this research study, we only generated the missing data matrix according to MCAR. We plan to improve the α-GAIN for imputing the missing data in MAR and MNAR pattern. Another potential future work is to investigate the hyperparameter a in the modified sigmoid function, Equation 3, i.e. test the performance of the α-GAIN on different hyperparameter a settings. Another future work is to design a better dynamic hint function, besides modified sigmoid function Equation 3, such that α-GAIN has better imputation accuracy and quality for high missing rate.

REFERENCES

[1] Ashish Bora, Eric Price, and Alexandros G. Dimakis. 2018. AmbientGAN: Generative models from lossy measurements. In *International Conference on Learning Representations*. https://openreview.net/forum?id=Hy7fDog0b

[2] Yann N. Dauphin, Harm de Vries, Junyoung Chung, and Yoshua Bengio. 2015. RMSProp and equilibrated adaptive learning rates for non-convex optimization. *CoRR* abs/1502.04390 (2015). http://dblp.uni-trier.de/db/journals/corr/corr1502.html#DauphinVCB15

[3] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative Adversarial Networks. https://doi.org/10.48550/ARXIV.1406.2661

[4] Yulei. He, Guangyu. Zhang, and Chiu-Hsieh. Hsu. 2021. *Multiple Imputation of Missing Data in Practice: Basic Theory and Analysis Strategies*. CRC Press. https://books.google.nl/books?id=0UYEugEACAAJ

[5] Joop. Hox, Mirjam. Moerbeek, and Rens. van de Schoot. 2017. *Multilevel Analysis: Techniques and Applications*. Routledge. https://doi.org/10.4324/9781315650982

[6] Joop. Hox, Mirjam. Moerbeek, and Rens. van de Schoot. 2018. Multilevel Analysis. https://github.com/MultiLevelAnalysis/Datasets-third-edition-Multilevel-book/blob/master/chapter%202/popularity/SPSS/popular2.sav.

[7] Xia Hua, Ting Rui, Xia Cai, Xinqing Wang, Haitao Zhang, and Dong Wang. 2020. Collaborative Generative Adversarial Network with Visual perception and memory reasoning. *Neurocomputing* 414 (2020), 101–119. https://doi.org/10.1016/j.neucom.2020.06.037

[8] IBM. 2022. SPSS Statstics. https://www.ibm.com/products/spss-statistics

[9] Yann LeCun and Corinna Cortes. 2010. MNIST handwritten digit database. http://yann.lecun.com/exdb/mnist/. (2010). http://yann.lecun.com/exdb/mnist/

[10] Steven Cheng-Xian Li, Bo Jiang, and Benjamin M. Marlin. 2019. MisGAN: Learning from Incomplete Data with Generative Adversarial Networks. *CoRR* abs/1902.09599 (2019). arXiv:1902.09599 http://arxiv.org/abs/1902.09599

[11] Roderick J. A. Little and Donald B. Rubin. 2002. *Statistical analysis with missing data*. Wiley. http://books.google.com/books?id=aYPwAAAAMAAJ

[12] John T. McCoy, Steve Kroon, and Lidia Auret. 2018. Variational Autoencoders for Missing Data Imputation with Application to a Simulated Milling Circuit. *IFAC-PapersOnLine* 51, 21 (2018), 141–146. https://doi.org/10.1016/j.ifacol.2018.09.406 5th IFAC Workshop on Mining, Mineral and Metal Processing MMM 2018.

[13] Seunghyoung Ryu, Minsoo Kim, and Hongseok Kim. 2020. Denoising Autoencoder-Based Missing Value Imputation for Smart Meters. *IEEE Access* 8 (2020), 40656–40666. https://doi.org/10.1109/ACCESS.2020.2976500

[14] Qinbao Song and Martin Shepperd. 2007. Missing Data Imputation Techniques. *IJBIDM* 2 (10 2007), 261–291. https://doi.org/10.1504/IJBIDM.2007.015485

[15] Daniel J. Stekhoven and Peter Bühlmann. 2012. MissForest—non-parametric missing value imputation for mixed-type data. *Bioinformatics* 28, 1 (Jan. 2012), 112–118. https://doi.org/10.1093/bioinformatics/btr597

[16] Stef Van Buuren and Karin Groothuis-Oudshoorn. 2011. mice: Multivariate imputation by chained equations in R. *Journal of statistical software* 45 (2011), 1–67.

[17] Jinsung Yoon, James Jordon, and Mihaela van der Schaar. 2018. GAIN: Missing Data Imputation using Generative Adversarial Nets. In *Proceedings of the 35th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 80)*. PMLR, 5689–5698. https://proceedings.mlr.press/v80/yoon18a.html

[18] Jinsung Yoon, James Jordon, and Mihaela van der Schaar. 2018. Generative Adversarial Imputation Networks (GAIN). https://github.com/jsyoon0823/GAIN.git.

[19] Hongyang Zhang, Susu Xu, Jiantao Jiao, Pengtao Xie, Ruslan Salakhutdinov, and Eric P. Xing. 2018. Stackelberg GAN: Towards Provable Minimax Equilibrium via Multi-Generator Architectures. https://doi.org/10.48550/ARXIV.1811.08010

A PROOFS AND THEORETICAL ANALYSIS

A.1 Uncertainties of Hint Information

To investigate the effects of hint generator in original GAIN, we make a definition and a proposition to analyse the information uncertainty mathematically.

Definition A.1. Consider an incomplete dataset in size of r rows and c columns, the number of data is represented by N where $N = r \times c$. Let N_{obs} denote the number of the observed data, N_{mis} denote the number of missing data, thus $N = N_{obs} + N_{mis}$. In the hint matrix which has the same dimension as the dataset, let N_{nh-obs} represent the number of non-hint observed data indicated by 0-elements. $N_{(0)}$ denotes the number of 0-elements which represents missing cells, $N_{(1)}$ denotes the number of 1-elements which represents observed cells, thus $N = N_{(0)} + N_{(1)}$. Let δ denote the uncertainty of the hint matrix, δ_{obs} denote the uncertainty of observed data indication, δ_{mis} denote the uncertainty of missing data indication, since element-wise multiplication only affects observed data indicators represented by 1, then:

$$N_{(0)} = N_{nh-obs} + N_{mis} \tag{5}$$

$$N_{(1)} = N_{obs} - N_{nh-obs} \tag{6}$$

$$\delta = \frac{N_{nh-obs}}{N} \tag{7}$$

$$\delta_{obs} = 0 \tag{8}$$

$$\delta_{mis} = \frac{N_{nh-obs}}{N_{(0)}} = \frac{N_{nh-obs}}{N_{nh-obs} + N_{mis}} \tag{9}$$

It can be witnessed that δ only depends on the proportion of the number of non-hint observed data. δ_{mis} can be determined in the range of $[0, 1]$. δ_{mis} tends to 0 while N_{mis} is much greater than N_{nh-obs}, δ_{mis} is equal to 0 if and only if N_{nh-obs} is equal to 0 and N_{mis} is not 0. On the contrary, δ_{mis} tends to 1 while N_{nh-obs} is much greater than N_{mis}, δ_{mis} is equal to 1 if and only if N_{mis} is equal to 0 and N_{nh-obs} is not 0.

Proposition A.2. *Missing rate r_m is a characteristic of real-world datasets, defined by the proportion of missing data in the dataset. Hint rate r_h is the probability of revealing the datum in each cell, which is used to control the hint information quantitatively. However, the uncertainty δ cannot be determined based on the original GAIN algorithm despite both r_m and r_h are determined. Assume that δ is equal to 0 in case: (i) r_h is greater than $1 - r_m$; (ii) all the observed data are revealed, which means that the cells of 1-elements in the hint matrix cover all the cells of 1-elements in the mask matrix and it leads to the N_{nh-obs} being equal to 0. The probability $P_{(\delta=0)}$ can be calculated as follows.*

$$P_{(\delta=0)} = C_{N_{obs}}^{N_{obs}} r_h^{N_{obs}} (1 - r_h)^0 = r_h^{(1-r_m)N} \tag{10}$$

This is the ideal probability calculation, which supposes that all the data is distributed independently and identically. Nevertheless, limited by the size of the mini-batch and the computer's pseudo-random behaviour, generating an incomplete dataset and a hint matrix in this experiment is more similar to stochastic sampling. The practical probability can be estimated by:

$$\hat{P}_{(\delta=0)} = \frac{C_{r_h N}^{N_{obs}}}{C_N^{N_{obs}}} = \frac{(r_h N)!(r_m N)!}{N!((r_h + r_m - 1)N)!} \tag{11}$$

According to the above equation, the probability $\hat{P}_{(\delta=0)}$ approach to 1 if r_h or r_m close to 1, which infers that the hint matrix in the original GAIN algorithm would be likely to imply the whole mask matrix to the discriminator while the experiment is conducting at the high missing rate or high hint rate.

B RESULTS

This section we tabulated the RMSEs and their standard deviation for Figure 3a, Figure 3b, Figure 4a and Figure 4b in Table 3, Table 4, Table 5 and Table 6, respectively.

Table 3: RMSE (mean ± standard deviation) for different missing rates for spam dataset (Figure 3a).

	missing rate								
model	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
GAIN	0.0558±0.0013	0.0575±0.0022	0.0597±0.0019	0.0601±0.0052	0.0616±0.0005	0.0912±0.0305	0.1005±0.0506	0.1052±0.0433	0.1061±0.0499
α-GAIN	0.0519±0.0023	0.0552±0.0012	0.0575±0.0038	0.0612±0.0073	0.0652±0.0122	0.0831±0.0237	0.0889±0.0376	0.0963±0.0498	0.3336±0.1152
α-GAIN (FI only)	0.0538±0.0017	0.0549±0.0012	0.0554±0.0007	0.0585±0.0021	0.0600±0.0051	0.0951±0.0407	0.0964±0.0361	0.0949±0.0440	0.1095±0.0425
α-GAIN (DH only)	0.0497±0.0012	0.0531±0.0013	0.0538±0.0006	0.0581±0.0011	0.0596±0.0025	0.0628±0.0088	0.0697±0.0163	0.0860±0.0357	0.1002±0.0413

Table 4: RMSE (mean ± standard deviation) for different missing rates for popularity dataset (Figure 3b).

	missing rate								
model	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
GAIN	0.1442±0.0156	0.1526±0.0265	0.1578±0.0105	0.2016±0.0124	0.2048±0.0108	0.2504±0.0183	0.3240±0.0234	0.3550±0.0397	0.3620±0.0340
α-GAIN	0.1399±0.0144	0.1568±0.0104	0.1602±0.0071	0.1864±0.0141	0.2033±0.0171	0.2426±0.0126	0.2765±0.0108	0.3153±0.0167	0.5016±0.0318
α-GAIN (FI only)	0.1268±0.0065	0.1428±0.0168	0.1491±0.0053	0.1770±0.0212	0.2005±0.0125	0.2570±0.0103	0.3130±0.0149	0.3392±0.0231	0.3593±0.0183
α-GAIN (DH only)	0.1361±0.0066	0.1467±0.0109	0.1756±0.0148	0.1809±0.0249	0.1860±0.0064	0.2289±0.0081	0.2817±0.0207	0.3386±0.0212	0.3758±0.0207

Table 5: RMSE (mean ± standard deviation) for different hint rates for spam dataset (Figure 4a).

	missing rate								
hint rate	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
0.1	0.0541±0.0012	0.0573±0.0027	0.0610±0.0063	0.0615±0.0029	0.0741±0.0135	0.0837±0.0230	0.0837±0.0301	0.1031±0.0346	0.3440±0.1288
0.3	0.0540±0.0083	0.0553±0.0014	0.0559±0.0016	0.0609±0.0067	0.0687±0.0110	0.0802±0.0237	0.1026±0.0372	0.1048±0.0484	0.3385±0.1283
0.5	0.0534±0.0028	0.0569±0.0014	0.0617±0.0016	0.0636±0.0124	0.0647±0.0100	0.0746±0.0239	0.0989±0.0476	0.1075±0.0381	0.3338±0.1198
0.7	0.0545±0.0098	0.0575±0.0010	0.0597±0.0078	0.0613±0.0053	0.0631±0.0115	0.0774±0.0233	0.0904±0.0371	0.0967±0.0313	0.3814±0.1405
0.9	0.0559±0.0010	0.0586±0.0049	0.0590±0.0020	0.0608±0.0024	0.0695±0.0184	0.0723±0.0196	0.0870±0.0322	0.0961±0.0360	0.3373±0.1256

Table 6: RMSE (mean ± standard deviation) for different hint rates for popularity dataset (Figure 4b).

	missing rate								
hint rate	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
0.1	0.1613±0.0130	0.1652±0.0149	0.1796±0.0134	0.1955±0.0144	0.2106±0.0142	0.2522±0.0236	0.2985±0.0179	0.3324±0.0149	0.4930±0.0299
0.3	0.1533±0.0124	0.1601±0.0106	0.1793±0.0102	0.1834±0.0077	0.1999±0.0121	0.2319±0.0159	0.2962±0.0133	0.3315±0.0138	0.5046±0.0275
0.5	0.1278±0.0116	0.1475±0.0102	0.1641±0.0079	0.1947±0.0222	0.2032±0.0105	0.2479±0.0071	0.2862±0.0167	0.3318±0.0151	0.4714±0.0305
0.7	0.1349±0.0167	0.1477±0.0163	0.1599±0.0123	0.1868±0.0109	0.1949±0.0179	0.2380±0.0099	0.2942±0.0180	0.3416±0.0113	0.4912±0.0289
0.9	0.1377±0.0112	0.1594±0.0119	0.1724±0.0145	0.1842±0.0114	0.1976±0.0128	0.2542±0.0074	0.2901±0.0205	0.3328±0.0101	0.4694±0.0197