HexaGAN: Generative Adversarial Nets for Real World Classification

Uiwon Hwang [1] **Dahuin Jung** [1] **Sungroh Yoon** [1 2]

Abstract

Most deep learning classification studies assume clean data. However, when dealing with the real world data, we encounter three problems such as 1) missing data, 2) class imbalance, and 3) missing label problems. These problems undermine the performance of a classifier. Various preprocessing techniques have been proposed to mitigate one of these problems, but an algorithm that assumes and resolves all three problems together has not been proposed yet. In this paper, we propose HexaGAN, a generative adversarial network framework that shows promising classification performance for all three problems. We interpret the three problems from a single perspective to solve them jointly. To enable this, the framework consists of six components, which interact with each other. We also devise novel loss functions corresponding to the architecture. The designed loss functions allow us to achieve state-of-the-art imputation performance, with up to a 14% improvement, and to generate high-quality class-conditional data. We evaluate the classification performance (F1-score) of the proposed method with 20% missingness and confirm up to a 5% improvement in comparison with the performance of combinations of state-of-the-art methods.

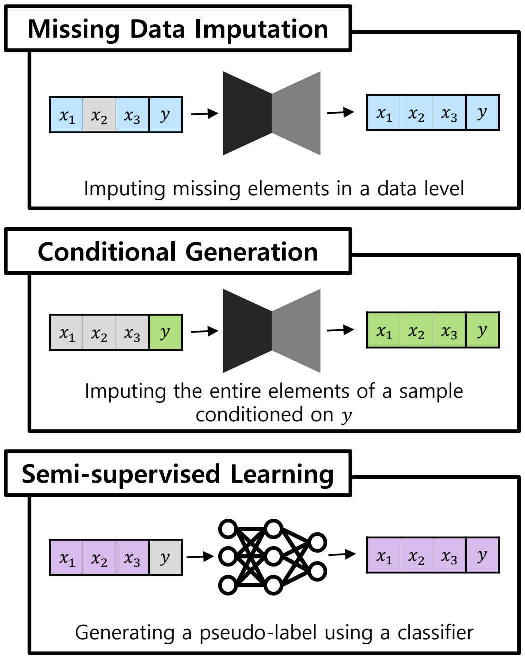
Missing Data Imputation
x_1 x_2 x_3 y
Imputing missing elements in a data level
Conditional Generation
Imputing the entire elements of a sample conditioned on y
Semi-supervised Learning
Generating a pseudo-label using a classifier

Figure 1. Tasks for the three main problems in real world classification. We define missing data imputation as a task that fills in missing data elements. Conditional generation can be defined as a task that imputes the entire elements in an instance conditioned on a certain class. Semi-supervised learning can be defined as a task that imputes missing labels.

1. Introduction

As deep learning models have achieved super-human performance in image classification tasks (He et al., 2016), there have been increasing attempts to apply deep learning models to more complicated tasks such as object detection (Ren et al., 2015), text classification (Zhang et al., 2015), and disease prediction (Hwang et al., 2017). However, real world data are often *dirty*, which means that the elements and labels are missing, or there is an imbalance between different classes of data. This prevents a classifier from being fully effective, and thus a preprocessing phase is required. Despite a considerable amount of research, no preprocessing technique has been proposed to address these three problems concurrently. Therefore, we first propose a framework which deals robustly with dirty data.

The types of data which are typically bedeviled with missing information include the user data employed in recommender systems (Koren et al., 2009), and in electronic health records (Miotto et al., 2016) utilizing deep learning based classifiers. Rubin (1976) identifies three main types of missing data: 1) Data are missing completely at random (MCAR). This type of missing data has no pattern which can be correlated with any other variable, whether observed or not. 2)

[1]Electrical and Computer Engineering, Seoul National University, Seoul, Korea [2]ASRI, INMC, Institute of Engineering Research, Seoul National University, Seoul, Korea. Correspondence to: Sungroh Yoon <sryoon@snu.ac.kr>.

Proceedings of the 36th International Conference on Machine Learning, Long Beach, California, PMLR 97, 2019. Copyright 2019 by the author(s).

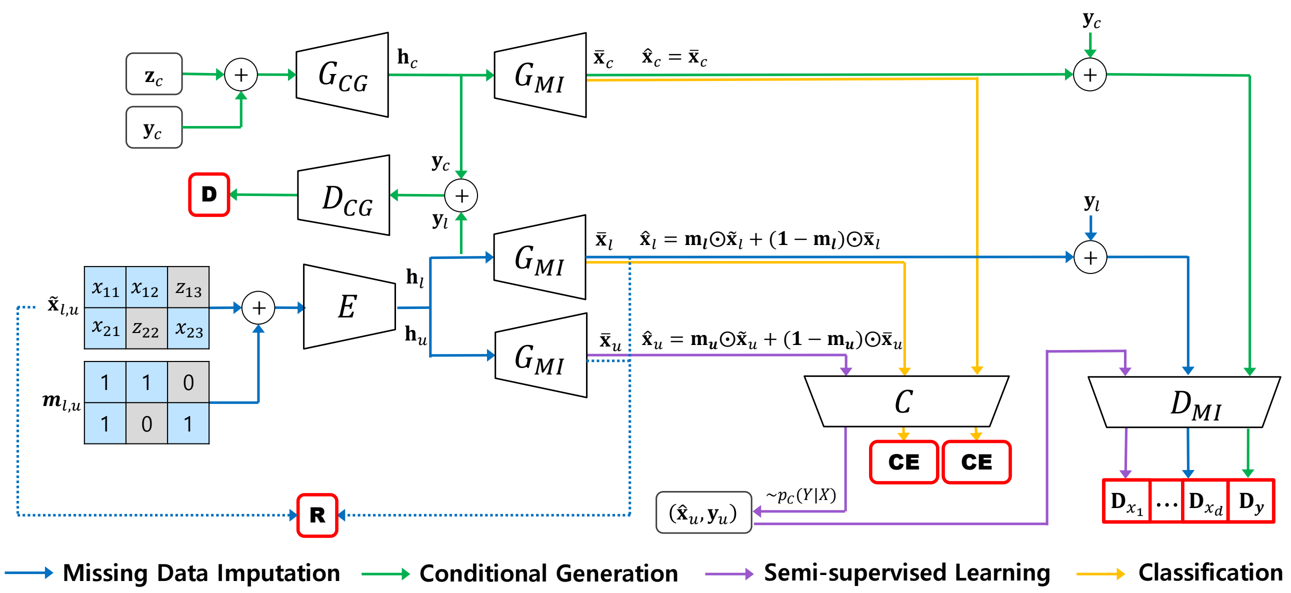
$\mathbf{z}_c$
$\mathbf{y}_c$
G_{CG}
$\mathbf{h}_c$
G_{MI}
$\bar{\mathbf{x}}_c$
$\hat{\mathbf{x}}_c = \bar{\mathbf{x}}_c$
$\mathbf{y}_c$
D
D_{CG}
$\mathbf{y}_c$
$\mathbf{y}_l$
$\mathbf{y}_l$
G_{MI}
$\bar{\mathbf{x}}_l$
$\hat{\mathbf{x}}_l = \mathbf{m}_l \odot \tilde{\mathbf{x}}_l + (\mathbf{1} - \mathbf{m}_l) \odot \bar{\mathbf{x}}_l$
$\tilde{\mathbf{x}}_{l,u}$
x_{11} x_{12} z_{13}
x_{21} z_{22} x_{23}
E
$\mathbf{h}_l$
$\mathbf{h}_u$
G_{MI}
$\bar{\mathbf{x}}_u$
$\hat{\mathbf{x}}_u = \mathbf{m}_u \odot \tilde{\mathbf{x}}_u + (\mathbf{1} - \mathbf{m}_u) \odot \bar{\mathbf{x}}_u$
$\mathbf{m}_{l,u}$
1 1 0
1 0 1
C
D_{MI}
CE
CE
$\sim p_C(Y|X)$
$(\hat{\mathbf{x}}_u, \mathbf{y}_u)$
R
$\mathbf{D}_{x_1}$ … $\mathbf{D}_{x_d}$ $\mathbf{D}_y$
Missing Data Imputation
Conditional Generation
Semi-supervised Learning
Classification

Figure 2. Overview of the HexaGAN model. Subscripts l, u, and c indicate that a vector is from labeled data, unlabeled data, and class-conditional data respectively. $\tilde{\mathbf{x}}$ denotes a data instance whose missing elements are replaced with noise. $\bar{\mathbf{x}}$ denotes a data generated by G_{MI}. $\hat{\mathbf{x}}$ denotes a data instance whose missing elements are filled with the generated values. $\mathbf{y}$ is a class label. Unlike $\mathbf{y}_l$ and $\mathbf{y}_c$, $\mathbf{y}_u$ is produced by C. $\mathbf{m}$ is a vector that indicates whether corresponding elements are missing or not. $\mathbf{h}$ is a vector in the hidden space. $\mathbf{R}$ is the reconstruction loss. $\mathbf{D}$ is the adversarial loss function between G_{CG} and D_{CG}. $\mathbf{D}_{x_i}$ represents the element-wise adversarial loss function. $\mathbf{D}_y$ represents the adversarial loss function for the label. **CE** represents the cross-entropy loss.

Data are missing at random (MAR). In this case, the pattern of missing data can be correlated with one or more observed variables. 3) Data are missing (but) not at random (MNAR). The pattern of this type of missing data can be related to both observed and unobserved variables. In this paper, we are concerned with MCAR data. The replacement of missing information within data is called *imputation* (Van Buuren, 2018). Imputation techniques include matrix completion (Hastie et al., 2015), k-nearest neighbors (Troyanskaya et al., 2001), multivariate imputation by chained equations (MICE) (Buuren & Groothuis-Oudshoorn, 2010), denoising autoencoders (Vincent et al., 2008), and methods based on generative adversarial networks (GAN) (Yoon et al., 2018; Shang et al., 2017). Poor or inappropriate imputation can mislead deep learning based techniques into learning the wrong data distribution.

Many real world datasets such as those related to anomaly detection (Chandola et al., 2009) and disease prediction (Khalilia et al., 2011) involve poorly balanced classes. The class imbalance problem can be overcome by techniques such as the synthetic minority oversampling technique (SMOTE) (Chawla et al., 2002) and adaptive synthetic (ADASYN) sampling (He et al., 2008). However, oversampling from the entire data distribution requires a large amount of memory. Cost sensitive loss (Sun et al., 2007) is also used to solve the class imbalance problem by differentiating cost weights to each class. However, cost sensitive loss tends to overfit to the minority classes (Elrahman & Abraham, 2013). We overcome the class imbalance problem by training a deep generative model to follow the true data distribution, and then generate samples of minority classes for each batch. This requires conditional generation, which we regard as imputation, as shown in Figure 1, in which entire elements are imputed according to the appropriate class label.

In deep learning, the amount of labeled training data has a significant impact on the performance. Insufficiency of labeled data is referred to as the missing label problem. It is encountered in real world applications such as natural language models (Turian et al., 2010) or healthcare systems (Beaulieu-Jones et al., 2016), where the cost of labeling is expensive. Related researchers have proposed semi-supervised methods by which to leverage unlabeled data. Semi-supervised learning is designed to make the best use of unlabeled data, using regularization and generative approaches. The regularization approach adds a regularization loss term, which is designed on the assumption that adjacent data points or the same architectural data points are likely to have the same label (Wang & Zhang, 2008; Laine & Aila, 2017; Grandvalet & Bengio, 2005; Miyato et al., 2015; Tarvainen & Valpola, 2017). Unlike the regularization approach, the generative approach enhances the performance of a classifier by utilizing raw unlabeled data in training the generative model (Kingma et al., 2014; Abbasnejad et al., 2017; Salimans et al., 2016; Springenberg, 2015; Dai et al., 2017).

As depicted in Figure 1, we define the missing data, class imbalance, and missing label problems in terms of imputation. Based on insight concerning the imputation, we find out that networks used for imputation can play multiple roles. Moreover, solving the three data problems simultaneously is more effective than solving them in a cascading form. In this paper, we propose a GAN framework consisting of six components to solve the three problems in real world classifications. We derive a new objective function for the imputation of missing data, and demonstrate that it performs better than the existing state-of-the-art imputation methods. We define conditional generation from the perspective of conditional imputation, and confirm that the proposed method works successfully by designing the imputation model to be a part of the framework. In order to deal with the missing label problem, we use semi-supervised learning, in which a classifier generates a synthetic class label for unlabeled data and a discriminator distinguishes fake from real labels.

In summary, our contributions are as follows:

- To the best of our knowledge, this is one of the first studies that defines the three problems (missing data, class imbalance, and missing label) in terms of imputation. Then, we propose HexaGAN to encourage thorough imputation of data with these three problems.
- To implement real world datasets into existing classifiers, we must apply suitable preprocessing techniques to the datasets. However, our framework is simple to use and works automatically when the absence of data elements and labels is indicated ($\mathbf{m}$ and m_y, See Section 3).
- We devise a combination of six components and the corresponding cost functions. More specifically, we propose a novel adversarial loss function and gradient penalty for element-wise imputation, confirming that our imputation performance produces stable, state-of-the-art results.
- In real world classification, the proposed method significantly outperforms cascading combinations of the existing state-of-the-art methods. As a result, we demonstrate that the components of our framework interplay to solve the problems effectively.

2. Generative Adversarial Networks

Generative models, which include GANs (Goodfellow et al., 2014), are capable of generating high-quality synthetic data for many applications. Although GANs are the most advanced than other techniques, model training can be unstable. Many studies have tried to stabilize GANs. Among them, Arjovsky et al. (2017) proposed the Wasserstein GAN (WGAN), which has a smoother gradient by introducing the Wasserstein (Earth Mover) distance. Several gradient penalties have also been proposed (Gulrajani et al., 2017; Mescheder et al., 2018) to make WGAN training more stable. In this paper, we modify the WGAN loss and zero-centered gradient penalty for missing data imputation. Experimentally, we show that the proposed method has more stable and better imputation performance than the existing vanilla GAN loss-based model.

GAIN (Yoon et al., 2018) is the first method to use a GAN for imputing MCAR data. The typical discriminator predicts whether each *instance* is real or fake. However, this task is difficult if all instances have missing data. Instead, GAIN labels each *element* of an instance as missing or not, so that the discriminator can discriminate between real and fake elements. Our imputation method shares some similarity with GAIN because both methods label elements as real or fake. The imputation performance of GAIN measured by our own implementation with the specific dataset is lower than that of the autoencoder, and the learning curve appears to be unstable. However, HexaGAN provides a stable imputation performance, and usability by including class-conditional generation to address the class imbalance problem, and through the use of semi-supervised learning.

TripleGAN (Li et al., 2017) is a GAN for semi-supervised learning in which a classifier, a generator, and a discriminator interact. The classifier creates pseudo-labels for unlabeled data, and image-label pairs are then passed to the discriminator. The classifier and discriminator are trained competitively. In this paper, we adopt the pseudo-labeling technique of TripleGAN to allow HexaGAN to perform semi-supervised learning.

3. Proposed Method

The HexaGAN framework is comprised of six components, as illustrated in Figure 2:

- E: the encoder, that transfers both labeled and unlabeled instances into the hidden space.
- G_{MI}: a generator that imputes missing data.
- D_{MI}: a discriminator for missing imputation, that distinguishes between missing and non-missing elements and labels.
- G_{CG}: a generator that creates conditional hidden vectors $\mathbf{h}_c$.
- D_{CG}: a discriminator for conditional generation, that

determines whether a hidden vector is from the dataset or has been created by G_{CG}.

- C: the classifier, that estimates class labels. This also works as the label generator.

HexaGAN operates on datasets containing instances $\mathbf{x}^1, ..., \mathbf{x}^n \in \mathbb{R}^d$, where n is the number of instances and d is the number of elements in an instance. The i-th element in a single instance x_i^j is a scalar, and some of these elements may be missing. The first n_l instances are labeled data, and the remaining $n - n_l$ instances are unlabeled data. There are class labels $\mathbf{y}^1, ..., \mathbf{y}^{n_l} \in \mathbb{R}^{n_c}$ corresponding to each instance, where n_c is the number of classes. Boolean vectors $\mathbf{m}^1, ..., \mathbf{m}^n \in \mathbb{R}^d$ indicate whether each element in an instance is missing or not. If m_i^j (the i-th element of a vector $\mathbf{m}^j$) is 0, x_i^j is missing. The boolean $m_y \in \mathbb{R}$ indicates whether an instance has a label or not. If m_y is 0, the label is missing. Thus, labeled instances exist as a set of $D_l = \{(\mathbf{x}^j, \mathbf{y}^j, \mathbf{m}^j, m_y^j = 1)\}_{j=1}^{n_l}$, and unlabeled instances exist as a set of $D_u = \{(\mathbf{x}^j, \mathbf{m}^j, m_y^j = 0)\}_{j=n_l+1}^{n}$.

3.1. Missing data imputation

Missing data imputation aims to fill in missing elements using the distribution of data represented by the generative model. In HexaGAN, missing data imputation is performed by E, G_{MI}, and D_{MI}. An instance received by D_{MI} is not labeled as real or fake, but each *element* is labeled as real (non-missing) or fake (missing).

From now on, we omit the superscript for a clearer explanation (i.e., x_i^j=x_i). First, we make a noise vector $\mathbf{z} \in \mathbb{R}^d$ with the same dimension as an input instance $\mathbf{x} \in (\mathbf{x}_l \cup \mathbf{x}_u)$ by sampling from a uniform distribution $U(0, 1)$. We replace the missing elements in the instance with elements of $\mathbf{z}$ to generate $\tilde{\mathbf{x}}$:

$$\tilde{\mathbf{x}} = \mathbf{m} \odot \mathbf{x} + (\mathbf{1} - \mathbf{m}) \odot \mathbf{z} \tag{1}$$

where $\odot$ is element-wise multiplication. The objective of our framework is to sample the patterns stored in the model that are the most suitable replacements for the missing data (i.e., to generate samples which follow $p(\mathbf{x}|\tilde{\mathbf{x}}, \mathbf{m})$). Then, $\tilde{\mathbf{x}}$ is concatenated with $\mathbf{m}$, and from the pair $(\tilde{\mathbf{x}}, \mathbf{m})$, the encoder E generates a hidden variable $\mathbf{h} = E(\tilde{\mathbf{x}}, \mathbf{m})$ in the hidden space, which has the dimension d_{h}.

The G_{MI} receives $\mathbf{h}$ and generates $\bar{\mathbf{x}} = G_{MI}(\mathbf{h})$. The missing elements in the input instance are imputed with the generated values, resulting in $\hat{\mathbf{x}}$ as follows:

$$\hat{\mathbf{x}} = \mathbf{m} \odot \mathbf{x} + (\mathbf{1} - \mathbf{m}) \odot \bar{\mathbf{x}} \tag{2}$$

The D_{MI} now determines whether each element of the pair $(\hat{\mathbf{x}}, \mathbf{y})$ is real or fake. The label for $\hat{\mathbf{x}}$ is $\mathbf{m}$. The D_{MI} calculates the adversarial losses by determining whether the missingness is correctly predicted for each element, which is then used to train E, G_{MI}, and D_{MI}. The adversarial loss $\mathcal{L}_{G_{MI}}$ which is used to train E and G_{MI}, and $\mathcal{L}_{D_{MI}}$ which is used to train D_{MI} can be expressed as follows:

Algorithm 1 Missing data imputation

```
input  : x - data with missing values sampled from $D_l$ and $D_u$;
         m - vector indicating whether elements are missing;
         z - noise vector sampled from $U(0, 1)$
output : $\hat{\mathbf{x}}$ - imputed data
  repeat
    Sample a batch of pairs $(\mathbf{x}, \mathbf{m}, \mathbf{z})$
    $\tilde{\mathbf{x}} \leftarrow \mathbf{m} \odot \mathbf{x} + (\mathbf{1} - \mathbf{m}) \odot \mathbf{z}$
    $\mathbf{h} \leftarrow E(\tilde{\mathbf{x}}, \mathbf{m})$
    $\bar{\mathbf{x}} \leftarrow G_{MI}(\mathbf{h})$
    $\hat{\mathbf{x}} \leftarrow \mathbf{m} \odot \mathbf{x} + (\mathbf{1} - \mathbf{m}) \odot \bar{\mathbf{x}}$
    Update $D_{MI}$ using stochastic gradient descent (SGD)
    $\nabla_{D_{MI}} \mathcal{L}_{D_{MI}} + \lambda_1 \mathcal{L}_{\text{GP}_{MI}}$
    Update $E$ and $G_{MI}$ using SGD
    $\nabla_E \mathcal{L}_{G_{MI}} + \alpha_1 \mathcal{L}_{\text{recon}}$
    $\nabla_{G_{MI}} \mathcal{L}_{G_{MI}} + \alpha_1 \mathcal{L}_{\text{recon}}$
  until training loss is converged
```

$$\mathcal{L}_{G_{MI}} = -\sum_{i=1}^{d} \mathbb{E}_{\hat{\mathbf{x}}, \mathbf{y}, \mathbf{m}} \left[(1 - m_i) \cdot D_{MI}(\hat{\mathbf{x}}, \mathbf{y})_i \right] \tag{3}$$

$$\mathcal{L}_{D_{MI}} = \sum_{i=1}^{d} \mathbb{E}_{\hat{\mathbf{x}}, \mathbf{y}, \mathbf{m}} \left[(1 - m_i) \cdot D_{MI}(\hat{\mathbf{x}}, \mathbf{y})_i \right] - \mathbb{E}_{\hat{\mathbf{x}}, \mathbf{y}, \mathbf{m}} \left[m_i \cdot D_{MI}(\hat{\mathbf{x}}, \mathbf{y})_i \right] \tag{4}$$

where $D_{MI}(\cdot)_i$ is the i-th output element of D_{MI}. The following theorem confirms that the proposed adversarial loss functions make the generator distribution converge to the desired data distribution.

Theorem 1 *A generator distribution $p(\mathbf{x}|\mathbf{m}_i = 0)$ is a global optimum for the min-max game of G_{MI} and D_{MI}, if and only if $p(\mathbf{x}|\mathbf{m}_i = 1) = p(\mathbf{x}|\mathbf{m}_i = 0)$ for all $\mathbf{x} \in \mathbb{R}^d$, except possibly on a set of zero Lebesgue measure.*

Proof of Theorem 1 is provided in Supplementary Materials.

Moreover, we add a reconstruction loss to the loss function of E and G_{MI} to exploit the information of non-missing elements, as follows:

$$\mathcal{L}_{\text{recon}} = \mathbb{E}_{\bar{\mathbf{x}}|\mathbf{x}, \mathbf{m}} \left[\sum_{i=1}^{d} m_i (x_i - \bar{x}_i)^2 \right] \tag{5}$$

For more stable GAN training, we modify a simplified version of the zero-centered gradient penalty proposed by

Mescheder et al. (2018) in an element-wise manner, and add the gradient penalty to the loss function of D_{MI}. The modified regularizer penalizes the gradients of each output unit of the D_{MI} on $p_{\mathcal{D}}(x_i)$:

$$\mathcal{L}_{\text{GP}_{MI}} = \sum_{i=1}^{d} \mathbb{E}_{p_{\mathcal{D}}(x_i)} \left[||\nabla_{\hat{\mathbf{x}}} D_{MI}(\hat{\mathbf{x}})_i||_2^2 \right] \tag{6}$$

We define $\hat{\mathbf{x}}$ in $p_{\mathcal{D}}(x_i)$ as data with m_i is 1 (i.e., $p_{\mathcal{D}}(x_i) = \{\hat{\mathbf{x}}^j | m_i^j = 1\}$). In other words, as suggested by Mescheder et al. (2018), we penalize D_{MI} only for data wherein the i-th element is not missing (real) in a batch. This helps balance an adversarial relationship between the generator and discriminator by forcing the discriminator closer to Nash Equilibrium.

Therefore, missing data imputation and model training are performed as described in Algorithm 1. We used 10 for both hyperparameters λ_1 and α_1 in our experiments.

3.2. Conditional generation

We define conditional generation for the class imbalance problem as the imputation of entire data elements on a given class label (i.e., generating $(x_1, ..., x_d)$ following $p(\mathbf{x}|\mathbf{y})$). Since we have G_{MI}, which is a generator for imputation, we can oversample data instances by feeding synthetic $\mathbf{h}$ into G_{MI}. Therefore, we introduce G_{CG} to generate a hidden variable $\mathbf{h}_c$ corresponding to the target class label $\mathbf{y}_c$, i.e., we sample $\mathbf{h}_c \sim p_{G_{CG}}(\mathbf{h}|\mathbf{y})$. We also introduce D_{CG} to distinguish pairs of generated hidden variables and target class labels $(\mathbf{h}_c, \mathbf{y}_c)$ (fake) from pairs of hidden variables for labeled data and corresponding class labels $(\mathbf{h}_l, \mathbf{y}_l)$ (real). G_{CG} and D_{CG} are trained with WGAN loss and zero-centered gradient penalty on $\mathbf{h}_l$ as follows:

$$\mathcal{L}_{G_{CG}} = -\mathbb{E}_{\mathbf{h}_c \sim p_{G_{CG}}(\mathbf{h}_c|\mathbf{y}_c)}[D_{CG}(\mathbf{h}_c, \mathbf{y}_c)] \tag{7}$$

$$\begin{aligned}\mathcal{L}_{D_{CG}} =& \mathbb{E}_{\mathbf{h}_c \sim p_{G_{CG}}(\mathbf{h}_c|\mathbf{y}_c)}[D_{CG}(\mathbf{h}_c, \mathbf{y}_c)] \\ &- \mathbb{E}_{\mathbf{h}_l \sim p_E(\mathbf{h}_l|x_l)}[D_{CG}(\mathbf{h}_l, \mathbf{y}_l)]\end{aligned} \tag{8}$$

$$\mathcal{L}_{\text{GP}_{CG}} = \mathbb{E}_{\mathbf{h}_l \sim p_E(\mathbf{h}_l|x_l)} \left[||\nabla_{\mathbf{h}_l} D_{CG}(\mathbf{h}_l, \mathbf{y}_l)||_2^2 \right] \tag{9}$$

G_{MI} maps generated $\mathbf{h}_c$ into a realistic $\hat{\mathbf{x}}_c$. Because $\mathcal{L}_{G_{CG}}$ is not enough to stably generate $\mathbf{h}_c$, we add the loss of G_{MI} from $\hat{\mathbf{x}}_c$. Since we defined conditional generation as imputation of all the elements, G_{CG} and D_{MI} are related adversarially. The label of $(\hat{\mathbf{x}}_c, \mathbf{y}_c)$ for D_{MI} is a $(d+1)$-dimensional zero vector.

In addition, the cross-entropy of $(\hat{\mathbf{x}}_c, \mathbf{y}_c)$ calculated from the prediction of C is also added to the loss function of G_{CG} to stably generate the data that is conditioned on the target class as follows:

$$\mathcal{L}_{\text{CE}}(\hat{\mathbf{x}}_c, \mathbf{y}_c) = -\mathbb{E}_{\hat{\mathbf{x}}_c|\mathbf{y}_c} \left[\sum_{k=1}^{n_c} \mathbf{y}_{c_k} \log(C(\hat{\mathbf{x}}_c)_k) \right] \tag{10}$$

where $C(\cdot)_k$ is the softmax output for the k-th class. Thus, D_{CG} and G_{CG} are trained according to:

$$\min_{D_{CG}} \mathcal{L}_{D_{CG}} + \lambda_2 \mathcal{L}_{\text{GP}_{CG}} \tag{11}$$

$$\min_{G_{CG}} \mathcal{L}_{G_{CG}} + \alpha_2 \mathcal{L}_{G_{MI}} + \alpha_3 \mathcal{L}_{\text{CE}}(\hat{\mathbf{x}}_c, y_c) \tag{12}$$

where λ_2, α_2, and α_3 denote hyperparameters, and we set λ_2 to 10, α_2 to 1, and α_3 to 0.01 in our experiments. Since the distribution of $\mathbf{h}_l$ moves according to the training of E, we set the number of update iterations of D_{CG} and G_{CG} per an update of E to 10, so that $\mathbf{h}_c$ follows the distribution of $\mathbf{h}_l$ well.

3.3. Semi-supervised classification

3.3.1. Pseudo-labeling

We define semi-supervised learning as imputing missing labels by the pseudo-labeling technique, TripleGAN (Li et al., 2017). Semi-supervised learning is achieved by the interaction of C and D_{MI}. C generates a pseudo-label $\mathbf{y}_u$ of an unlabeled instance $\hat{\mathbf{x}}_u$, i.e., $\mathbf{y}_u$ is sampled from the classifier distribution $p_C(\mathbf{y}|\mathbf{x})$. Then, the data-label pair $(\hat{\mathbf{x}}_u, \mathbf{y}_u)$ enters D_{MI}. The last element of the D_{MI} output, $D_{MI}(\cdot)_{d+1}$, determines whether the label is real or fake. The label for pseudo-labeling is m_y. C and D_{MI} are trained according to the following loss functions:

$$L_C = -\mathbb{E}_{\mathbf{y}_u|\hat{\mathbf{x}}_u \sim p_C} \left[D_{MI}(\hat{\mathbf{x}}_u, \mathbf{y}_u)_{d+1} \right] \tag{13}$$

$$\begin{aligned}L_{D_{MI}}^{d+1} =& \mathbb{E}_{\mathbf{y}_u|\hat{\mathbf{x}}_u \sim p_C} \left[D_{MI}(\hat{\mathbf{x}}_u, \mathbf{y}_u)_{d+1} \right] \\ &- \mathbb{E}_{\mathbf{y}|\hat{\mathbf{x}} \sim p_{data}} \left[D_{MI}(\hat{\mathbf{x}}, \mathbf{y})_{d+1} \right]\end{aligned} \tag{14}$$

where p_{data} denotes the data distribution of y conditioned on $\hat{\mathbf{x}}$. $L_{D_{MI}}^{d+1}$ is added to the loss of D_{MI}, so that i in Equation 4 expands from d to $d+1$. If G_{MI} learns the true data distribution, then we can postulate that p_{data} follows the true conditional distribution. We should note that the adversarial loss is identical to the loss function of WGAN between C and D_{MI}. Therefore, C plays a role as a label generator, and $D_{MI}(\cdot)_{d+1}$ acts as a label discriminator.

Through adversarial learning, we expect that the adversarial loss enhances the performance of C. It can be shown that C minimizing the adversarial loss L_C is equivalent to optimizing the output distribution matching (ODM) cost (Sutskever et al., 2015).

Theorem 2 *Optimizing the adversarial losses for C and $D_{MI}(\cdot)_{d+1}$ imposes an unsupervised constraint on C. Then, the adversarial losses for semi-supervised learning in HexaGAN satisfy the definition of the ODM cost.*

Proof of Theorem 2 is provided in Supplementary Materials.

According to the properties of the ODM cost, the global optimum of supervised learning is also a global optimum

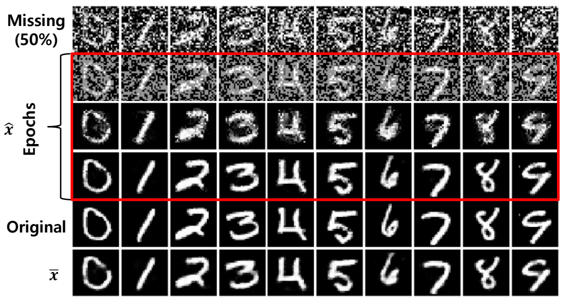

Figure 3. Imputation results with the MNIST dataset. 1st row: MNIST images with 50% missing randomly as inputs of HexaGAN. 2nd∼4th rows (red box): images imputed by HexaGAN ($\hat{\mathbf{x}}$) at 1, 10, and 100 epochs. 5th row: original images (no missing element). 6th row: images generated by G_{MI} for imputation ($\bar{\mathbf{x}}$).

of semi-supervised learning. Therefore, intuitively, L_C and $L^{d+1}_{D_{MI}}$ serve as guides for finding the optimum point of the supervised loss.

3.3.2. Classification of HexaGAN

In order to train C, the two models E and G_{MI} impute the missing values of data instances $\hat{\mathbf{x}}_l$. G_{CG} produces hidden vectors $\mathbf{h}_c$ conditioned on the minority classes so that the number of data in the minority classes in each batch is equal to the number of data instances in the majority class of each batch, and G_{MI} generates class-conditional data $\hat{\mathbf{x}}_c$. Then, the cross-entropy between $\hat{\mathbf{x}}_{l,c} \in (\hat{\mathbf{x}}_l \cup \hat{\mathbf{x}}_c)$ and $\mathbf{y}_{l,c} \in (\mathbf{y}_l \cup \mathbf{y}_c)$ is calculated to train C. Unlabeled data $\hat{\mathbf{x}}_u$ is used to optimize L_C, the loss for pseudo-labeling, thereby training a more robust classifier.

Therefore, C is trained according to:

$$\min_C \mathcal{L}_{\text{CE}}(\hat{\mathbf{x}}_{l,c}, \mathbf{y}_{l,c}) + \alpha_4 L_C \tag{15}$$

where we used 0.1 for α_4 in our experiments. The entire training procedure of HexaGAN is presented in Supplementary Materials.

4. Experiments

Here, we present the performance of the proposed method. We used datasets from the UCI machine learning repository (Dheeru & Karra Taniskidou, 2017), including real world datasets (breast, credit, wine) and a synthetic dataset (madelon). Detailed descriptions are presented in Supplementary Materials. We also used a handwritten digit dataset (MNIST). First, we show the imputation performance of HexaGAN. Then, we show the quality of conditional generation using our framework. Finally, we present the classification performance of our proposed model, assuming the problems in real world classification.

We basically assume 20% missingness (MCAR) in the elements and labels of the UCI dataset and 50% in the elements of the MNIST dataset to cause missing data and missing label problems. Every element was scaled to a range of [0,1]. We repeated each experiment 10 times and used 5-fold cross validation. As the performance metric, we calculated the root mean square error (RMSE) for missing data imputation and the F1-score for classification. We analyzed the learning curve and found that the modified zero-centered gradient and RMSprop promote stability in HexaGAN. The details are described in Supplementary Materials. The architecture of HexaGAN can also be found in Supplementary Materials.

Table 1. Performance comparison with other imputation methods (RMSE)

Method	Breast	Credit	Wine	Madelon	MNIST
Zeros	0.2699	0.2283	0.4213	0.5156	0.3319
Matrix	0.0976	0.1277	0.1772	0.1456	0.2540
K-NN	0.0872	0.1128	0.1695	0.1530	0.2267
MICE	0.0842	0.1073	0.1708	0.1479	0.2576
Autoencoder	0.0875	0.1073	0.1481	0.1426	0.1506
GAIN	0.0878	0.1059	0.1406	0.1426	0.1481
HexaGAN	**0.0769**	**0.1022**	**0.1372**	**0.1418**	**0.1452**

4.1. Imputation performance

4.1.1. Comparison with real world datasets

We used UCI datasets and the MNIST dataset to evaluate the imputation performance. Table 1 shows the imputation performance of zero imputation, matrix completion, k-nearest neighbors, MICE, autoencoder, GAIN, and HexaGAN. In our experiments, we observed that HexaGAN outperforms the state-of-the-art methods on all datasets (up to a 14% improvement). Two deep generative models, GAIN and HexaGAN, use both the reconstruction loss and the adversarial loss. GAIN shows the same or lower performance than the autoencoder on certain datasets, whereas HexaGAN consistently outperforms the autoencoder on all datasets. This shows that the novel adversarial loss boosts the imputation performance.

4.1.2. Qualitative analysis

Figure 3 visualizes the imputation performance with the MNIST dataset. Since MNIST is an image dataset, we designed HexaGAN with convolutional and deconvolutional neural networks. The first row of Figure 3 shows MNIST data with 50% missing as the input for HexaGAN. The next three rows show $\hat{\mathbf{x}}$ after 1, 10, and 100 epochs, and it can be seen that higher quality imputed data are generated as the number of epochs increases. The next row presents the original data with no missing value, and the last row shows $\bar{\mathbf{x}}$ generated by G_{MI}. This suggests that the proposed method imputes missing values with very high-quality data. The

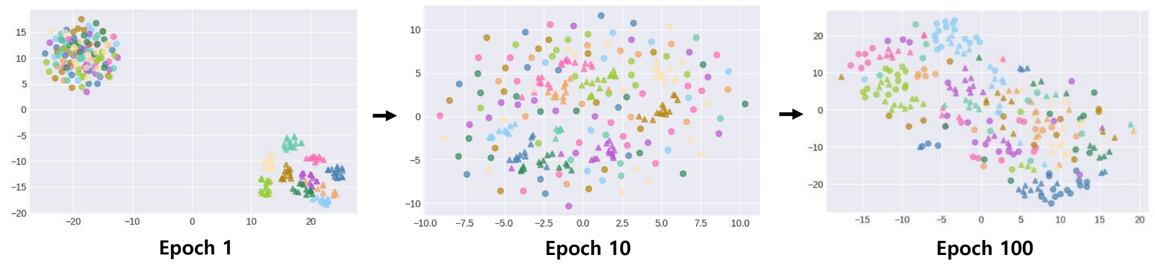

Figure 4. tSNE analysis with the MNIST dataset at 1, 10, and 100 epochs. The circles stand for $\mathbf{h}_l$ (hidden vectors from E). The triangles denotes $\mathbf{h}_c$ (hidden vectors from G_{CG}). Different colors represent different class labels.

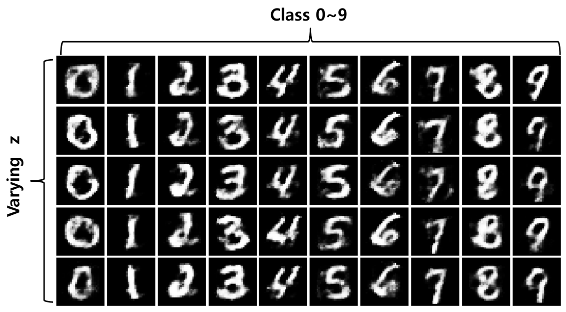

Figure 5. Class-conditional generation results with the MNIST dataset. Each row visualizes generated images conditioned on 0$\sim$9. Each column shows images generated by all different $\mathbf{z}$s.

RMSE value using the convolutional architecture is 0.0914.

4.2. Conditional generation performance

4.2.1. TSNE ANALYSIS

We used tSNE (Maaten & Hinton, 2008) to analyze $\mathbf{h}_l$ generated by E and $\mathbf{h}_c$ generated by G_{CG}. Figure 4 shows the changes of $\mathbf{h}_l$ (circle) and $\mathbf{h}_c$ (triangle) according to the iteration. Each color stands for a class label. At epoch 1, $\mathbf{h}_l$ and $\mathbf{h}_c$ have very different distributions, and form respective clusters. At epoch 10, the cluster of $\mathbf{h}_c$ is overlapped by the cluster of $\mathbf{h}_l$. At epoch 100, E learns the manifold of the hidden representation, so that $\mathbf{h}_l$ is gathered by class and $\mathbf{h}_c$ follows the distribution of $\mathbf{h}_l$ well. That is, G_{CG} creates a high-quality $\mathbf{h}_c$ that is conditioned on a class label. The complete version of the tSNE analysis is given in Supplementary Materials.

4.2.2. QUALITATIVE ANALYSIS

To evaluate the performance of conditional generation, we used the same architecture as in Section 4.1.2 and generated synthetic MNIST images conditioned on 10 class labels. Figure 5 presents the generated MNIST images. Each row shows the results of conditioning the class labels $0 \sim 9$, and each column shows the results of changing the noise vector $\mathbf{z}$. It can be seen that G_{CG} and G_{MI} produce realistic images of digits and that various image shapes are generated according to $\mathbf{z}$. Images conditioned on 9 in the second and fifth rows look like 7. This can be interpreted as a phenomenon in which the hidden variables for 9 and 7 are placed in adjacent areas on the manifold of the hidden space.

4.3. Classification performance

HexaGAN works without any problem for multi-class classification, but for the convenience of the report, we tested only binary classifications. The breast and credit datasets are imbalanced with a large number of negative samples. The wine dataset has three classes, and it was tested by binarizing the label 1 as negative, and labels 2 and 3 as positive to calculate an F1-score. The wine dataset was imbalanced with a large number of positive samples. Madelon is a balanced synthetic dataset that randomly assigns binary labels to 32 clusters on 32 vertices of a 5-dimensional hypercube.

4.3.1. ABLATION STUDY

The components affecting the classification performance of HexaGAN are G_{MI} to fill in missing data, G_{CG} to perform conditional generation, and $D_{MI}(\cdot)_{d+1}$ to enable semi-supervised learning. Table 2 compares the classification performance depending on the removal of these components. In the case of MLP, which is equivalent to HexaGAN without any of these three components, missing data were filled in with values sampled uniformly from [0,1].

As a result, MLP shows the worst performance. When HexaGAN contains G_{CG} (from the second row to the fourth row), the biggest performance improvement is shown in the credit data which is the most imbalanced. The more the components included in HexaGAN, the higher the classification performance obtained. HexaGAN with all components shows the highest performance on every dataset. Our delicately devised architecture improves the classification performance by up to 36%. It offers the advantage that any classifier that is state-of-the-art in a controlled environment can be plugged into the proposed framework, and the

Table 2. Ablation study of HexaGAN (F1-score)

Method	Breast	Credit	Wine	Madelon
MLP (HexaGAN w/o G_{MI} & G_{CG} & $D_{MI_{d+1}}$)	0.9171 ±0.0101	0.3404 ±0.0080	0.9368 ±0.0040	0.6619 ±0.0017
HexaGAN w/o G_{CG} & $D_{MI_{d+1}}$	0.9725 ±0.0042	0.4312 ±0.0028	0.9724 ±0.0065	0.6676 ±0.0038
HexaGAN w/o G_{CG}	0.9729 ±0.0007	0.4382 ±0.0075	0.9738 ±0.0135	0.6695 ±0.0043
HexaGAN w/o $D_{MI_{d+1}}$	0.9750 ±0.0030	0.4604 ±0.0097	0.9770 ±0.0037	0.6699 ±0.0022
HexaGAN	**0.9762 ±0.0021**	**0.4627 ±0.0040**	**0.9814 ±0.0059**	**0.6716 ±0.0019**

Table 3. Classification performance (F1-score) comparison with other combinations of state-of-the-art methods

Method	Breast	Credit	Wine	Madelon
MICE + CS + TripleGAN	0.9417 ±0.0044	0.3836 ±0.0052	0.9704 ±0.0043	0.6681 ±0.0028
GAIN + CS + TripleGAN	0.9684 ±0.0102	0.4076 ±0.0038	0.9727 ±0.0046	0.6690 ±0.0027
MICE + SMOTE + TripleGAN	0.9434 ±0.0060	0.4163 ±0.0029	0.9756 ±0.0037	0.6712 ±0.0008
GAIN + SMOTE + TripleGAN	0.9672 ±0.0063	0.4401 ±0.0031	0.9735 ±0.0063	0.6703 ±0.0032
HexaGAN	**0.9762 ±0.0021**	**0.4627 ±0.0040**	**0.9814 ±0.0059**	**0.6716 ±0.0019**

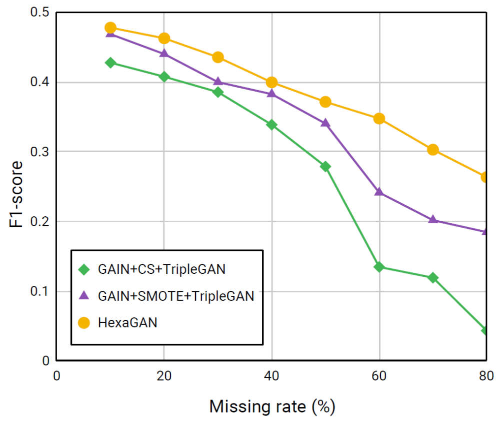

Figure 6. Classification performance (F1-score) comparison with respect to the missing rate with the credit dataset

classifier will perform at its highest capacity.

4.3.2. Comparison with other combinations

In this experiment, we compared the classification performance of HexaGAN with those of combinations of state-of-the-art methods for the three problems. For missing data imputation, we used MICE, which showed the best performance among machine learning based methods, and GAIN, which showed the best performance among deep generative models. For class imbalance, we used the cost sensitive loss (CS) and oversampled the minority class in a batch using SMOTE. We adopted the TripleGAN for semi-supervised learning. The classifier of TripleGAN used the same architecture as C of HexaGAN for a fair comparison.

As shown in Table 3, HexaGAN shows significantly better performance than the combinations of existing methods in cascading form (up to a 5% improvement). In addition, the madelon dataset is balanced; thus, comparing HexaGAN without G_{CG} (the third row of Table 2) with the combination of MICE, CS, and TripleGAN (the first row of Table 3) and the combination of GAIN, CS, and TripleGAN (the second row of Table 3) shows the classification performance with respect to imputation methods. We confirm that the imputation method of HexaGAN guarantees better classification performance than the other imputation methods.

4.3.3. Classification performance with respect to missing rate

Figure 6 compares the classification performance of HexaGAN with those of competitive combinations for various missing rates in the credit dataset. We used the combination of GAIN, CS, and TripleGAN and the combination of GAIN, SMOTE, and TripleGAN as benchmarks. According to the results, HexaGAN outperforms the benchmarks for all missing rates. Moreover, our method shows a larger performance gap compared to the benchmarks for high missing rates. This means that HexaGAN works robustly in situations in which only little information is available.

5. Conclusion

To interactively overcome the three main problems in real world classification (missing data, class imbalance, and missing label), we define the three problems from the perspective of missing information. Then, we propose a HexaGAN framework wherein six neural networks are actively correlated with others, and design several loss functions that maximize the utilization of any incomplete data. Our proposed method encourages more powerful performance in both imputation and classification than existing state-of-the-art methods. Moreover, HexaGAN is a one-stop solution that automatically solves the three problems commonly presented in real world classification. For future work, we plan to extend HexaGAN to time series datasets such as electronic health records.

Acknowledgements

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (Ministry of Science and ICT) [No. 2018R1A2B3001628, No. 2014M3C9A3063541], Korea Health Technology RD Project through the Korea Health Industry Development Institute (KHIDI) funded by the Ministry of Health & Welfare [HI15C3224], AIR Lab (AI Research Lab) in Hyundai Motor Company through HMC-SNU AI Consortium Fund, and Brain Korea 21 Plus Project in 2019.

References

Abbasnejad, M. E., Dick, A., and van den Hengel, A. Infinite variational autoencoder for semi-supervised learning. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 781–790. IEEE, 2017.

Arjovsky, M., Chintala, S., and Bottou, L. Wasserstein generative adversarial networks. In *International Conference on Machine Learning*, pp. 214–223, 2017.

Beaulieu-Jones, B. K., Greene, C. S., et al. Semi-supervised learning of the electronic health record for phenotype stratification. *Journal of biomedical informatics*, 64:168–178, 2016.

Buuren, S. v. and Groothuis-Oudshoorn, K. mice: Multivariate imputation by chained equations in r. *Journal of statistical software*, pp. 1–68, 2010.

Chandola, V., Banerjee, A., and Kumar, V. Anomaly detection: A survey. *ACM computing surveys (CSUR)*, 41(3): 15, 2009.

Chawla, N. V., Bowyer, K. W., Hall, L. O., and Kegelmeyer, W. P. Smote: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, 16:321–357, 2002.

Dai, Z., Yang, Z., Yang, F., Cohen, W. W., and Salakhutdinov, R. R. Good semi-supervised learning that requires a bad gan. In *Advances in Neural Information Processing Systems*, pp. 6510–6520, 2017.

Dheeru, D. and Karra Taniskidou, E. UCI machine learning repository, 2017. URL `http://archive.ics.uci.edu/ml`.

Elrahman, S. M. A. and Abraham, A. A review of class imbalance problem. *Journal of Network and Innovative Computing*, 1(2013):332–340, 2013.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. Generative adversarial nets. In *Advances in neural information processing systems*, pp. 2672–2680, 2014.

Grandvalet, Y. and Bengio, Y. Semi-supervised learning by entropy minimization. In *Advances in neural information processing systems*, pp. 529–536, 2005.

Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., and Courville, A. C. Improved training of wasserstein gans. In *Advances in Neural Information Processing Systems*, pp. 5767–5777, 2017.

Hastie, T., Mazumder, R., Lee, J. D., and Zadeh, R. Matrix completion and low-rank svd via fast alternating least squares. *The Journal of Machine Learning Research*, 16 (1):3367–3402, 2015.

He, H., Bai, Y., Garcia, E. A., and Li, S. Adasyn: Adaptive synthetic sampling approach for imbalanced learning. In *Neural Networks, 2008. IJCNN 2008.(IEEE World Congress on Computational Intelligence). IEEE International Joint Conference on*, pp. 1322–1328. IEEE, 2008.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Hwang, U., Choi, S., and Yoon, S. Disease prediction from electronic health records using generative adversarial networks. *arXiv preprint arXiv:1711.04126*, 2017.

Khalilia, M., Chakraborty, S., and Popescu, M. Predicting disease risks from highly imbalanced data using random forest. *BMC medical informatics and decision making*, 11(1):51, 2011.

Kingma, D. P., Mohamed, S., Rezende, D. J., and Welling, M. Semi-supervised learning with deep generative models. In *Advances in neural information processing systems*, pp. 3581–3589, 2014.

Koren, Y., Bell, R., and Volinsky, C. Matrix factorization techniques for recommender systems. *Computer*, (8): 30–37, 2009.

Laine, S. and Aila, T. Temporal ensembling for semi-supervised learning. In *International Conference on Learning Representations*, 2017.

Li, C., Xu, T., Zhu, J., and Zhang, B. Triple generative adversarial nets. In *Advances in Neural Information Processing Systems 30*, pp. 4088–4098. 2017.

Maaten, L. v. d. and Hinton, G. Visualizing data using t-sne. *Journal of machine learning research*, 9(Nov): 2579–2605, 2008.

Mescheder, L., Geiger, A., and Nowozin, S. Which training methods for gans do actually converge? In *International Conference on Machine Learning*, pp. 3478–3487, 2018.

Miotto, R., Li, L., Kidd, B. A., and Dudley, J. T. Deep patient: an unsupervised representation to predict the future of patients from the electronic health records. *Scientific reports*, 6:26094, 2016.

Miyato, T., Maeda, S.-i., Koyama, M., Nakae, K., and Ishii, S. Distributional smoothing with virtual adversarial training. *arXiv preprint arXiv:1507.00677*, 2015.

Ren, S., He, K., Girshick, R., and Sun, J. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pp. 91–99, 2015.

Rubin, D. B. Inference and missing data. *Biometrika*, 63(3): 581–592, 1976.

Salimans, T., Goodfellow, I., Zaremba, W., Cheung, V., Radford, A., and Chen, X. Improved techniques for training gans. In *Advances in Neural Information Processing Systems*, pp. 2234–2242, 2016.

Shang, C., Palmer, A., Sun, J., Chen, K.-S., Lu, J., and Bi, J. Vigan: Missing view imputation with generative adversarial networks. In *Big Data (Big Data), 2017 IEEE International Conference on*, pp. 766–775. IEEE, 2017.

Springenberg, J. T. Unsupervised and semi-supervised learning with categorical generative adversarial networks. *arXiv preprint arXiv:1511.06390*, 2015.

Sun, Y., Kamel, M. S., Wong, A. K., and Wang, Y. Cost-sensitive boosting for classification of imbalanced data. *Pattern Recognition*, 40(12):3358–3378, 2007.

Sutskever, I., Jozefowicz, R., Gregor, K., Rezende, D., Lillicrap, T., and Vinyals, O. Towards principled unsupervised learning. *arXiv preprint arXiv:1511.06440*, 2015.

Tarvainen, A. and Valpola, H. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. In *Advances in neural information processing systems*, pp. 1195–1204, 2017.

Troyanskaya, O., Cantor, M., Sherlock, G., Brown, P., Hastie, T., Tibshirani, R., Botstein, D., and Altman, R. B. Missing value estimation methods for dna microarrays. *Bioinformatics*, 17(6):520–525, 2001.

Turian, J., Ratinov, L., and Bengio, Y. Word representations: a simple and general method for semi-supervised learning. In *Proceedings of the 48th annual meeting of the association for computational linguistics*, pp. 384–394. Association for Computational Linguistics, 2010.

Van Buuren, S. *Flexible imputation of missing data*. Chapman and Hall/CRC, 2018.

Vincent, P., Larochelle, H., Bengio, Y., and Manzagol, P.-A. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th international conference on Machine learning*, pp. 1096–1103. ACM, 2008.

Wang, F. and Zhang, C. Label propagation through linear neighborhoods. *IEEE Transactions on Knowledge and Data Engineering*, 20(1):55–67, 2008.

Yoon, J., Jordon, J., and van der Schaar, M. Gain: Missing data imputation using generative adversarial nets. In *Proceedings of the 35th International Conference on Machine Learning*, pp. 5689–5698. PMLR, 2018.

Zhang, X., Zhao, J., and LeCun, Y. Character-level convolutional networks for text classification. In *Advances in neural information processing systems*, pp. 649–657, 2015.