Received August 14, 2021, accepted August 29, 2021, date of publication September 7, 2021, date of current version October 1, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3111105

NNR-GL: A Measure to Detect Co-Nonlinearity Based on Neural Network Regression Regularized by Group Lasso

MIHO OHSAKI[1], (Member, IEEE), NAOYA KISHIMOTO[2], HAYATO SASAKI[1], RYOJI IKEURA[2], SHIGERU KATAGIRI[1], (Life Fellow, IEEE), KEI OHNISHI[3], (Member, IEEE), YAKUB SEBASTIAN[4], AND PATRICK THEN[5], (Member, IEEE)

[1]Graduate School of Science and Engineering, Doshisha University, Kyoto 610-0321, Japan
[2]Faculty of Science and Engineering, Doshisha University, Kyoto 610-0321, Japan
[3]Graduate School of Computer Science and Systems Engineering, Kyushu Institute of Technology, Fukuoka 820-8502, Japan
[4]College of Engineering, IT and Environment, Charles Darwin University, Northern Territory 0810, Australia
[5]Faculty of Engineering, Computing and Science, Swinburne University of Technology, Sarawak 93350, Malaysia

Corresponding author: Miho Ohsaki (mohsaki@mail.doshisha.ac.jp)

This work was supported in part by Japan Society for the Promotion of Science (JSPS) Grants-in-Aid for Scientific Research (KAKENHI) under Grant 21K12018, and in part by the COVID-19 Research Project of Doshisha University.

ABSTRACT For finding keys to understand and elucidate a phenomenon, it is essential to detect dependences among variables, and so measures for that have been proposed. Correlation coefficient and its variants are most common, but they only detect a linear dependence (co-linearity) between two variables. Some recent measures can detect a nonlinear dependence (co-nonlinearity) by means of kernelization or segmentation. They are supposed to handle two variables only and open to discussion with regard to performance in detection and difficulty in setup. There is room for a novel measure based on Neural Networks (NNs), since usual NNs aim at prediction but not at variable dependence detection. For the high-performance detection of co-nonlinearities among multi variables, we propose a measure called NNR-GL based on Neural Network Regression (NNR) regularized by Group Lasso (GL). NNR-GL embodies the detection through multi-input single-output regression by NNR and regularization on the input layer by GL. NNR-GL then calculates how strong the detected co-nonlinearities are by unifying the regression performance and the weights on input variables. We conducted experiments using artificial data to examine the behaviors and fundamental effectiveness of NNR-GL. The performance was estimated by a comprehensive detection performance criterion (CDP-AUC in short), which is the mean of area under curves representing true positive and true negative detections. NNR-GL achieved the values of CDP-AUC from 0.7472 to 0.9681, where 0 means complete failure and 1 means complete success in detection. These values were consistently higher than those from 0.5972 to 0.9259 of the conventional measures for all the different conditions of dependence, data size, and noise rate. Consequently, the effectiveness and robustness of NNR-GL were clearly confirmed.

INDEX TERMS Machine learning, knowledge discovery, nonlinear dependence, measure to detect co-nonlinearity, regularization, robustness, neural network regression, group lasso.

I. INTRODUCTION

In this study, we propose and evaluate a novel neural-network-based measure that detects nonlinear dependences among multi variables.[1] Two backgrounds motivated us, where one is of variable dependence detection, and the other is of machine learning and knowledge discovery. Section I provides the first background, the second background, and the motivation and objective in Sections I-A, I-B, and I-C, respectively.

A. BACKGROUND OF VARIABLE DEPENDENCE DETECTION

Detecting dependences among variables is a common issue in various fields as the first step toward understanding and elucidating phenomena. Measures for that have been pro-

The associate editor coordinating the review of this manuscript and approving it for publication was Gianmaria Silvello.

[1] "dependence" not "dependency" is used following the use of "dependence" as a technical term in the related papers. "dependences" is used intentionally to mean multiple relationships among multiple variables.

TABLE 1. Interests and standpoints of machine learning (ML) and knowledge discovery (KD). What we focus on are in bold letters.

Interest of ML	A ML or MLs solve a problem in an inductive way with high performance instead of humans.	
Interest of KD	A human or humans solve a problem assisted by ML that provides suggestive results. There are roughly two standpoints depending on the needs and expertise of humans.	
	Standpoint of Analysis	It is not difficult for me, a human, to directly find and understand relationships such as dependences among variables. Hence, for rigorous analysis, I apply ML to variables selected by myself.
	Standpoint of Awareness	**It is difficult for me, a human, to directly find and understand relationships such as dependences among variables.** **Hence, for this help, I apply ML to variables before rigorous analysis.**

posed and applied to scientific and engineering disciplines. We actually found a large number of such applications by a survey for only the last 5 years, some of which are about the following: Medicine on cancer, Alzheimer's disease, and Covid-19 [1]–[3]. Brain science and engineering using electroencephalogram, electromyogram, and so on [4], [5]. Genomics and proteomics on emergent properties, clonal fate, and protein types [6]–[8]. Chemistry, physics, and material science on laser devices, ion energy, and photoelectrochemical power [9]–[11]. Environmental and earth science on climate and ocean dynamics [12]–[14]. Automotive and transportation engineering including self-driving techniques [15]–[17].

Once dependence is detected using a measure, it can become new knowledge on the focal phenomenon. It can also contribute to the selection of important variables and the improvement of regression/classification performance. Steps in variable dependence detection are illustrated in Fig. 1. In the first step, potential dependences among a wide variety of variables are detected that give hints for new hypotheses on a target phenomenon. In the second step, these dependences are narrowed down and brushed up to promising dependences based on knowledge specific to the domain. In the third step, the promising dependences are interpreted and formulated rigorously on the basis of domain knowledge. This result becomes newly established knowledge on the phenomenon. A traditional way starts from the second or third step for limited numbers and types of variables, assuming a wealth of domain knowledge. This way is important, of course, but will not be sufficient for perceiving unexpected dependences behind various variables. We hence consider a measure for the first step to detect potential dependences that are difficult to perceive only with known domain knowledge.

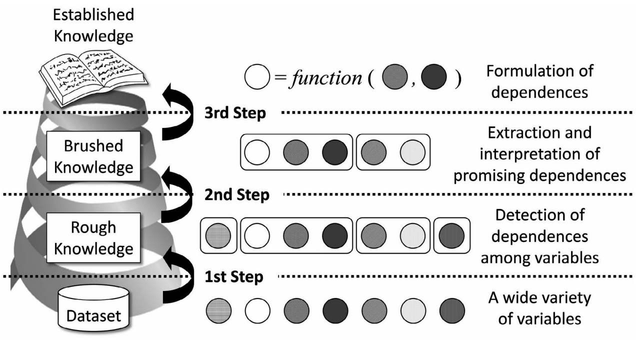

Established Knowledge
○ = *function* (●, ●)
Formulation of dependences
3rd Step
Brushed Knowledge
Extraction and interpretation of promising dependences
2nd Step
Rough Knowledge
Detection of dependences among variables
1st Step
Dataset
A wide variety of variables

FIGURE 1. Steps to established knowledge on variable dependences.

As an example of the stepwise process, suppose clinical data accumulated in hospitals. Applying a measure in the first step enables to detect potential dependences related to health risks from variables including patients' environments, lifestyles, clinical examinations, diagnoses, and treatments. That provides awareness of dependences hardly noticeable with known medical knowledge only. The potential dependences can be analyzed and formulated in the second and third steps to reach novel medical knowledge. Another example is car driving support, where sensor data on users, devices, and environments are available. In the first step, a measure detects potential dependences causing traffic accidents. Their analysis and formulation lead to new knowledge on safe driving in the second and third steps. The same applies to other fields.

Most of the conventional measures aim at the detection of a dependence between two variables. Correlation coefficient and its variants are most common, but because of their assumption of a linear dependence (co-linearity), they cannot detect a nonlinear dependence (co-nonlinearity) [18]–[21]. In recent years, measures able to detect a co-nonlinearity have been proposed. Some of them are based on mapping and correlation coefficient/covariance [22]–[26]. The others are based on segmentation and mutual information [27], [28]. Their abilities in expressing a co-nonlinearity and difficulties in setting are still controversial. Both the linear and nonlinear measures cannot directly detect dependences among more than two variables.

B. BACKGROUND OF MACHINE LEARNING AND KNOWLEDGE DISCOVERY

Looking at Machine Learning (ML) and its use for finding new knowledge i.e. Knowledge Discovery (KD), they share technical features but pursue different interests as in Table 1. For the interest of ML, Neural Networks (NNs) including deep learning attract great attention and achieve high performances in nonlinear prediction. Such NNs implicitly model dependences among variables, but do not explicitly provide the modeled dependences. In terms of variable dependence detection, it would be worth utilizing NNs for the interest of KD from the standpoint of awareness.

As far as we surveyed, surprisingly we did not find NNs that directly aim at assisting awareness except a few related work [29]. Specifically, NN-based measures representing how strong variable dependences were not found, despite the high potential of NNs in nonlinear modeling. The main reason would be that NNs are a black box mixing input variables up in a way not understandable in the domain context (what is called "implicit distributed representation"). In our past research collaboration with medical experts, they wanted to know clear relationships of original variables that were clinical test results having medical meanings. They did not want to get into the features transformed inside of NNs that were difficult to understand medically. This episode is just our experience but gives a suggestion that focusing on original variables (equivalently, the input layer of a NN) is essential for awareness and understanding.

C. MOTIVATION AND OBJECTIVE

The background in Section I-A indicates the need for a novel measure to detect nonlinear dependences among multi variables in the first step in Fig. 1. That in Section I-B indicates the high but not demonstrated potential of NNs in variable dependence detection and the way to demonstrate it for the awareness in KD in Table 1.

These indications encourage us to propose a NN-based measure called NNR-GL, in which Neural Network Regression (NNR) [30], [31] models co-nonlinearities among multi-input single-output variables, and Group Lasso (GL) [32], [33] selects contributable input variables by the regularization on the NNR's input layer. Thanks to GL on the input layer only, NNR-GL provides explicit localized representation on which input and output variables are dependent and accepts any kinds of NNRs. As the measure value, NNR-GL outputs a quantity representing the strength of input-output co-nonlinearity by unifying the performance of regression and the weight on each input variable. To analytically evaluate NNR-GL, we conduct experiments in comparison with the conventional measures using artificial datasets with known correct dependences. NNR-GL, of which rough idea partially appeared in our past study [34], is now thoroughly proposed and evaluated in our present study.

The main contributions of this paper are summarized as the following **#1**, **#2**, and **#3**.

#1: From a broad perspective, the detection of unexpected complex dependences behind various variables is necessary as the first step to clarifying phenomena. Our study, which lies in the interdiscipline between ML/KD and other sciences and engineering, provides people in these areas a novel way to achieve this first step. Furthermore, the study expands the utilization of NNs from prediction in place of humans to dependence detection for human awareness.

#2: As a concrete solution, our proposed measure NNR-GL makes the detection of multi nonlinear dependences possible in an accurate and robust manner. The ideas of NNR-GL (namely, nonlinear modeling by NNR, variable selection by GL, the way to quantify detected dependences, and robustness by averaging) are simple but applicable to various NNs.

#3: The experimental results in our study ensure the effectiveness of NNR-GL. They are helpful as the baseline performances of NNR-GL and the conventional measures when one wants to use these measures for his/her task. Unlike the past studies, our study introduces an evaluation methodology that is quantitative and both-sided for true positives and true negatives. This methodology can be used in future related work.

In this paper, Section I provides the background and objective as above. Section II reviews conventional measures for variable dependence detection and their remaining problems. Section III proposes the novel measure NNR-GL to detect nonlinear dependences among multi variables. Section IV designs an evaluation experiment to analytically examine the fundamental effectiveness of NNR-GL with artificial datasets. Section V reports the experiment and discusses the performances of NNR-GL and the conventional measures. As a stepping stone toward practicality, Section VI reports a pilot experiment to apply NNR-GL to real benchmark datasets. Finally, Section VII concludes the paper and gives some directions for future work.

II. CONVENTIONAL MEASURES

Our proposed measure NNR-GL is for the purpose of detecting nonlinear dependences among variables by introducing machine learning techniques NNR and GL. Thus, in Section II on related work, variable dependence is defined at first in Section II-A. Next, the conventional measures for variable dependence detection are reviewed in Section II-B. Finally, NNs, NNR, Lasso, GL, and their combinations are reviewed in Section II-C.

A. DEFINITION OF VARIABLE DEPENDENCE

In general, the independence between two random variables X and Y is defined as Equ. (1) using the joint probability mass or density function $p_{XY}(x, y)$, the marginal probability mass or density function of X $p_X(x)$, and that of Y $p_Y(y)$ [35], [36]. There is another definition Equ. (2) based on Fourier transform, i.e. the characteristic functions $c^{XY}(s, t)$, $c^X(s)$, and $c^Y(t)$. The greater the difference between the left and right sides of Eqs. (1) or (2), the stronger the dependence.

$$p_{XY}(x, y) = p_X(x)\, p_Y(y), \quad \forall x, y \tag{1}$$

$$c^{XY}(s, t) = c^X(s)\, c^Y(t), \quad \forall s, t \tag{2}$$

The measures of a dependence between two variables can be categorized viewing from three aspects. The first aspect is the definition of dependence, whether it is based on probabilities or characteristic functions. The second aspect is the formulation of dependence, how the left and right side difference is described mathematically such as the norm of subtraction and the logarithm of ratio. The third aspect is the assumption of dependence, what shape is assumed for the dependence function, that is linear or nonlinear. Generally, it is emphasized to unlock hidden complex dependences, and so we discuss conventional measures as to the third aspect.

TABLE 2. Variable dependence measures categorized based on shape assumption. Our focal points are in bold letters.

Linear	Correlation Coefficient (CC) and its variants (See Section II-B1).	
Nonlinear	Mapping-based	Distance Correlation Coefficient (DCC) and its variants (See Section II-B2). Hilbert Schmidt Independence Criterion (HSIC) and its variants (See Section II-B2).
	Segmentation-based	Maximal Information Coefficient (MIC) and its variants (See Section II-B3).
	NN-based	**No explicit measures, and so we are motivated (See Section II-C).**

B. MEASURES TO DETECT VARIABLE DEPENDENCE

Variable dependence measures are categorized into linear and nonlinear in Table 2. Based on our survey, the sub-categories of nonlinear measures are mapping-based, segmentation-based, and (potentially) NN-based. The linear measures are Correlation Coefficient (CC) and its variants. The elemental and dominant nonlinear measures based on mapping are Distance Correlation Coefficient (DCC), Hilbert Schmidt Independence Criterion (HSIC), and their variants. Those based on segmentation are Maximal Information Coefficient (MIC) and its variants. Those based on NNs are not found so far, and thus we propose a NN-based measure later in Section III. CC, DCC, HSIC, and MIC are explained hereinafter in this Section II-B. Note that their definitions for the population X and Y are skipped, but the definitions for the sample x and y are provided. In Section II-C, the existing NNs regularized by GL are discussed, which are not measures but technically related to NNR-GL.

1) LINEAR MEASURES

CC is the most basic measure that assumes a linear dependence between two variables [18]–[21]. It estimates the difference between the left and right sides of Equ. (1), from the viewpoint of how the variables co-vary around a linear line. Equ. (3) formulates the definition of CC, where x and y are variables. x_i is the ith observation of x, $\overline{x}$ is the sample mean of x, and the same applies to y_i and $\overline{y}$. N is the number of sample points. $\mathrm{COV}_{\mathrm{smp}}(x, y)$ is the sample covariance of x and y. CC ranges from -1 to $+1$, and its larger absolute value means a stronger dependence. CC and its variants cannot detect nonlinear dependences and which variable causes which one.

$$\mathrm{CC}(x, y) = \frac{\mathrm{COV}_{\mathrm{smp}}(x, y)}{\sqrt{\mathrm{COV}_{\mathrm{smp}}(x, x)}\sqrt{\mathrm{COV}_{\mathrm{smp}}(y, y)}}$$
$$\mathrm{COV}_{\mathrm{smp}}(x, y) = \frac{1}{N}\sum_{i=1}^{N}(x_i - \overline{x})(y_i - \overline{y}) \qquad (3)$$

2) NONLINEAR MEASURES BASED ON MAPPING

Distance Correlation Coefficient (DCC) is a measure based on characteristic functions [22], [23]. It calculates the L_2 norm distance between the left and right sides of Equ. (2) for each of x and y, x and x, and y and y. DCC then outputs the normalized distance of x and y as the quantity representing dependence. In Equ. (4), $c_N^{xy}(s, t)$, $c_N^x(s)$, and $c_N^y(t)$ are the characteristic functions. $\mathrm{DIST}_{\mathrm{smp}}(x, y)$ is the distance estimated with the N sample points. DCC ranges from 0 to $+1$, and its larger value means a stronger dependence. It can detect co-nonlinearity with no assumption, but cannot explicitly specify the family of nonlinear functions.

$$\mathrm{DCC}(x, y) = \frac{\mathrm{DIST}_{\mathrm{smp}}(x, y)}{\sqrt{\mathrm{DIST}_{\mathrm{smp}}(x, x)}\sqrt{\mathrm{DIST}_{\mathrm{smp}}(y, y)}}$$
$$\mathrm{DIST}_{\mathrm{smp}}(x, y) = \frac{1}{N}\sum_{s,t=1}^{N} \|c_N^{xy}(s, t) - c_N^{x}(s)c_N^{y}(t)\|_2^2 \qquad (4)$$

Another mapping-based measure is Hilbert Schmidt Independence Criterion (HSIC) [24]–[26]. It detects a nonlinear dependence between two variables via kernelization. The difference between the left and right sides of Equ. (1) is estimated by the covariance in the reproducing kernel Hilbert space. HSIC is defined as Equ. (5), where $\phi(x, \theta_\phi)$ is the mapping function of x with its hyperparameter θ_ϕ, and $\psi(y, \theta_\psi)$ is that of y with its hyperparameter θ_ψ. The actual calculation is done using the inner products of the mapping functions or the kernel functions $k(x, x')$ and $l(y, y')$. The range of HSIC depends on kernel functions and hyperparameters, but can be normalized to -1 to $+1$. The larger the absolute value of HSIC, the stronger the dependence. Although HSIC is able to express various families of nonlinear functions, it matters how to select kernel functions and set their hyperparameters.

$$\mathrm{HSIC}(x, y) = \mathrm{COV}_{\mathrm{smp}}(\phi(x, \theta_\phi),\ \psi(y, \theta_\psi))$$
$$k(x, x') = <\phi(x, \theta_\phi),\ \phi(x', \theta_\phi)>$$
$$l(y, y') = <\psi(y, \theta_\psi),\ \psi(y', \theta_\psi)> \qquad (5)$$

3) NONLINEAR MEASURES BASED ON SEGMENTATION

Maximal Information Coefficient (MIC) performs segmentation of an original variable space [27], [28]. It then detects a nonlinear dependence by accumulating the mutual information between two variables over all the segments. Seeing Equ. (6), MIC can be understood to be the mean of piecewise log ratios of the left and right sides of Equ. (1). It has hyperparameters determining the maximal grid size and the maximal segment size. The parameters to optimize via training are the kth segmentation pattern for x and the lth segmentation pattern for y. $\mathrm{MI}_{\mathrm{smp}}(SEG_k, SEG_l)$ is the sample mutual information when the sets of segments are SEG_k and SEG_l. $p_{XY,kl}(x, y)$, $p_{X,k}(x)$, and $p_{Y,l}(y)$ are the probabilities estimated based on the ratio of sample points in SEG_k and SEG_l. $\mathrm{MIC}(x, y)$ is the maximal mutual information obtained with the optimal sets of segments SEG_{k*} and SEG_{l*}.

MIC ranges from 0 to $+1$, where the larger value means stronger dependence. It can express any families of nonlinear functions representing a dependence, but may cause overfitting if all the data are used for segmentation pattern search. The advantage that MIC detects any of one-to-one, one-to-many, many-to-one, and many-to-many relationships backfires in identifying which variable causes which one. The versatility of MIC is said to be controversial [37].

$$\begin{aligned}\mathrm{MIC}(x, y) &= \mathrm{MI}_{\mathrm{smp}}(SEG_{k*}, SEG_{l*})\\ \mathrm{MI}_{\mathrm{smp}}(SEG_k, SEG_l)&\\ &= \sum_{\substack{x \in SEG_k\\ y \in SEG_l}} p_{XY,kl}(x, y) \log \frac{p_{XY,kl}(x, y)}{p_{X,k}(x) p_{Y,l}(y)} \qquad (6)\end{aligned}$$

The measures CC, DCC, HSIC, and MIC are established basic ones. Consequently, recent related studies are on the analysis, assistance, expansion, and utilization of these measures. There are studies that analyzed the characteristics and behaviors of the measures theoretically and/or empirically [38]–[41]. For the assistance of the measures, other studies proposed algorithms to accelerate measure value calculation [42], [43]. Other studies aim at expanding and utilizing the measures in a general manner. They proposed methods based on the measures for the following: statistical tests [44]–[46], robustness improvement [47]–[49], algorithmic strategies for multi dependence detection [50], [51], feature selection [52]–[56], and feature extraction [57], [58]. The other studies are to utilize the measures in specific domains, the applications in short, which are as given in Section I-A. As suggested by the many related studies, CC, DCC, HSIC, and MIC are still the gold standards and so should be the competitors to our proposed measure.

C. NEURAL NETWORKS WITH GROUP LASSO

There is a superficially subtle yet fundamental distinction between NNs for prediction and NN-based measures for variable dependence detection. NNs with Lasso have been proposed in the former sense but not in the latter sense that we pursue. However, they are discussed below because of their technical aspects common to our study. The simple L_1 regularization Lasso is not efficient to selectively strengthen important neurons, since Lasso adjusts weights on edges regardless which edges are connected to which neurons. GL can solve this problem by grouping edges connected to the same neuron and adjusting the weights groupwise. Hence, NNs with GL attract attentions as to sparse modeling and variable selection aiming at better prediction.

For sparse modeling in image recognition, Zhao *et al.* [59] proposed a Deep NN (DNN) classifier that consists of sub-networks representing modalities. The weights in each sub-network are grouped and regularized by GL. Similar attempts were done in the literatures [60]–[62]. In speech recognition, Ochiai *et al.* [63] proposed a hybrid of DNN and Hidden Markov Model, where the weights corresponding to each neuron in hidden layers are grouped and regularized by GL. These methods successfully made the DNNs sparse and improved the performances. The effectiveness of NNs with GL was ensured, but their purpose and way (prediction by a NN with GL on all the layers) differ from ours (variable dependence detection utilizing a NN with GL on the input layer).

For variable selection, there are studies to regularize the input layer of a NN with GL. They used ‘‘feature selection’’ to refer to selecting input variables, despite features represented by hidden layers were out of their scope. That is confusing, and so we use ‘‘variable selection’’ instead. Li *et al.* [64] employed GL and $L_{1/2}$ regularization on the input layers of multilayer feedforward NNs including NNR and Autoencoder (AE). Han *et al.* [65] made a similar attempt regarding AE. Zhang *et al.* [66] proposed a smooth differentiable GL on the input layer for efficient training. These methods remove input variables independent of the output variable under the fixed input and output variables. If a strong dependence exists among input variables, a part of them are randomly selected as their representatives by GL. We call this problem ‘‘multi-co-nonlinearity’’ (or ‘‘multi-col-nonlinearity’’) named after multicolinearity (or multicollinearity) [18], [19]. The reproducibility of such selection might be open to question. As is obvious, the methods do not concern detecting nonlinear dependences; they are not co-nonlinearity measures.

III. PROPOSAL OF NNR-GL

The background in Section I and the related work in Section II motivated and led us to propose NNR-GL. In Section III, we concretize NNR-GL by providing the underlying ideas and algorithm design in Section III-A and the formulation and definition in Section III-B. Additionally, we answer potential questions that the readers may have about NNR-GL in Section III-C.

A. UNDERLYING IDEAS AND ALGORITHM DESIGN

Let us begin with the summary of what is suggested in Section II. Aiming at variable dependence detection, there are linear and nonlinear (mapping-based and segmentation-based) measures, but no NN-based ones. The past successes of NNs with GL in prediction encourage us to devise a novel NN-based measure. The core of the measure should be a NNR with GL regularizing the input layer. Unlike the

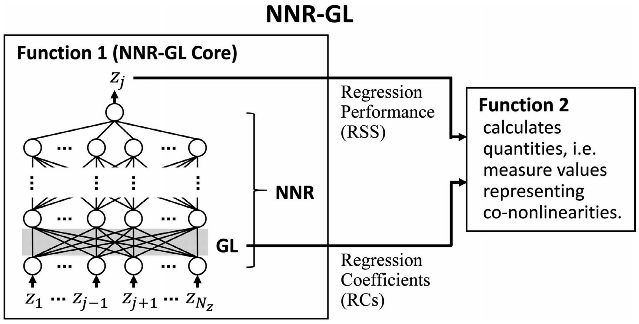

NNR-GL
Function 1 (NNR-GL Core)
z_j
NNR
GL
$z_1 \cdots z_{j-1}$ $z_{j+1} \cdots z_{N_z}$
Regression Performance (RSS)
Regression Coefficients (RCs)
Function 2 calculates quantities, i.e. measure values representing co-nonlinearities.

FIGURE 2. Conceptual design of the proposed measure NNR-GL.

```
Function 1: Iterative Nonlinear Regression
Input: $D_{tr}$, $D_{va}$, $D_{te}$
Output: Sets of RCs and RSS
 1: % Train and validate NNR-GL Core while changing the
    output variable and initialization.
 2: for $hs \in$ all hyperparameter settings do
 3:     for $j = 1$ to $N_z$ do
 4:         for $k = 1$ to $N_{rnd}$ do
 5:             Under $hs$, train NNR-GL Core initialized with $k$
                to regress the $j$th variable $z_j$ on $z_1, \cdots, z_{j-1}, z_{j+1},$
                $\cdots, z_{N_z}$ using $D_{tr}$.
 6:             Validate the trained NNR-GL Core using $D_{va}$.
 7:             Record the best hyperparameter setting $bhs$ and
                the best parameter setting $bps$ for $j$ and $k$.
 8:         end for
 9:     end for
10: end for
11: % Test NNR-GL Core for each output variable and initialization.
12: for $j = 1$ to $N_z$ do
13:     for $k = 1$ to $N_{rnd}$ do
14:         Under the $bhs$ and $bps$ corresponding to $j$ and $k$,
            apply the trained and validated NNR-GL Core to
            regress the $j$th variable $z_j$ on $z_1, \cdots, z_{j-1}, z_{j+1}, \cdots,$
            $z_{N_z}$ using $D_{te}$.
15:         Extract RCs and RSS from the trained, validated,
            and tested NNR-GL Core.
16:     end for
17: end for
18: Output the sets of RCs and RSS for all $j$ and $k$.
```

existing NNs with GL, the measure should exchange input and output variables. Motivated by the above, we propose a NN-based measure called NNR-GL that detects nonlinear dependences among multi variables with high performance. We design NNR-GL to have Function 1 (iterative nonlinear regression) and Function 2 (measure value calculation) as shown in Fig. 2.

1) FUNCTION 1

NNR-GL Core plays the main role here. In training and validation phases, NNR-GL Core models nonlinear dependences between multi input variables and a single output variable. It makes a contrast of the weights or the regression coefficients (RCs) on input variables according to their contributions to regression. In a test phase, the trained and validated NNR-GL Core estimates the regression performance or the residual sum of squares (RSS). This process is done individually to each variable as the output. If high reproducibility is required, regression is repeated and averaged over different initializations expecting a kind of ensemble effect. Consequently, the sets of RCs and RSS are obtained for all the different output variables and initializations.

```
Function 2: Measure Value Calculation
Input: Sets of RCs and RSS
Output: $MV_{te} = [\ qntty_{jm} \mid j, m = 1, 2, \cdots, N_z\ ]$
 1: % Calculate the measure values.
 2: for $j = 1$ to $N_z$ do
 3:     for $m = 1$ to $N_z$ do
 4:         for $k = 1$ to $N_{rnd}$ do
 5:             if $j == m$ then
                    Set the quantity $qntty_{jmk}$, which represents the
                    dependence between the $j$th and $m$th variables
                    under the $k$th initialization, to 1.
 6:             else
 7:                 Calculate $qntty_{jmk}$ by unifying the RC and RSS
                    corresponding to $j$, $m$, and $k$.
 8:                 Overwrite the smaller with the larger out of
                    $qntty_{jmk}$ and $qntty_{mjk}$.
 9:             end if
10:         end for
11:         Calculate the mean of $qntty_{jmk}$ over the $N_{rnd}$ initial-
            izations and set $qntty_{jm}$ to this mean.
12:     end for
13: end for
14: Output the measure value matrix $MV_{te}$ consisting of
    $qntty_{jm}$ for $j, m = 1, 2, \cdots, N_z$.
```

There are two reasons why changing the output variable. One reason is to solve the multi-co-nonlinearity problem, which was discussed at the end of Section II. Executing NNR-GL Core for different output variables reveals dependences possibly masked by multi-co-nonlinearity. Regarding the other reason, suppose $x_3 = f(x_1)+g(x_2)$. The dependence between x_3 and x_1 and that between x_3 and x_2 are detectable when x_3 is regressed on x_1 and x_2. Meanwhile, these dependences are less or not detectable when x_1 or x_2 is regressed on the others, because the inverse mapping of nonlinear $f(x_1)$ and $g(x_2)$ is one-to-many. To solve this problem, NNR-GL runs NNR-GL Core for different output variables and reveals dependences possibly masked by inverse mapping. In other words, NNR-GL takes into account which variable causes which one. In the example, the dependence between x_3 and x_1 is adopted when x_3 is regressed, but it is discarded when x_1 is regressed.

Function 1 is embodied as in the pseudocode. It receives training, validation, and test sets D_{tr}, D_{va}, and D_{te} and returns the sets of RCs and RSS. For the hyperparameter setting hs, the jth output variable, and the kth initialization, NNR-GL Core gets through training using D_{tr} and validation using D_{va} (Lines 1 to 10). The best hyperparameter setting bhs and the best parameter setting bps are obtained for each output variable with each initialization. NNR-GL Core with the settings bhs and bps moves onto test using D_{te}. It models dependences between the output variable z_j and the input variables z_1 to z_{N_z} except z_j (Lines 11 to 17). In the end, Function 1 outputs the sets of RCs and RSS for all the output variables and initializations.

2) FUNCTION 2

Given the sets of RCs and RSS from Function 1, NNR-GL unifies the RC and RSS for each pair of an input variable and an output one into a quantity representing their co-nonlinearity. This pairing clarifies each input-output co-nonlinearity, even if there exist multi-co-nonlinearities in input variables. The quantity for variables x and x' is obtained both when x is regressed and when x' is regressed. Out of such quantities, NNR-GL picks up the larger one representing the dependence not in inverse mapping but in forward mapping. For reproducibility, quantities under the same condition are averaged over their different initializations. NNR-GL then outputs the sets of quantities as the measure values of all the dependences.

As in the pseudocode, Function 2 receives the sets of RCs and RSS and returns the measure value matrix. The RC and RSS, which were obtained when the jth variable was output and the mth variable was one of inputs under the kth initialization, are unified into the quantity $qntty_{jmk}$ (Lines 1 to 13). This quantity represents the co-nonlinearity generalized for D_{te} between the jth and mth variables. This calculation includes selecting the larger one of $qntty_{jmk}$ and $qntty_{mjk}$ and averaging it over initializations. Finally, Function 2 outputs the measure value matrix MV_{te} of which element is $qntty_{jm}$ for all the combinations of variables.

B. FORMULATION AND DEFINITION

We formulate the model structure, objective function, and optimization of NNR-GL Core in Function 1. NNR-GL Core accepts any types of NNs for regression, but a Multilayer Perceptron (MLP) is used for simplicity. NNR-GL Core has $N_L + 1$ layers, where the 0th is the input layer, the 1st to $(N_L - 1)$th are the hidden layers, and the N_Lth is the output layer. In the input layer, there are N_z neurons corresponding to input variables $z_1, z_2, \cdots, z_{N_z}$. In Fig.2, the regression target variable z_j is excluded from the input layer for ease of understanding. However, z_j is included here for a general formulation, and its weight is fixed to 0 to be excluded parametrically. In the lth layer, there are $N_z^{(l)}$ neurons, $z_1^{(l)}$, $z_2^{(l)}, \cdots, z_{N_z^{(l)}}^{(l)}$. $w_{mn}^{(l)}$ is the weight on the edge from the mth neuron in the $(l-1)$th layer to the nth neuron in the lth layer. $\mathbf{W}^{(l)}$ given in Equ. (7) is the weight matrix containing $w_{mn}^{(l)}$. $\mathbf{W}$, which gathers $\mathbf{W}^{(l)}$ over $1 \leq l \leq N_L$, is the weight matrix to set by training. As mentioned above, $w_{jn}^{(1)}$ is fixed to 0. The numbers of layers and neurons are hyperparameters to set by validation.

$$\mathbf{W}^{(l)} = \begin{bmatrix} w_{11}^{(l)}, & w_{12}^{(l)}, & \cdots, & w_{1N_z^{(l)}}^{(l)} \\ w_{21}^{(l)}, & w_{22}^{(l)}, & \cdots, & w_{2N_z^{(l)}}^{(l)} \\ \ddots & \ddots & w_{mn}^{(l)} & \ddots \\ w_{N_z^{(l-1)}1}^{(l)}, & w_{N_z^{(l-1)}2}^{(l)}, & \cdots, & w_{N_z^{(l-1)}N_z^{(l)}}^{(l)} \end{bmatrix} \quad (7)$$

The objective function of NNR-GL Core is defined in Equ.(8), which is composed of the terms as to regression performance and variable selection. λ is a hyperparameter balancing the two terms. s and N_{tr} are the index and number of training sample points, respectively. z_{js} is the sth observation of the output variable z_j. $\hat{z}_{js}(\mathbf{W})$ is the prediction of z_{js} by NNR-GL Core with the weight matrix $\mathbf{W}$. The first term is the RSS between z_{js} and $\hat{z}_{js}(\mathbf{W})$. The second term is the GL regularization to pick up contributable ones out of input variables z_1 to z_{N_z} except z_j. The weights $w_{m1}^{(1)}$, $w_{m2}^{(1)}$, $\cdots$, $w_{mN_z^{(1)}}^{(1)}$ on the edges connected to the mth neuron of the input layer are grouped into a weight vector $\mathbf{w}_m^{(1)}$. The L_2 norm $||\mathbf{w}_m^{(1)}||_2$ fuses the weights and can be understood as the RC of the mth input variable z_m. The sum of $||\mathbf{w}_m^{(1)}||_2$ is the L_1 norm of such RCs and yields the effect of variable selection.

$$J_j(\mathbf{W}) = \frac{1}{N_{tr}} \sum_{s=1}^{N_{tr}} (z_{js} - \hat{z}_{js}(\mathbf{W}))^2 + \lambda \sum_{m=1,\ \neq j}^{N_z} ||\mathbf{w}_m^{(1)}||_2 \quad (8)$$

NNR-GL Core is trained by minimizing the objective function. Any types of optimization methods are acceptable, but here we use the most standard ones, Backpropagation and Stochastic Gradient Descent [30], [31]. Equ. (9) shows the parameter update rule. The partial derivative of the first term $J_j^{\text{1st term}}(\mathbf{W})$ of the objective function is derived by the chain rule, where $\mathbf{z}$ denotes the outputs of all the neurons. The partial derivative of the second term is directly and easily derived for not only MLP but also other NNs. After training and validating, the weight vector for each neuron of the input layer i.e. RC is obtained. RSS is also obtained via testing.

$$\mathbf{W} \leftarrow \mathbf{W} - \rho \frac{\partial J_j(\mathbf{W})}{\partial \mathbf{W}}$$
$$\frac{\partial J_j(\mathbf{W})}{\partial \mathbf{W}} = \frac{J_j^{\text{1st term}}(\mathbf{W})}{\partial \mathbf{z}} \frac{\partial \mathbf{z}}{\partial \mathbf{W}} + \lambda \frac{\mathbf{w}_m^{(1)}}{||\mathbf{w}_m^{(1)}||_2} \quad (9)$$

In Equ.(10), we define the quantity representing a co-nonlinearity calculated in Function 2. It is a multiplication of normalized RC and RSS. $\mathbf{W}_x^2$ is the normalized RC, that is the normalized squared L_2 norm of the weight vector of the focal input variable x. $\mathbf{w}_m^{(1)}$ is the weight vector of the mth input variable, and M is the number of input variables $N_z - 1$. $\mathbf{W}_x^2$ is normalized to sum to 1 or to make the maximum to 1. $\mathbf{R}_y^2$ is the normalized RSS, that is the coefficient of determination for the output variable y. y_i is the ith observation, $\hat{y}_i$ is the ith prediction, $\bar{y}$ is the mean of y_i, and N_{te} is the size of a test dataset. $\mathbf{R}_y^2$ basically ranges from 0 to 1. There is a possibility that $\mathbf{R}_y^2$ happens to be less than 0 when regression fails. Such a negative value means no dependence, and so NNR-GL replaces it with 0. Finally, $\mathbf{W}_x^2$ and $\mathbf{R}_y^2$ are square-rooted and multiplied into the quantity or the measure value. Averaging it over initializations can raise the reproducibility of detection.

$$\mathrm{NNR\text{-}GL}(x, y) = \sqrt{\mathbf{W}_x^2}\sqrt{\mathbf{R}_y^2}$$

$$\mathbf{W}_x^2 = \begin{cases} \dfrac{||\mathbf{w}_x^{(1)}||_2^2}{\sum\limits_{m=1}^{M} ||\mathbf{w}_m^{(1)}||_2^2} \\ \text{or} \\ \dfrac{||\mathbf{w}_x^{(1)}||_2^2}{\max\limits_{m} ||\mathbf{w}_m^{(1)}||_2^2} \end{cases} \quad \mathbf{R}_y^2 = 1 - \frac{\sum\limits_{i=1}^{N_{te}} (y_i - \hat{y}_i)^2}{\sum\limits_{i=1}^{N_{te}} (y_i - \bar{y})^2} \tag{10}$$

C. ANSWERS TO POTENTIAL QUESTIONS

We here answer potential questions on our measure NNR-GL. Questions and answers (QAs) concerning meaningfulness and significance are QAs 1 to 3, and those concerning technical aspects are QAs 4 to 7 as follows.

QA1: "NNs do not provide the explicit mathematical functions of dependences. Is NNR-GL based on NNs really meaningful?" This question intends the second and third steps to brush up and formulate promising dependences (See Section I-A). These steps should be taken after the detection of potential dependences done by measures including NNR-GL in the first step for human awareness.

QA2: "There exist NNs with GL. Is NNR-GL really novel?" NNs with GL actually have been studied [59]–[66]. They are NNs themselves aiming at sparse modeling or variable selection, but not measures aiming at variable dependence detection (See Section II-C). They make non-contributable neurons perish for better prediction. In contrast, NNR-GL is a measure containing a NN with GL in Function 1 to detect co-nonlinearities, followed by Function 2 to derive quantities representing the detected co-nonlinearities.

QA3: "What is the difference between NNR-GL and conventional nonlinear measures?" NNR-GL is free from the problems which the conventional mapping-based and segmentation-based approaches suffer from, i.e. the limitation to two variables and the limited families of nonlinear functions. Modeling by conventional measures is done pairwise, on the other hand, modeling by NNR-GL is done output-variable-wise. NNR-GL can simultaneously model nonlinear dependences among multi variables with high performance. The concise mechanism of GL regularizing only the input layer enables NNR-GL to accept various kinds of NNRs.

QA4: "Why does not NNR-GL take the weights over all the neurons and layers into account?" The reason why not all the weights are regularized by GL is as below. If so, neurons highly graded by GL in the input layer may happen to be degraded by GL in the subsequent layers and vice versa. GL on only the input layer localizes variable selection to avoid such cancellation. Moreover, it is easy to embed and so enables NNR-GL to accept various NNRs. The reason why not all the weights are included in the measure value is the following. Neurons in the subsequent layers are indirectly but commonly connected to neurons in the input layer. Their weights are common for the input variables and unnecessary in measure value calculation.

QA5: "Is it reasonable to use RCs and RSS for variable dependence detection?" When the least squares is used in single-input linear regression, RC becomes the covariance between input and output variables x and y divided by the variance of x [67], [68]. RC is thus equivalent to the CC between x and y when y has a variance of 1. Remember that CC is a measure formulating the difference between the left and right sides of Equ. (1). Therefore, RC is a kind of measures. RSS normalized by the total sum of squares is called the coefficient of determination. It equals the square of CC between x and y [67], [68], and so RSS can be a measure as well as RC. The above applies to multi-input nonlinear regression and supports the use of RCs and RSS.

QA6: "NNR-GL requires training, validation, and test sets split from a dataset. Is NNR-GL really better than measures with no split?" This issue is not specific to NNR-GL. The ultimate goal of a measure is to detect dependences in a population (namely, generalized dependences) using a finite sample. For that, going through training, validation, and test is essential. Strict assumptions such as linearity make a measure free from training (optimizing the shape of a dependence function) and validation (optimizing the family of dependence functions). In return, the measure sacrifices a chance to detect generalized dependences that are unexpected and nonlinear. To overcome this problem, NNR-GL goes through training, validation, and test without too many assumptions.

QA7: "How much is the computational complexity of NNR-GL?" Most of the computational complexity depends on what NNR is used in NNR-GL. One needs to select a NNR suitable for the speed required in his/her specific task, especially considering if real-time processing is required or not. The number of combinations affects the time for computation, too. For the conventional measures, it is N_z^2 due to their pairwise processing. For NNR-GL, it is $N_z \times N_{rnd}$, where N_z and N_{rnd} are the numbers of variables and initializations, respectively. The experimental results given later in Section V indicated that the number around 3 was sufficient for N_{rnd}. Therefore, the number of combinations for NNR-GL becomes considerably smaller than that for the conventional measures, when dealing with many variables.

IV. EXPERIMENTAL DESIGN

To fairly evaluate our proposed measure NNR-GL through comparison with the conventional measures, the experiment should be well designed. Section IV discusses the design of experiment in various aspects below: the direction and outline in Section IV-A, the competitors and settings in Section IV-B, the datasets in Section IV-C, and the evaluation criteria and methods in Section IV-D.

A. DIRECTION AND OUTLINE

Generally, the evaluation of variable dependence measures should be done stepwise [27]. At first, it is examined whether a measure behaves as expected and detects the known correct dependences (that is, fundamental effectiveness is examined). Artificially synthesized datasets meet this purpose. Such datasets enable to mathematically design variable dependences and to check the success or failure in detection using the known dependences. Next, it is examined whether a measure detects known correct dependences for real data (practical effectiveness). Using benchmark datasets on real domains with known dependences is appropriate for that. Finally, it is examined whether a measure detects unknown useful dependences in real world problems (practical usefulness). The present experiment examines the fundamental effectiveness of NNR-GL.

B. COMPETITORS AND SETTINGS

The hyperparameters, parameters, and settings of NNR-GL are listed in Table 3. To identify the baseline of performance, we decided to use the most traditional NN structure for NNR-GL, which was a MLP with sigmoid activation functions. NNR-GL is compared to the four representative measures reviewed in Section II-B, namely CC, DCC, HSIC, and MIC.

CC does not involve any hyperparameters and parameters because of its linearity [18], [21]. DCC is based on the norm of the distance of characteristic functions. It has no hyperparameters and parameters to preset or optimize [22]. HSIC is based on the covariance of data mapped by kernelization. The width of a Gaussian kernel function σ and the test statistic a are hyperparameters, which we set to the recommended values [25]. HSIC has no parameters. MIC is based on the mutual information over the segments dividing a variable space. Its hyperparameters are the coefficients α and c to determine the maximal grid size and segment size. We use their recommended values [27]. MIC has parameters to determine the segmentation pattern, and how to set them is discussed in Section IV-C.

The hyperparameters of NNR-GL are the number of layers $(N_L + 1)$, the number of neurons in a hidden layer N_N, and the weight on the GL regularization term λ. $(N_L + 1)$ and N_N are optimized by Grid Search of which detail is in Table 3. λ is fixed to 0.1 that performed best in preliminary experiments. With regard to training, the learning rate and the maximum epoch are fixed to 0.01 and 15000 respectively, based on preliminary experiments. The weights are initialized using a Gaussian distribution with the mean 0 and the standard deviation 1. Unlike the conventional measures, NNR-GL is applicable to both single and multi input variables. We try both and call NNR-GL with a single input variable NNR-GL(S) and that with multi ones NNR-GL(M).

TABLE 3. The hyperparameters, parameters, and settings of the conventional and proposed measures. Regarding †, we follow the literature [27].

Measure	Hyperparameter: Setting	Parameter: Setting
CC	None	None
DCC	None	None
HSIC	Kernel width σ: Median distance	None
	Test statistic a: 0.50	
MIC	Coefficient α for the maximal grid size: 0.60	Segmentation Pattern: Set using a test set. †
	Coefficient c for the maximal segment size: 75.00	
NNR-GL	# of layers $(N_L + 1)$: Searched from 4 to 20 with a step of 4 using a validation set.	Weight on edges: Set using a training set.
	# of neurons in a hidden layer N_N: Searched from 20 to 100 with a step of 20 using a validation set.	
	Weight on regularization λ: 0.10	

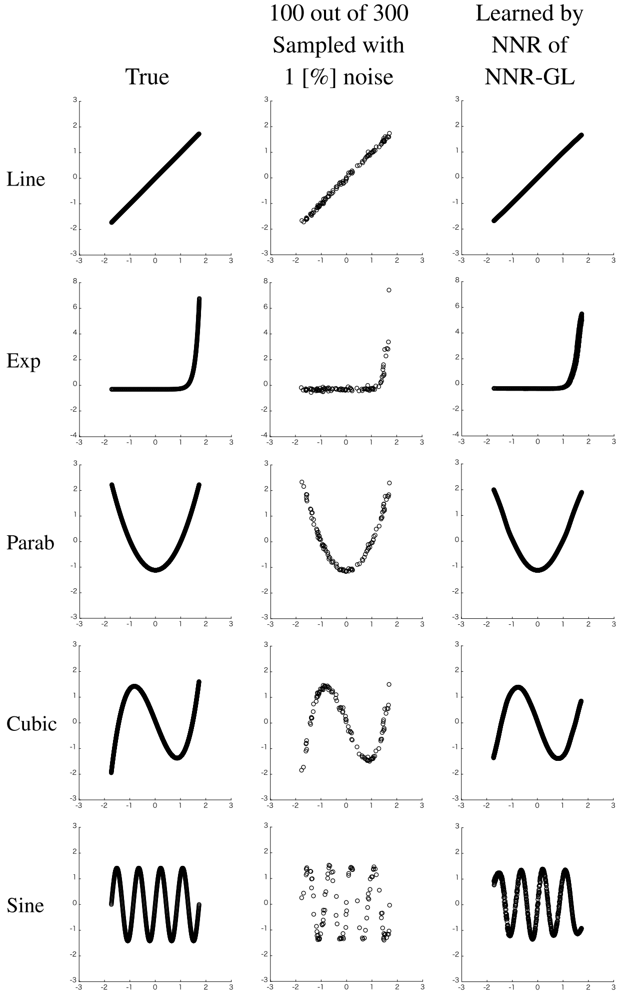


FIGURE 3. The left shows the true dependences between horizontal x_1 and vertical x_2. The middle are the graphs plotting the 100 points for training out of 300 points sampled from a true dependence with 1 [%] noise. The right shows the learned dependences by NNR of NNR-GL.

For a fair comparison, NNR-GL and the competitive measures were implemented on the common platform MATLAB. For CC, we used its library equipped in MATLAB. For DCC, we downloaded its MATLAB program [69], which was not developed by the proposers of DCC. We carefully did the code review, added a function for normalization, and then used this program. For HSIC, we downloaded and used its MATLAB program developed by the proposers of HSIC [70]. The same applies to MIC [71]. We developed a

TABLE 4. Artificial datasets where there is a dependence $x_2 = f(x_1)$ with independent x_3.

Dependence	x_1: Input of the dependence function	x_2: Output of the dependence function	x_3: Independent to the others
Line	$x_1 \sim U(0, 1)$	$x_2 = \text{Line}(x_1) = x_1$	$x_3 \sim U(0, 1)$
Exp	$x_1 \sim U(0, 10)$	$x_2 = \text{Exp}(x_1) = 10^{x_1}$	$x_3 \sim U(0, 10)$
Parab	$x_1 \sim U(-0.5, 0.5)$	$x_2 = \text{Parab}(x_1) = 4x_1^2$	$x_3 \sim U(-0.5, 0.5)$
Cubic	$x_1 \sim U(-1.3, 1.1)$	$x_2 = \text{Cubic}(x_1) = 4x_1^3 + x_1^2 - 4x_1$	$x_3 \sim U(-1.3, 1.1)$
Sine	$x_1 \sim U(0, 1)$	$x_2 = \text{Sine}(x_1) = sin(8\pi x_1)$	$x_3 \sim U(0, 1)$

program of NNR-GL ourselves using the standard commands of MATLAB.

C. DATASETS

To analytically examine the fundamental effectiveness of NNR-GL, we synthesize artificial datasets of which variable dependences are known, basically following Reshef *et al.* [27]. Data size is a dominant factor affecting detection performance, and so the robustness of each measure to data size is investigated. The robustness to another dominant factor, observation noise, is investigated as well. In the literature [27], they added simulated observation noise only to the output variable of a dependence function. It is more realistic to add noise to both input and output variables, and we do so.

Each dataset consists of the dependent variables x_1 and x_2 and the independent one x_3. There are five dependence functions mapping x_1 to x_2 as in Table 4. Line is linear, and the others are nonlinear with Exp for exponential, Parab for parabolic, Cubic for cubic, and Sine for sinusoidal. The true dependences are visualized in the left of Fig. 3. The values of x_1 are generated at even intervals. By substituting them into a dependence function, the values of x_2 are obtained. The values of x_3 are generated in the same way of x_1. Assuming the generated points (x_1, x_2, x_3) to be a population, N points are extracted from the population according to a uniform distribution. N is set to 3000, 1500, 300, or 150 [points]. To simulate real sampling, observation noise following a Gaussian distribution with the mean 0 and the standard deviation *SD* is added to the x_1, x_2, and x_3 of the extracted points. *SD* is varied from 0 to 40 with the step size 5 [%] of the variable range. For example, the training set of 100 points, which are split out of 300 sampled points with 1 [%] noise, is plotted for each dependence in the middle of Fig. 3.

A dataset is evenly split into three sets for training (parameter setting), validation (hyperparameter setting), and test (generalized detection performance estimation). Only a test set is used for CC and DCC, since they have no parameters and hyperparameters. A test set is used for HSIC as well. HSIC has no parameters but has hyperparameters fixed to the recommended values [25]. In our opinion regarding MIC, a training set should be used to set the segmentation pattern which is a kind of parameters. However, that was not mentioned in the literature [27]. We presume that they used a dataset for both training and test, and we follow this. The hyperparameters of MIC are fixed to the recommended values [27]. For NNR-GL, parameter setting, hyperparameter setting, and generalized detection performance estimation are done using training, validation, and test sets, respectively.

D. EVALUATION CRITERIA AND METHODS

The following evaluation criteria seem reasonable, because the correct dependences of artificial datasets are known: True Positive (TP) representing that a measure correctly identifies an existent dependence and True Negative (TN) representing that a measure correctly identifies a non-existent dependence. However, TP and TN require the binarization of measure values. For detailed analysis, it is better to estimate how close measure values are to TP and TN without binarization. We hence define two criteria called True Positive Certainty (TPC) and True Negative Certainty (TNC). TPC estimates whether a measure value is sufficiently large when a dependence exists. It equals the measure value itself. TNC estimates whether a measure value is sufficiently small when a dependence does not exist. It is (1 − the measure value).

To discuss the robustness to data size and noise rate, the curves of TPC and TNC of a measure are drawn over noise rates for each data size as in Fig. 4. In general, a signal is buried in strong noise; an existent dependence becomes difficult to detect when the noise rate is high. A TPC curve appears like an inverted sigmoid function, of which higher position represents the better TP detection. It is rare to misrecognize noise as a signal, but it becomes slightly more frequent for stronger noise. A TNC curve appears like an inverted and compressed sigmoid function, of which higher position represents the better TN detection. We qualitatively estimate performances based on the TPC and TNC curves.

Moreover, we calculate the Area Under Curve (AUC) of a curve over noise rates for each data size, as a quantitative estimation. We call the AUC of a TPC curve TPC-AUC, and

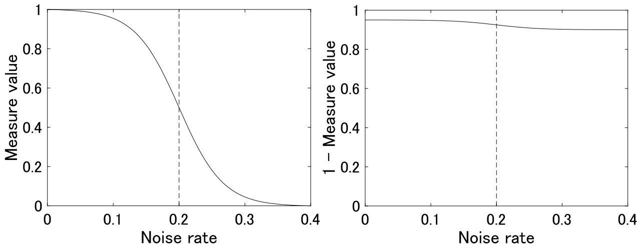


FIGURE 4. Illustration of TPC curve (left) and TNC curve (right) plotted over an increasing noise rate. The higher a curve locates at, the better the performance to detect true positives or true negatives.

that of TNC TNC-AUC. The upper limit of integration in AUC calculation, i.e. a vertical line in Fig. 4, is determined fairly for measures by the procedures below. The noise rates on the horizontal axis are picked up, where a TPC curve falls down to the measure values 0.6, 0.5, and 0.4 on the vertical axis. These noise rates are accumulated and averaged over all the measures. The upper limit for both TPC-AUC and TNC-AUC is set to this average commonly to all the measures. The mean of TPC-AUC and TNC-AUC is also used as a comprehensive detection performance criterion called CDP-AUC. Detection performances are quantitatively discussed based on TPC-AUC, TNC-AUC, and CDP-AUC.

The past studies [27], [37] qualitatively discussed TP detection performance by visualization. They did not consider TN detection performance, and such a one-sided view might lead to overestimate measures that output a large measure value even if there is no dependence. They also did not have a quantitative discussion. We design the evaluation criteria to overcome these past issues.

V. EVALUATION EXPERIMENT ON FUNDAMENTAL EFFECTIVENESS USING ARTIFICIAL DATASETS

We carried out the experiment designed in Section IV and report its details here in Section V. Sections V-A and V-B are devoted to the purpose and conditions and the results and discussion, respectively.

A. PURPOSE AND CONDITIONS

We conducted a set of experiments to examine whether NNR-GL detects co-nonlinearities correctly, compared to the conventional measures. Artificial datasets with known true variable dependences were used to estimate the correctness of detection. The experimental design and conditions were as given in Section IV. In short, the dependences Line, Exp, Parab, Cubic, and Sine were assumed between x_1 and x_2 accompanied with independent x_3. Datasets were sampled with observation noise from each dependence, where the data size was 3000 to 150, and the noise rate was 0 to 40 [%]. Each dataset was divided into training, validation, and test sets.

The competitors CC, DCC, HSIC, and MIC were applied to each pair of x_1, x_2, and x_3. There were two conditions for NNR-GL: NNR-GL(S) applied to a single-input variable similarly to the competitors and NNR-GL(M) applied to multi-input variables. The hyperparameters and parameters of measures were optimized or set to the recommended values. Three different random initializations were tried for NNR-GL. Detection performances were estimated based on TPC and TNC curves and their AUCs.

B. RESULTS AND DISCUSSION

1) BEHAVIORS OF NNR AND GL

Prior to discussing the performance of NNR-GL, we confirm the behaviors of NNR and GL inside of NNR-GL. A part of the results learned by NNR is visualized in the right of Fig. 3. This was obtained under the multi-input NNR-GL(M), using a training set of 100 sample points out of 300 with 1 [%] noise. The learned dependences in the right look similar to the true ones in the left; NNR succeeded in modeling true dependences behind sampled data in the middle. To save the space, we briefly report that NNR worked well for the other data sizes and noise rates.

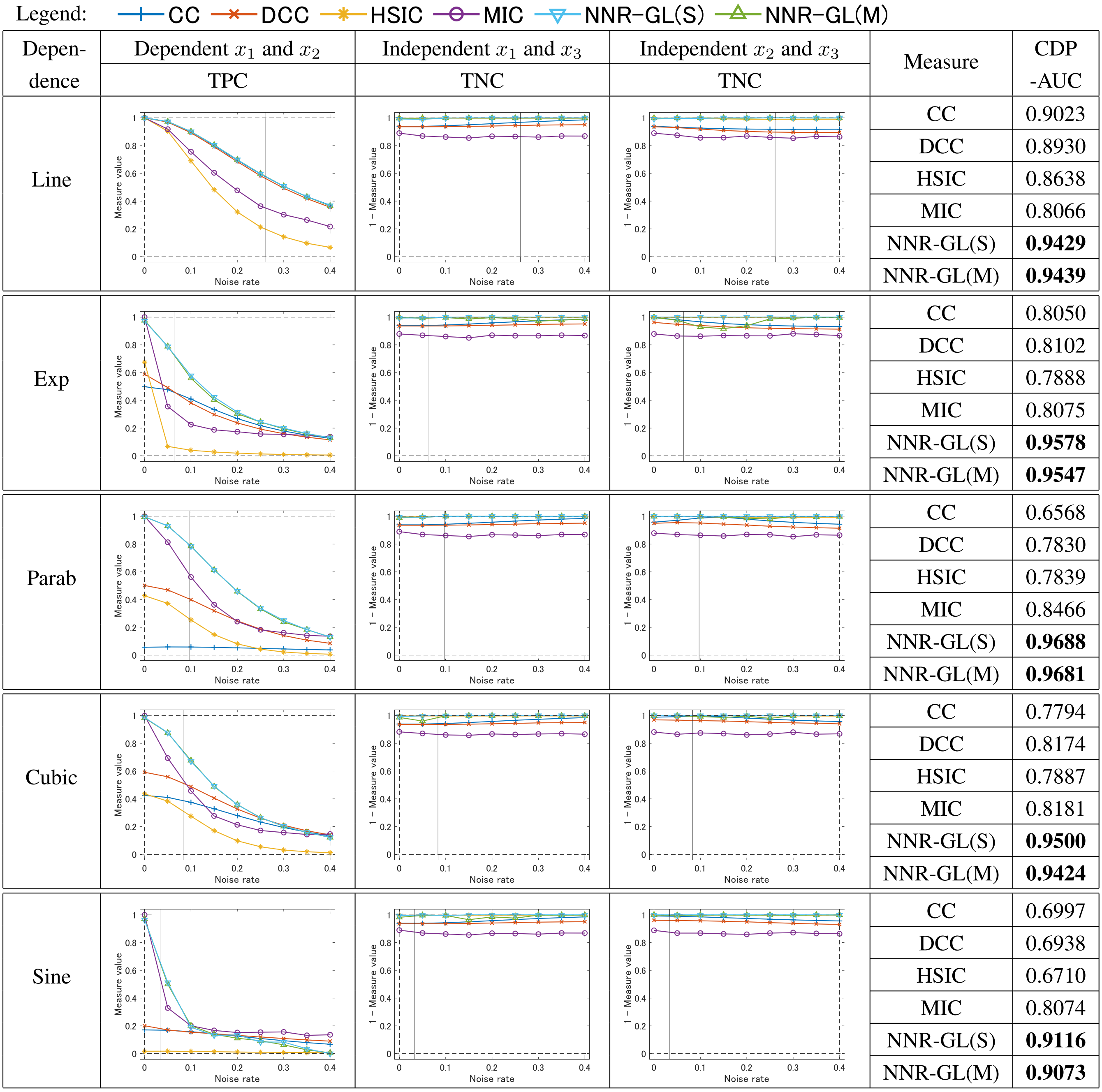
TABLE 5. **The weights on input variables optimized by GL of NNR-GL, under 1000 sample points out of 3000 with 0 [%] noise for training. R1, R2, and R3 correspond to the 3 random initializations.**

Dependence	Out	x_1		x_2		x_3	
	In	x_2	x_3	x_1	x_3	x_1	x_2
Line	R1	**0.2092**	0.0008	**0.2194**	0.0003	0.0005	0.0005
	R2	**0.2385**	0.0009	**0.2092**	0.0003	0.0006	0.0005
	R3	**0.2030**	0.0008	**0.1947**	0.0004	0.0004	0.0004
Exp	R1	0.4864	0.0015	**0.8525**	0.0021	0.0003	0.0001
	R2	0.4784	0.0001	**0.7975**	0.0017	0.0002	0.0002
	R3	0.4777	0.0012	**0.8007**	0.0011	0.0005	0.0005
Parab	R1	0.0005	0.0011	**0.7278**	0.0008	0.0000	0.0005
	R2	0.0002	0.0005	**0.6753**	0.0012	0.0012	0.0002
	R3	0.0015	0.0032	**0.7650**	0.0001	0.0004	0.0008
Cubic	R1	0.2159	0.0042	**1.3536**	0.0005	0.0008	0.0010
	R2	0.2035	0.0052	**1.2921**	0.0005	0.0005	0.0009
	R3	0.2190	0.0074	**0.4563**	0.0011	0.0015	0.0009
Sine	R1	0.0514	0.0102	**1.2876**	0.0002	0.0005	0.0007
	R2	0.0427	0.0006	**1.1260**	0.0008	0.0007	0.0000
	R3	0.0457	0.0055	**1.1753**	0.0006	0.0005	0.0007

Let us mention about hyperparameter setting. As is common for NNs, the hyperparameter values optimized by validation in the experiment differed depending on variable dependence, data size, and noise rate. As a guide, we provide the average number of total layers including input, hidden, and output ones (# of layers) and that of neurons in each hidden layer (# of neurons). These numbers are hyperparameters with the most significant effect on regression performance. In the case of large and noiseless data with the data size 3000 and the noise rate 0 [%], # of layers were from 4 to 16, and # of neurons were from 60 to 86 for all the 5 dependences. In the case of small and noisy data with 150 and 10 [%], # of layers were from 4 to 13, and # of neurons were from 20 to 73. It is not possible to say definitely, but these numbers would be reasonable for our datasets which were not image or sound but numerical.

Regarding GL, we focus on NNR-GL(M) using a training set of 1000 sample points out of 3000 with 0 [%] noise, because this condition clearly demonstrates the behaviors of GL. A part of the results optimized by GL is summarized in Table 5. For example, 0.2092 in the left top is the L_2 norm of weights on the edges connected to the input variable x_2. This numeric value, namely the RC of x_2, was obtained in the regression of x_1 on x_2 and x_3 with the random initialization R1. Although differences appear to some extent depending on initialization, the trend of the results is basically consistent. Therefore, we discuss the overall trend regardless of initialization.

Logically speaking, the RC of x_1 should be the largest to represent its contribution to regression when the output vari-



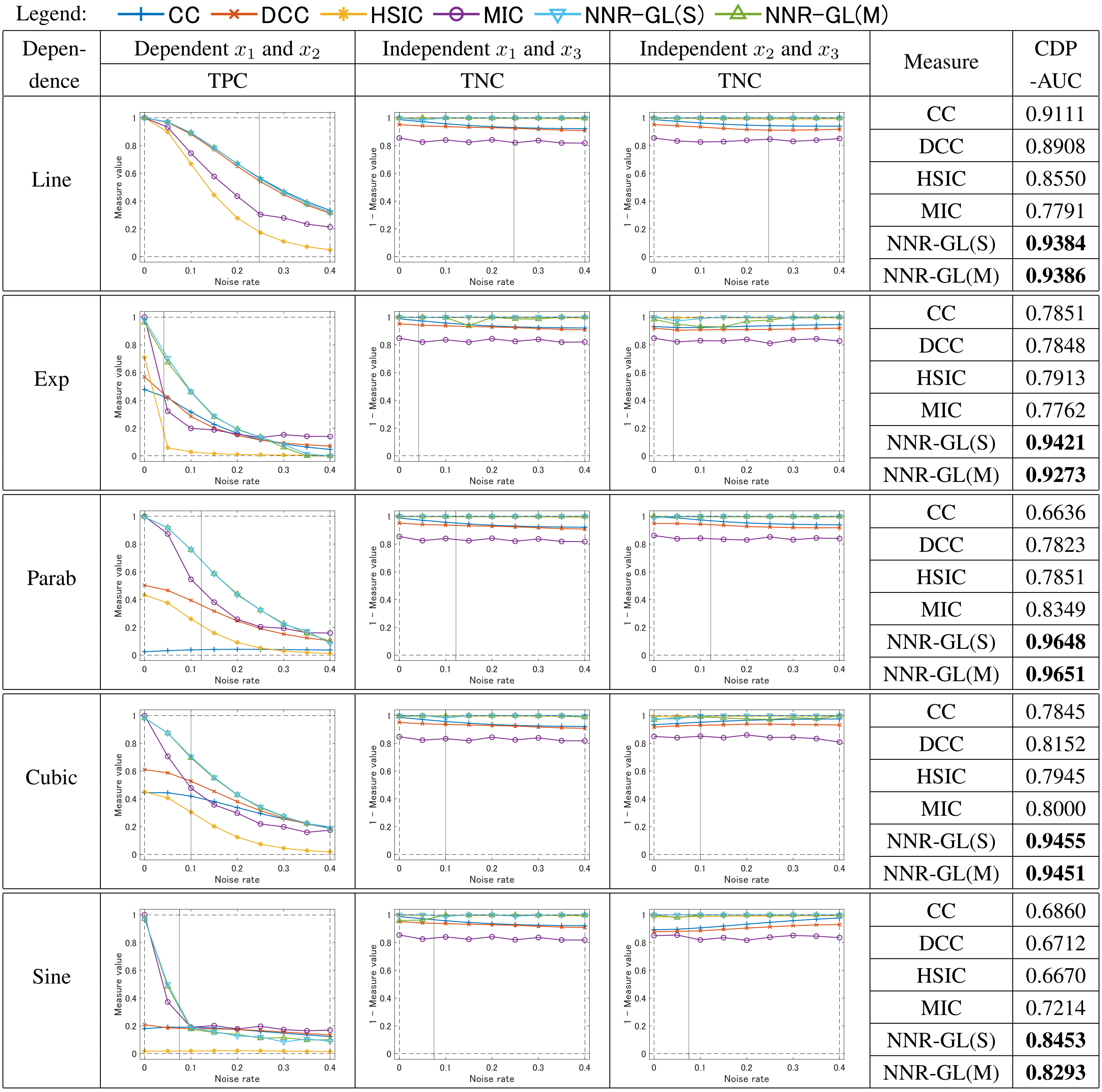
Dependence	Measure	CDP-AUC
Line	CC	0.9023
	DCC	0.8930
	HSIC	0.8638
	MIC	0.8066
	NNR-GL(S)	**0.9429**
	NNR-GL(M)	**0.9439**
Exp	CC	0.8050
	DCC	0.8102
	HSIC	0.7888
	MIC	0.8075
	NNR-GL(S)	**0.9578**
	NNR-GL(M)	**0.9547**
Parab	CC	0.6568
	DCC	0.7830
	HSIC	0.7839
	MIC	0.8466
	NNR-GL(S)	**0.9688**
	NNR-GL(M)	**0.9681**
Cubic	CC	0.7794
	DCC	0.8174
	HSIC	0.7887
	MIC	0.8181
	NNR-GL(S)	**0.9500**
	NNR-GL(M)	**0.9424**
Sine	CC	0.6997
	DCC	0.6938
	HSIC	0.6710
	MIC	0.8074
	NNR-GL(S)	**0.9116**
	NNR-GL(M)	**0.9073**

FIGURE 5. The TPC curves, TNC curves, and comprehensive detection performances CDP-AUCs obtained under the data size of 3000 (1000 for training, 1000 for validation, and 1000 for test). The higher CDP-AUCs between the proposed and conventional measures are in boldface.

able is x_2, because $x_2 = f(x_1)$. Moreover, the RC of x_2 should be the largest only for Line due to linearity. As expected, the RCs of x_1 when x_2 is regressed on x_1 and x_3 are the largest for any dependences in Table 5. In case of Line, the RCs of x_2 when x_1 is regressed on x_2 and x_3 are the largest, too. In the other conditions, this trend gets somewhat blurred but still appears with the decrease of sample size and the increase of noise rate. GL worked well to differentiate dependent and independent variables.

2) EFFECTIVENESS OF NNR-GL

We move onto discussing the detection performance of NNR-GL. Figs. 5 to 8, which correspond to data sizes from 3000 to 150, show the TPC curves, TNC curves, and CDP-AUCs of the proposed and conventional measures. The results are aligned from top to bottom according to the dependences Line, Exp, Parab, Cubic, and Sine. They are aligned from left to right according to the variable combinations x_1 and x_2, x_1 and x_3, and x_2 and x_3. The dependent x_1 and x_2 have

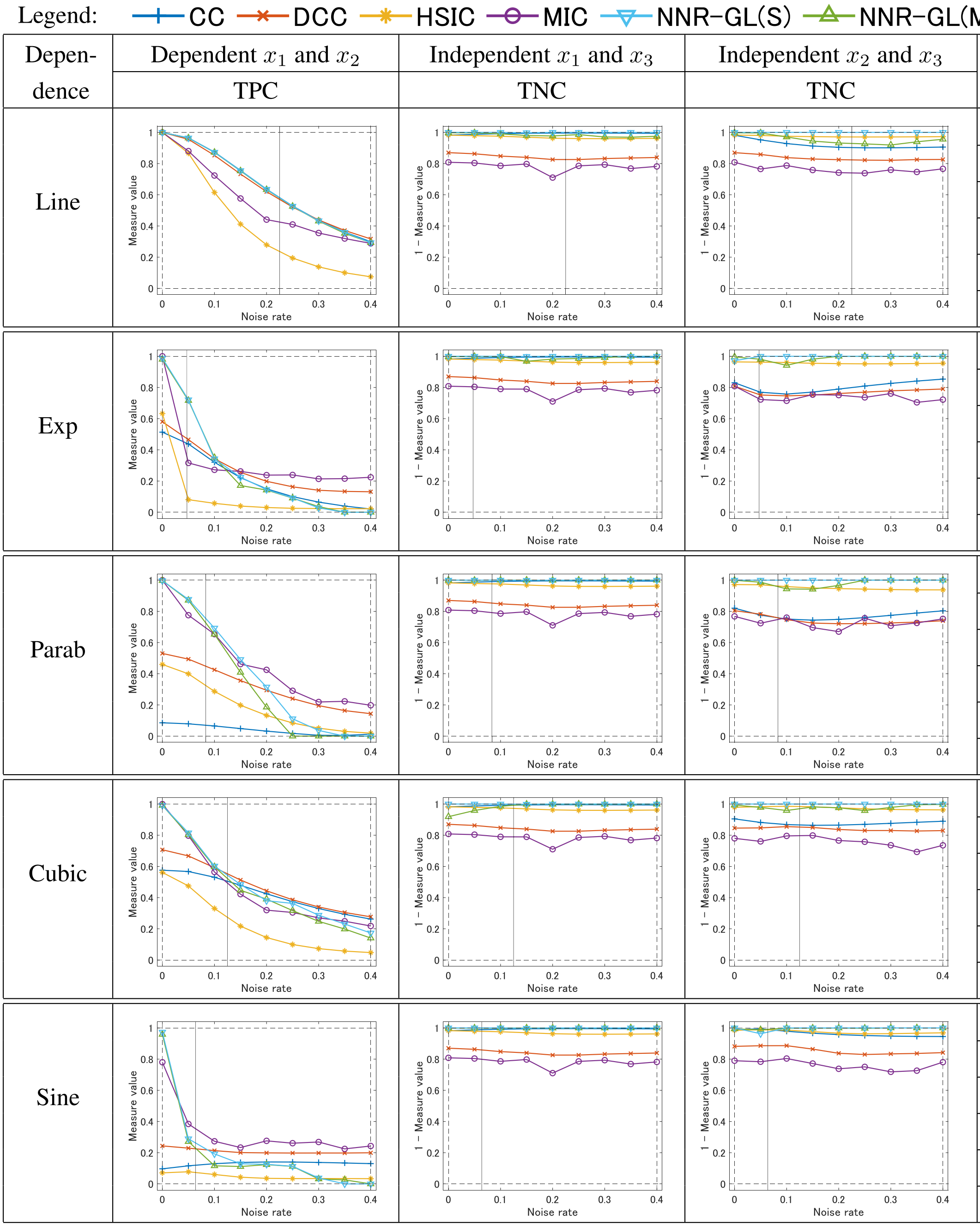


Dependence	Measure	CDP-AUC
Line	CC	0.9111
	DCC	0.8908
	HSIC	0.8550
	MIC	0.7791
	NNR-GL(S)	**0.9384**
	NNR-GL(M)	**0.9386**
Exp	CC	0.7851
	DCC	0.7848
	HSIC	0.7913
	MIC	0.7762
	NNR-GL(S)	**0.9421**
	NNR-GL(M)	**0.9273**
Parab	CC	0.6636
	DCC	0.7823
	HSIC	0.7851
	MIC	0.8349
	NNR-GL(S)	**0.9648**
	NNR-GL(M)	**0.9651**
Cubic	CC	0.7845
	DCC	0.8152
	HSIC	0.7945
	MIC	0.8000
	NNR-GL(S)	**0.9455**
	NNR-GL(M)	**0.9451**
Sine	CC	0.6860
	DCC	0.6712
	HSIC	0.6670
	MIC	0.7214
	NNR-GL(S)	**0.8453**
	NNR-GL(M)	**0.8293**

FIGURE 6. The TPC curves, TNC curves, and comprehensive detection performances CDP-AUCs obtained under the data size of 1500 (500 for training, 500 for validation, and 500 for test). The higher CDP-AUCs between the proposed and conventional measures are in boldface.

TPC curves representing TP detection and no TNC curves. The independent x_1 and x_3 have TNC curves representing TN detection and no TPC curves, and the independent x_2 and x_3 have the same. Refer to Section IV-D on how to read TPC and TNC curves. The values of CDP-AUC, which is the mean of TPC-AUC and TNC-AUC over the variable combinations, are listed in the rightmost. NNR-GL behaved quite similarly under three different initializations, and so we provide only the mean measure values of NNR-GL over initializations.

In principle, NNR-GL(M) which is the original use of NNR-GL can detect multi dependences simultaneously and efficiently. However, there is a possibility that NNR-GL(M) is disadvantaged in learning, compared to NNR-GL(S). The reason is that NNR-GL(S) devotes its all learning resources to a single-to-single dependence, while NNR-GL(M) assigns those to multi-to-single dependences. Despite this possibility, the curves of NNR-GL(S) and NNR-GL(M) are almost the same in Figs. 5 to 8. Concretely, the downward triangles for NNR-GL(S) and the upward triangles for NNR-GL(M) over-



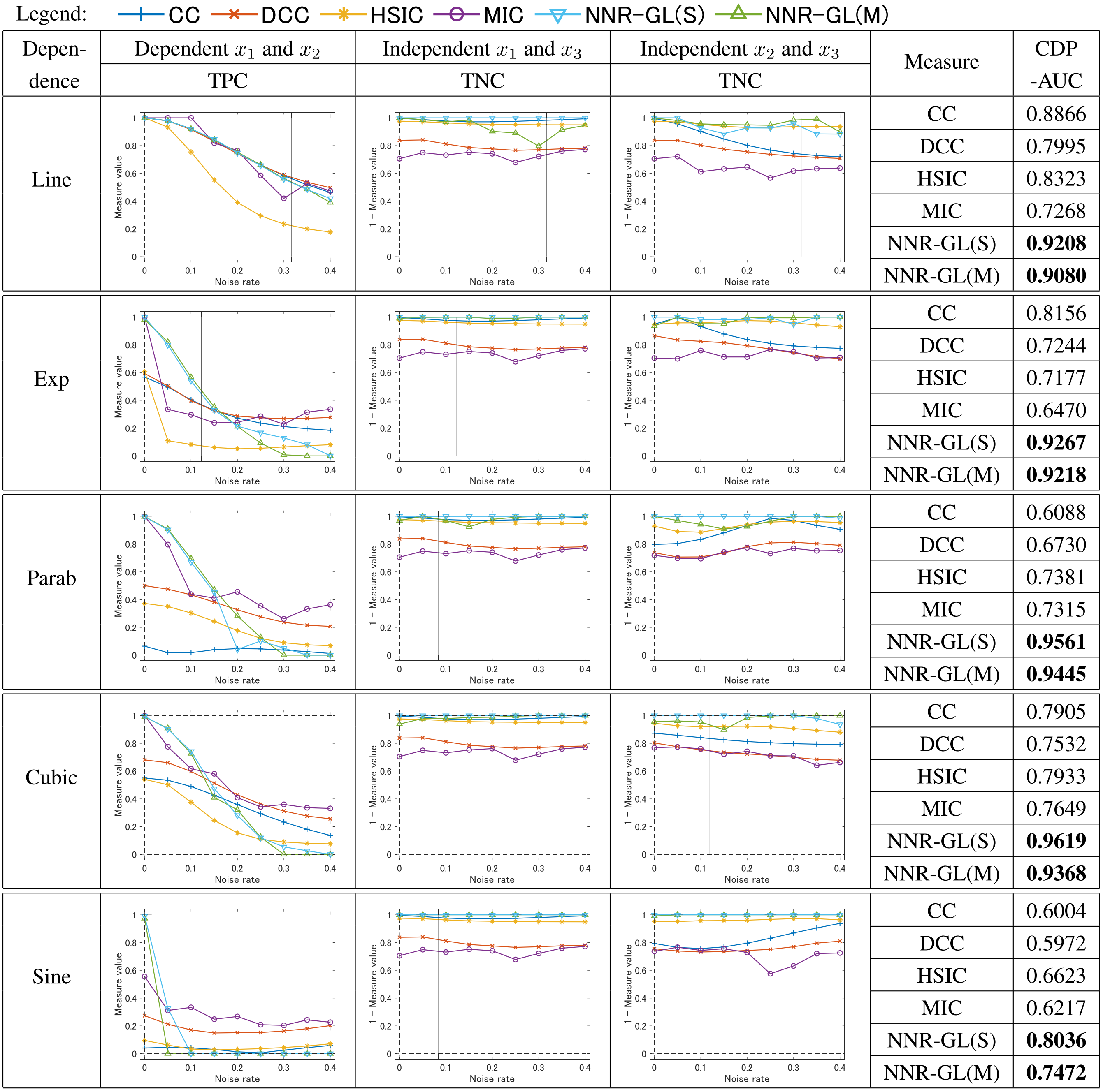
Dependence	Measure	CDP-AUC
Line	CC	0.9259
	DCC	0.8437
	HSIC	0.8611
	MIC	0.7611
	NNR-GL(S)	**0.9506**
	NNR-GL(M)	**0.9363**
Exp	CC	0.7534
	DCC	0.7230
	HSIC	0.7668
	MIC	0.7432
	NNR-GL(S)	**0.9468**
	NNR-GL(M)	**0.9448**
Parab	CC	0.6149
	DCC	0.7085
	HSIC	0.7777
	MIC	0.7819
	NNR-GL(S)	**0.9536**
	NNR-GL(M)	**0.9423**
Cubic	CC	0.8106
	DCC	0.7894
	HSIC	0.8073
	MIC	0.7884
	NNR-GL(S)	**0.9353**
	NNR-GL(M)	**0.9106**
Sine	CC	0.6942
	DCC	0.6623
	MIC	0.6805
	HSIC	0.7249
	NNR-GL(S)	**0.8708**
	NNR-GL(M)	**0.8684**

FIGURE 7. The TPC curves, TNC curves, and comprehensive detection performances CDP-AUCs obtained under the data size of 300 (100 for training, 100 for validation, and 100 for test). The higher CDP-AUCs between the proposed and conventional measures are in boldface.

lap and look like stars. The weights on multi input variables were successfully optimized to make NNR-GL(M) equivalent to NNR-GL(S), as we expected. Hereinafter, we simply regard NNR-GL(S) and NNR-GL(M) as the same NNR-GL.

In the left graph of Fig. 5 where the data size is 3000 (1000 for training, 1000 for validation, and 1000 for test), the TPC curve of NNR-GL is located in the highest position for all of Line, Exp, Parab, Cubic, and Sine. In the middle and right graphs, the TNC curves of NNR-GL lie highest as well. In the rightmost table, the values of CDP-AUC of NNR-GL(S) and NNR-GL(M) are the highest. NNR-GL outperformed the other measures consistently in terms of all of TP detection, TN detection, and the kind of dependences. NNR well modeled a variety of co-nonlinearities, and GL on the NNR's input layer well differentiated dependent and independent variables. That brought the high TP and TN detection performances of NNR-GL.

Paying attention to the other measures in Fig. 5, the TPC curve of CC takes the highest position for Line as well as



Dependence	Measure	CDP-AUC
Line	CC	0.8866
	DCC	0.7995
	HSIC	0.8323
	MIC	0.7268
	NNR-GL(S)	**0.9208**
	NNR-GL(M)	**0.9080**
Exp	CC	0.8156
	DCC	0.7244
	HSIC	0.7177
	MIC	0.6470
	NNR-GL(S)	**0.9267**
	NNR-GL(M)	**0.9218**
Parab	CC	0.6088
	DCC	0.6730
	HSIC	0.7381
	MIC	0.7315
	NNR-GL(S)	**0.9561**
	NNR-GL(M)	**0.9445**
Cubic	CC	0.7905
	DCC	0.7532
	HSIC	0.7933
	MIC	0.7649
	NNR-GL(S)	**0.9619**
	NNR-GL(M)	**0.9368**
Sine	CC	0.6004
	DCC	0.5972
	HSIC	0.6623
	MIC	0.6217
	NNR-GL(S)	**0.8036**
	NNR-GL(M)	**0.7472**

FIGURE 8. The TPC curves, TNC curves, and comprehensive detection performances CDP-AUCs obtained under the data size of 150 (50 for training, 50 for validation, and 50 for test). The higher CDP-AUCs between the proposed and conventional measures are in boldface.

NNR-GL. It appears unstably in the lower positions for the other dependences. The TNC curves of CC stay in the third highest position for all the dependences. Reflecting the TPC and TNC trends, the values of CDP-AUC differ depending on the kind of dependences. These results are a matter of course, since the model of CC is fixed to be linear. The TPC curve of DCC takes the highest position for Line, but it takes the third for the other dependences. Its TNC curves stay in the second lowest position for all the dependences. With respect to CDP-AUC, DCC is slightly better and more stable than CC. The model of DCC is nonlinear but in a limited function family. This limitation would lead to these results. For almost all the dependences, the TPC curve of HSIC takes the lowest position, while its TNC curves stay in the highest or close to the highest. Reflecting those, the values of CDP-AUC are comparatively low in spite of the broader family of nonlinear functions that HSIC has. There is a possibility that the hyperparameters fixed to the recommended values hindered HSIC from its best performance. The TPC curve of MIC takes the fourth position for Line and the second position for the

TABLE 6. Results for Pima diabetes dataset. The detected sets matching the correct ones are highlighted in bold.

Measure	Detected sets of dependent variables
CC	$\{x_1, x_3, x_8\}$, $\{x_2, x_5\}$, $\{x_4, x_6\}$, $\boldsymbol{\{x_7\}}$
DCC	$\{x_1, x_3, x_8\}$, $\{x_2, x_5\}$, $\{x_4, x_6\}$, $\boldsymbol{\{x_7\}}$
HSIC	$\{x_1, x_3, x_8\}$, $\{x_4, x_5, x_6\}$, $\{x_2\}$, $\boldsymbol{\{x_7\}}$
MIC	$\boldsymbol{\{x_1, x_8\}}$, $\{x_4, x_6\}$, $\{x_2\}$, $\{x_3\}$, $\{x_5\}$, $\boldsymbol{\{x_7\}}$
NNR-GL	$\boldsymbol{\{x_1, x_8\}}$, $\boldsymbol{\{x_2, x_3, x_4, x_5, x_6\}}$, $\boldsymbol{\{x_7\}}$

TABLE 7. Results for US-130 Hospital dataset. The highest performances are highlighted in bold.

Measure	P_{ave}	R_{ave}
CC	0.0641	0.2500
DCC	0.0641	0.2500
HSIC	0.1818	0.6000
MIC	0.0444	0.1833
NNR-GL	**0.6000**	**1.0000**

other dependences. Its TNC curves stay in the lowest position for all the dependences. Due to those, the values of CDP-AUC are not so high. The flexible nonlinear model of MIC would end up with somewhat overfitting to TPs by using the same data for segmentation and detection.

Comparing Figs. 5 to 8, the TPC and TNC curves of all the measures gradually get down and unstable as the data size decreases from 3000 to 150. However, their trends are consistent throughout the figures; the TPC and TNC curves of NNR-GL are located in the highest for all the dependences and data sizes. Under some conditions with the data size of 300 or 150, the TPC curves of NNR-GL quickly fall outside of the upper limit of noise rate. That suggests that NNR-GL refrained from the overdetection of dependences under too high noise and led to better TN detection. In contrast, the measures of which TPC curves stay higher outside of the upper limit tend to have lower performances in TN detection. The success of NNR-GL in Figs. 7 and 8, of which data sizes are 300 and 150, is noteworthy. The result suggests that overfitting for small data was avoided by GL and the use of training, validation, and test sets, which are equipped in NNR-GL but not in the other measures.

Here, the findings above are summarized and concluded. In both TP and TN detections, NNR-GL outperformed the conventional measures. Its performance was stable to initialization, robust to noise rate, and robust to data size, commonly for all the dependences. Therefore, the fundamental effectiveness of NNR-GL was confirmed.

VI. PILOT EXPERIMENT USING REAL BENCHMARK DATASETS

In Section V, the experiment and analysis using artificial data suggested that NNR-GL works robustly and better than the conventional measures. It is time to broaden the horizons to the practicality of NNR-GL. Full-scale experiments will be in the next stage of our research, but we started case studies using real benchmark datasets. Section VI reports those. One case study is given in Section VI-A, and the other is given in Section VI-B. In Section VI-C, we discuss the perspectives found in all the experiments in the present research.

A. CASE STUDY USING PIMA DIABETES DATASET

To try the potential of NNR-GL for real world problems, we demonstrated a pilot experiment including two case studies using real benchmark datasets. This paper aims to propose and analytically evaluate NNR-GL, and thus we briefly report the pilot experiment. NNR-GL(M) was employed here, since it worked as well as NNR-GL(S) in the evaluation experiment in Section V. The conditions of NNR-GL and the competitive measures were the same in the evaluation experiment, too. The best performances of NNR-GL and the competitive measures were estimated in the following manner, which was common to the two case studies. For each measure, the threshold of measure values was set to find the correct sets of dependent variables as much as possible. Variables exceeding the threshold were detected and assigned to the corresponding detected set of dependent variables.

The first dataset was Pima Diabetes Dataset in the website [72], of which 8 variables except the class and 768 sample points were used. We prepared the correct sets of dependent variables based on common sense and medical literature survey using [73]. There were 3 correct sets of dependent variables, $\{x_1, x_8\}$, $\{x_2, x_3, x_4, x_5, x_6\}$, and $\{x_7\}$. The first set means that the number of pregnancies x_1 and age x_8 are dependent. That was judged as correct based on the common sense. The second set contains glucose concentration x_2, blood pressure x_3, triceps skin thickness x_4, insulin level x_5, and body mass index x_6. Their dependence was supported by a collection of medical literature [73]. The third set consists of diabetes pedigree x_7, which was a kind of genetic factor possibly less dependent on the other variables. We applied NNR-GL and the competitive measures and obtained the detected sets of dependent variables. As shown in Table 6, NNR-GL detected all the correct sets, but the other measures did not.

B. CASE STUDY USING US-130 HOSPITAL DATASET

The second dataset was US-130 Hospital Dataset in the website [74], of which 10 variables were picked up referring the literature [75]. This dataset was split into 10 subsets consisting of 1000 sample points for 10 trials. We converted the following 6 symbol variables into the sets of indicator variables representing the absence of symbol with 0 and the presence with 1: race, admission source, specialty of the admitting physician, primary diagnosis, hemoglobin A1c, and readmission rate. These sets were obviously the correct sets of dependent variables and so targeted.

We estimated the performances of NNR-GL and the competitive measures using Equ. (11). The number of trials T is

10. $S_t^{(d)}$ denotes the set of variable sets detected by a measure in the tth trial. $S^{(c)}$ denotes the set of the 6 correct variable sets. By setting $S_t = S_t^{(d)}$, the term in the summation becomes the precision obtained in the tth trial. Hence, Equ. (11) is the averaged precision P_{ave} over all trials. Similarly, Equ. (11) is the averaged recall R_{ave} when $S_t = S^{(c)}$. In Table 7, NNR-GL achieved the highest P_{ave} and R_{ave} compared to the other measures.

$$\frac{1}{T}\sum_{t=1}^{T}\left(\frac{|S_t^{(d)} \cap S^{(c)}|}{|S_t|}\right) \tag{11}$$

where

$$S_t = S_t^{(d)} \text{ for } P_{ave} \text{ and } S_t = S^{(c)} \text{ for } R_{ave}$$

C. PERSPECTIVES FOUND IN THE EXPERIMENTS

There were only two case studies under limited conditions in the pilot experiment. However, the fact that NNR-GL outperformed the conventional measures in both case studies indicates the potential of NNR-GL in real world problems. Furthermore, we found three perspectives in the pilot experiment and also the evaluation experiment in Section V. The first perspective is inspired by the success of NNR-GL in detecting common senses and indicator variable sets. We come up with the following. By adaptively updating the groups in GL, NNR-GL will be able to gather detected dependent variables into a group as prior information. Under this prior information like a common sense, NNR-GL will detect other latent dependences in the next turn. This spiral process will carve out important novel dependences.

For ease of understanding the experimental results, we manually incorporated the detected dependences into their sets in Table 6. The same will be needed when NNR-GL is applied to real world problems. Therefore, the second perspective is a framework that automatically organizes the sets of detected dependent variables. In addition, it will be helpful for human awareness to accompany which variable is the representative of each of these sets. Although this idea was partially achieved in our past study [34], it should be improved and evaluated in detail.

It was experimentally confirmed that NNR-GL works well for the first step to detect potential dependences in Fig. 1. To more ensure the fundamental effectiveness of NNR-GL, we are planning to conduct additional experiments using different types of dependences and noise distributions. To go beyond the first step, in other words, to bridge the first step to the second and third ones, the third perspective is a way to mathematically formulate the detected dependences. The inverse mapping problem in Section III-A gives us a hint that causalities "which variables cause which ones" can be detected by bidirectional regressions.

Our idea is to achieve this causality detection by assuming the following: As inputs, variables yielding one-to-one or many-to-one mapping (namely, forward mapping) should be "causes." Variables yielding one-to-many or many-to-many mapping (inverse mapping) should be "results." We actually introduced the part of this idea to identify representative variables in the past study [34], but a thorough investigation on that should be one of our future work. Furthermore, we think that the number of turning points on a detected dependence helps to formulate the dependence function, because this number suggests the shape and degree of the function. Recently, we started working on the three perspectives above.

VII. CONCLUSION

The detection of variable dependences is essential for a broad range of disciplines. To detect nonlinear dependences among multi variables, we proposed a measure called NNR-GL. It consists of nonlinear modeling by Neural Network Regression and variable selection by Group Lasso, accompanied by detected dependence quantification and averaging for robustness. For evaluating the fundamental effectiveness of NNR-GL, we demonstrated several experiments. NNR-GL was applied to artificial datasets with several dependences under different data sizes and noise rates, and its correctness of detection was estimated. A criterion CDP-AUC, which is the overall representation of true positive and true negative detections, was used for the estimation. The values of CDP-AUC by NNR-GL were 0.7472 to 0.9681. They were higher than the values of CDP-AUC by the conventional measures, which were 0.5972 to 0.9259, for all the experimental conditions. It was confirmed that NNR-GL can detect co-nonlinearities correctly and robustly to data size and noise rate. Our future work will be the improvement and extension of NNR-GL and the confirmation of its practicality for real world problems.

REFERENCES

[1] M. Vallières, E. Kay-Rivest, L. J. Perrin, X. Liem, C. Furstoss, H. J. W. L. Aerts, N. Khaouam, P. F. Nguyen-Tan, C.-S. Wang, K. Sultanem, J. Seuntjens, and I. E. Naqa, "Radiomics strategies for risk assessment of tumour failure in head-and-neck cancer," *Sci. Rep.*, vol. 7, no. 1, Aug. 2017, Art. no. 10117, doi: 10.1038/s41598-017-10371-5.

[2] B. Yang *et al.*, "Clinical and molecular characteristics of COVID-19 patients with persistent SARS-CoV-2 infection," *Nature Commun.*, vol. 12, no. 1, Jun. 2021, Art. no. 3501, doi: 10.1038/s41467-021-23621-y.

[3] Y. Liu, L. Fan, C. Zhang, T. Zhou, Z. Xiao, L. Geng, and D. Shen, "Incomplete multi-modal representation learning for Alzheimer's disease diagnosis," *Med. Image Anal.*, vol. 69, Apr. 2021, Art. no. 101953, doi: 10.1016/j.media.2020.101953.

[4] H. Ke, D. Chen, X. Li, Y. Tang, T. Shah, and R. Ranjan, "Towards brain big data classification: Epileptic EEG identification with a lightweight VGGNet on global MIC," *IEEE Access*, vol. 6, pp. 14722–14733, Mar. 2018, doi: 10.1109/ACCESS.2018.2810882.

[5] T. Liang, Q. Zhang, X. Liu, C. Lou, X. Liu, and H. Wang, "Time-frequency maximal information coefficient method and its application to functional corticomuscular coupling," *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 28, no. 11, pp. 2515–2524, Nov. 2020, doi: 10.1109/TNSRE.2020.3028199.

[6] T. M. Norman, M. A. Horlbeck, J. M. Replogle, A. Y. Ge, A. Xu, M. Jost, L. A. Gilbert, and J. S. Weissman, "Exploring genetic interaction manifolds constructed from rich single-cell phenotypes," *Science*, vol. 365, no. 6455, pp. 786–793, Aug. 2019, doi: 10.1126/science.aax4438.

[7] H. Wang, Y. Ding, J. Tang, and F. Guo, "Identification of membrane protein types via multivariate information fusion with Hilbert–Schmidt independence criterion," *Neurocomputing*, vol. 383, no. 28, pp. 257–269, Mar. 2020, doi: 10.1016/j.neucom.2019.11.103.

[8] C. Weinreb, A. Rodriguez-Fraticelli, F. D. Camargo, and A. M. Klein, "Lineage tracing on transcriptional landscapes links state to fate during differentiation," *Science*, vol. 367, no. 6479, Feb. 2020, Art. no. eaaw3381, doi: 10.1126/science.aaw3381.

[9] F. Moens, S. Konstantinidis, and D. Depla, ''The target material influence on the current pulse during high power pulsed magnetron sputtering,'' *Frontiers Phys.*, vol. 5, Oct. 2017, Art. no. 51, doi: 10.3389/fphy.2017.00051.

[10] M. Umehara, H. S. Stein, D. Guevarra, P. F. Newhouse, D. A. Boyd, and J. M. Gregoire, ''Analyzing machine learning models to accelerate generation of fundamental materials insights,'' *NPJ Comput. Mater.*, vol. 5, no. 1, Mar. 2019, Art. no. 34, doi: 10.1038/s41524-019-0172-5.

[11] O. Spitz, A. Herdt, J. Wu, G. Maisons, M. Carras, C.-W. Wong, W. Elsäßer, and F. Grillot, ''Private communication with quantum cascade laser photonic chaos,'' *Nature Commun.*, vol. 12, no. 1, Jun. 2021, Art. no. 3327, doi: 10.1038/s41467-021-23527-9.

[12] J. Runge, P. Nowack, M. Kretschmer, S. Flaxman, and D. Sejdinovic, ''Detecting and quantifying causal associations in large nonlinear time series datasets,'' *Sci. Adv.*, vol. 5, no. 11, Nov. 2019, Art. no. eaau4996, doi: 10.1126/sciadv.aau4996.

[13] X. Ma, W. Liu, R. J. Allen, G. Huang, and X. Li, ''Dependence of regional ocean heat uptake on anthropogenic warming scenarios,'' *Sci. Adv.*, vol. 6, no. 45, Nov. 2020, Art. no. eabc0303, doi: 10.1126/sciadv.abc0303.

[14] R. Culberg, D. M. Schroeder, and W. Chu, ''Extreme melt season ice layers reduce firn permeability across Greenland,'' *Nature Commun.*, vol. 12, no. 1, Apr. 2021, Art. no. 2336, doi: 10.1038/s41467-021-22656-5.

[15] Y. Xing, C. Lv, Z. Zhang, H. Wang, X. Na, D. Cao, E. Velenis, and F.-Y. Wang, ''Identification and analysis of driver postures for in-vehicle driving activities and secondary tasks recognition,'' *IEEE Trans. Comput. Social Syst.*, vol. 5, no. 1, pp. 95–108, Mar. 2018, doi: 10.1109/TCSS.2017.2766884.

[16] T. Wen, D. Dong, Q. Chen, L. Chen, and C. Roberts, ''Maximal information coefficient-based two-stage feature selection method for railway condition monitoring,'' *IEEE Trans. Intell. Transp. Syst.*, vol. 20, no. 7, pp. 2681–2690, Jul. 2019, doi: 10.1109/TITS.2018.2881284.

[17] Y. Xing, C. Lv, and D. Cao, ''Personalized vehicle trajectory prediction based on joint time-series modeling for connected vehicles,'' *IEEE Trans. Veh. Technol.*, vol. 69, no. 2, pp. 1341–1352, Feb. 2020, doi: 10.1109/TVT.2019.2960110.

[18] D. E. Farrar and R. R. Glauber, ''Multicollinearity in regression analysis: The problem revisited,'' *Rev. Econ. Statist.*, vol. 49, no. 1, pp. 92–107, Feb. 1967.

[19] E. R. Mansfield and B. P. Helms, ''Detecting multicollinearity,'' *Amer. Statistician*, vol. 36, no. 3, pp. 158–160, Aug. 1982, doi: 10.1080/00031305.1982.10482818.

[20] C. H. Mason and W. D. Perreault, Jr., ''Collinearity, power, and interpretation of multiple regression analysis,'' *J. Marketing Res.*, vol. 28, no. 3, pp. 268–280, Aug. 1991, doi: 10.1177/002224379102800302.

[21] T. Anderson, *An Introduction to Multivariate Statistical Analysis*, 3rd ed. New York, NY, USA: Wiley, 2003.

[22] G. J. Székely, M. L. Rizzo, and N. K. Bakirov, ''Measuring and testing dependence by correlation of distances,'' *Ann. Appl. Stat.*, vol. 35, no. 6, pp. 2769–2794, Dec. 2007, doi: 10.1214/009053607000000505.

[23] G. J. Székely and M. L. Rizzo, ''Brownian distance covariance,'' *Ann. Appl. Statist.*, vol. 3, no. 4, pp. 1236–1265, Dec. 2009, doi: 10.1214/09-AOAS312.

[24] A. Gretton, O. Bousquet, A. Smola, and B. Schölkopf, ''Measuring statistical dependence with Hilbert–Schmidt norms,'' in *Proc. ALT*, Singapore, 2005, pp. 63–77, doi: 10.1007/11564089_7.

[25] A. Gretton, K. Fukumizu, C. H. Teo, L. Song, B. Schölkopf, and A. J. Smola, ''A kernel statistical test of independence,'' in *Proc. NIPS*, Vancouver, BC, Canada, 2007, pp. 585–592.

[26] D. Sejdinovic, B. Sriperumbudur, A. Gretton, and K. Fukumizu, ''Equivalence of distance-based and RKHS-based statistics in hypothesis testing,'' *Ann. Appl. Statist.*, vol. 41, no. 5, pp. 2263–2291, Oct. 2013, doi: 10.1214/13-AOS1140.

[27] D. N. Reshef, Y. A. Reshef, H. K. Finucane, S. R. Grossman, G. McVean, P. J. Turnbaugh, E. S. Lander, M. Mitzenmacher, and P. C. Sabeti, ''Detecting novel associations in large data sets,'' *Science*, vol. 334, no. 6062, pp. 1518–1524, Dec. 2011, doi: 10.1126/science.1205438.

[28] J. Zhao, Y. Zhou, X. Zhang, and L. Chen, ''Part mutual information for quantifying direct associations in networks,'' *Proc. Nat. Acad. Sci. USA*, vol. 113, no. 18, pp. 5130–5135, May 2016, doi: 10.1073/pnas.1522586113.

[29] B. Lusch, J. N. Kutz, and S. L. Brunton, ''Deep learning for universal linear embeddings of nonlinear dynamics,'' *Nature Commun.*, vol. 9, no. 1, Nov. 2018, Art. no. 4950, doi: 10.1038/s41467-018-07210-0.

[30] C. Bishop, *Pattern Recognition and Machine Learning*. New York, NY, USA: Springer, 2006. [Online]. Available: https://www.microsoft.com/en-us/research/people/cmbishop/prml-book/

[31] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, 2nd ed. New York, NY, USA: Springer, 2017. [Online]. Available: https://web.stanford.edu/~hastie/ElemStatLearn/

[32] M. Yuan and Y. Lin, ''Model selection and estimation in regression with grouped variables,'' *J. Roy. Stat. Soc. B, Stat. Methodol.*, vol. 68, no. 1, pp. 49–67, Feb. 2006, doi: 10.1111/j.1467-9868.2005.00532.x.

[33] L. Meier, S. van de Geer, and P. Bühlmann, ''The group lasso for logistic regression,'' *J. Roy. Stat. Soc. B, Stat. Methodol.*, vol. 70, no. 1, pp. 53–71, Feb. 2008, doi: 10.1111/j.1467-9868.2007.00627.x.

[34] M. Ohsaki, H. Sasaki, N. Kishimoto, S. Katagiri, and P. H. H. Then, ''Discovery of sets and representatives of variables in co-nonlinear relationships by neural network regression and group lasso,'' in *Proc. IEEE BIBM*, Madrid, Spain, Dec. 2018, pp. 2287–2294, doi: 10.1109/BIBM.2018.8621207.

[35] M. Kac, *Statistical Independence in Probability, Analysis, and Number Theory*. New York, NY, USA: Wiley, 1959.

[36] G. S. Goodman, ''Statistical independence and normal numbers: An aftermath to Mark Kac's Carus monograph,'' *Amer. Math. Monthly*, vol. 106, no. 2, pp. 112–126, Feb. 1999, doi: 10.1080/00029890.1999.12005018.

[37] N. Simon and R. Tibshirani, ''Comment on 'detecting novel associations in large data sets' by Reshef et al, science Dec 16, 2011,'' Jan. 2014, *arXiv:1401.7645*. [Online]. Available: http://arxiv.org/abs/1401.7645

[38] J. B. Kinney and G. S. Atwal, ''Equitability, mutual information, and the maximal information coefficient,'' *Proc. Nat. Acad. Sci. USA*, vol. 111, no. 9, pp. 3354–3359, Mar. 2014, doi: 10.1073/pnas.1309933111.

[39] C. F. O. Mendes and M. W. Beims, ''Distance correlation detecting Lyapunov instabilities, noise-induced escape times and mixing,'' *Phys. A, Stat. Mech. Appl.*, vol. 512, pp. 721–730, Dec. 2018, doi: 10.1016/j.physa.2018.08.028.

[40] A. Pérez-Suay and G. Camps-Valls, ''Sensitivity maps of the Hilbert–Schmidt independence criterion,'' *Appl. Soft Comput.*, vol. 70, pp. 1054–1063, Sep. 2018, doi: 10.1016/j.asoc.2017.04.024.

[41] D. Edelmann, K. Fokianos, and M. Pitsillou, ''An updated literature review of distance correlation and its applications to time series,'' *Int. Stat. Rev.*, vol. 87, no. 2, pp. 237–262, Aug. 2019, doi: 10.1111/insr.12294.

[42] S. Wang, Y. Zhao, Y. Shu, H. Yuan, J. Geng, and S. Wang, ''Fast search local extremum for maximal information coefficient (MIC),'' *J. Comput. Appl. Math.*, vol. 327, pp. 372–387, Jan. 2018, doi: 10.1016/j.cam.2017.05.038.

[43] D. Cao, Y. Chen, J. Chen, H. Zhang, and Z. Yuan, ''An improved algorithm for the maximal information coefficient and its application,'' *Roy. Soc. Open Sci.*, vol. 8, no. 2, Feb. 2021, Art. no. 201424, doi: 10.1098/rsos.201424.

[44] K. Fokianos and M. Pitsillou, ''Consistent testing for pairwise dependence in time series,'' *Technometrics*, vol. 59, no. 2, pp. 262–270, Apr. 2017, doi: 10.1080/00401706.2016.1156024.

[45] N. Pfister, P. Bühlmann, B. Schölkopf, and J. Peters, ''Kernel-based tests for joint independence,'' *J. Roy. Stat. Soc. B, Stat. Methodol.*, vol. 80, no. 1, pp. 5–31, Jan. 2018, doi: 10.1111/rssb.12235.

[46] X. Tian, J. He, and Y. Shi, ''Statistical dependence test with Hilbert–Schmidt independence criterion,'' *J. Phys., Conf. Ser.*, vol. 1601, Aug. 2020, Art. no. 032008, doi: 10.1088/1742-6596/1601/3/032008.

[47] D. N. Reshef, Y. A. Reshef, P. C. Sabeti, and M. Mitzenmacher, ''An empirical study of the maximal and total information coefficients and leading measures of dependence,'' *Ann. Appl. Statist.*, vol. 12, no. 1, pp. 123–155, Mar. 2018, doi: 10.1214/17-AOAS1093.

[48] S. Romano, N. X. Vinh, K. Verspoor, and J. Bailey, ''The randomized information coefficient: Assessing dependencies in noisy data,'' *Mach. Learn.*, vol. 107, no. 3, pp. 509–549, Mar. 2018, doi: 10.1007/s10994-017-5664-2.

[49] N. N. Kachouie and W. Deebani, ''Association factor for identifying linear and nonlinear correlations in noisy conditions,'' *Entropy*, vol. 22, no. 4, Apr. 2020, Art. no. 440, doi: 10.3390/e22040440.

[50] T. Wang and W. Li, ''Kernel learning and optimization with Hilbert–Schmidt independence criterion,'' *Int. J. Mach. Learn. Cyber.*, vol. 9, pp. 1707–1717, Oct. 2018, doi: 10.1007/s13042-017-0675-7.

[51] T. Górecki, M. Krzyśko, and W. Wołyński, ''Independence test and canonical correlation analysis based on the alignment between kernel matrices for multivariate functional data,'' *Artif. Intell. Rev.*, vol. 53, no. 1, pp. 475–499, Jan. 2020, doi: 10.1007/s10462-018-9666-7.

[52] B. B. Damodaran, N. Courty, and S. Lefèvre, ''Sparse Hilbert Schmidt independence criterion and surrogate-kernel-based feature selection for hyperspectral image classification,'' *IEEE Trans. Geosci. Remote Sens.*, vol. 55, no. 4, pp. 2385–2398, Apr. 2017, doi: 10.1109/TGRS.2016.2642479.

[53] H. Lyu, M. Wan, J. Han, R. Liu, and C. Wang, "A filter feature selection method based on the maximal information coefficient and gram-Schmidt orthogonalization for biomedical data mining," *Comput. Biol. Med.*, vol. 89, pp. 264–274, Oct. 2017, doi: 10.1016/j.compbiomed.2017.08.021.

[54] K. Zheng and X. Wang, "Feature selection method with joint maximal information entropy between features and class," *Pattern Recognit.*, vol. 77, pp. 20–29, May 2018, doi: 10.1016/j.patcog.2017.12.008.

[55] S. Liaghat and E. G. Mansoori, "Filter-based unsupervised feature selection using Hilbert–Schmidt independence criterion," *Int. J. Mach. Learn. Cybern.*, vol. 10, no. 9, pp. 2313–2328, Sep. 2019, doi: 10.1007/s13042-018-0869-7.

[56] K. Zheng, X. Wang, B. Wu, and T. Wu, "Feature subset selection combining maximal information entropy and maximal information coefficient," *Appl. Intell.*, vol. 50, no. 2, pp. 487–501, Feb. 2020, doi: 10.1007/s10489-019-01537-x.

[57] L. Abdi and A. Ghodsi, "Discriminant component analysis via distance correlation maximization," *Pattern Recognit.*, vol. 98, Feb. 2020, Art. no. 107052, doi: 10.1016/j.patcog.2019.107052.

[58] D. Xie, H. Sun, and J. Qi, "A new feature extraction method based on improved variational mode decomposition, normalized maximal information coefficient and permutation entropy for ship-radiated noise," *Entropy*, vol. 22, no. 6, Jun. 2020, Art. no. 620, doi: 10.3390/e22060620.

[59] L. Zhao, Q. Hu, and W. Wang, "Heterogeneous feature selection with multi-modal deep neural networks and sparse group LASSO," *IEEE Trans. Multimedia*, vol. 17, no. 11, pp. 1936–1948, Nov. 2015, doi: 10.1109/TMM.2015.2477058.

[60] Y. Li, C.-Y. Chen, and W. Wasserman, "Deep feature selection: Theory and application to identify enhancers and promoters," *J. Comput. Biol.*, vol. 23, no. 5, pp. 322–336, May 2016, doi: 10.1089/cmb.2015.0189.

[61] W. Wen, C. Wu, Y. Wang, Y. Chen, and H. Li, "Learning structured sparsity in deep neural networks," in *Proc. NIPS*, Barcelona, Spain, 2016, pp. 2082–2090.

[62] S. Scardapane, D. Comminiello, A. Hussain, and A. Uncini, "Group sparse regularization for deep neural networks," *Neurocomputing*, vol. 241, pp. 81–89, Jun. 2017, doi: 10.1016/j.neucom.2017.02.029.

[63] T. Ochiai, S. Matsuda, H. Watanabe, and S. Katagiri, "Automatic node selection for deep neural networks using group lasso regularization," in *Proc. IEEE ICASSP*, New Orleans, LA, USA, Mar. 2017, pp. 5485–5489.

[64] F. Li, J. M. Zurada, Y. Liu, and W. Wu, "Input layer regularization of multilayer feedforward neural networks," *IEEE Access*, vol. 5, pp. 10979–10985, Jun. 2017, doi: 10.1109/ACCESS.2017.2713389.

[65] K. Han, Y. Wang, C. Zhang, C. Li, and C. Xu, "Autoencoder inspired unsupervised feature selection," in *Proc. IEEE ICASSP*, Calgary, AB, Canada, Apr. 2018, pp. 2941–2945, doi: 10.1109/ICASSP.2018.8462261.

[66] H. Zhang, J. Wang, Z. Sun, J. M. Zurada, and N. R. Pal, "Feature selection for neural networks using group lasso regularization," *IEEE Trans. Knowl. Data Eng.*, vol. 32, no. 4, pp. 659–673, Apr. 2020, doi: 10.1109/TKDE.2019.2893266.

[67] D. J. Ozer, "Correlation and the coefficient of determination," *Psychol. Bull.*, vol. 97, no. 2, pp. 307–315, Mar. 1985, doi: 10.1037/0033-2909.97.2.307.

[68] J. L. Rodgers and W. A. Nicewander, "Thirteen ways to look at the correlation coefficient," *Amer. Statist.*, vol. 42, no. 1, pp. 59–66, Feb. 1988, doi: 10.1080/00031305.1988.10475524.

[69] (2021). *Distance Correlation*. [Online]. Available: https://jp.mathworks.com/matlabcentral/fileexchange/39905-distance-correlation

[70] (2021). *A Kernel Statistical Test of Independence*. [Online]. Available: http://people.kyb.tuebingen.mpg.de/arthur/indep.htm

[71] (2021). *Minepy—Maximal Information-Based Nonparametric Exploration*. [Online]. Available: https://minepy.readthedocs.io/en/latest/index.html

[72] (2021). *Kaggle Machine Learning and Data Science Community*. [Online]. Available: https://www.kaggle.com/uciml/pima-indians-diabetes-database

[73] (2021). *U.S. National Library of Medicine*. [Online]. Available: https://www.ncbi.nlm.nih.gov/pubmed/

[74] (2021). *UCI Machine Learning Repository*. [Online]. Available: https://archive.ics.uci.edu/ml/datasets/diabetes+130-us+hospitals+for+years+1999-2008

[75] B. Strack, J. P. DeShazo, C. Gennings, J. L. Olmo, S. Ventura, K. J. Cios, and J. N. Clore, "Impact of HbA1c measurement on hospital readmission rates: Analysis of 70,000 clinical database patient records," *BioMed Res. Int.*, vol. 2014, Apr. 2014, Art. no. 781670, doi: 10.1155/2014/781670.

MIHO OHSAKI (Member, IEEE) received the B.E., M.E., and Dr. Eng. degrees from Kyushu Institute of Design (now, Kyushu University), in 1994, 1996, and 1999, respectively. From 1999 to 2004, she was an Assistant Professor at Shizuoka University. In 2004, she started working with Doshisha University, where she is currently a Professor. Her research interests include machine learning, knowledge discovery, and their application to biomedical data analysis. She is a member of IPSJ and JSAI.

NAOYA KISHIMOTO received the B.E. degree from Doshisha University, in 2019. He currently works with Capcom Co., Ltd. He is a User Interface Developer of action role-playing games.

HAYATO SASAKI received the B.E. and M.E. degrees from Doshisha University, in 2017 and 2019, respectively. He currently works with Yahoo Japan Corporation. His research interests include machine learning and deep learning for natural language processing.

RYOJI IKEURA received the B.E. degree from Doshisha University, in 2020. He currently serves as a Buddhist Monk for Junkyoji Temple.

SHIGERU KATAGIRI (Life Fellow, IEEE) received the Dr. Eng. degree in information engineering from Tohoku University, in 1982. From 1982 to 1986, he worked with the Electrical Communication Laboratories, Nippon Telegraph and Telephone Public Corporation (currently, NTT). In 1986, he moved to the Advanced Telecommunications Research Institute International (ATR), and returned to the NTT Communication Science Laboratories, in 1999. Since 2006, he has been with Doshisha University, where he is currently a Professor of the Graduate School of Science and Engineering. He is an NTT Research and Development Fellow. He has played several roles in academic communities, including the Chair of the IEEE James L. Flanagan Speech and Audio Processing Award Committee, the Chair of the IEEE Kansai Section, and a member of the Science Council of Japan.

KEI OHNISHI (Member, IEEE) received the B.E., M.E., and D.E. degrees from Kyushu Institute of Design, Japan, in 1998, 2000, and 2003, respectively. He worked as a Postdoctoral Researcher with the University of Illinois at Urbana–Champaign, Kyushu Institute of Technology, and the Human Media Creation Center/Kyushu. Since October 2007, he has been an Associate Professor at Kyushu Institute of Technology. His research interests include evolutionary computation and bio-inspired algorithms. He is a member of ISAL, SOFT, and JPNSEC.

YAKUB SEBASTIAN received the Ph.D. degree in computer science from Monash University, Australia. He is currently a Lecturer in information technology at Charles Darwin University, Australia. His primary research interests include literature-based discovery, scientometric, and data science. He is a member of Australian Computer Society.

PATRICK THEN (Member, IEEE) is a Professor and the Head of School of Information and Communication Technologies in the Faculty of Engineering, Computing and Science, Swinburne University of Technology, Sarawak. He is also the Director of the Centre for Digital Futures, Swinburne University of Technology. He has established industry collaborations and has been managing and leading projects funded by industry and government agencies at the national and international level. His research interests include big data, data mining, health informatics, and the Internet of Things. He is a fellow of the Society for Design and Process Science, USA, and a Senior Member of Australian Computer Society.