Bayesian Neural Networks for High-Dimensional Nonlinear Variable Selection with a Double Parallel Monte Carlo Algorithm

Faming Liang

University of Florida

September 22, 2016

Abstract

Recent advances in high-throughput biotechnologies have provided an unprecedented opportunity for biomarker discovery, which, from a statistical point of view, can be cast as a variable selection problem. This problem is challenging due to the high-dimensional and non-linear nature of omics data, and it generally suffers three difficulties: (i) an unknown functional form, (ii) variable selection consistency, and (iii) highly-demanding computation. To circumvent the first difficulty, we employ a feed-forward neural network to approximate the unknown nonlinear function motivated by its universal approximation ability. To circumvent the second difficulty, we conduct structure selection (with variable selection being induced) for the neural network by choosing appropriate prior distributions that lead to the consistency of variable selection. We propose to resolve the computational issue by implementing the population stochastic approximation Monte Carlo algorithm, a parallel adaptive Markov Chain Monte Carlo (MCMC) algorithm, on the OpenMP platform which provides a linear speedup for the simulation. The numerical results indicate that the proposed method can execute very fast on a multicore computer and work very well for identification of relevant variables for general high-dimensional nonlinear systems. The proposed method is successfully applied to selection of anticancer drug response genes for the cancer cell line encyclopedia (CCLE) data.

Past work on Parallel and Distributed Computing

- Song, Q., Wu, M. and Liang, F. (2014). Weak Convergence Rates of Population versus Single-Chain Stochastic Approximation MCMC Algorithms. *Advances in Applied Probability*, **46**, 1059-1083.
- Song, Q. and Liang, F. (2015). A Split-and-Merge Bayesian Variable Selection Approach for Ultra-high dimensional Regression. *JRSSB*, 77, 947-972.
- Liang, F., Kim, J. and Song, Q. (2016). A Bootstrap Metropolis-Hastings Algorithm for Bayesian Analysis of Big Data. *Technometrics*, **58**, 304-318.
- Liang, F., Shi, R. and Mo, Q. (2016). A Split-and-Merge Approach for Singular Value Decomposition of Large-Scale Matrices. *Statistics and Its Interface*, in press.

Double Parallel Monte Carlo

- Parallel Markov chain Monte Carlo:
  - The iterative nature of MCMC makes it hard to be parallelized (Asynchronous Gibbs Sampler ...)
  - Many short runs versus a single long run: "Many short runs can only diagnose nonconvergence when you can quickly get from the starting distribution to every interesting feature of the equilibrium distribution. It only works in toy problems where you already know the answer." (Charles Geyer, http://users.stat.umn.edu/ geyer/mcmc/one.html)
  - Population SAMC is more efficient than single chain SAMC (Song, Wu and Liang, 2014) due to the interactions between different chains.
- Data Parallel: The results from subdata are difficult to combine.
  - Consensus Monte Carlo
  - Wasserstein posterior (WASP) Monte Carlo
  - Data Parallel SAMC (working paper)

Outline of Today's Talk

- A Brief Review of pop-SAMC
- Application of pop-SAMC: BNN for high-dimensional nonlinear variable selection
- Data parallel for Bayesian variable selection

Pop-SAMC Algorithm

Suppose that we are interested in sampling from a distribution,

$$f(x) = c\psi(x), \quad x \in \mathcal{X},$$

where $\mathcal{X}$ is the sample space and c is an unknown constant.

Pop-SAMC Algorithm

Let $E_1, \ldots, E_m$ denote a partition of the sample space $\mathcal{X}$. For example, the sample space can be partitioned according to the energy function of $f(x)$, i.e., $U(x) = -\log \psi(x)$, into the following subregions: $E_1 = \{x : U(x) \leq u_1\}$, $E_2 = \{x : u_1 < U(x) \leq u_2\}$, $\ldots$, $E_{m-1} = \{x : u_{m-2} < U(x) \leq u_{m-1}\}$ and $E_m = \{x : U(x) \geq u_m\}$, where $u_1 < u_2 < \cdots < u_{m-1}$ are user-specified numbers.

Given the partition, pop-SAMC seeks to draw samples from the distribution

$$f_w(x) \propto \sum_{i=1}^{m} \frac{\pi_i \psi(x)}{w_i} I(x \in E_i),$$

where $w_i = \int_{E_i} \psi(x) dx$, and π_i's specify the desired sampling frequencies for each of the subregions and they satisfy the constraints: $\pi_i > 0$ for all i and $\sum_{i=1}^{m} \pi_i = 1$.

Pop-SAMC Algorithm

1. (Population sampling) For $i = 1, \ldots, \kappa$, simulate a sample $x_{t+1}^{(i)}$ by running, for one step, the Metropolis-Hastings algorithm which starts with $x_t^{(i)}$ and admit the stationary distribution:
$$f_{\theta_t}(x) \propto \sum_{i=1}^{m} \frac{\psi(x)}{e^{\theta_{t,i}}} I(x \in E_i),$$
where $\theta_t = (\theta_{t,1}, \ldots, \theta_{t,m})$ and $\theta_{t,i}$ denotes the working estimate of $\log(w_i/\pi_i)$ at iteration t. Denote the population of samples by $\mathbf{x}_{t+1} = (x_{t+1}^{(1)}, \ldots, x_{t+1}^{(\kappa)})$.
2. (θ-updating) Set $\theta_{t+1} = \theta_t + a_{t+1}\mathbf{H}(\theta_t, \mathbf{x}_{t+1})$, where $\mathbf{H}(\theta_t, \mathbf{x}_{t+1}) = \sum_{i=1}^{\kappa}(\mathbf{z}_{t+1}^{(i)} - \boldsymbol{\pi})/\kappa$, $\mathbf{z}_{t+1}^{(i)} = (I(x_{t+1}^{(i)} \in E_1), \ldots, I(x_{t+1}^{(i)} \in E_m))$, and $\boldsymbol{\pi} = (\pi_1, \ldots, \pi_m)$.

Pop-SAMC Algorithm: Convergence & Asymptotic Normality

Song, Wu and Liang (2014) have proven the following results:

- $\theta_t \to \theta_*$, a.s., where $\theta_* = (\theta_*^{(1)}, \ldots, \theta_*^{(m)})$ is given by

$$\theta_*^{(i)} = C + \log\left(\int_{E_i} \psi(x)dx\right) - \log(\pi_i), \quad i = 1, \ldots, m,$$

  and C is a constant.

- Conditioned on the event $\{\theta_t \to \theta_*\}$,

$$\frac{\theta_t - \theta_*}{\sqrt{a_t}} \Rightarrow N(0, \Sigma),$$

  where Σ denotes a positive definite matrix.

Pop-SAMC Algorithm: Efficiency

Let $\{\theta_t^p\}$ and $\{\theta_t^s\}$ denote the estimates resulted from the pop-SAMC and single-chain SAMC, respectively. Song, Wu and Liang (2014) showed that $(\theta_t^p - \theta_*)/\sqrt{a_t}$ and $(\theta_{\kappa t}^s - \theta_*)/\sqrt{\kappa a_{\kappa t}}$ have the same asymptotic distribution with the ratio of convergence rates

$$\frac{a_t}{\kappa a_{\kappa t}} = \kappa^{\zeta - 1},$$

where $\zeta \in (0.5, 1]$ is given by the gain factor sequence

$$a_t = \frac{t_0}{t^\zeta}, \quad t \geq 1.$$

Hence, when $\zeta < 1$, the pop-SAMC algorithm is asymptotically more efficient than the single-chain SAMC algorithm. When $\zeta = 1$, the two algorithms have the same asymptotic efficiency.

Pop-SAMC Algorithm: Parallel Implementation

OpenMP is particularly suitable for a parallel implementation of the pop-SAMC algorithm.

- The **fork** step, which works on population sampling, costs the major portion of the CPU and the parallel execution provides a linear speedup for the simulation.
- The **join** step works on θ-updating, where distribution of the updated θ_t to different threads is avoided due to its shared memory mode. As shown in our examples, the pop-SAMC algorithm can execute very quickly on OpenMP.

High Dimensional Variable Selection: Linear System

During the past decade, substantial progresses have been obtained for linear systems for which the regression function can be described by a generalized linear model.

Frequentist methods: usually regularization-based

- LASSO (Tibshirani, 1996)
- elastic net (Zou and Hastie, 2005)
- SCAD (Fan and Li, 2001)
- MCP (Zhang, 2010)
- rLasso (Song and Liang, 2015)

High Dimensional Variable Selection: Linear System

Bayesian methods: Posterior Consistency or global convergence

- Bayesian subset modeling (Liang et al., 2013)
- Split-and-merge (Song and Liang, 2015)
- Non-local prior (Johnson and Rossel, 2012)

High Dimensional Variable Selection: Non-Linear System

Dictionary Approach: It is to consider a dictionary of nonlinear features and then use a regularization method to select relevant elements of the dictionary by assuming that the true regression function of the nonlinear system can be approximated by a linear combination of nonlinear features included in the dictionary.

- Additive model (Ravikumar et al., 2009): Each nonlinear feature is expressed as a basis function of a single variable. It fails to model interactions between variables.
- Lin and Zhang (2006) encodes more complex interactions among the variables, e.g., the features defined by a second-degree polynomial kernel. The size of the dictionary grows more than exponentially as one considers high-order interactions.

High Dimensional Variable Selection: Non-Linear System

Tree-based Approach: It makes use of the internals of the decision tree structure in variable selection. All observations begin in single root node and are split into two groups based on whether $X_k \geq c$ or $X_k < c$, where X_k is a chosen splitting variable and c is a chosen splitting point. The two groups form the left daughter node and the right daughter node, respectively. Then additional binary splits can be chosen for each of the two daughter nodes. Variable selection can be made based on the splitting variables.

- Random Forest
- Dynamic Tree
- Bayesian additive regression trees (BART)

High Dimensional Variable Selection: Non-Linear System

Our Approach: Bayesian feed-forward neural networks, which have properties:

- Universal Approximation Ability: a feedforward neural network is capable of approximating any continuous functions on compact subsets to any desired degree of accuracy.
- Posterior Consistency: The true density of the nonlinear system can be consistently estimated by the density of the models sampled from the posterior distribution of feed-forward neural networks.
- Population Stochastic Approximation Monte Carlo (pop-SAMC): It is a parallel adaptive MCMC algorithm, and can be implemented on the OpenMP platform.
- Variable selection: It selects variables based on the frequency how often the variable appears in the neural networks simulated from the posterior distribution. In BNN, the nonlinearity of the system and the interaction effects between different variables are modeled by including a hidden layer.

Feed-forward neural networks

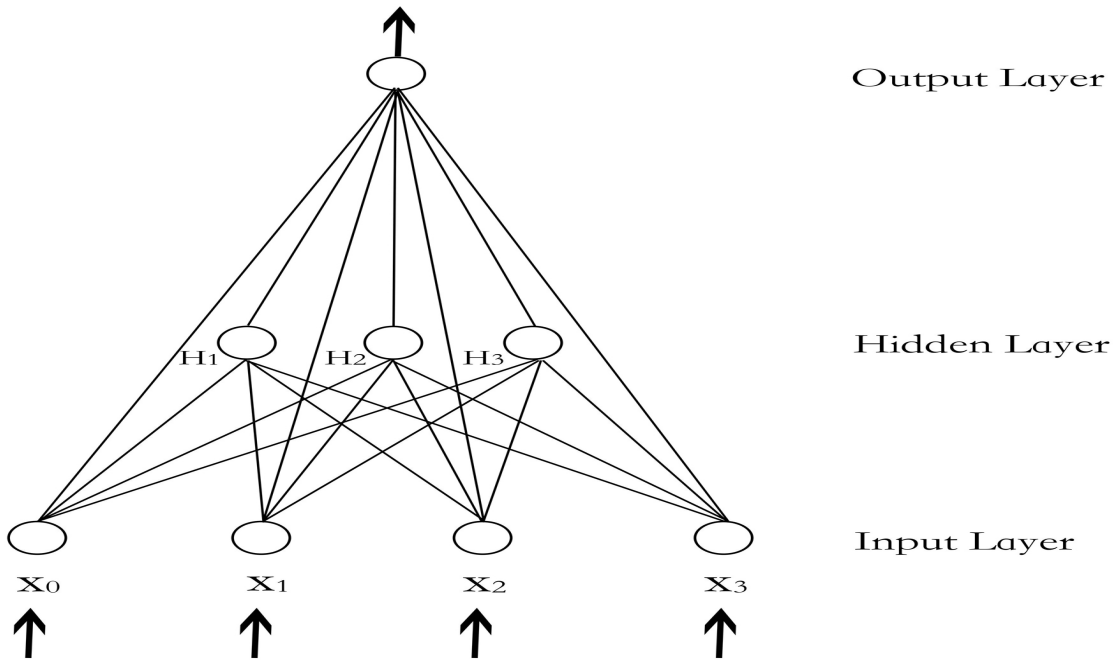

Figure: A fully connected one hidden layer MLP network with three input units (x_1, x_2, x_3), one bias unit (x_0), three hidden units (H_1, H_2, H_3), and one output unit (O). The arrows show the direction of data feeding.

Feed-forward neural networks

Let a_o denote the output of the network, and let $\psi_o(\cdot)$ denote the activation function of the output unit; that is,

$$a_o = \psi_o\Big(\sum_{i=0}^{P} I_{s_{io}} w_{io} x_i + \sum_{j=1}^{H} I_{s_{jo}} w_{jo} a_j\Big),$$

where w_{io} denotes the connection weight from input unit i to the output unit, w_{jo} denotes the connection weight from hidden unit j to the output unit, and $I_{s_{.o}}$ is the corresponding indicator for the effectiveness of the connection.

We set $\psi_o(z)$ to be the identity function (i.e., $\psi_o(z) = z$) for normal regression problems, and set $\psi_o(z)$ to be the sigmoid function for binary classification problems.

Feed-forward neural networks

Let $\psi_h(\cdot)$ denote the activation function of the hidden unit, and let a_j denote the output of the hidden unit j; that is,

$$a_j = \psi_h(\sum_{i=0}^{P} I_{s_{ij}} w_{ij} x_i) \stackrel{\triangle}{=} \psi_h(z_j), \tag{1}$$

where $I_{s_{ij}}$ is the indicator for whether the connection from input unit i to hidden unit j is included in the network; and, if included, w_{ij} denotes the connection weight from input unit i to hidden unit j.

We set $\psi_h(z)$ to be the hyperbolic tangent function, i.e., $\psi_h(z) = \tanh(z)$. An alternative choice of $\psi_h(z)$ is the sigmoid function $\psi_h(z) = 1/(1 + e^{-z})$.

Feed-forward neural networks

- For normal regression, we generally assume the response variable $y \sim N(\mu^*(\mathbf{x}), \sigma^2)$, where $\mathbf{x} = (x_0, x_1, \ldots, x_P)$, $\mu^*(\mathbf{x})$ is an unknown nonlinear function, and σ^2 denotes the variance of y.
- For binary classification, we generally assume that the response variable y is a Bernoulli random variable with the success probability $\psi_0(\mu^*(\mathbf{x}))$. In this case, a_0 works as an approximator of the success probability.
- Note that when $I_{s_{jo}} = 0$ for all $j = 1, \ldots, H$, the neural network model is reduced to a linear regression or logistic regression, depending on the choice ψ_o.

Feed-forward neural networks

Remarks:

- Given the universal approximation ability of feed-forward neural networks, the problem of variable selection for nonlinear systems is reduced to selecting appropriate variables for $\mu(\boldsymbol{\beta}, \mathbf{x})$ such that it can provide an adequate approximation to $\mu^*(\mathbf{x})$.
- Here, stemming from the universal approximation property, we have implicitly assumed that $\mu^*(\mathbf{x})$ can be well approximated by a *parsimonious neural network model* with relevant variables, and this parsimonious model is called the *true model* in the context of the paper.

BNN: Prior Setting

Let γ denote a BNN model, and let $\boldsymbol{\beta}_\gamma$ denote the corresponding connection weights.

- Conditional on γ, $\boldsymbol{\beta}_\gamma$ follows $N(0, \mathbf{V}_\gamma)$, where $\mathbf{V}_\gamma$ is a $|\gamma| \times |\gamma|$ covariance matrix, and $|\gamma|$ is the number of nonzero elements of γ.
- For any valid neural network,

$$\pi(\gamma) \propto \lambda_n^{|\gamma|}(1-\lambda_n)^{K_n-|\gamma|} I(3 \leq |\gamma| \leq \bar{r}_n, \gamma \in \mathcal{G}), \qquad (2)$$

  where $\bar{r}_n$ is the maximum network size allowed in the simulation, λ_n can be read as an approximate prior probability for each connection to be included in the network, and $\mathcal{G}$ is the set of valid neural networks.
- In general, we set the hyperparameter $\lambda_n \to 0$ as $K_n \to \infty$, which provides an automatic control for the multiplicity involved in variable selection.

BNN: Posterior Consistency

For BNN, we define the the Hellinger distance

$$d(p, p^*)^2 = \int \int |p(y, \mathbf{x}|\gamma, \boldsymbol{\beta}_\gamma)^{1/2} - p^*(y, \mathbf{x})^{1/2}|^2 \nu_y(dy)\nu_{\mathbf{x}}(d\mathbf{x}).$$

Theorem 1. Under mild conditions, we have proved that

(i) For some $c_1 > 0$, and for all sufficiently large n,

$$P^*\{\pi[d(p, p^*) > \epsilon_n | D^n] \geq e^{-0.5c_1 n\epsilon_n^2}\} \leq e^{-0.5c_1 n\epsilon_n^2}.$$

(ii) For some $c_1 > 0$, and for all sufficiently large n,

$$E^*_{D^n}\pi[d(p, p^*) > \epsilon_n | D^n] \leq e^{-c_1 n\epsilon_n^2}.$$

BNN: Posterior Consistency

- The key to the proof of Theorem 1 is to bound the Hellinger distance by a function of γ and $\boldsymbol{\beta}_\gamma$. Thanks to the mathematical tractability of the activation function $\tanh(\cdot)$, which is bounded and has a bounded derivative function, the distance function has a simple analytic bound. Then the prior distribution can be elicited to have an asymptotic focus on a neighborhood of the true model, which, as a consequence, leads to the posterior consistency.
- Since the sigmoid function has the same property as $\tanh(\cdot)$, i.e., being bounded and having a bounded derivative, Theorem 1 also holds for the networks with the sigmoid hidden unit activation function.

BNN: Variable Selection

For a variable x_i, we define its marginal inclusion probability as

$$q_i = \sum_{\gamma} e_{i|\gamma} \pi(\gamma|\mathcal{D}^n),$$

where $\pi(\gamma|\mathcal{D}^n) = \int \pi(\gamma, \boldsymbol{\beta}_\gamma|\mathcal{D}^n) d\boldsymbol{\beta}_\gamma$ is the marginal probability mass function of the model γ, and $e_{i|\gamma} = I(\sum_{j=1}^{H} I_{s_{ij}} I_{s_{jo}} + I_{s_{io}} > 0)$ is the indicator for whether x_i contributes to the output in the model γ.

The proposed approach is to choose the variables for which the marginal inclusion probability is greater than a threshold value $\hat{q}$; that is, setting $\widehat{\gamma}_{\hat{q}} = \{\mathbf{x}_j : q_j > \hat{q}, j = 1, 2, \ldots, P_n\}$ as an estimator of the set $\{x_i : e_{i|\gamma_*} = 1, i = 1, \ldots, P_n\}$.

BNN: Identifiability condition

Let $A_{\epsilon_n} = \{\gamma : d(\hat{f}(y|x,\gamma), f(y|x,\gamma_*)) \leq \epsilon_n\}$. Define

$$\rho(\epsilon_n) = \sum_{\gamma \in A_{\epsilon_n}} \sum_{1 \leq k \leq K_n} |e_{c_k|\gamma_*} - e_{c_k|\gamma}| \pi(\gamma|D^n),$$

which measures the distance between the true model and sample models in the ϵ_n-neighborhood A_{ϵ_n}. Then the identifiability condition can be stated as follows:

$$\rho(\epsilon_n) \to 0, \quad \text{as } n \to \infty \text{ and } \epsilon_n \to 0, \tag{3}$$

that is, when n is sufficiently large, if a model has the same density function as the true model then the model must coincide with the true model.

BNN: Consistency of Variable Selection

Theorem 2: Under the identifiability condition,

(i) For any $\delta > 0$ and sufficiently large n,

$$P\left(\max_{1\le i\le P_n} |q_j - e_{i|\gamma_*}| \ge 2\sqrt{\delta_n + e^{-0.5cn\epsilon_n^2}}\right) \le P_n e^{-0.5cn\epsilon_n^2}.$$

(ii) (Sure screening) For all sufficiently large n,

$$P(\gamma_* \subset \hat{\gamma}_{\hat{q}}) \ge 1 - |\gamma_*| e^{-0.5cn\epsilon_n^2},$$

for some constant $c > 0$ and some $\hat{q} \in (0, 1)$, preferably one not close to 0 or 1.

(iii) (Consistency) For all sufficiently large n,

$$P(\gamma_* = \hat{\gamma}_{0.5}) \ge 1 - K_n e^{-0.5cn\epsilon_n^2}.$$

BNN: Data Parallel

Let $q_1^{(s)}, \ldots, q_p^{(s)}$ denote the marginal inclusion probabilities obtained from the sth sub-dataset. Then a meta analysis method can be used to combine them to a single set of p-values. For example, we can set

$$q_i = \Phi\left(\sum_{s=1}^{S} w_s \Phi^{-1}(q_i^{(s)}) / \sqrt{\sum_{s=1}^{S} w_s^2}\right), \quad i = 1, \ldots, p,$$

where w_s denotes the weight assigned to sub-dataset s.
Then $q_1, \ldots, q_p$ can be used for variable selection as described previously.

Multiple Modes of BNN model

The posterior of BNN can have multiple modes:

(i) the output is invariant to relabeling of hidden units;

(ii) since the activation function $\tanh(\cdot)$ is used for the hidden units, the output is invariant to a simultaneous sign change of the weights on the connections from the input units to the hidden units and the weights on the connections from the hidden units to the output unit.

To resolve this issue, we impose the following constraint:

$$I_{s_{1o}} w_{1o} \geq I_{s_{2o}} w_{2o} \geq \cdots \geq I_{s_{Ho}} w_{Ho} \geq 0,$$

that is, all the weights on the effective connections from the hidden units to the output unit are restricted to be non-negative and non-increasing (arranged from the first hidden unit to the last one).

Examples

1. $y = x_0 + 2\tanh(x_1 + 2x_2) + 2x_3 + \sigma\epsilon$,
2. $y = \frac{10x_2}{1+x_1^2} + 5\sin(x_3x_4) + 2x_5 + \epsilon$,
3. $$y = \begin{cases} 1, & e^{x_1} + x_2^2 + 5\sin(x_3x_4) - 3 > 0, \\ 0, & \text{otherwise}, \end{cases}$$

where $\sigma = 0.5$, $x_0 = 1$, $\epsilon \sim N(0, 1)$, and x_i's for $i = 1, \ldots, 500$ are generated via the equation

$$x_i = (e + z_i)/2, \quad i = 1, \ldots, P, \tag{4}$$

where e and z_i are independently generated from $N(0, 1)$; that is, all variables are highly correlated with a correlation coefficient of 0.5.

Example 1

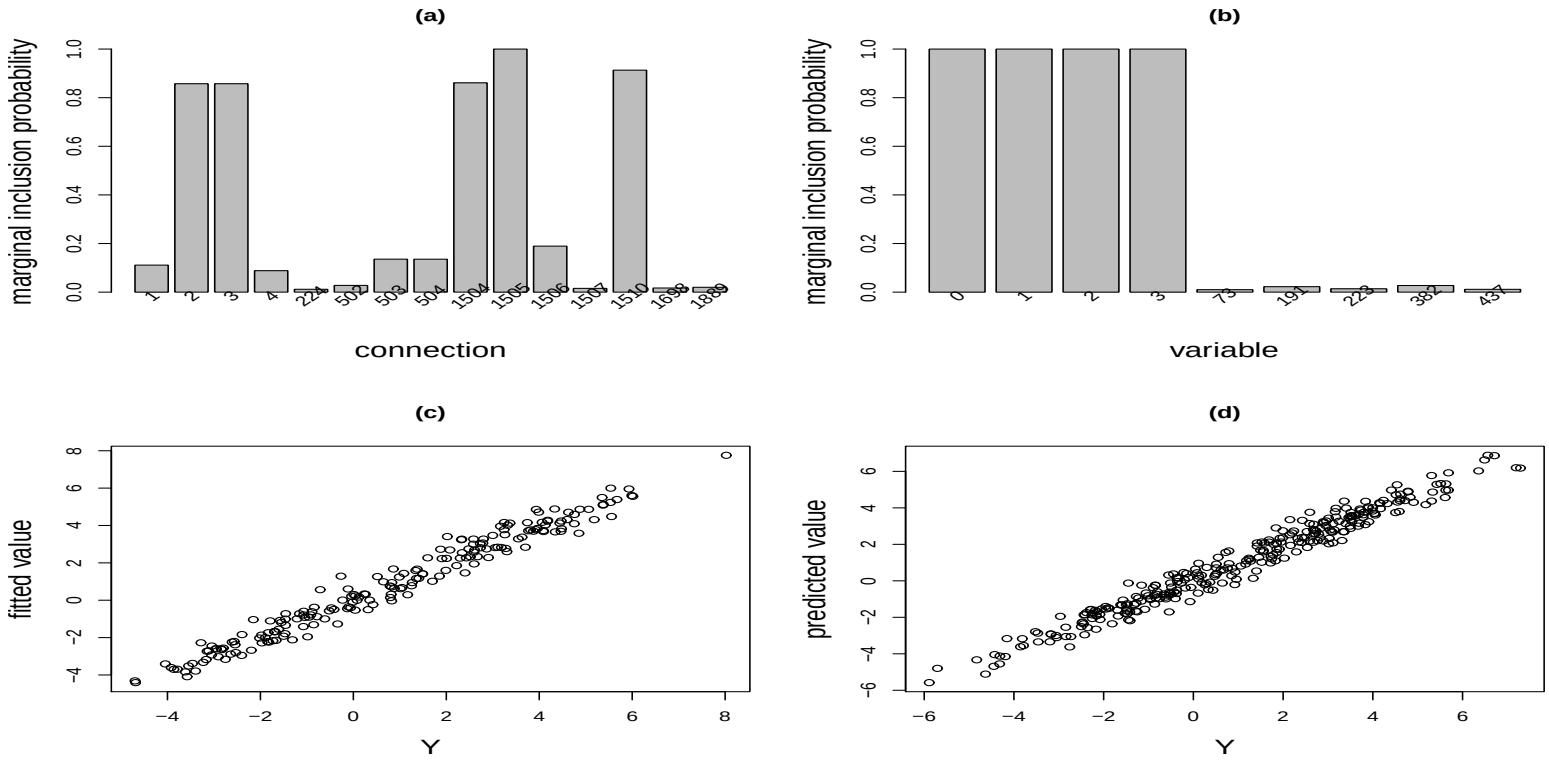

Figure: Example 1: (a) Marginal inclusion probabilities of the connections with the marginal inclusion probability greater than 0.01; (b) marginal inclusion probabilities of the covariates with the marginal inclusion probability greater than 0.01; (c) scatter plot of Y and the fitted value $\hat{Y}$ for training data; and (d) scatter plot of Y and the predicted value $\hat{Y}$ for test data.

Examples 2

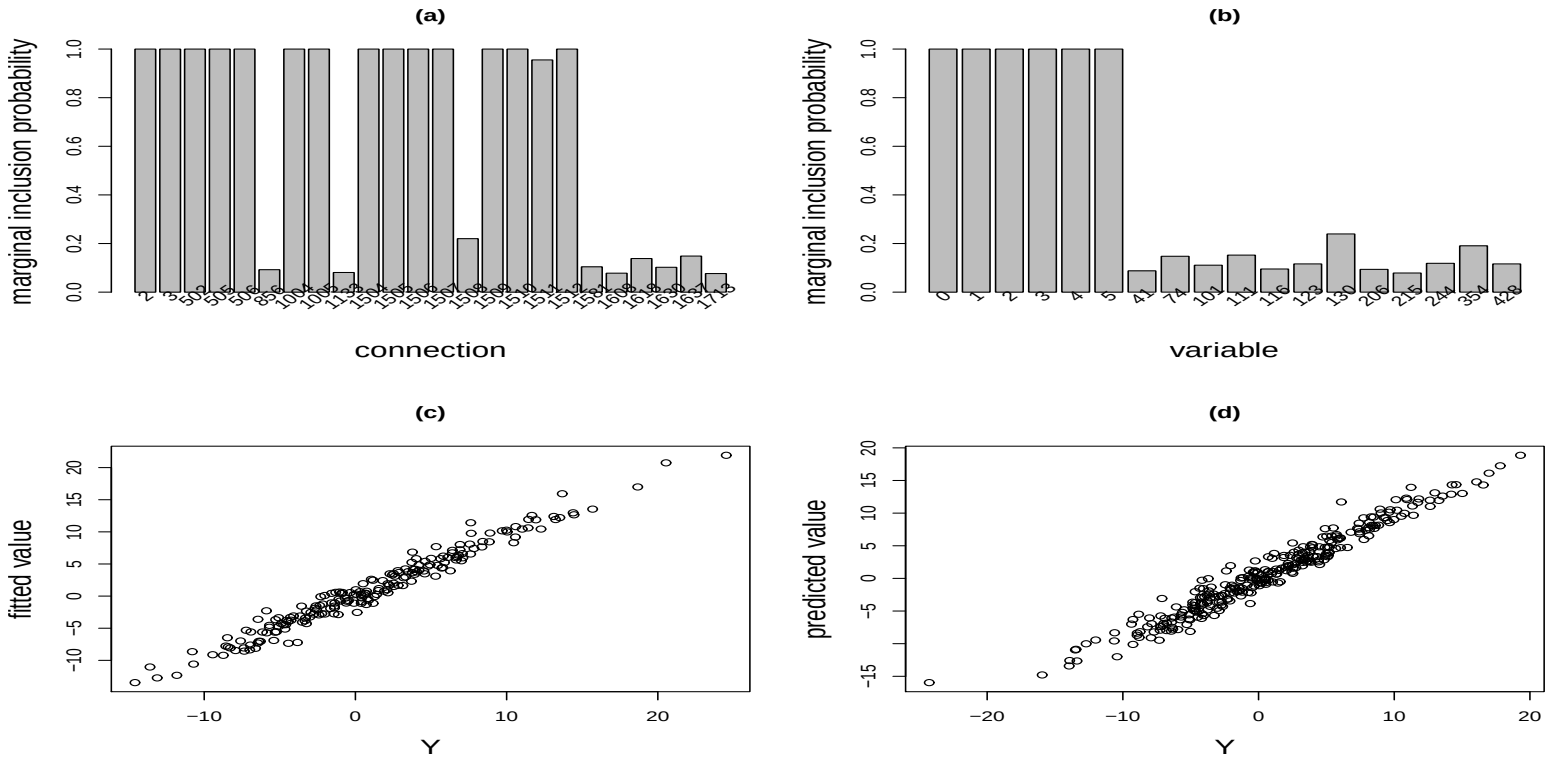

Figure: Example 2: (a) marginal inclusion probabilities of the connections with the marginal inclusion probability greater than 0.075; (b) marginal inclusion probabilities of the covariates with the marginal inclusion probability greater than 0.075; (c) scatter plot of Y and the fitted value $\hat{Y}$ for training data; and (d) scatter plot of Y and the predicted value $\hat{Y}$ for test data.

Examples 2

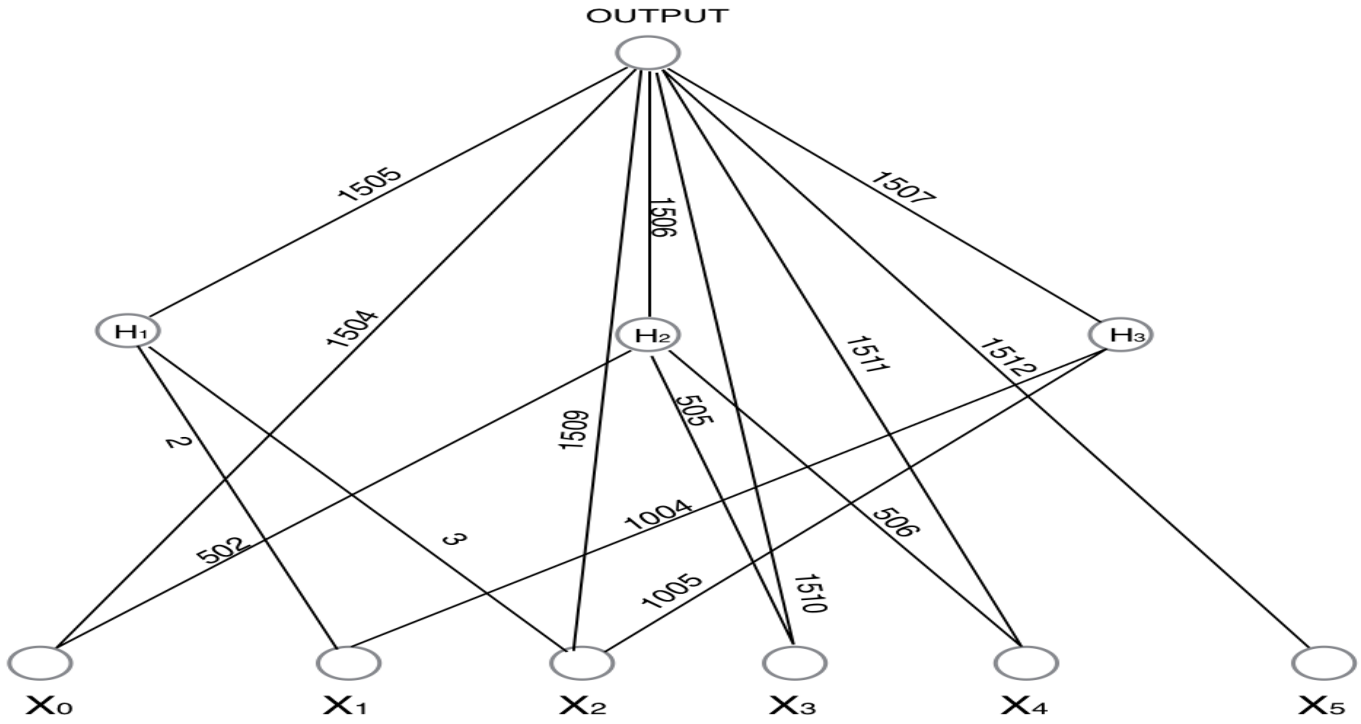

Figure: Median probability network produced by BNN for the simulated nonlinear regression example: the corresponding marginal inclusion probabilities are shown in Figure 3(a).

Example 2

Table: Comparison of BNN, GAM, random forest, and BART in variable selection and nonlinear prediction for Example 2: "MPM" denotes the median probability model marginal inclusion probability greater than 0.5; "MSFE" is for mean-squared fitting error, "MSPE" denotes the mean squared prediction error for the mean response. The results are averaged based 10 datasets.

Methods	Setting	$\overline{\lvert \mathbf{s}_j^* \rvert}$	fsr	nsr	MSFE	MSPE
BNN	FDR(0.05)	5.3 (0.21)	0.057	0	1.61 (0.15)	2.05 (0.23)
	MPM	5.4 (0.22)	0.074	0		
GAM		41.3 (6.77)	0.898	0.16	3.78 (0.37)	6.09 (0.29)
RF		3.7 (0.30)	0.49	0.62	1.60 (0.02)	9.53 (0.26)
BART	20 trees	5.9 (2.87)	0.64	0.58	2.79 (0.17)	8.45 (0.34)
	35 trees	8.0 (4.34)	0.75	0.60	1.54 (0.09)	8.57 (0.42)
	50 trees	4.3 (2.53)	0.56	0.62	0.82 (0.07)	8.34 (0.38)

Example 2

Table: Effects of the number of hidden units on the performance of BNN.

Methods	Setting	$\overline{\lvert \mathbf{s}_j^* \rvert}$	fsr	nsr	MSFE	MSPE
BNN(H=3)	FDR(0.05)	5.3 (0.21)	0.057	0	1.61 (0.15)	2.05 (0.23)
	MPM	5.4 (0.22)	0.074	0		
BNN(H=5)	FDR(0.05)	5.5 (0.22)	0.091	0	1.62 (0.14)	2.05 (0.29)
	MPM	5.3 (0.33)	0.094	0.04		
BNN(H=7)	FDR(0.05)	5.5 (0.17)	0.091	0	1.48 (0.09)	1.87 (0.18)
	MPM	5.2 (0.29)	0.077	0.04		

Example 3

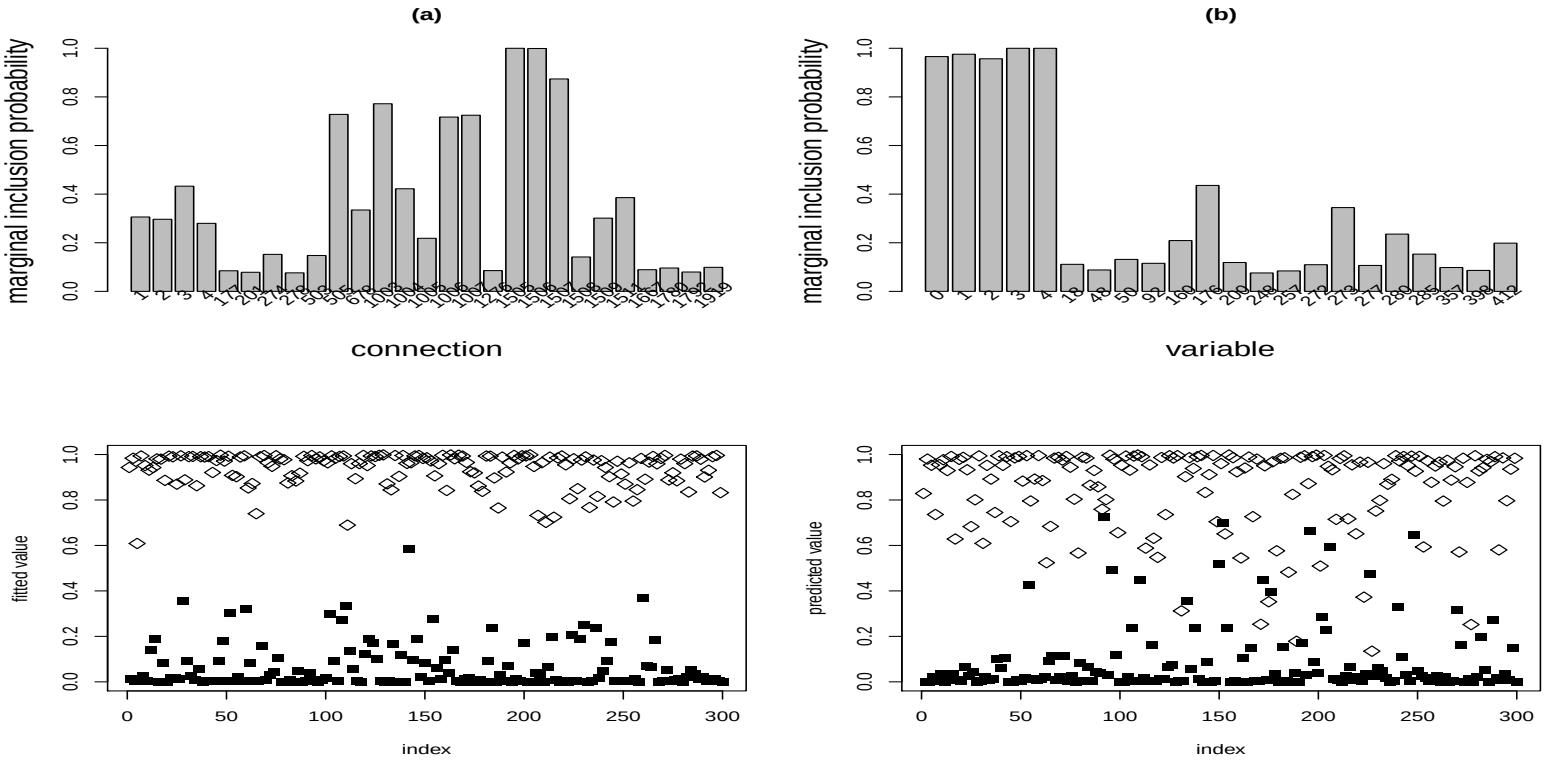

Figure: Classification Example: (a) marginal inclusion probabilities of the connections with the marginal inclusion probability greater than 0.075; (b) marginal inclusion probabilities of the covariates with the marginal inclusion probability greater than 0.075; (c) fitted value of Y (open diamond for true $Y = 1$, filled square for true $Y = 0$); and (d) predicted value of Y (open diamond for true $Y = 1$, filled square for true $Y = 0$).

Example 3

Table: Comparison of BNN, GAM, random forest, and BART in variable selection and class prediction for the simulated classification example.

Methods	Setting	$\overline{\lvert \mathbf{s}_j^* \rvert}$	fsr	nsr	Fitting(%)	Prediction(%)
BNN	FDR(0.05)	4.8 (0.55)	0.188	0.025	4.53 (0.73)	8.45 (0.73)
	MPM	5.5 (0.67)	0.291	0.025		
GAM		13.5 (3.68)	0.73	0.10	12.13 (1.34)	15.57 (1.97)
RF	250 trees	8.4 (1.73)	0.70	0.375	27.1 (1.77)	21.3 (2.10)
	500 trees	7.2 (1.16)	0.56	0.20	24.6 (1.94)	20.1 (1.72)
	750 trees	7.5 (2.66)	0.57	0.20	22.3 (1.89)	19.57 (2.10)
BART	20 trees	2.8 (0.33)	0.0	0.30	9.90 (1.04)	17.72 (1.83)
	35 trees	3.0 (0.26)	0.0	0.25	6.83 (0.89)	17.00 (1.34)
	50 trees	3.0 (0.30)	0.0	0.25	5.33 (0.75)	15.57 (1.42)
	75 trees	3.3 (0.33)	0.03	0.20	4.47 (0.55)	16.90 (1.58)

CCLE Data

The CCLE dataset consisted of 8-point dose-response curves for 24 chemical compounds across over 400 cell lines. For each cell line, it consisted of the expression data of 18,926 genes. Our goal is to identify the genes that respond to the chemical compounds, which is fundamental to elucidate the response mechanism of anticancer drugs and critical to precision medicine for selecting right drugs for right patients.

We used the area under the dose-response curve, which is termed as activity area to measure the sensitivity of drug to a given cell line.

Three Drugs

We gave a detailed analysis for three drugs, topotecan, 17-AAG and paclitaxel. The gene selection results for the other drugs are briefly reported later.

- Topotecan (trade name Hycamtin) is a chemotherapeutic agent that is a topoisomerase inhibitor. It has been used to treat ovarian cancer, lung cancer and other cancer types. The number of cell lines is $n = 491$.
- 17-AAG is a derivative of the antibiotic geldanamycin that is being studied in the treatment of cancer, specific young patients with certain types of leukemia or solid tumors, especially kidney tumors. The number of cell lines is $n = 490$.
- Paclitaxel is a drug used to treat ovarian, breast, lung, pancreatic and other cancers. The number of cell lines is $n = 490$.

Marginal Feature Screening

(a) Apply the nonparanormal transformation (Liu et al., 2009) to the data to get the transformed variables $\tilde{Y}$ and $\tilde{X}_1, \ldots, \tilde{X}_P$.

(b) For each $k = 1, \ldots, P$, calculate the Henze-Zirkler test statistic

$$\omega(\tilde{Y}, \tilde{X}_k) = \frac{n}{1+2\beta^2} + \frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{n}\exp\left(-\frac{\beta^2}{2}D_{ij}\right) - \frac{2}{1+\beta^2}\sum_{i=1}^{n}\exp\left(-\frac{\beta^2}{2(1+\beta^2)}D_i\right),$$

where $\beta = (1.25n)^{1/6}/\sqrt{2}$ is the smoothing parameter, $D_{ij} = (\tilde{x}_{ki} - \tilde{x}_{kj})^2 + (\tilde{y}_i - \tilde{y}_j)$, $D_i = \tilde{x}_{ki}^2 + \tilde{y}_i^2$, and $\tilde{x}_{ki}$ and $\tilde{y}_i$ denote the kth elements of $\tilde{X}_k$ and $\tilde{Y}$, respectively.

(c) Select $p' = \lceil n/\log(n) \rceil$ genes with the largest value of $\omega(\cdot, \cdot)$ for further analysis.

Gene Selection

Table: The supscript * indicates the genes for which the interaction with the drug or the drug target gene has been reported in the PubMed Articles available at http://www.ncbi.nlm.nih.gov/pubmed/.

Drug	Target	Number	Genes
17-AAG	HSP90	5	NQO1*, ATP6V0E1, ZFP30, RPUSD4, MMP24
Paclitaxel	TUBB1	3	BCL2L1*, SSRP1, SPATA5L1
Topotecan	TOP2	3	SLFN11*, HSPB8, CD63

Gene Selection

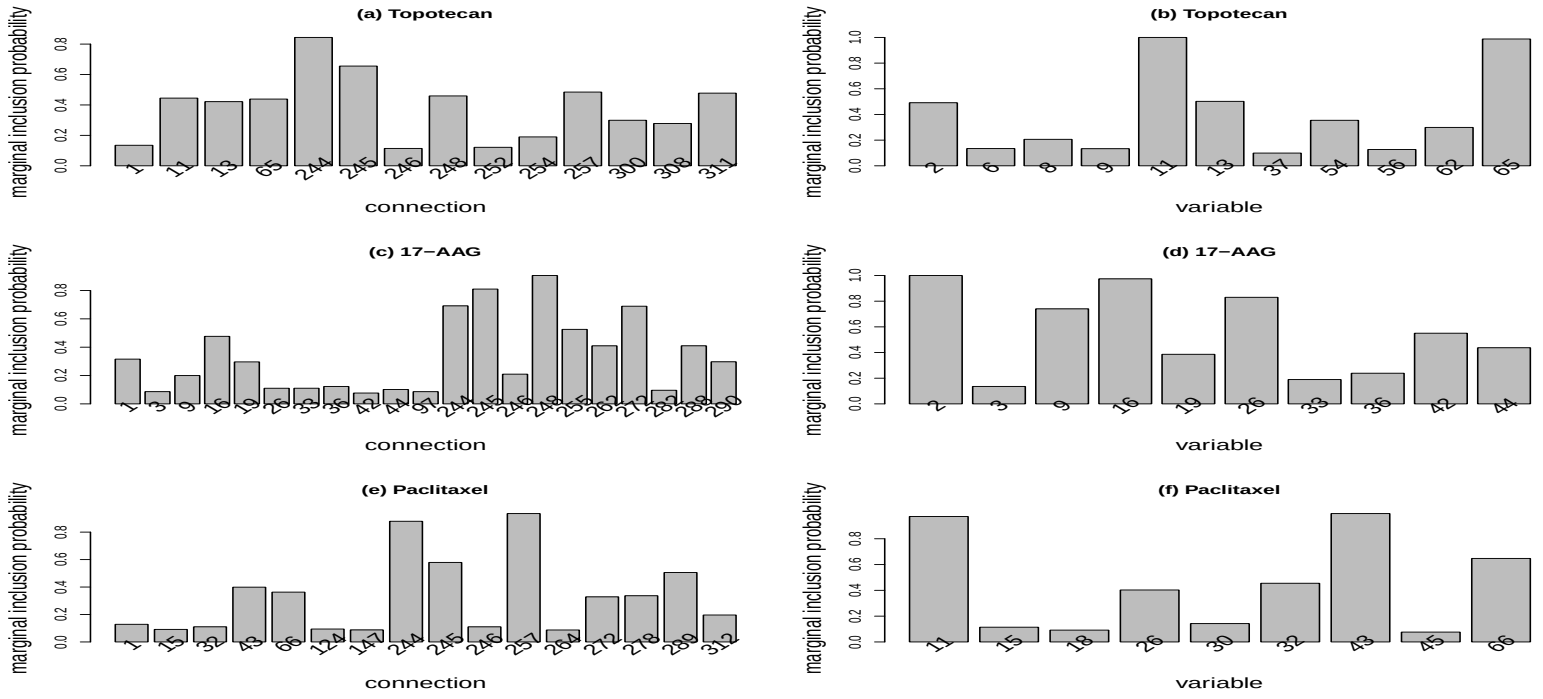

Figure: Marginal inclusion probabilities of the network connections and the corresponding genes with the marginal inclusion probability greater than 0.075. The upper panel is for Topotecan, the selected genes are SLFN11 (bar 11), HSPB8 (bar 65) and CD63 (bar 13); The middle panel is for 17-AAG, the selected genes are NQO1 (bar 2), ATP6V0E1 (bar 16), ZFP30 (bar 26), RPUSD4 (bar 9) and MMP24 (bar 42); The lower panel is for paclitaxel, the selected genes are BCL2L1 (bar 43), SSRP1 (bar 11) and SPATA5L1 (bar 66).

Gene Selection

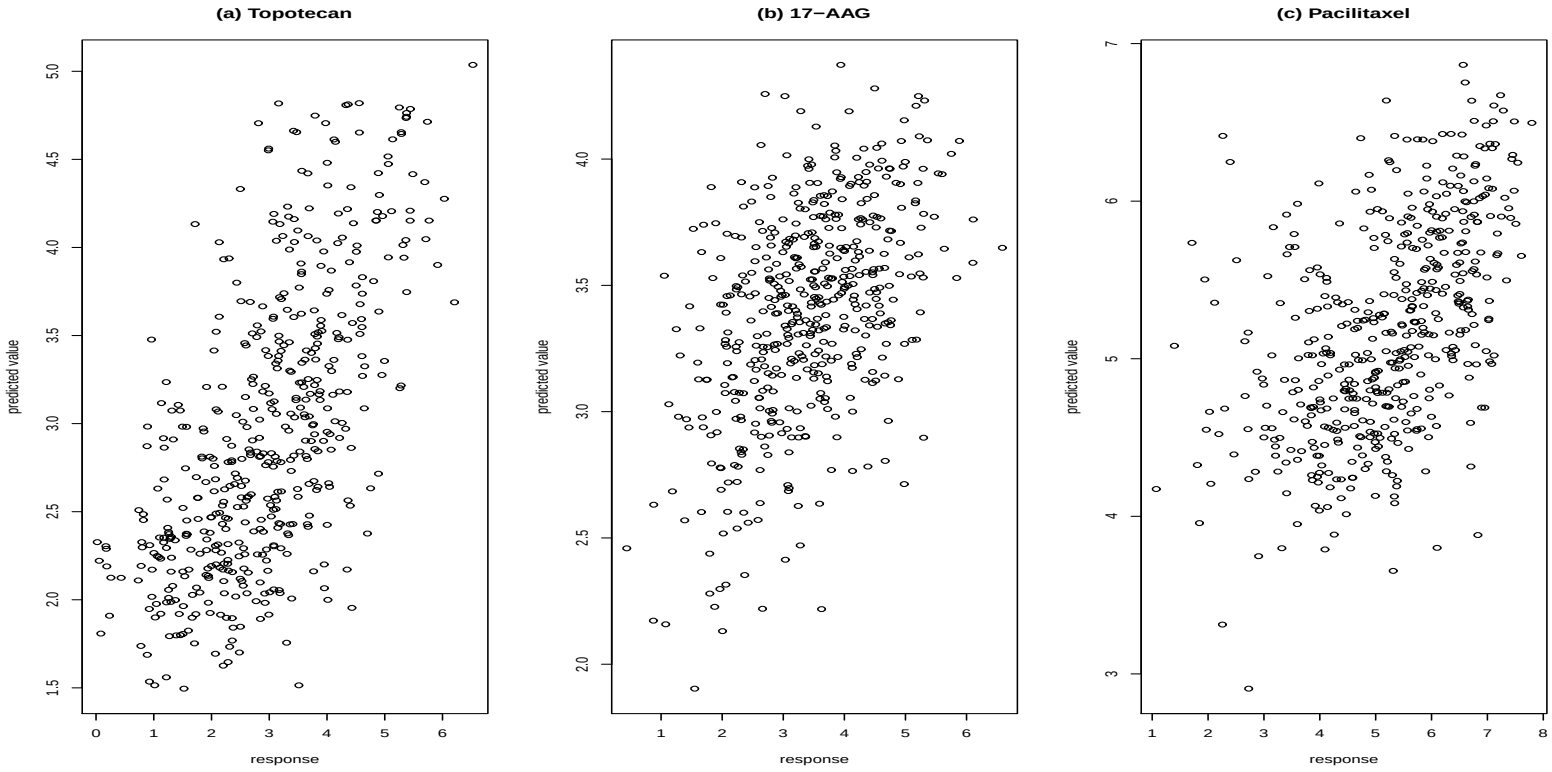

Figure: Scatter plots of predicted values by BNN versus observed response values for three drugs: (a) Topotecan, (b) 17-AAG, and (c) Pacilitaxel.

Gene Selection

Table: Comparison of BNN with GAM, RF, and BART in gene selection for the drug Topotecan, 17-AAG and Paclitaxel: "Fitting" denotes the mean squared fitting error, and #gene denotes the number of selected genes.

Method	Topotecan		17-AAG		Paclitaxel	
	Fitting	#gene	Fitting	#gene	Fitting	#gene
GAM	0.77	32	0.54	54	0.84	48
RF	0.90	80	0.78	80	1.15	40
BART	0.47	8	0.34	7	0.59	9
BNN	0.78	3	0.67	5	1.08	3

Drug response genes selected by BNN for CCLE data

Drug	Target	Number	Genes
17-AAG	HSP90	5	NQO1*, ATP6V0E1, ZFP30, RPUSD4, MMP24
AEW541	IGF1R	4	SLC44A1, UST*, TMEM229B, SP1*
AZD0530	ABL	2	APOO, SPINT2*
AZD6244	MEK	11	SPRY2*, PLEKHG4B, CSF1*, MYEOV, KIF1B, RNF125, CMTM7, LYZ, PDZD2, EML1, TMC6
Erlotinib	EGFR	11	GJB3, EVPL, MGC4294*, STX2, ARHGAP27, C1orf172, BSPRY, HSD17B8, FUT1, TUBB2A, PTPN6
Irinotecan	TOP2	3	SLFN11*, ARHGAP19, CPSF6
L-685458	GS	9	GHRLOS2*, CHERP, EIF4EBP2, RHOC*, RAB5C, RBBP5, DCAF12, DNAJB1, CTSL1*
Lapatinib	EGFR	3	GRB7*, PRSS16, EPHA1*
LBW242	XIAP	8	RIPK1*, SLC7A9, GORASP2, CCDC54, TMEM177, DCP2, RCOR3, KRTAP4-6
Nilotinib	ABL	11	FBXO46, CNOT7, C7orf29, KDM3B, C4orf29, PTPN14, RHOC, PLEC, FAM129B, AHNAK2, CD38*
Nutlin-3	MDM2	10	KDM5A, DDB2*, RBM15, CDS1*, RAB28, CCDC30, FEM1A, ATXN7, SIRT2*, OSTC
Paclitaxel	TUBB1	3	BCL2L1*, SSRP1, SPATA5L1
Panobinostat	HDAC	5	EIF4EBP2, MYOF, LARP6, TGFB2*, AXL*

Drug response genes selected by BNN for CCLE data

Drug	Target	Number	Genes
PD-0325901	MEK	11	SPRY2*, RNF125, HHLA3, KLF3, C9orf167, PLEKHG4B, CYR61*, HIVEP1, ODZ2, OSBPL3, DUSP1*
PD-0332991	CDK4	4	PUM2, COX18, AVPI1, CAV1*
PF2341066	c-MET	12	ELF2, MET*, LRMP, HGF*, CBFA2T3, TSEN34, ANP32A, TM4SF1, KIF2A, ENAH, RPA2, IT-PRIPL1
PHA-665752	c-MET	7	SYAP1, MRPL24, INHBB, SPCS2, TNFRSF1B, MICB, HGF*
PLX4720	RAF	12	PLEKHH3, MEX3C, VPS33B, ANKAR*, ACP5, B3GALNT1, TNFAIP2, ADORA1*, PHACTR1, PLP2, IL20RA, PSORS1C1
RAF265	RAF	6	PIK3CD*, SFPQ, SYT17, RGS6, C15orf57, TMEM79
Sorafenib	RTF	12	LAIR1, MDM4, RBBP5, RBM12, WFS1, FBXO46, C9orf3, RPL22, FN1, CD48, BLM*, NCBP2
TAE684	ALK	2	ARID3A, SP1*
TKI258	FGFR	4	WFDC3, SDC4*, INPP5B, FECH
Topotecan	TOP2	3	SLFN11*, HSPB8, CD63
ZD-6474	EGFR	3	APOO, NMI*, KLF2*

Discussion

- We have proposed an innovative method of variable selection for general high-dimensional nonlinear systems, established the consistency of the proposed method, and successfully applied the proposed method to personalized medicine.
- The computational issue involved in the proposed method is resolved by implementing a parallel adaptive MCMC algorithm on the OpenMP platform.
- Feed-forward neural networks have been traditionally viewed as a blackbox approximator to an objective function. However, the proposed research indicates that this view is not completely correct for BNN: The structures of the networks sampled from the posterior distribution indicate the relevance of the selected covariates to the output variable.

Discussion (continued)

- The multiple-hidden-layer neural network with a small central layer has been widely used for reducing the dimensionality of image data, where the outputs of the central layer units can be viewed as the nonlinear principal component vectors of the high-dimensional data. This is termed as deep learning in computational science. From the perspective of nonlinear dimension reduction, the proposed research, which focuses on variable selection, can be viewed as a complementary work to deep learning.

Acknowledgments

- NSF grants.
- NIH R01GM117597