An Introduction to Stochastic Deep Learning

Faming Liang

Purdue University

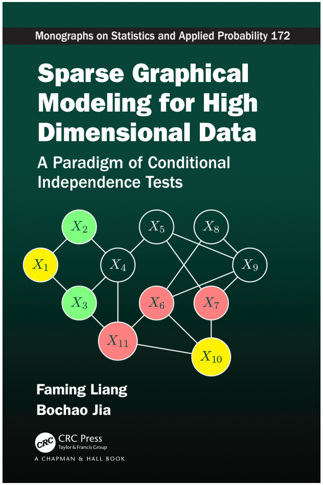
November 28, 2023

Abstract

We have developed a new type of stochastic neural network (StoNet), which is formulated as a composition of many simple linear/logistic regression models, and designed an adaptive stochastic gradient MCMC algorithm for its training. The StoNet fits into the framework of statistical modeling, allowing us not only to address fundamental issues in deep learning, such as structural interpretability and uncertainty quantification, but also to provide a platform for transferring the theory and methods developed for linear models to deep learning. We showcase the integration of reproducing kernel methods into deep neural networks to enhance their training and prediction performance. Furthermore, we demonstrate how to use the StoNet to perform nonlinear sufficient dimension reduction and causal inference on high-dimensional data. Lastly, we illustrate how to leverage the StoNet to handle special types of data, such as those with missing values or measurement errors, and how to use it to perform statistical inference for conventional deep neural networks. This talk is based on joint work with Yan Sun, Siqi Liang, and Yaxin Fang.

A New Book

It offers a comprehensive framework for mastering the complexities of sparse graphical modeling for high-dimensional data through the use of conditional independent tests. These tests are strategically conducted within a Markov neighborhood, ensuring both the low dimensionality of the conditioning set and their equivalence to the original high-dimensional conditional independence tests.


Monographs on Statistics and Applied Probability 172
Sparse Graphical Modeling for High Dimensional Data
A Paradigm of Conditional Independence Tests
X_1 X_2 X_3 X_4 X_5 X_6 X_7 X_8 X_9 X_{10} X_{11}
Faming Liang
Bochao Jia
CRC Press
Taylor & Francis Group
A CHAPMAN & HALL BOOK

Motivations

Deep learning has played a crucial role in the recent advancements of data science. However, when viewed from the perspective of statistical modeling, it suffers from several fundamental issues:

- overparameterization
- miscalibration
- uninterpretability in structure

which have posed great challenges in understanding the performance of deep learning theoretically.

Motivations

On the other hand, statistics, during its century-long history, has developed principles and methods to address above issues for traditional statistical models particularly linear models.

Question:
Whether and how can we bridge the gap between linear models and DNNs such that the theory and methods developed for linear models can be transferred to DNNs to have the above issues adequately addressed?

Related work: Some authors have tried to understand the performance of the DNNs from the perspective of kernel learning. They proved the transition to linearity of the DNN model and constancy of the neural tangent kernel (NTK) when the DNNs are sufficiently wide.

Basic Idea: StoNet

We reformulate the DNN as a composition of many simple linear or logistic regressions by adding random noise to the feeding value of each hidden unit, while maintaining its universal approximation power.

As a result, the stochastic neural network (StoNet) falls into the framework of statistical modeling, which does not only allow us to address many fundamental issues in deep learning, such as overparameterization, miscalibration, and structural uninterpretability, but also provides us with a platform for transferring the theory and methods from linear models to deep learning.

StoNet: Structure

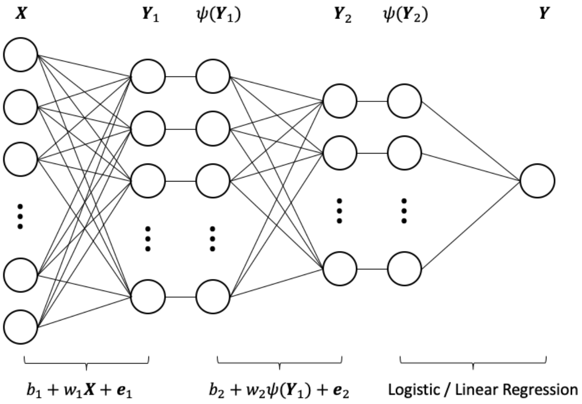


Figure 1: An illustrative plot for the structure of a StoNet with two hidden layers.

Mathematical Formulation: DNN

Consider a DNN model with h hidden layers. We can rewrite the DNN in the following form

$$\begin{aligned} \tilde{\boldsymbol{Y}}_1 &= \boldsymbol{b}_1 + \boldsymbol{w}_1 \boldsymbol{X}, \\ \tilde{\boldsymbol{Y}}_i &= \boldsymbol{b}_i + \boldsymbol{w}_i \Psi(\tilde{\boldsymbol{Y}}_{i-1}), \quad i = 2, 3, \ldots, h, \\ \boldsymbol{Y} &= \boldsymbol{b}_{h+1} + \boldsymbol{w}_{h+1} \Psi(\tilde{\boldsymbol{Y}}_h) + \boldsymbol{e}_{h+1}, \end{aligned} \tag{1}$$

where $\boldsymbol{e}_{h+1} \sim N(0, \sigma_{h+1}^2 I_{d_{h+1}})$ is Gaussian random error.

Mathematical Formulation: StoNet

The StoNet, as a probabilistic deep learning model, is given by

$$
\begin{aligned}
\boldsymbol{Y}_1 &= \boldsymbol{b}_1 + \boldsymbol{w}_1 \boldsymbol{X} + \boldsymbol{e}_1, \\
\boldsymbol{Y}_i &= \boldsymbol{b}_i + \boldsymbol{w}_i \Psi(\boldsymbol{Y}_{i-1}) + \boldsymbol{e}_i, \quad i = 2, 3, \ldots, h, \\
\boldsymbol{Y} &= \boldsymbol{b}_{h+1} + \boldsymbol{w}_{h+1} \Psi(\boldsymbol{Y}_h) + \boldsymbol{e}_{h+1},
\end{aligned} \tag{2}
$$

where $\boldsymbol{Y}_1, \boldsymbol{Y}_2, \ldots, \boldsymbol{Y}_h$ can be viewed as latent variables. Further, we assume that $\boldsymbol{e}_i \sim N(0, \sigma_i^2 I_{d_i})$ for $i = 1, 2, \ldots, h, h+1$.

In words, the StoNet has been formulated as a composition of many simple linear/logistic regressions, which makes its structure more designable and interpretable.

StoNet: an Approximator to DNN-1

Let $\boldsymbol{\theta}_i = (\boldsymbol{w}_i, \boldsymbol{b}_i)$, let $\boldsymbol{\theta} = (\boldsymbol{\theta}_1, \boldsymbol{\theta}_2 \cdots, \boldsymbol{\theta}_{h+1})$ denote the parameter vector of StoNet.

Assumption 1

(i) *Θ is compact, i.e., Θ is contained in a d_θ-ball centered at 0 with radius r;*

(ii) *$\mathbb{E}(\log \pi(\boldsymbol{Y}|\boldsymbol{X}, \boldsymbol{\theta}))^2 < \infty$ for any $\boldsymbol{\theta} \in \Theta$;*

(iii) *the activation function $\psi(\cdot)$ is c'-Lipschitz continuous for some constant c';*

(iv) *the network's depth h and widths d_i's are both allowed to increase with n;*

(v) *$\sigma_{n,1} \le \sigma_{n,2} \le \cdots \le \sigma_{n,h+1}$, $\sigma_{n,h+1} = O(1)$, and $d_{h+1}(\prod_{i=k+1}^{h} d_i^2) d_k \sigma_{n,k}^2 \prec \frac{1}{h}$ for any $k \in \{1, 2, \ldots, h\}$.*

Loss Surface

Theorem 1

Suppose Assumption 1 holds. Then the StoNet (2) and the neural network (1) have asymptotically the same loss function, i.e.,

$$\sup_{\boldsymbol{\theta}\in\Theta}\left|\frac{1}{n}\sum_{i=1}^{n}\log\pi(\boldsymbol{Y}^{(i)},\boldsymbol{Y}_{mis}^{(i)}|\boldsymbol{X}^{(i)},\boldsymbol{\theta})-\frac{1}{n}\sum_{i=1}^{n}\log\pi(\boldsymbol{Y}^{(i)}|\boldsymbol{X}^{(i)},\boldsymbol{\theta})\right|\xrightarrow{p}0,\quad as\quad n\to\infty, \tag{3}$$

where $\boldsymbol{Y}_{mis}=(\boldsymbol{Y}_1,\boldsymbol{Y}_2,\ldots,\boldsymbol{Y}_h)$ *denotes the collection of all latent variables in the StoNet (2).*

Loss Surface

Let $Q^*(\boldsymbol{\theta}) = \mathbb{E}(\log \pi(\boldsymbol{Y}|\boldsymbol{X}, \boldsymbol{\theta}))$, where the expectation is taken with respect to the joint distribution $\pi(\boldsymbol{X}, \boldsymbol{Y})$. By Assumption 1-($i$)&($ii$) and the law of large numbers,

$$\frac{1}{n}\sum_{i=1}^{n} \log \pi(\boldsymbol{Y}^{(i)}|\boldsymbol{X}^{(i)}, \boldsymbol{\theta}) - Q^*(\boldsymbol{\theta}) \xrightarrow{p} 0 \tag{4}$$

holds uniformly over Θ. Further, we assume the following condition hold for $Q^*(\boldsymbol{\theta})$:

Assumption 2

(i) $Q^*(\boldsymbol{\theta})$ is continuous in $\boldsymbol{\theta}$ and uniquely maximized at $\boldsymbol{\theta}^*$; (ii) for any $\epsilon > 0$, $sup_{\boldsymbol{\theta}\in\Theta\setminus B(\epsilon)} Q^*(\boldsymbol{\theta})$ exists, where $B(\epsilon) = \{\boldsymbol{\theta} : \|\boldsymbol{\theta} - \boldsymbol{\theta}^*\| < \epsilon\}$, and $\delta = Q^*(\boldsymbol{\theta}^*) - sup_{\boldsymbol{\theta}\in\Theta\setminus B(\epsilon)} Q^*(\boldsymbol{\theta}) > 0$.

StoNet: an Approximator to DNN-2

Theorem 2

Suppose Assumptions 1 and 2 hold, and $\pi(\boldsymbol{Y}, \boldsymbol{Y}_{mis}|\boldsymbol{X}, \boldsymbol{\theta})$ *is continuous in* $\boldsymbol{\theta}$*. Let*
$\hat{\boldsymbol{\theta}}_n = \arg\max_{\boldsymbol{\theta}\in\Theta}\{\frac{1}{n}\sum_{i=1}^n \log\pi(\boldsymbol{Y}^{(i)}, \boldsymbol{Y}_{mis}^{(i)}|\boldsymbol{X}^{(i)}, \boldsymbol{\theta})\}$*. Then*

$$\|\hat{\boldsymbol{\theta}}_n - \boldsymbol{\theta}^*\| \xrightarrow{p} 0, \quad \textit{as } n \to \infty.$$

This theorem implies that the DNN (1) can be trained by training the StoNet (2). The two models are asymptotically equivalent as the sample size n becomes large.

StoNet Training-I: Imputation-Regularized Optimization (Liang et al.,2018)

- **Imputation**: For each sample $(\boldsymbol{X}^{(i)}, \boldsymbol{Y}^{(i)})$, draw $\boldsymbol{Y}_{\mathrm{mis}}^{(i,t+1)}$ from $\pi(\boldsymbol{Y}_{\mathrm{mis}}|\boldsymbol{Y}^{(i)}, \boldsymbol{X}^{(i)}, \hat{\boldsymbol{\theta}}_n^{(t)}, \boldsymbol{\sigma}_n^2)$, where t indexes the iterations.
- **Regularized optimization**: Based on the pseudo-complete data $(\boldsymbol{Y}, \boldsymbol{Y}_{\mathrm{mis}}^{(t+1)}, \boldsymbol{X})$, update $\hat{\boldsymbol{\theta}}_n^{(t)}$ by minimizing a penalized loss function, i.e., setting

$$\hat{\boldsymbol{\theta}}_n^{(t+1)} = \arg\min_{\boldsymbol{\theta}} \Big\{ -\frac{1}{n}\sum_{i=1}^{n} \log \pi(\boldsymbol{Y}^{(i)}, \boldsymbol{Y}_{\mathrm{mis}}^{(i,t+1)} | \boldsymbol{X}^{(i)}, \boldsymbol{\theta}, \boldsymbol{\sigma}_n^2) + P_{\lambda_n}(\boldsymbol{\theta}) \Big\}, \tag{5}$$

where the penalty function $P_{\lambda_n}(\boldsymbol{\theta})$ is chosen such that $\hat{\boldsymbol{\theta}}_n^{(t+1)}$ forms a consistent estimator of

$$\boldsymbol{\theta}_*^{(t+1)} = \arg\max_{\boldsymbol{\theta}} \mathbb{E}_{\boldsymbol{\theta}_n^{(t)}} \log \pi(\boldsymbol{Y}, \boldsymbol{Y}_{\mathrm{mis}} | \boldsymbol{X}, \boldsymbol{\theta}, \boldsymbol{\sigma}_n^2) \tag{6}$$

where $\boldsymbol{\theta}_*^{(t+1)}$ is called the working true parameter at iteration $t+1$.

StoNet Training-I: a note

Solving (5) corresponds to solving a series of linear regressions by noting that the joint distribution $\pi(\boldsymbol{Y}_{\mathrm{mis}}, \boldsymbol{Y}|\boldsymbol{X}, \boldsymbol{\theta}_n, \boldsymbol{\sigma}_n^2)$ can be decomposed as follows by its Markov structure:

$$\begin{aligned}\pi(\boldsymbol{Y}_{\mathrm{mis}}, \boldsymbol{Y}|\boldsymbol{X}, \boldsymbol{\theta}_n, \boldsymbol{\sigma}_n^2) &= \pi(\boldsymbol{Y}|\boldsymbol{Y}_h, \boldsymbol{\theta}_n, \boldsymbol{\sigma}_n^2) \\ &\times \pi(\boldsymbol{Y}_h|\boldsymbol{Y}_{h-1}, \boldsymbol{\theta}_n, \boldsymbol{\sigma}_n^2) \cdots \pi(\boldsymbol{Y}_1|\boldsymbol{X}, \boldsymbol{\theta}_n, \boldsymbol{\sigma}_n^2),\end{aligned} \tag{7}$$

and, furthermore, the components of $\boldsymbol{Y}_i \in \mathbb{R}^{d_i}$ are mutually independent conditional on $\boldsymbol{Y}_{i-1}$ for $i = 1, 2, \dots, h+1$.

The IRO algorithm leads to two interleaved Markov chains:

$$\boldsymbol{\theta}_n^{(0)} \to \boldsymbol{Y}_{\mathrm{mis}}^{(1)} \to \boldsymbol{\theta}_n^{(1)} \to \boldsymbol{Y}_{\mathrm{mis}}^{(2)} \to \cdots ,$$

whose convergence has been studied in Liang et al. (2018).

StoNet Training-I: Convergence

Let $\gamma^* = \{k : \theta^*_k \neq 0\}$ be the set of indexes of non-zero elements of $\boldsymbol{\theta}^*$ and select the connections by setting $\widehat{\gamma}_n^{(t)} = \{k : |\hat{\theta}_{k,n}^{(t)}| > c\sqrt{r_n}\}$ for some constant c, where θ^*_k and $\hat{\theta}_{k,n}^{(t)}$ denote the k-th component of $\boldsymbol{\theta}^*$ and $\hat{\boldsymbol{\theta}}_n^{(t)}$, respectively.

Theorem 3

Suppose that the Lasso penalty is imposed on $\boldsymbol{\theta}$. Under appropriate assumptions, the following results hold:

(i) $\|\hat{\boldsymbol{\theta}}_n^{(t)} - \boldsymbol{\theta}^*\| \xrightarrow{p} 0$ *for sufficiently large n and sufficiently large t and almost every observed dataset D_n.*

(ii) $P(\widehat{\gamma}_n^{(t)} = \gamma^*) \to 1$ *as $n \to \infty$ and $t \to \infty$.*

Similar to the sparse learning theory of linear models, Theorem 3 shows that if an θ-min condition is satisfied, then the true structure of the StoNet can be recovered.

StoNet Training-I: Corollary

With IRO, we have given a constructive proof for the consistency of sparse StoNets based on the sparse learning theory of linear models. Let

$$\widehat{\boldsymbol{\theta}}_n^* = \arg\max_{\boldsymbol{\theta}} \left\{ \frac{1}{n} \sum_{i=1}^{n} \log \pi(\boldsymbol{Y}^{(i)}, \boldsymbol{Y}_{\mathrm{mis}}^{(i)} | \boldsymbol{X}^{(i)}, \boldsymbol{\theta}, \boldsymbol{\sigma}^2) + \frac{1}{n} P_\lambda(\boldsymbol{\theta}) \right\}, \qquad (8)$$

$$\widehat{\boldsymbol{\theta}}_{\mathrm{DNN,n}}^* = \arg\max_{\boldsymbol{\theta}} \left\{ \frac{1}{n} \sum_{i=1}^{n} \log \pi(\boldsymbol{Y}^{(i)} | \boldsymbol{X}^{(i)}, \boldsymbol{\theta}) + \frac{1}{n} P_\lambda(\boldsymbol{\theta}) \right\}. \qquad (9)$$

By Theorem 1,

Corollary 4

If the Lasso penalty $P_\lambda(\boldsymbol{\theta})$ is employed, then the estimator (9) is also consistent in both parameter estimation and structure selection, i.e., $\|\widehat{\boldsymbol{\theta}}_{\mathrm{DNN,n}}^* - \boldsymbol{\theta}^*\| \xrightarrow{p} 0$ and $P(\widehat{\boldsymbol{\gamma}}_{\mathrm{DNN,n}} = \boldsymbol{\gamma}^*) \to 1$ as $n \to \infty$, where $\widehat{\boldsymbol{\gamma}}_{\mathrm{DNN,n}}$ is selected with the same threshold as given in Theorem 3.

StoNet Training-II

By Theorem 2, the StoNet can be trained by solving the equation

$$\mathbb{E}[H(\boldsymbol{Y}_{mis}, \boldsymbol{\theta})] = \int H(\boldsymbol{Y}_{mis}, \boldsymbol{\theta})\pi(\boldsymbol{Y}_{mis}|\boldsymbol{\theta}, \boldsymbol{X}, \boldsymbol{Y})d\boldsymbol{Y}_{mis} = 0, \quad (10)$$

where $H(\boldsymbol{Y}_{mis}, \boldsymbol{\theta}) = \nabla_{\boldsymbol{\theta}} \log \pi(\boldsymbol{Y}, \boldsymbol{Y}_{mis}|\boldsymbol{X}, \boldsymbol{\theta}) - \nabla P_{\lambda_n}(\boldsymbol{\theta})$.

Equation (10) further implies that StoNet can be trained by an adaptive stochastic gradient MCMC algorithm.

StoNet Training-II: adaptive stochastic gradient MCMC

(i) (*Sampling*) Simulate the latent variables $\boldsymbol{Y}_{mis}^{(k+1)} := (\boldsymbol{Y}_1^{(k+1)}, \boldsymbol{Y}_2^{(k+1)}, \ldots, \boldsymbol{Y}_h^{(k+1)})$ from $\pi(\boldsymbol{Y}_{mis}|\boldsymbol{\theta}^{(k)}, \boldsymbol{X}, \boldsymbol{Y})$ using a stochastic gradient MCMC algorithm, e.g., SGLD or SGHMC.

(ii) (*Parameter updating*) Update the parameters of the StoNet by one stochastic gradient descent (SGD) step:

$$\boldsymbol{\theta}^{(k+1)} = \boldsymbol{\theta}^{(k)} + \gamma_{k+1} H(\boldsymbol{Y}_{mis}^{(k+1)}, \boldsymbol{\theta}^{(k)}).$$

Convergence of Adaptive SGHMC

Theorem 5

Under some regularity conditions, if we set $\epsilon_k = C_\epsilon/(c_e + k^\alpha)$ *and* $\gamma_k = C_\gamma/(c_g + k^\alpha)$ *for some constants* $\alpha \in (0,1)$, $C_\epsilon > 0$, $C_\gamma > 0$, $c_e \geq 0$ *and* $c_g \geq 0$, *then there exists an iteration* k_0 *and a constant* $\lambda_0 > 0$ *such that for any* $k > k_0$,

$$\mathbb{E}(\|\boldsymbol{\theta}^{(k)} - \boldsymbol{\theta}^*\|^2) \leq \lambda_0 \gamma_k, \tag{11}$$

where $\boldsymbol{\theta}^*$ *denotes a solution to the equation (10).*

Convergence of SGHMC

Theorem 6

Under some regularity conditions, for any $k \in \mathbb{N}$, we have

$$\mathcal{W}_2(\mu_{\boldsymbol{D},T_k}, \pi_{\boldsymbol{D}}) \le C\sqrt{\mathcal{H}_\rho(\mu_0, \pi_{\boldsymbol{D}})}e^{-\mu_* T_k}$$
$$+ \sqrt{C_5 \log(T_k)}\left(\sqrt{\tilde{C}(k)} + \left(\frac{\tilde{C}(k)}{2}\right)^{1/4}\right) + \sqrt{C_6 T_k \sum_{j=1}^{k-1} \epsilon_{j+1}^2},$$

which can be made arbitrarily small by choosing a large enough value of T_k and small enough values of ϵ_1 and γ_1.

Prediction Uncertainty Quantification

Let $\boldsymbol{Y}_i^{(t)}$ denote the imputed latent variable at layer i, corresponding to the input vector $\boldsymbol{z}$. Let $\boldsymbol{\mu}_i^{(t)}$ and $\boldsymbol{\Sigma}_i^{(t)}$ denote, respectively, the mean and covariance matrix of $\boldsymbol{Y}_i^{(t)}$. By Eve's law, for any layer $i \in \{2, 3, \ldots, h+1\}$,

$$\boldsymbol{\Sigma}_i^{(t)} = \mathbb{E}(\mathrm{Var}(\boldsymbol{Y}_i^{(t)} | \boldsymbol{Y}_{i-1}^{(t)})) + \mathrm{Var}(\mathbb{E}(\boldsymbol{Y}_i^{(t)} | \boldsymbol{Y}_{i-1}^{(t)}))$$

where the respective variances can be calculated by the Lasso+OLS.

Prediction Uncertainty Quantification

Given $\widehat{\Sigma}_i^{(t)}$'s, the 95% prediction interval of $\mu_j(\boldsymbol{z}, \boldsymbol{\theta}^*)$, the j-th component of $\mu(\boldsymbol{z}, \boldsymbol{\theta}^*)$, can be constructed in the following procedure:

(i) For each StoNet estimate $\hat{\boldsymbol{\theta}}^{(t)} \in \mathcal{S}$, calculate the variance of the training error by $\hat{\varsigma}^{2(t)}_{h+1,j} = \frac{1}{n}\sum_{k=1}^{n}(\mu_j(\boldsymbol{x}^{(k)}, \hat{\boldsymbol{\theta}}^{(t)}) - y_j^{(k)})^2$.

(ii) For each StoNet estimate $\hat{\boldsymbol{\theta}}^{(t)} \in \mathcal{S}$, construct the prediction interval

$$\begin{aligned}\Big(\mu_j(\boldsymbol{z}, \hat{\boldsymbol{\theta}}^{(t)}) - 1.96\sqrt{\widehat{\Sigma}^{(t)}_{h+1,j} + \hat{\varsigma}^{2(t)}_{h+1,j}},\\ \mu_j(\boldsymbol{z}, \hat{\boldsymbol{\theta}}^{(t)}) + 1.96\sqrt{\widehat{\Sigma}^{(t)}_{h+1,j} + \hat{\varsigma}^{2(t)}_{h+1,j}}\Big),\end{aligned} \tag{12}$$

where $\widehat{\Sigma}^{(t)}_{h+1,j}$ denotes the (j,j)-th diagonal element of $\widehat{\Sigma}^{(t)}_{h+1}$.

(iii) Output the final 95% prediction interval of $\mu_j(\boldsymbol{z}, \boldsymbol{\theta}^*)$ by averaging m intervals obtained in step (ii).

Prediction Uncertainty Quantification: Example

Consider a neural network model:

$$y = \tanh(2\tanh(2x_1 - x_2)) + 2\tanh(2\tanh(x_3 - 2x_4) - \tanh(2x_5)) + 0x_6 + \ldots \tag{13}$$

where $\epsilon \sim N(0,1)$, $\boldsymbol{x} = (x_1, x_2, \ldots, x_{20})$, $x_i \sim N(0,1)$ for $i = 1, 2, \ldots, 20$, and x_i's are correlated with a mutual correlation coefficient of 0.5.

(a) 2-hidden-layer StoNet

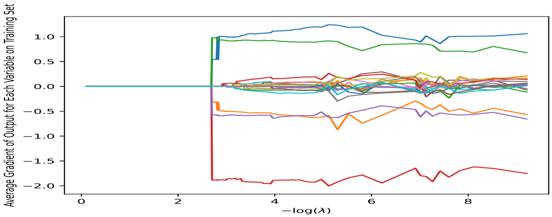
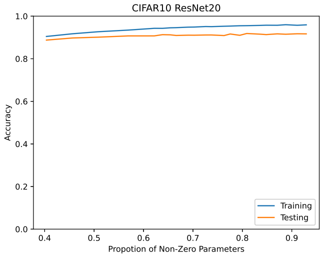


Figure 2: Variable selection paths by the StoNet.

Large-scale Network Compression



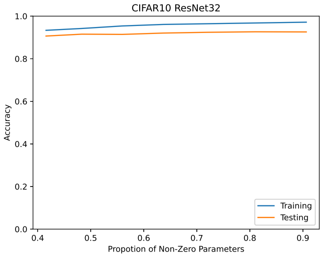


Figure 3: Training (blue) and test (orange) accuracy versus sparsity levels for sparse ResNet20 (left) and ResNet32 (right) on the CIFAR-10 data.

Structural Interpretability: Sufficient Dimensional Reduction

Suppose that we are able to learn a StoNet, which maps $\boldsymbol{X}$ to $\boldsymbol{Y}$ via some stochastic hidden layers and possesses a layer-wise Markovian structure such that

$$\pi(\boldsymbol{Y}, \boldsymbol{Y}_h, \boldsymbol{Y}_{h-1}, \ldots, \boldsymbol{Y}_1 | \boldsymbol{X}) = \pi(\boldsymbol{Y} | \boldsymbol{Y}_h) \pi(\boldsymbol{Y}_h | \boldsymbol{Y}_{h-1}) \cdots \pi(\boldsymbol{Y}_1 | \boldsymbol{X}), \tag{14}$$

where each conditional distribution is modeled by a linear or logistic regression (on transformed outputs of the previous layer).

The layer-wise Markovian structure implies $\boldsymbol{Y} \perp\!\!\!\perp \boldsymbol{X} | \boldsymbol{Y}_h$, and the simple regression structure of $\pi(\boldsymbol{Y} | \boldsymbol{Y}_h)$ successfully gets around the identifiability issue of the latent variable $\boldsymbol{Z} := \boldsymbol{Y}_h$ that has been suffered by some other deep learning-based methods.

Sufficient Dimension Reduction (SDR)

Let $\boldsymbol{Y} \in \mathbb{R}^d$ be response variables, and let $\boldsymbol{X} = (X_1, \ldots, X_p)^T \in \mathbb{R}^p$ be explanatory variables. The goal of SDR is to find a lower-dimensional representation $\boldsymbol{Z} \in \mathbb{R}^q$, as a function of $\boldsymbol{X}$ for some $q < p$, such that

$$P(\boldsymbol{Y}|\boldsymbol{X}) = P(\boldsymbol{Y}|\boldsymbol{Z}), \quad \text{or equivalently} \quad \boldsymbol{Y} \perp\!\!\!\perp \boldsymbol{X}|\boldsymbol{Z}, \tag{15}$$

where $\perp\!\!\!\perp$ denotes conditional independence.

SDR: Linear Setting

Under the linear setting, SDR is to find a few linear combinations of $\boldsymbol{X}$ that are sufficient to describe the conditional distribution of $\boldsymbol{Y}$ given $\boldsymbol{X}$, i.e., finding a projection matrix $\boldsymbol{B} \in \mathbb{R}^{p \times q}$ such that

$$\boldsymbol{Y} \perp\!\!\!\perp \boldsymbol{X} \big| \boldsymbol{B}^T \boldsymbol{X}. \tag{16}$$

Inverse regression Methods:

- sliced inverse regression (SIR)
- sliced average variance estimation (SAVE)
- parametric inverse regression
- contour regression
- directional regression

These methods require strict assumptions on the joint distribution of $(\boldsymbol{X}, \boldsymbol{Y})$ or the conditional distribution of $\boldsymbol{X}|\boldsymbol{Y}$, which limit their use in practice.

SDR: Nonlinear Setting

Under the nonlinear setting, SDR is to find a nonlinear function $f(\cdot)$ such that

$$\boldsymbol{Y} \perp\!\!\!\perp \boldsymbol{X} \Big| f(\boldsymbol{X}). \tag{17}$$

A common strategy to achieve nonlinear SDR is to apply the kernel trick to the existing linear SDR methods, where the variable $\boldsymbol{X}$ is first mapped to a high-dimensional feature space via kernels and then inverse or forward regression methods are performed.

Nonlinear SDR methods: Kernel trick

- kernel sliced inverse regression (KSIR)
- kernel dimension reduction (KDR)
- manifold kernel dimension reduction (MKDR)
- generalized sliced inverse regression (GSIR)
- generalized sliced average variance estimator (GSAVE)
- least square mutual information estimation (LSMIE)

A drawback shared by these methods is that they require to compute the eigenvectors or inverse of an $n \times n$ matrix. Therefore, these methods lack the scalability necessary for big data problems.

SDR: Deep Learning

In Kapla et al. (2021), the authors assume that the response variable $\boldsymbol{Y}$ on the predictors $\boldsymbol{X}$ is fully captured by a regression

$$\boldsymbol{Y} = g(\boldsymbol{B}^T\boldsymbol{X}) + \epsilon, \tag{18}$$

for an unknown function $g(\cdot)$ and a low rank parameter matrix $\boldsymbol{B}$, where $g(\cdot)$ is approximated by a DNN.

This method might be invalid unless the estimate of $g(\cdot)$ is consistent, but the consistency does not generally hold for the fully connected neural networks trained without constraints.

SDR: Deep Learning

Banijamali et al. (2018) learn the latent variable $\boldsymbol{Z}$ by optimizing three DNNs to approximate the distributions $p(\boldsymbol{Z}|\boldsymbol{X})$, $p(\boldsymbol{X}|\boldsymbol{Z})$ and $p(\boldsymbol{Y}|\boldsymbol{Z})$, respectively, under the framework of variational autoencoder. Again, $\boldsymbol{Z}$ suffers from the identifiability issue due to the universal approximation ability of the DNN.

SDR: A Validating Example

- The dataset consists of 100 samples. Each sample is generated from the model $\boldsymbol{Y} = \cos(\boldsymbol{X}^T \boldsymbol{b}) + \epsilon$, where $\boldsymbol{X} \in \mathbb{R}^{20}$ follows a multivariate Gaussian distribution, and ϵ follows a generalized Gaussian distribution $GN(0, \sqrt{1/2}, 0.5)$.
- Kapla et al. (2021) projected the data to one-dimensional space by working with a refinement network of 20-1-512-1. Let $\boldsymbol{Z}_1$ and $\boldsymbol{Z}_2$ denote two SDR vectors produced by the method in two independent runs with different initializations of network weights. An independence test $\boldsymbol{Z}_1 \perp\!\!\!\perp \boldsymbol{Z}_2$ returns a p-value of 0.4068, which suggests that the two SDR vectors are independent.
- StoNet was applied to the same dataset with structure 20-10-1-1. The test $\boldsymbol{Z}_1 \perp\!\!\!\perp \boldsymbol{Z}_2$ returns a p-value of 0.012, which suggests that the two SDR vectors are not independent.

SDR: Classification

Table 1: Mean misclassification rates on test sets over 20 independent trials for some binary classification examples.

Datasets	q	StoNet	LSMIE	GSIR	GSAVE	KDR	SIR	SAVE	PCA
thyroid	1	**0.0687(.0068)**	0.2860(.0109)	**0.0640(.0063)**	0.0913(.0102)	0.2847(.0110)	0.1373(.0117)	0.3000(.0110)	0.3013(.0110)
	2	**0.0693(.0068)**	0.1733(.0113)	**0.0667(.0071)**	0.0947(.0103)	0.2713 (.0128)	0.1373(.0130)	0.3000(.0118)	0.1467(.0143)
breastcancer	2	**0.2578(.0074)**	0.2812(.0110)	0.2772(.0091)	0.2740(.0069)	**0.2714(.0102)**	0.2818(.0125)	0.2870(.0075)	0.2857(.0129)
	4	**0.2682(.0113)**	**0.2760(.0118)**	**0.2740(.0100)**	**0.2805(.0076)**	**0.2740(.0105)**	**0.2831(.0110)**	0.2922(.0147)	**0.2766(.0097)**
flaresolar	2	**0.3236(.0040)**	0.3770(.0177)	**0.3305(.0034)**	**0.3308(.0033)**	0.4161(.0138)	**0.3312(.0052)**	0.4860(.0127)	**0.3313(.0046)**
	4	**0.3239 (.0043)**	0.3346(.0043)	0.3400(.0040)	0.3336(.0038)	0.3673(.0108)	**0.3328(.0049)**	0.4302(.0133)	0.3612(.0036)
heart	3	**0.1625(.0076)**	**0.1725(.0073)**	**0.1645(.0069)**	**0.1731(.0060)**	0.1870(.0064)	**0.1720(.0088)**	0.1910(.0053)	0.1920 (.0123)
	6	**0.1625(.0062)**	**0.1695(.0073)**	**0.1650(.0068)**	0.1754(.0063)	**0.1715(.0075)**	0.1770(.0100)	**0.1720(.0073)**	0.1830(.0102)
german	5	**0.2368(.0050)**	0.25(.0052)	**0.2325(.0058)**	**0.2323(.0050)**	**0.2430(.0050)**	**0.2367(.0068)**	0.2703(.0072)	0.2777(.0070)
	10	**0.2356(.0047)**	0.2443(.0056)	**0.2327(.0046)**	**0.2312(.0047)**	**0.2347(.0075)**	**0.2360(.0068)**	0.2447(.0062)	**0.2350(.0051)**
waveform	5	**0.1091(.0010)**	0.1336(.0013)	0.1140(.0015)	**0.1095(.0016)**	0.1269(.0031)	0.1453(.0018)	0.1427(.0020)	0.1486(.0013)
	10	**0.1079(.0012)**	0.1369(.0018)	0.1117(.0009)	**0.1070(.0013)**	0.1254(.0030)	0.1444(.0017)	0.1417(.0020)	0.1430(.0020)

SDR: MNIST Example

Table 2: Misclassification rates on the test set for the MNIST example, where the best misclassification rates achieved by different methods at each dimension q are specified by bold face. The CPU time (in seconds) was recorded on a computer of 2.2 GHz.

q	StoNet	LSMIE	GSIR	GSAVE	Autoencoder	PCA
392	**0.0456**	-	0.0596	0.0535	0.1965	0.1002
196	**0.0484**	-	0.0686	0.0611	0.2268	0.0782
98	**0.0503**	-	0.0756	0.0696	0.2733	0.0843
49	**0.0520**	-	0.0816	0.0764	0.3112	0.0889
10	**0.0825**	-	0.0872	0.0901	0.4036	0.1644
Average Time(s)	96.18	$> 24 hours$	16005.59	22154.11	1809.18	5.11

SDR: Regression

Table 3: Mean MSE and Pearson correlation on the test sets (and their standard deviations in the parentheses) over 10 trails for the Relative location of CT slices on axial axis dataset.

	StoNet		Autoencoder		PCA	
q	MSE	Corr	MSE	Corr	MSE	Corr
192	**0.0002(.0000)**	**0.9986(.0001)**	0.0079(.0015)	0.9267(.0147)	0.0027(.0000)	0.9755(.0001)
96	**0.0002(.0000**	**0.9985(.0001)**	0.0106(.0024)	0.9002(.0237)	0.0026(.0000)	0.9756(.0001)
48	**0.0002(.0000)**	**0.9982(.0001)**	0.0143(.0035)	0.8562(.0399)	0.0034(.0000)	0.9682(.0001)
24	**0.0002(.0000)**	**0.9980(.0001)**	0.0185(.0033)	0.8168(.0364)	0.0042(.0000)	0.9612(.0001)
12	**0.0002(.0000)**	**0.9980(.0001)**	0.0233(.0027)	0.7579(.0338)	0.0053(.0000)	0.9499(.0001)
6	**0.0002(.0000)**	**0.9980(.0001)**	0.0304(.0024)	0.6668(.0300)	0.0102(.0001)	0.9023(.0002)
3	**0.0004(.0000)**	**0.9965(.0002)**	0.0384(.0030)	0.5529(.0538)	0.0209(.0001)	0.7858(.0004)

K-StoNet

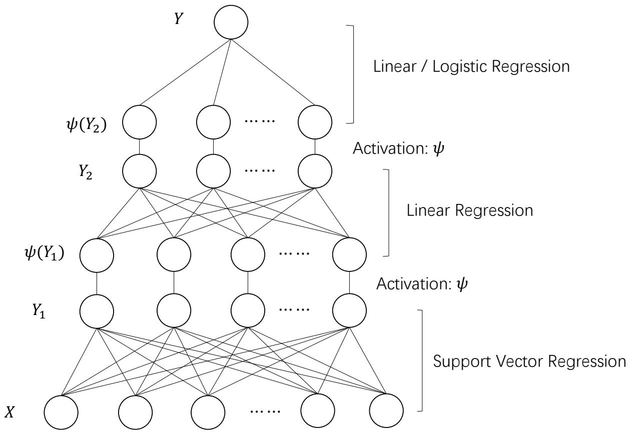


Figure 4: Structure of K-StoNet

Attractive features of K-StoNet

K-StoNet overcomes the issues on local trap and uncertainty quantification suffered by the DNN in a coherent way. The new model incorporates support vector regression (SVR) as the first hidden layer and reformulates the neural network as a latent variable model.

- The SVR layer maps the input vector from its original space into an infinite dimensional feature space, ensuring all local minima on the loss surface are globally optimal.
- The latent variable modeling resolves the parameter optimization and statistical inference issues associated with the neural network: it breaks the high-dimensional nonconvex optimization problem into a series of low-dimensional convex optimization problems, enabling the prediction uncertainty of the neural network easily assessed.

Attractive features of K-StoNet (continuation)

- The new model can be easily trained using the imputation-regularized optimization (IRO) algorithm (Liang et al., 2018), which usually converges in tens of epochs.
- The introduction of the SVR layer with a universal kernel enables K-StoNet to work with a smaller network, while ensuring the universal approximation power to hold.

In summary, K-StoNet provides a new neural network model *which possesses a theoretical guarantee to asymptotically converge to the global optimum and enables the prediction uncertainty easily assessed*.

A full row rank example

The dataset was generated from a two-hidden layer neural network with structure 1000-5-5-1:

$$\boldsymbol{y} = \boldsymbol{w}_3 \tanh(\boldsymbol{w}_2 \tanh(\boldsymbol{w}_1 \boldsymbol{x})) + \epsilon, \tag{19}$$

where the input variables $x_1, \ldots, x_{1000}$ have a mutual correlation coefficient of 0.5, $\boldsymbol{w}_1 \in \mathbb{R}^{5\times 1001}$, $\boldsymbol{w}_2 \in \mathbb{R}^{5\times 6}$ and $\boldsymbol{w}_3 \in \mathbb{R}^{1\times 6}$ represent the weights at different layers of the neural network, $\tanh(\cdot)$ is the hyperbolic tangent function, and the random error $\epsilon \sim N(0,1)$. Each element of $\boldsymbol{w}_i$'s was randomly sampled from the set $\{-2,-1,1,2\}$. The full dataset consisted of 1000 training samples and 1000 test samples.
Since $\boldsymbol{w}_i$'s for $i = 1,2,3$ are all full-row rank matrices, there will be no local traps for SGD.

A full row rank example

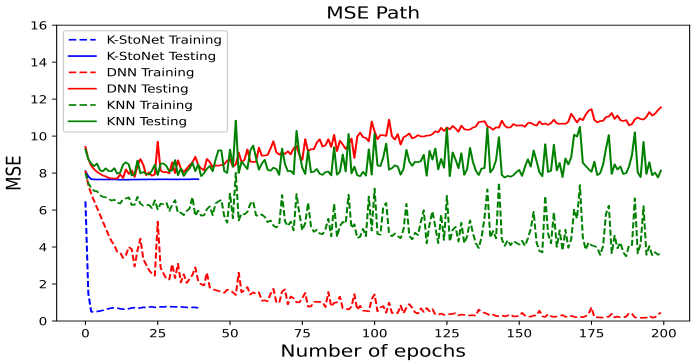


Figure 5: MSE paths produced by K-StoNet, KNN, and DNN for one simulated dataset.

A full row rank example

- For K-StoNet, we tried a model with one hidden layer and 5 hidden units. The model was trained by IRO for 40 epochs.
- K-StoNet converges to the global optimum very fast, usually within a few epochs; while the DNN needs over 100 epochs.
- K-StoNet is less bothered by over-fitting, whose prediction performance is stable after convergence has been reached. However, the DNN tends to be over fitted, whose prediction becomes worse and worse as training goes on.

A measurement error example

This example mimics the typical scenario that DNN works in. We generated 500 training samples and 500 test samples from a nonlinear regression model:

$$Y = \frac{5X_2}{1+X_1^2} + 5\sin(X_3X_4) + 2X_5 + \epsilon,$$

where $\epsilon \sim N(0,1)$, and X_i's have a mutual correlation coefficient of 0.5. Then each explanatory variable was perturbed by adding a random measurement error independently simulated from $N(0, 0.5)$, making the true input-output mapping unknown and complicated.

This is a typical large-n-small-p problem that deep learning works on.

A measurement error example

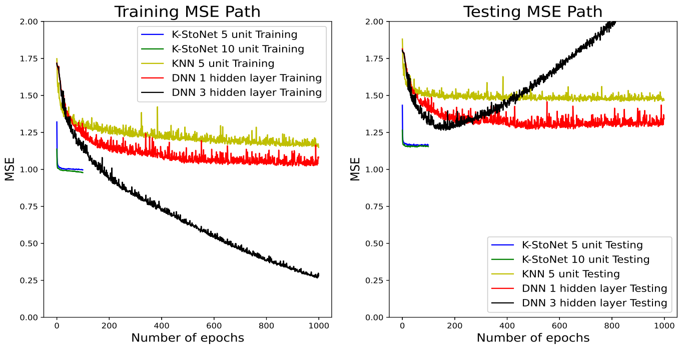


Figure 6: MSE paths produced by K-StoNet, KNN and DNN on a measurement error example.

A measurement error example

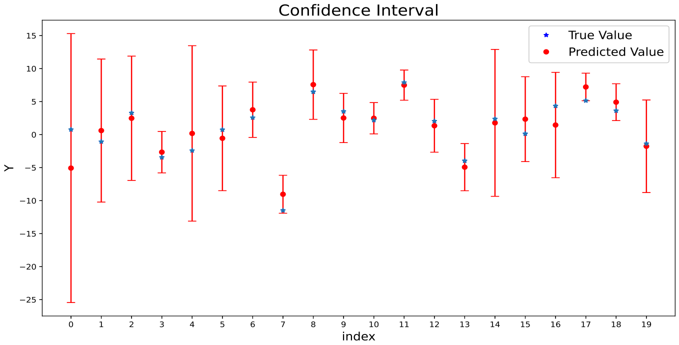


Figure 7: 95% confidence intervals produced by K-StoNet for 20 test points. The overall covrage rate for 500 test points is 93.812% with a standard deviation of 0.781%.

MNIST

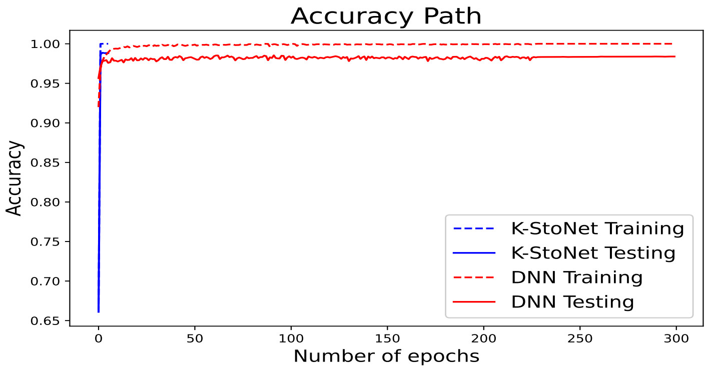


Figure 8: Training and test accuracy versus epochs produced by K-StoNet and DNN (LeNet-300-100) for the MNIST data, where K-StoNet achieved a prediction accuracy of 98.87%, and LeNet-300-100 achieved a prediction accuracy of 98.38% (without data augmentation being used in the experiments)

Causal inference: challenges

Challenges with causal inference for high-dimensional complex data:

(i) High dimensionality of covariates, which is common in social media data, environmental and healthcare data, and genomic data

(ii) Unknown functional forms for the outcome and treatment models

(iii) Missing values in covariates

Sparse StoNet addresses all the three challenges in a coherent way!

Causal inference: flexible structure of StoNet

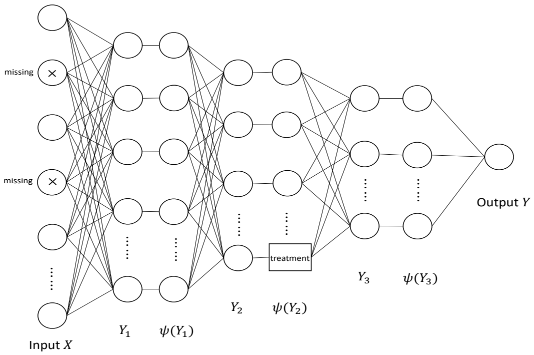


Figure 9: Causal-StoNet Structure: the treatment is included as a visible unit (rectangle) in a middle layer, and $\boldsymbol{Y}_2$ *denotes the latent variable of that layer but with the unit directly feeding to the treatment rectangle excluded;* 'x' represents possible missing values.

Causal inference: Estimation

Under appropriate assumptions, we can show the following results for the Causal-StoNet:

(a) (Propensity score function) With probability greater than $1 - \exp\{cn\epsilon_n^2\}$ for some constant c,

$$\mathbb{E}_{\boldsymbol{x}}[(\hat{p}(\boldsymbol{x}; \hat{\boldsymbol{\theta}}_n) - p^*(\boldsymbol{x}))^2] = o(n^{-1/2}).$$

(b) (Outcome function) With probability greater than $1 - \exp\{cn\epsilon_n^2\}$ for some constant c,

$$\mathbb{E}_{\boldsymbol{x}}(|\hat{\mu}(\boldsymbol{x}, A; \hat{\boldsymbol{\theta}}_n) - \mu^*(\boldsymbol{x})|^2) = o(n^{-1/2}).$$

(c) (Structure selection) $P(\hat{\gamma}(\hat{\boldsymbol{\theta}}_n) = \boldsymbol{\gamma}_*) \xrightarrow{p} 1$, where $\boldsymbol{\gamma}_*$ specifies the structure of the underlying true sparse deep neural network.

Causal inference: Estimation

(d) $V_\tau^{-1/2}\sqrt{n}(\hat{\tau}_n - \tau^*) \xrightarrow{d} \mathcal{N}(0,1)$ as $n \to \infty$, where τ^* denotes the true value of the average treatment effect, and $\hat{\tau}_n$ denotes the double robust estimator constructed with $\hat{\mu}(\boldsymbol{x}, A, \hat{\boldsymbol{\theta}}_n)$ and $\hat{p}(\boldsymbol{x}, \hat{\boldsymbol{\theta}}_n)$.

(e) (Uniformly valid inference) For $c_\alpha = \Phi^{-1}(1-\alpha/2)$,

$$\sup_{P_n \in \mathcal{P}_n} \left| \mathbb{P}_{P_n}\left[\tau^* \in \left\{\hat{\tau}_n \pm c_\alpha \sqrt{\hat{V}_\tau / n}\right\}\right] - (1-\alpha)\right| \to 0, \quad \text{as } n \to \infty,$$

where $\hat{V}_\tau$ denotes a consistent estimator of V_τ, and $\mathcal{P}_n$ denotes the class of data generating processes.

Causal inference: Example

The Causal-StoNet is compared with the baseline methods on 10 simulated dataset and 10 synthetic datasets by Atlantic Causal Inference Conference (ACIC) 2019 Data Challenge. Each of the datasets contains 200 covariates, and the true ATE is known.

Table 4: MAE of the ATE estimates by different methods, where the number in the parentheses is the standard deviation of the MAE

	Simulation		ACIC	
	In-sample	Out-of-sample	In-sample	Out-of-sample
Causal-StoNet	**0.0130(0.0027)**	**0.0504(0.0101)**	**0.0501(0.0118)**	**0.0542(0.0132)**
DSE	0.0876(0.0113)	0.0964(0.0169)	0.0776(0.0193)	0.1632(0.0251)
ARBE	0.0999(0.0095)	0.1012(0.0184)	0.0729(0.0166)	0.1335(0.0179)
TMLE(Lasso)	0.1111(0.0102)	0.1343(0.0237)	0.0869(0.0164)	0.0867(0.0165)
TMLE(ensemble)	0.0506(0.0067)	0.0813(0.0177)	0.1140(0.0394)	0.1316(0.0429)
X-Learner	0.0535(0.0168)	0.0928(0.0138)	0.1634(0.0434)	0.1805(0.0489)
Dragonnet	0.0340(0.0124)	0.0612(0.0147)	0.0800(0.0170)	0.0857(0.0269)

Post-StoNet for Deep Learning

In real applications, use of large-scale deep neural networks, such as residual networks and transformer, has been a common practice. To make statistical inference for these large-scale models, we propose a post-StoNet modeling procedure, without significantly changing the current practice of large-scale models.

The proposed procedure is as follows:

(i) Transform the explanatory variables by calculating the output of the last-hidden-layer of a well-trained DNN.

(ii) Learn a simple sparse StoNet(e.g. with one hidden layer only) using the transformed data and their response on the validation data.

Post-StoNet for Deep Learning: Classification

Table 5: Calibration results for CIFAR10 data, where standard deviations of the respective measures are given in parentheses.

Network	Size	Method	ACC	NLL	ECE
DenseNet40	176K	No Post Calibration	**92.88%(0.19%)**	0.3076(0.0094)	0.0434(0.0019)
		Matrix Scaling	92.73%(0.20%)	0.2226(0.0052)	0.0132(0.0026)
		Temp. Scaling	**92.88%**(0.19%)	0.2194(0.0055)	0.0117(0.0016)
		Post-StoNet	92.79%(0.17%)	**0.2175**(0.0034)	**0.0047**(0.0010)
ResNet110	1.7M	No Post Calibration	**93.23%(0.37%)**	0.3113(0.0220)	0.0444(0.0030)
		Matrix Scaling	92.96%(0.32%)	0.2127(0.0097)	0.0145(0.0023)
		Temp. Scaling	**93.23%**(0.37%)	0.2077(0.0092)	0.0122(0.0016)
		Post-StoNet	93.22%(0.31%)	**0.2045**(0.0086)	**0.0070**(0.0014)
WideResNet-28-10	36M	No Post Calibration	**95.76%(0.13%)**	0.1710(0.0077)	0.0258(0.0014)
		Matrix Scaling	95.71%(0.14%)	0.1475(0.0042)	0.0104(0.0014)
		Temp. Scaling	**95.76%**(0.13%)	0.1489(0.0055)	0.0120(0.0017)
		Post-StoNet	95.63%(0.08%)	**0.1448**(0.0031)	**0.0089**(0.0007)

Modularized Deep Learning

The structure of the StoNet is very flexible, for which one can replace each regression by a more sophisticated model such that large-scale models can be easily learned for a complex system. The modularized DNN is naturally used to model the following complex system:

$$\begin{aligned} \boldsymbol{Y}_1 &= \mu_1(X) + \boldsymbol{e}_1 \\ \boldsymbol{Y} &= \mu_2(\boldsymbol{Y}_1) + \boldsymbol{e}_2 \\ \boldsymbol{Z} &= \boldsymbol{Y}_1 + \boldsymbol{e}_3 \end{aligned} \tag{20}$$

where $\boldsymbol{X}$, $\boldsymbol{Z}$, and $\boldsymbol{Y}$ represent the input, intermediate and output variables, respectively; $\mu_1(\cdot)$ and $\mu_2(\cdot)$ are complex functions represented by neural networks.

Drug effect prediction with CCLE Data

We used the mutation data as the input $\boldsymbol{X}$, the gene expression data as the intermediate observations $\boldsymbol{Z}$, and the drug effect data as the output Y. For the drug PLX4720, the dimensions of $\boldsymbol{X}$, $\boldsymbol{Y}$ and $\boldsymbol{Z}$ are 1638, 96 and 1, respectively. We modeled $\mu_1(\cdot)$ and $\mu_2(\cdot)$ by shallow neural networks.

Table 6: Mean squared prediction error (MSPE) in 5-fold cross validation for PLX4720, where DNN_1 is with expression data only, DNN_2 is with mutation data only, and DNN_3 is with expression+mutation data.

Model	Split 1	Split 2	Split 3	Split 4	Split 5	Average
StoNet	0.8043	0.7655	0.7668	0.6533	0.8768	**0.7734 (0.0362)**
DNN_1	0.9061	0.8602	0.8584	0.8860	0.8579	0.8737 (0.0097)
DNN_2	1.1732	1.1012	1.4205	1.3769	1.3951	1.2934 (0.0651)
DNN_3	1.0572	1.2708	1.5788	1.4851	1.4853	1.3754 (0.0943)

CCLE Data Example: Training

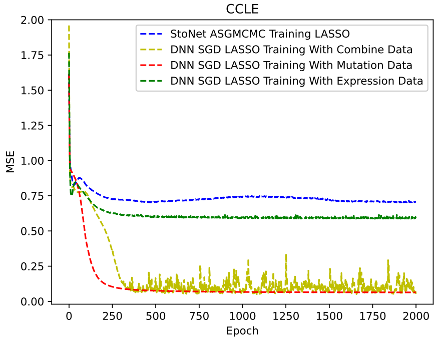


Figure 10: Evolutionary paths of the mean squared fitting errors produced by different methods in one fold of the cross-validation experiment.

CCLE Data Example: Testing

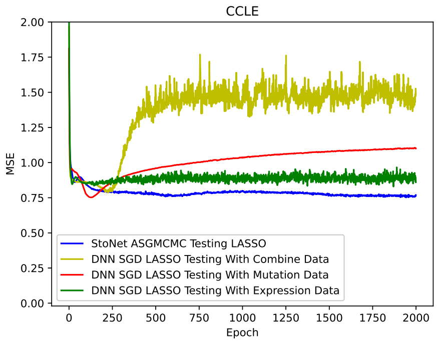


Figure 11: Evolutionary paths of the mean squared prediction errors produced by different methods in one fold of the cross-validation experiment.

Discussion

The StoNet provides a new formulation of the deep neural network as a composition of many regressions:

- The StoNet bridges from linear models to deep learning, enabling the sparse learning theory to be transferred from linear models to deep neural networks.
- The StoNet reformulates the DNN as a latent variable model. It breaks the high-dimensional nonconvex neural network training problem into a series of lower-dimensional convex optimization problems, and enables the prediction uncertainty easily assessed.
- The compositional regression structure enables the StoNet to model many complex systems in a natural way, easing statistical analysis of the data collected therefrom.
- The StoNet has many applications in statistics: sufficient dimension reduction, causal inference, missing data imputation, data integration, measurement error, etc.

Reference

- Liang, F., Jia, B., Xue, J., Li, Q. and Luo, Y. (2018). An imputation-regularized optimization approach for high-dimensional missing data problems and beyond. *Journal of the Royal Statistical Society, Series B* **80**(5), 899-926.
- Sun, Y. and Liang, F. (2022). A kernel-expanded stochastic neural network. *Journal of the Royal Statistical Society, Series B*, **84**, 547-578.
- Sun, Y., Song, Q., and Liang, F. (2022). Consistent Sparse Deep Learning: Theory and Computation. *Journal of the American Statistical Association*, 117 (540), 1981-1995.
- Liang, S., Sun, Y., and Liang, F. (2022). Nonlinear sufficient dimension reduction with a stochastic neural network. *NeurIPS 2022* (see also arXiv:2210.04349).
- Sun, Y. and Liang, F. (2023). Statistical Inference for Deep Learning via Stochastic Modeling. Working Manuscript.
- Fang, Y. and Liang, F. (2023). Causal StoNet: Causal Inference for High-Dimensional Complex Data. Working Manuscript.

Acknowledgments

- The talk is based on the joint works with my graduate students Yan Sun, Siqi Liang, and Yaxin Fang.
- NSF grants
- NIH grants