

GAMIN: Generative Adversarial Multiple Imputation Network for Highly Missing Data

Seongwook Yoon
Korea University
Seoul, Republic of Korea
swyoon@mpeg. korea. ac. kr

Sanghoon Sull
Korea University
Seoul, Republic of Korea
sul1@korea. ac. kr

Abstract

We propose a novel imputation method for highly missing data. Though most existing imputation methods focus on moderate missing rate, imputation for high missing rate over 80% is still important but challenging. As we expect that multiple imputation is indispensable for high missing rate, we propose a generative adversarial multiple imputation network (GAMIN) based on generative adversarial network (GAN) for multiple imputation. Compared with similar imputation methods adopting GAN, our method has three novel contributions: 1) We propose a novel imputation architecture which generates candidates of imputation. 2) We present a confidence prediction method to perform reliable multiple imputation. 3) We realize them with GAMIN and train it using novel loss functions based on the confidence. We synthesized highly missing datasets using MNIST and CelebA to perform various experiments. The results show that our method outperforms baseline methods at high missing rate from 80% to 95%.

1. Introduction

In many cases of measurements, data is obtained as missing data which is partially observed due to various reasons. To deal with such missing data, imputation transforms missing data into complete data to further process it without discarding missing values. There have been many imputation methods and those are still being studied, recently using deep learning in particular.

There are various straightforward methods such as zero imputation, mean imputation and deck imputation. Zero imputation substitutes each missing value with zero and mean imputation substitutes each missing value with the mean of observed values. Deck imputation performs selection from the stack of candidate imputations [1]. One of the most promising non-straightforward approaches is regression imputation which estimates missing values based on

observed values. Recently, most of imputation methods use regression imputation because it can appropriately handle multivariate missing data and learn the statistics of missing values.

Imputation methods can be divided into single imputation and multiple imputation methods. The single imputation method suggests one complete data per one missing data. On the other hand, the multiple imputation method suggests a set of multiple complete data per one missing data. In this case, each imputation is processed independently, and the corresponding results are consolidated into one result [2].

Learning from dataset is one of the most important part of regression imputation methods [3, 4, 5, 6]. In deep learning literature, there are two major approaches to learn imputation from large dataset. First, deep autoencoders are used in supervised learning from complete dataset [7], excluding the unsupervised version [8]. Second, based on the recently popular generative adversarial network (GAN) [9], a generator imputes missing data and a discriminator predicts missingness from the imputation in [10]. In the imputation method of [11], a generator imputes missing data to deceive the discriminator which discriminates between the imputation and fake complete data generated by another GAN, called misGAN. Likewise, [12] performed multivariate time series missing data imputation using a discriminator which discriminates between the imputation and real missing data.

In this paper, we focus on imputation of highly missing data which is a challenging problem. Depending on how much a given dataset is structured or distributed, multiple imputation is desirable for highly missing dataset, because high missing rate makes the distribution of the missing values multi-modal. For example, the digits of MNIST [13] can be hardly recognized at 90% missing so that even a human can be confused between several candidate digits.

We also adopted GAN for missing data imputation. While the imputation architectures of [10] and [11] are identical, we propose a novel imputation architecture which co-

operates with a generator. While [11] also have another generator to produce fake complete data at training procedure, the generator of our imputation architecture operates also at test procedure. This architecture motivates a novel confidence prediction method. Therefore, our method has three major proposals: 1) Novel imputation architecture, 2) confidence prediction based on the imputation architecture, and 3) learning and inference methods for our multiple imputation network, GAMIN. To clarify, these three are designed for unsupervised learning of multiple imputation for highly missing dataset.

In Section 2, we describe the details of related work, especially for [10] and [11]. In Section 3, we formulate our problem. In Section 4, we analyze missingness and imputation architecture in general. In Section 5, we describe the details of GAMIN including our imputation architecture and confidence prediction. Finally, in Section 6, we show the experimental results for MNIST [13] and CelebA [14] datasets.

2. Related work

In this section, we briefly describe GAIN [10] and misGAN [11]. The imputation architectures of two methods are identical, but the learning procedure using GAN is different. In GAIN [10], the imputer is considered as a generator and the discriminator tries to discriminate whether each component of an input is imputed or not. A hint mechanism is utilized to adjust the difficulty of the discrimination based on the theoretical analysis for MCAR case. The algorithm works well on low dimensional datasets with low missing rate. Though it also works at MNIST of 50% missing rate, it converges to zero imputation or mean imputation for highly missing datasets. Also, multiple imputations sampled by repeating diverse latent variables are not various enough to predict the multi-modality of imputation well.

On the other hand, the imputation method in [11] works better for highly missing datasets. The paper proposed a GAN architecture for a missing dataset, called as misGAN, and an imputation method using it. The generator of misGAN consists of a data generator which generates fake complete data and a mask generator which generates the fake mask indicating which component is missed. Then, the fake missing data is generated by masking the fake complete data with the fake mask. To train those generators, a discriminator tries to discriminate between fake missing data and real missing data. For imputation, another pair of generator and discriminator are used. The generator for imputation imputes missing data to deceive the corresponding discriminator which discriminates between imputation and fake complete data.

3. Problem formulation

In this section, we will describe the formulation of our problem and its scope. First, we adopt Rubin's formulation for missingness [2]. As we deal with d dimensional real-valued data vectors, a complete data \mathbf{x} is a d dimensional random vector drawn from a distribution $P(\mathbf{x})$. A missing data vector \mathbf{x}_m can be represented using the missing mechanism of conditional distribution $P(\mathbf{x}_m|\mathbf{x})$.

Our dataset is assumed to be the collection of i.i.d. samples driven from the distribution $P(\mathbf{x}_m|\mathbf{x})P(\mathbf{x})$. However, as we are interested in unsupervised learning, training dataset $\mathbf{D} = \{\mathbf{x}_m^i\}_{i=1}^N$ is the collection only of the missing data vectors. Also, it should be indicated which components are missing. Furthermore, some characteristics of missingness such as missing rate are assumed to be known to adjust hyper-parameters or architectures.

If an imputation method is stochastic, multiple imputations can be obtained simply by repeating it. However, we want to consider the confidence of each imputation, because some of stochastic imputations are not good imputations. Then, to conform the Rubin's rule [2] which treats each imputation equivalently, we can select top- k imputations by ordering stochastic imputations according to the confidence. Therefore, our imputation method with the confidence prediction can be treated equally as a general multiple imputation method. We need to model not only the stochastic imputation $P(\mathbf{x}|\mathbf{x}_m)$ but also the top- k imputations $P(\{\mathbf{x}_{t(i)}\}_{i=1}^k|\{\mathbf{x}_i\}_{i=1}^s)$, where \mathbf{x}_i is the i -th stochastic imputation of total s stochastic imputations and $t(i)$ is the index of the selected top- i imputation.

4. Models for missingness and imputation

In this section, we describe a general models for missingness and imputation using the concept of substitution.

4.1. Model for missingness

Figure 1. Missing data \mathbf{x}_m results from the missing progress that masks complete data \mathbf{x} with mask \mathbf{m} and substitution \mathbf{y} using the masking function M .

As formulated in [2], a missing data vector can be represented with the substitution of the missing components as shown in Figure 1. There are two ways of indicating which

component is missing from the two outputs of the model. We can simply use the binary mask m to indicate missingness, or we need to carefully select the substitution vector y of which field Y is not a subset of the field of data vector R to imply missingness without the mask. For both cases, the missing data can be represented using the masking function M as below:

$$x_m = M(x, m, y)^{(i)} = \begin{cases} x^{(i)} & \text{if } m^{(i)} = 1 \\ y^{(i)} & \text{if } m^{(i)} = 0 \end{cases} \quad (1)$$

where $x^{(i)}$, $m^{(i)}$, $y^{(i)}$, and $M(x, m, y)^{(i)}$ are the i -th components of x , m , y , and $M(x, m, y)$, respectively. Note that we use the extended form of masking function which broadcasts scalar inputs for x and y . For example, $M(x, m, 0)$ is equal to $M(x, m, \mathbf{0})$, where $\mathbf{0}$ is a d dimensional zero vector.

There are three types of missingness: Missing completely at random (MCAR), missing at random (MAR) and missing not at random (MNAR). In this paper, we focus on MCAR case which can be represented as $P(m|x) = P(m)$, because it can be said that highly missing dataset is mostly caused by either accidental exterior factors or control of measurement system before the actual measurement. Both cases imply that the complete data is independent with the missingness.

4.2. Model for imputation

Figure 2. After the missing data is substituted with another substitution y_{im} , an imputer computes imputation \hat{x} from the substituted data \tilde{x} and the mask m .

As the outputs of the missing model are necessary to indicate the missing components, the input of the imputer also needs to indicate the missing components. However, if the substitution field Y is not a real number, such as 'nan', to indicate missing components, there is no proper arithmetic operation to compute imputation. Thus, an imputation method should replace the substituted components with another substitution again as below:

$$\tilde{x} = M(x_m, m, y_{im}), \quad (2)$$

where y_{im} is another substitution for the imputation of which field is equal to that of data and \tilde{x} is the substituted data. Since the missingness cannot be identified after the substitution, we need to input the mask m as well as the substituted data \tilde{x} .

There can be various substitution methods and each method can be interpreted as one of simple prior imputation methods. For example, substituting with zero corresponds to a zero imputation method and substituting with mean corresponds to a mean imputation method. The uniform random substitution method used in [10] and [11] can be interpreted as imputation based on uninformative prior knowledge. Based on this interpretation, our method is motivated to select the substitution y that is a good candidate of imputation. Note that the substitution should be random to perform stochastic imputation.

Figure 3. The imputer processes the conditional generation \tilde{x} into the imputation \hat{x} to preserve observed components. The conditional generation is typically done without the mask.

Furthermore, the imputer has a constraint that the values of observed components should be preserved through imputation. Thus, another substitution is again used to make the imputer satisfy the constraint as below:

$$\hat{x} = M(\tilde{x}, m, \tilde{x}) = M(x, m, \tilde{x}), \quad (3)$$

where \tilde{x} is conditional generation in the imputer and \hat{x} is the imputation as the final output of the imputer.

The rest of the imputer actually regresses the missing values, as shown in Figure 3. As the regression is done without the mask, the regression behaves like a general conditional generator that generates imputation given the substituted data \tilde{x} . The reason why it needs to be done without the mask is because of observation preserving objective for unsupervised learning. The objective is to minimize the difference between the regressed values of the observed components and their actual observed values as below:

$$L_{ob}(\tilde{x}) = d(M(x_m, m, \mathbf{0}), M(\tilde{x}, m, \mathbf{0})), \quad (4)$$

where L_{ob} represents an observation identity loss function for the objective and d is a difference function between two d dimensional vectors. If we input the mask, however, the objective can be satisfied too easily without learning features about missingness.

Though the observation preserving objective works well at low missing rate, an additional adversarial objective [11] is additionally needed. The objective is to deceive a discriminator by making the imputation \hat{x} realistic as below:

$$L_{adv}(\hat{x}; D_I) = L_{adv}(M(\tilde{x}, m, \tilde{x}); D_I), \quad (5)$$

where L_{adv} is the adversarial loss for imputation discriminator D_1 . Thus, the conditional generator tries to translate missing components suitable to observed components, which implies imputation. In [11], even without the observation preserving objective, the conditional generator learns imputation well.

4.3. Performance metric for multiple imputation

For multiple imputation, performance metric would follow Rubin’s rules [2]. Then, we need to calculate the average RMSE of each imputation. However, RMSE is insufficient performance metric for the case of high missing rate. For example, simple imputation methods such as mean imputation outperform most sophisticated imputation methods with respect to RMSE. However, actual imputation examples of sophisticated ones are noticeably better than those of simple ones. To reduce the inconsistency between RMSE and actual results, [11] introduced Fréchet Inception Distance (FID) [15]. However, although it can measure the quality and diversity of imputations themselves, it ignores the match between missing data and imputation. For example, even if the imputations are randomly permuted, FID score will not change. Thus, checking FID is useful but is not sufficient to evaluate the performance of imputation.

An alternative of RMSE is the generalized energy distance [16] which leverages distances between multiple imputations. This is useful for the dataset with multiple groundtruths. However, if we synthesize the missing dataset by simulating a missing mechanism, the dataset would consist of pairs of missing data and its single groundtruth complete data. Instead, we introduce new performance metric, top-k RMSE, which calculates the average of minimum RMSEs among multiple imputations. This metric checks whether at least one of multiple imputation are matched with groundtruth. In addition, we also need to consider about the purpose of imputation. For example, if MNIST is used for classification, we check the classification accuracy after imputation. In this case, instead of Rubin’s rule, we can use top-k classification accuracy.

5. Proposed network: GAMIN

Although our method is motivated by the imputation method of [11], there are three major changes in addition to several minor changes. Firstly, we changed the imputation architecture to make unconditional generator be directly involved in imputation process. Secondly, we propose a novel confidence prediction method and top-k imputation method. Finally, we trained GAMIN using new loss functions considering the confidence. Additionally, we removed the mask generation and used the mask of missing data itself. Also, we propose conditional generation for fake complete data which is more efficient for imputation.

5.1. Imputation architecture

Figure 4. Our imputation architecture with visual examples. Prior imputation: Missing data x_m is substituted with a candidate imputation g generated by the unconditional generator. Conditional generation: The prior imputation \tilde{x} is translated to a more probable imputation. Observation preservation: Imputation \hat{x} gathers missing components from the conditional generation \tilde{x} and observed components from the prior imputation \tilde{x} . Examples for MNIST dataset: Note that the missing values are colored red and we did not pick one of the best results.

The imputation architecture of GAMIN is shown in Figure 4. the first substitution vector g is the unconditional generation of fake complete data. Thus, as we can consider it as a candidate imputation, we can also treat the substituted vector \tilde{x} as a prior imputation. The more probable the candidate g is, the more easily the conditional generator perform imputation. Also, it can be said that the candidate g which is a fake complete data is more probable than the substitution vectors sampled from uninformative uniform distribution of other substitution methods.

However, there is a case such that the candidate g is well-structured complete data but more different from the true imputation than an uniformly generated sample. When missing rate is sufficiently low, it is easy for the conditional generator to impute the prior imputation \tilde{x} well by simple refinement. However, the desired behavior of the conditional generator in such case is somehow complex. So, it becomes hard for the conditional generator to perform imputation with a relatively bad prior imputation most of which components are from the bad candidate.

In short, a major problem of our imputation architecture is the behavior of the conditional generator with bad prior imputation. In order to make the learning procedure robust and adaptive to this problem, we propose to predict confidence of each prior imputation and design both inference and learning methods using the confidence.

5.2. Confidence prediction

Our imputation architecture has a good property for the ideal case: The better prior imputation is inputted, the less the imputer works to change it. For example, if a true imputation is inputted by chance, the imputer does not need to do anything. So, the motivation of our confidence prediction is that the confidence can be measured by how much the imputer changes the prior imputation \tilde{x} .

As the distribution of the candidate g is clustered, the amount of work can also be clustered. In MNIST [13], for example, if a missing data and a candidate belong to the same digit or similar digits like 7 and 9, respectively, the imputer may change the input less.

We define the amount of work simply as the difference between the input and the output of the imputer. As the imputer consists of a conditional generator and a masking function, we need to measure how much they work, respectively, as below:

$$W = d(\tilde{x}, \check{x}) + d(\check{x}, \hat{x}), \quad (6)$$

where α and β is positive real numbers. The first term is the difference between the input and output of the conditional generator and the second term is the difference between those of substitution preserving the observation in Figure 4. If the difference can be represented with component-wise sum such as absolute error or squared error, the two terms can be rewritten as below:

$$\begin{aligned} d(\tilde{x}, \check{x}) &= d(M(x_m, m, g), \check{x}) = d_o(x_m, \check{x}) + d_m(g, \check{x}), \\ d(\check{x}, \hat{x}) &= d(\check{x}, M(x_m, m, \check{x})) = d_o(\check{x}, x_m). \end{aligned} \quad (7)$$

where d_m is the difference at missing components and d_o is the difference at observed components. Thus, the amount of work W is rewritten as below:

$$W = d_m(g, \check{x}) + (\alpha + \beta) d_o(x_m, \check{x}). \quad (8)$$

The first term is difference between missing components of prior imputation and those of conditional generation. This can be interpreted as how much the conditional generator believes the prior imputation. The second term is exactly the same as the observation identity loss Eq. (4), which means partial clue of true imputation performance. In short, the amount of work can estimate the behaviour and true imputation error of the conditional generator to predict confidence.

As mentioned above, we need to repeat the imputation and confidence prediction to perform multiple imputation. So, the confidence prediction must consider the set of multiple imputations to determine confidence relatively. The confidence for a given pair of a missing data and its candidate can be modeled as follows:

$$C(\tilde{x}; x_m, g) = \frac{1}{Z(x_m)} \exp[-W], \quad (9)$$

where α is a positive real number and $Z(x_m)$ is a normalization term for given missing data. The reason for the exponential form is to model the Gaussian distribution of the imputation \hat{x} given the closest true imputation x as below:

$$\begin{aligned} C(\tilde{x}; x_m, g) &= P(\tilde{x}|x_m, g) \\ &= \prod_i P(\tilde{x}^i|x^{(i)}, x_m, g) P(x^{(i)}|x_m, g) \\ &= \text{Norm}_{\tilde{x}}[x, \cdot] P(x|x_m) \\ &= \exp[-d(\tilde{x}, x)], \end{aligned} \quad (10)$$

assuming that the missing data x_m and the candidate g are independent and i -th other true imputation $P(x^{(i)}|x_m)$ is equal for all i , because Rubin's rule implies that true multiple imputations are equivalent. As mentioned above, the true imputation error $d(\tilde{x}, x)$ can be approximated using the amount of work W .

We need to choose the parameter α and the normalization term. Instead, in test procedure, top-k imputations can be selected by ranking w.r.t the amount of work. In training procedure, however, the selection should be soft and the actual value is needed for loss calculation. Assuming that the number of stochastic imputations s is enough for all of stochastic imputations to be included, the normalization term can be approximated as below:

$$Z(x_m) = \sum_{i=1}^s \exp[-W_i], \quad (11)$$

where W_i is the amount of work of i -th stochastic imputation. Then, the confidence becomes the softmax function of $-W_i$ s. Thus, if we want to emphasize the difference, the absolute differences of $-W_i$ s should be larger than 1.

5.3. Loss function using confidence

We designed two types of loss functions: First, we weight the observation identity loss of Eq. (4) and the adversarial loss Eq. (5) using the confidence in Eq. (9) as below:

$$L_{ob}^c = \sum_{i=1}^s C(\tilde{x}_i) L_{ob}(\tilde{x}_i) = \sum_{i=1}^s C(\tilde{x}_i) d_o(x_m, \tilde{x}_i), \quad (12)$$

$$L_{adv}^c = \sum_{i=1}^s C(\tilde{x}_i) L_{adv}(\tilde{x}_i; D_1), \quad (13)$$

where \tilde{x}_i and \check{x}_i is the imputation and conditional generation of i -th repetition, respectively.

Second, we need to maximize the confidence of good imputations. Instead of directly maximizing the confidence of good imputations, we can weight the amount of work. In this respect, Eq. (12) already maximizes the confidence

from the second term of the amount of work defined in Eq. (8). Thus, the loss function for the first term of the amount of work is as follows:

$$L_c = \sum_{i=1}^s C(\tilde{x}_i) d_m(g_i, \tilde{x}_i). \quad (14)$$

The overall imputation losses are as below:

$$L_{con} = L_{ob}^c + \lambda_1 L_{adv}^c + \lambda_2 L_c, \quad (15)$$

$$L_{unc} = \lambda_3 L_{ob}^c + \lambda_4 L_c, \quad (16)$$

where L_{con} and L_{unc} is the imputation loss functions for the conditional generator and unconditional generator, respectively.

5.4 Discriminators

Figure 5. Two discriminators in GAMIN: D_G discriminates between fake complete data and missing data after masking with a constant α . D_I discriminates between imputation and fake complete data. The two bidirectional arrows indicate that these discrimination are independent and switchable.

Like [11], there are two discriminators used to train our model as shown in Figure 5. The generation discriminator D_G criticizes the unconditional generator for fake complete data generation which is also used at the candidate imputation. It discriminates between the masked missing data $M(x_m, m, \alpha)$ and the masked generations $M(g, m, \alpha)$. The substitution constant α should be chosen carefully to imply missingness, even though it cannot indicate actual missing components. Thus, there should be another generation loss function for the unconditional generator in addition to Eq. (16) as below:

$$L_g = \sum_{i=1}^s L_{adv}(g_i; D_G). \quad (17)$$

The imputation discriminator D_I criticizes both generator and imputer. It discriminates between the imputation \tilde{x} and fake complete data g . The actual form of adversarial loss L_{adv} is determined according to the type of GAN structure.

5.5 Conditional substitution

Indeed, the unconditional generation has some practical limitations w.r.t the quality of generation. Also, especially for using the unconditional generation as a candidate imputation, diversity should be adjusted properly to make a good candidate frequent. We can solve this problem if we replace the unconditional generation with the conditional generation using the missing data. Then, both generators become conditional generators but the conditional generator for the candidate imputation learns not to impute but should learn more qualified and related candidates. However, as the candidate imputation should be stochastic, we need to input not only the missing data x_m but also random latent variables. Therefore, the input of the conditional generator z_x can be as below:

$$z_x = h(M(x_m, m, z)), \quad (18)$$

where h is an additional function to adjust randomness, such as a dimension reduction function.

6. Experimental results

In this section, we describe synthesizing missing datasets, implementation details, and our experimental results. Additionally, we show ablation studies about several variations of our method.

We evaluated our method mainly on MNIST and additionally on CelebA. First, we need to synthesize probable types of missingness among MCAR case for MNIST. Like [11], we simulated the independent dropout missing and the square observation. MNIST dropout missing datasets are synthesized at 80%, 85%, 90% and 95% missing rates, respectively. For square observation, we synthesized the squares of 11x11 (84.57%) and 9x9 (89.67%), respectively. On the other hand, we maintained the color of each pixel by the pixel-wise dropout missing for CelebA.

We chose the hyper-parameters for the confidence prediction as $\lambda_1=1$, $\lambda_2=1$ and $\lambda_3=100$. Also, we set the hyper-parameters for loss functions to $\lambda_1=0.2$, $\lambda_2=0.1$, $\lambda_3=0.1$ and $\lambda_4=0.1$. We repeated stochastic imputation ten times ($s=10$) in mini-batch training (batch size is 5) and selected the top-3 imputations from twenty stochastic imputations in test time ($s=20$). We implemented the two baseline methods, GAIN cite09 and misGAN cite10 based on their available codes, and randomly sampled three multiple imputation results.

We used three performance metrics: RMSE, top-k MSE and top-k classification accuracy. As mentioned earlier, RMSE is an average of RMSEs of each multiple imputation following Rubin's rule and top-k RMSE is calculated as an average of minimum RMSEs of each multiple imputation.

Figure 6. Imputation results of 90% dropout missing: (a) Input of which missing components are colored red, (b) groundtruth, (c) zero-imputation, (d) mean-imputation, (e) multiple imputations of GAIN [10], (f) multiple imputations of misGAN [11], and (g) top-3 imputation of GAMIN.

6.1. MNIST missing dataset

We used a 128 dimensional latent variable and 256-512 fully connected (fc) layers for the unconditional generator, 784-784 fc layers for the conditional one, and 512-256 fc layers for two discriminators. We used least square adversarial loss for 0 and 1. Our network was trained by Adam optimizer with learning rate 105 for 150 epochs. Figure 6 shows the imputation results of various imputation methods at dropout missing of 90%. Clearly, the zero imputation method and the mean imputation method cannot produce meaningful imputations. GAIN [10] produces several noisy imputations or mean-like imputations. On the other hand, misGAN [11] produces more clean images but it tends to neglect correspondence between imputations and groundtruth.

Missing rate	80%	85%	90%	95%
GAIN [10]	0.250 (0.239)	0.266 (0.253)	0.281 (0.269)	0.307 (0.304)
misGAN [11]	0.255 (0.233)	0.282 (0.256)	0.324 (0.290)	0.368 (0.343)
Ours	0.224 (0.209)	0.251 (0.234)	0.272 (0.252)	0.306 (0.287)

Table 1. RMSE results of MNIST dropout missing

To investigate performance quantitatively, Table 1 shows the results of RMSE and top-3 RMSE. We represented those two values as 'RMSE (top-3 RMSE)' where the values were calculated on MNIST images normalized from 0 to 1. Note that the uncertainties of RMSE, which is the variance of

three RMSEs, are omitted because it is too small (<0.005) due to the large size of the dataset. As shown in Table 1, our method outperforms other two methods. Table 2 shows the results of top-k classification by checking if at least one of classification of k multiple imputations is correct. We trained LeNet [17] on complete MNIST dataset and the classification accuracy is over 99%.

Missing rate	80%	85%	90%	95%
GAIN [10]	43.5 58.5 66.3	31.1 45.2 52.8	19.6 28.8 34.2	14.8 21.1 24.9
misGAN [11]	61.5 78.5 86.3	45.7 64.5 74.3	29.1 46.1 56.9	11.6 20.8 27.5
Ours	70.7 82.6 87.8	57.3 72.5 80.0	42.4 57.9 66.9	23.6 36.7 46.4

Table 2. Classification results of MNIST dropout missing: The top-1, top-2 and top-3 accuracy are listed in order.

Figure 7 shows the imputation results of various imputation methods at square observation of approximately 90% missing rate. In this case, GAIN produces nearly zero-like imputations and misGAN produces less various image than dropout missing case. Our method also shows some failures, but it produces more diverse imputations.

Figure 7. Imputation results of 9x9 square observation: (a) Input of which missing components are colored red, (b) groundtruth, (c) zero-imputation, (d) mean-imputation, (e) multiple imputation of GAIN [10], (f) multiple imputation of misGAN [11], and (g) top-3 imputation of GAMIN.

Table 3 shows the quantitative results of MNIST square observation dataset. Since GAIN [10] converges to zero imputation, we did not show its results. While misGAN [11] works better at the square observation than at the dropout

missing, our method works better at the dropout missing for the same missing rates.

Square size	11	9
misGAN [11]	0.257 (0.243) 55.9 / 62.6 / 66.2	0.307 (0.294) 25.9 / 34.3 / 38.3
Ours	0.253 (0.229) 49.8 / 62.9 / 70.4	0.286 (0.264) 33.1 / 44.9 / 53.2

Table 3. Results of MNIST square observation dataset

6.2. CelebA missing dataset

Figure 8. Imputation results: (a) Input of which missing components are colored black, (b) groundtruth, (c) multiple imputation of misGAN [11], and (d) top-3 imputation of GAMIN.

We used the U-net [18] for the conditional generator and convolutional neural networks for unconditional generator and discriminators, which are used in misGAN [11]. Also, for fair comparison, we used Wasserstein adversarial loss and all the hyperparameters of [11]. Figure 8 and Table 4 show the imputation results and quantitative results, respectively, for dropout missing CelebA of 90%. Both results show that our method also works better for CelebA dataset.

misGAN [11]	0.129 (0.121)
Ours	0.115 (0.108)

Table 4. Results of CelebA missing dataset of 90%

6.3. Ablation study

In order to investigate the effect of individual parts of our method, we performed experiments for some variations of our method. In the first variation represented as ‘w/o subs’, we replaced our substitution method with the uniform substitution method of GAIN [10] and misGAN [11]. In the second variation represented as ‘w/o loss’, we removed the losses about the confidence. However, we applied the test-time confidence prediction to make top-3 imputation for all of variations except the variation ‘w/o

conf’. Finally, we performed experiments about variation ‘w/ cond’ applying the conditional substitution method in Section 5.5. Table 4 shows the results of these experiments at MNIST dropout missing of 90%. The conditional substitution method enhances the performance for all metrics. We also observed that the confidence prediction greatly improves performance. However, two variations without our substitution method and the loss function produce good performance at classification accuracy, but they produce less diverse imputations than the proposed method.

metric	RMSE	Class accuracy
w/o subs	0.280 (0.271)	47.7 / 62.8 / 70.6
w/o loss	0.275 (0.256)	42.4 / 58.4 / 67.3
w/o conf	0.304 (0.274)	22.6 / 37.8 / 48.9
w/ cond	0.270 (0.251)	46.1 / 62.6 / 71.6
Ours	0.272 (0.252)	42.4 / 57.9 / 66.9

Table 5. Results of variations at 90% MNIST dropout missing dataset

Though we focus on multiple imputation at high missing rate, our method can also be applied at relatively low missing rate. Thus, we synthesized a 50% MNIST dropout missing dataset. Since the unimodality of the dataset might reduce the benefit of our method, Table 6 shows that our method yields slightly lower performance with respect to RMSE result. However, the other metrics and actual imputation results are better or equal to those of GAIN [10] and misGAN [11].

GAIN [10]	0.130 (0.122) 94.4 / 97.1 / 98.0
misGAN [11]	0.136 (0.125) 95.0 / 97.4 / 98.1
Ours	0.140 (0.130) 95.5 / 97.5 / 98.2

Table 6. Results of 50% MNIST dropout missing dataset

7. Future work

We proposed GAMIN which learns multiple imputation by confidence prediction using a novel imputation method. We evaluated our method with highly missing image datasets which are sufficiently structured. However, we need to improve our method for low structured or complex missingness. Furthermore, we plan to make an additional absolute confidence prediction to enhance the reliability of our method.

Acknowledgement

This work was partly supported by the Samsung Electronics.

References

- [1] Rebecca R Andridge and Roderick JA Little. A review of hot deck imputation for survey non-response. *International statistical review*, 78(1):40–64, 2010.
- [2] Roderick JA Little and Donald B Rubin. *Statistical analysis with missing data*, volume 793. John Wiley & Sons, 2019.
- [3] Melissa J Azur, Elizabeth A Stuart, Constantine Frangakis, and Philip J Leaf. Multiple imputation by chained equations: what is it and how does it work? *International journal of methods in psychiatric research*, 20(1):40–49, 2011.
- [4] Daniel J Stekhoven and Peter Bühlmann. Missforest—non-parametric missing value imputation for mixed-type data. *Bioinformatics*, 28(1):112–118, 2011.
- [5] Rahul Mazumder, Trevor Hastie, and Robert Tibshirani. Spectral regularization algorithms for learning large incomplete matrices. *Journal of machine learning research*, 11(Aug):2287–2322, 2010.
- [6] Pedro J García-Laencina, José-Luis Sancho-Gómez, and Aníbal R Figueiras-Vidal. Pattern classification with missing data: a review. *Neural Computing and Applications*, 19(2):263–282, 2010.
- [7] Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th international conference on Machine learning*, pages 1096–1103. ACM, 2008.
- [8] Lovedeep Gondara and Ke Wang. Multiple imputation using deep denoising autoencoders. *arXiv preprint arXiv:1705.02737*, 2017.
- [9] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.
- [10] Jinsung Yoon, James Jordon, and Mihaela Van Der Schaar. Gain: Missing data imputation using generative adversarial nets. *arXiv preprint arXiv:1806.02920*, 2018.
- [11] Steven Cheng-Xian Li, Bo Jiang, and Benjamin Marlin. Misan: Learning from incomplete data with generative adversarial networks. *arXiv preprint arXiv:1902.09599*, 2019.
- [12] Yonghong Luo, Xiangrui Cai, Ying Zhang, Jun Xu, et al. Multivariate time series imputation with generative adversarial networks. In *Advances in Neural Information Processing Systems*, pages 1596–1607, 2018.
- [13] Yann LeCun and Corinna Cortes. MNIST handwritten digit database. 2010.
- [14] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Large-scale celebfaces attributes (celeba) dataset.
- [15] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *Advances in Neural Information Processing Systems*, pages 6626–6637, 2017.
- [16] Simon Kohl, Bernardino Romera-Paredes, Clemens Meyer, Jeffrey De Fauw, Joseph R Ledsam, Klaus Maier-Hein, SM Ali Eslami, Danilo Jimenez Rezende, and Olaf Ronneberger. A probabilistic u-net for segmentation of ambiguous images. In *Advances in Neural Information Processing Systems*, pages 6965–6975, 2018.
- [17] Yann LeCun, Léon Bottou, Yoshua Bengio, Patrick Haffner, et al. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.
- [18] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.