On Statistical Inference with Sparse Deep Learning

October 14, 2024

Outline

- Introduction on sparse deep learning
- Theory on sparse deep learning:
  - posterior consistency
  - structure selection consisteny
  - asymptotic normality of prediction
- Training algorithms
- Numerical experiments
- Conclusion

Deep Neural Networks

Deep neural networks (DNNs) have achieved great successes in pattern recognition and speech processing during the past decade, however, they are often over-parameterized, which leads to

- formidable computational challenges;
- mis-calibration.

For some problems, the DNNs might consist of hundreds of layers and billions of parameters. For some networks, only 5% of parameters are enough to achieve acceptable models.

Complexity Reduction

Three related questions:

- Is a sparse DNN able to approximate the target mapping with a desired accuracy?
- How to train and determine the structure of a sparse DNN?
- How to quantify uncertainty of the prediction with a sparse DNN?

We answer the three questions in a coherent way: We propose a frequentist-like method for learning sparse DNNs, justify its consistency under the Bayesian framework, and establish asymptotic normality of prediction.

We expect our theory can tame powerful neural networks into the framework of statistical modeling: By selecting an appropriate network size according to the training sample size, the proposed method can generally improve the generalization and calibration of the DNN while controlling the training error to a reasonable level.

Approximation Theory: Notation

- Let $D_n = (\boldsymbol{x}^{(i)}, y^{(i)})_{i=1,\ldots,n}$ denote a dataset of n $i.i.d$ observations, where $\boldsymbol{x}^{(i)} \in R^{p_n}$, $y^{(i)} \in R$, and p_n is assumed to grow with the sample size n.
- We study the approximation theory of sparse DNNs under the probabilistic framework of generalized linear models:

$$f(y|\mu^*(\boldsymbol{x})) = \exp\{A(\mu^*(\boldsymbol{x}))y + B(\mu^*(\boldsymbol{x})) + C(y)\}.$$

- We approximate $\mu^*(\boldsymbol{x})$ using a DNN, which has $H_n - 1$ hidden layers and L_h hidden units in the h-th layer, where $L_{H_n} = 1$ for the output layer and $L_0 = p_n$ for the input layer.

Approximation Theory: Notation

- The DNN forms the nonlinear mapping

$$\mu(\boldsymbol{w}, \boldsymbol{b}, \boldsymbol{x}) = \boldsymbol{w}^{H_n} \psi^{H_n - 1} \left[\cdots \psi^1 \left[\boldsymbol{w}^1 \boldsymbol{x} + \boldsymbol{b}^1 \right] \cdots \right] + \boldsymbol{b}^{H_n},$$

  where $\boldsymbol{\beta} = \{w_{ij}^h, b_k^h : h \in \{1, 2, ..., H_n\}, i, k \in \{1, ..., L_h\}, j \in \{1, ..., L_{h-1}\}\}$ denotes the collection of weights and biases, which consists of $K_n = \sum_{h=1}^{H_n} \left(L^{h-1} \times L^h + L^h \right)$ elements in total.
- Let $\boldsymbol{\gamma}^{\boldsymbol{w}^h}$ and $\boldsymbol{\gamma}^{\boldsymbol{b}^h}$ denote the matrix and vector of the indicator variables associated with $\boldsymbol{w}^h$ and $\boldsymbol{b}^h$, respectively. Let $\boldsymbol{\gamma} = \{\gamma_{ij}^{\boldsymbol{w}^h}, \gamma_k^{\boldsymbol{b}^h}\}$ and $\boldsymbol{\beta}_{\boldsymbol{\gamma}} = \{w_{ij}^h, b_k^h\}$, which specify, respectively, the structure and associated parameters for a sparse DNN.

Approximation Theory: Notation

- We define the *true DNN* as

$$(\boldsymbol{\beta}^*, \boldsymbol{\gamma}^*) = \underset{(\boldsymbol{\beta},\boldsymbol{\gamma})\in\mathcal{G}_n, \|\mu(\boldsymbol{\beta},\boldsymbol{\gamma},\boldsymbol{x})-\mu^*(\boldsymbol{x})\|_{L^2(\Omega)}\leq\varpi_n}{\arg\min} |\boldsymbol{\gamma}|, \qquad (1)$$

  where $\mathcal{G}_n := \mathcal{G}(C_0, C_1, \varepsilon, p_n, H_n, L_1, L_2, \ldots, L_{H_n})$ denotes the space of valid sparse networks satisfying condition A.2 (given below) for the given values of H_n, p_n, and L_h's, and ϖ_n is some sequence converging to 0 as $n \to \infty$.

Approximation Theory: Assumptions

A.1 The input $\boldsymbol{x}$ is bounded by 1 entry-wisely, i.e. $\boldsymbol{x} \in \Omega = [-1, 1]^{p_n}$, and the density of $\boldsymbol{x}$ is bounded in its support Ω uniformly with respect to n.

A.2 The true sparse DNN model satisfies the following conditions:

A.2.1 The network structure satisfies: $r_n H_n \log n + r_n \log \overline{L} + s_n \log p_n \leq C_0 n^{1-\varepsilon}$, where $0 < \varepsilon < 1$ is a small constant, $r_n = |\gamma^*|$ denotes the connectivity of γ^*, $\overline{L} = \max_{1 \leq j \leq H_n - 1} L_j$ denotes the maximum hidden layer width, s_n denotes the input dimension of γ^*.

A.2.2 The network weights are polynomially bounded: $\|\beta^*\|_\infty \leq E_n$, where $E_n = n^{C_1}$ for some constant $C_1 > 0$.

A.3 The activation function ψ is Lipschitz continuous with a Lipschitz constant of 1.

Approximation Theory: Remarks

Assumption A.2 specifies the class of DNN models we are considering. They are sparse, while still being able to approximate many types of functions arbitrarily well as the training sample size becomes large, i.e., $\lim_{n\to\infty} \varpi_n = 0$. Here are some examples:

- (Affine functions) For the functions that can be represented by an affine system, Bolcskei et al. (2019) proved that if the connection weights are bounded in absolute value by some polynomial $g(r_n)$, then the approximation error $\varpi_n = O(r_n^{-\alpha^*})$ for some constant α^*.

  For affine functions, we can verify that Assumption A.2 is satisfied with $r_n = O(n^{(1-\epsilon)/2})$, $\|\boldsymbol{\beta}_*\|_\infty \prec n^C$ for some constant C, and the approximation error $\varpi_n = O(n^{-\varsigma})$ for some constant $\varsigma > 0$.

Approximation Theory: Remarks

- (Piecewise smooth functions) Petersen and Voigtlaender(2018) showed that for a wide class of piecewise smooth functions with a fixed input dimension, a fixed depth ReLU network can achieve an ϖ_n-approximation with $\log(r_n) = O(-\log \varpi_n)$ and $\log E_n = O(-\log \varpi_n)$. This result satisfies condition A.2 by setting $\varpi_n = n^{-\varsigma}$ for some constant $\varsigma > 0$.
- (Bounded α-Hölder smooth function) Schmidt-Hieber (2017) and Polson and Rockova (2018) proved that any bounded α-Hölder smooth function $\mu_*(x)$ can be approximated by a sparse ReLU DNN with the network approximation error $\varpi_n = O(\log(n)^{\alpha/p_n} n^{-\alpha/(2\alpha+p_n)})$ for some $H_n \asymp \log n \log p_n$, $L_j \asymp p_n n^{p_n/(2\alpha+p_n)} / \log n$, $r_n = O(p_n^2 \alpha^{2p_n} n^{p_n/(2\alpha+p_n)} \log p_n)$, and $E_n = C$ for some fixed constant $C > 0$. This result satisfies condition A.2 as long as $p_n^2 \ll \log n$.

Approximation Theory: Remarks

The fundamental difference between the existing neural network approximation theory and ours:

- In the existing neural network approximation theory, no data is involved and a small network can potentially achieve an arbitrarily small approximation error by allowing the connection weights to take values in an unbounded space, see e.g. Theorem 2.2 in Bolcskei et al. (2019).
- In our theory, the network approximation error, the network size, and the bound of connection weights are all linked to the training sample size. A small network approximation error is required only when the training sample size is large; otherwise, over-fitting might be a concern from the point of view of statistical modeling.

Approximation Theory: Mixture Gaussian prior

We consider a mixture Gaussian prior

$$\gamma_{ij}^{\boldsymbol{w}^h} \sim Bernoulli(\lambda_n), \quad \gamma_k^{\boldsymbol{b}^h} \sim Bernoulli(\lambda_n), \tag{2}$$

$$\begin{aligned} w_{ij}^h|\gamma_{ij}^{\boldsymbol{w}^h} &\sim \gamma_{ij}^{\boldsymbol{w}^h} N(0, \sigma_{1,n}^2) + (1 - \gamma_{ij}^{\boldsymbol{w}^h}) N(0, \sigma_{0,n}^2), \\ b_k^h|\gamma_k^{\boldsymbol{b}^h} &\sim \gamma_k^{\boldsymbol{b}^h} N(0, \sigma_{1,n}^2) + (1 - \gamma_k^{\boldsymbol{b}^h}) N(0, \sigma_{0,n}^2), \end{aligned} \tag{3}$$

where $h \in \{1, 2, ..., H_N\}, i \in \{1, ..., L_{h-1}\}, j, k \in \{1, ..., L_h\}$, and $\sigma_{0,n}^2 < \sigma_{1,n}^2$ are prespecified constants. Marginally, we have

$$\begin{aligned} w_{ij}^h &\sim \lambda_n N(0, \sigma_{1,n}^2) + (1 - \lambda_n) N(0, \sigma_{0,n}^2), \\ b_k^h &\sim \lambda_n N(0, \sigma_{1,n}^2) + (1 - \lambda_n) N(0, \sigma_{0,n}^2). \end{aligned} \tag{4}$$

Typically, we set $\sigma_{0,n}^2$ to be a very small value while $\sigma_{1,n}^2$ to be relatively large.

Approximation Theory: Posterior Consistency

Theorem 1

Suppose Assumptions A.1-A.3 hold. If the mixture Gaussian prior (4) is set appropriately, then there exists an error sequence $\epsilon_n^2 = O(\varpi_n^2) + O(\zeta_n^2)$ *such that* $\lim_{n\to\infty} \epsilon_n = 0$ *and* $\lim_{n\to\infty} n\epsilon_n^2 = \infty$, *and the posterior distribution satisfies*

$$\begin{aligned} &P^*\left\{\pi[d(p_\beta, p_{\mu^*}) > 4\epsilon_n | D_n] \geq 2e^{-cn\epsilon_n^2}\right\} \leq 2e^{-cn\epsilon_n^2}, \\ &E^*_{D_n}\pi[d(p_\beta, p_{\mu^*}) > 4\epsilon_n | D_n] \leq 4e^{-2cn\epsilon_n^2}, \end{aligned} \tag{5}$$

for sufficiently large n, *where* $\zeta_n^2 = [r_n H_n \log n + r_n \log \overline{L} + s_n \log p_n]/n$, p_{μ^*} *denotes the underlying true data distribution, and* p_β *denotes the data distribution reconstructed by the Bayesian DNN based on its posterior samples.*

Remarks on Theorem 1

- Theorem 1 provides a posterior contraction rate ϵ_n for the sparse BNN. The contraction rate contains two components, ϖ_n and ζ_n, which represent the network approximation error and the network estimation error, respectively.
- Theorem 1 implies that given a training sample size n, the proposed method can learn a sparse neural network with at most $O(n/\log(n))$ connections. The sparse BNN has nice theoretical properties, such as posterior consistency, variable selection consistency, and asymptotically optimal generalization bounds.
- Liang et al. (2018) employed a spike-and-slab prior, for which sampling from the posterior is hard.

Consistency of DNN structure selection

Let $\nu(\boldsymbol{\gamma}, \boldsymbol{\beta})$ be an operator that maps any neural network to a unique network defined in an equivalent class via appropriate transformations such as nodes permutation, sign changes, weight rescaling, etc. For each connection c_i, we define its marginal posterior inclusion probability by

$$q_i = \int \sum_{\boldsymbol{\gamma}} e_{i|\nu(\boldsymbol{\gamma},\boldsymbol{\beta})} \pi(\boldsymbol{\gamma}|\boldsymbol{\beta}, D_n)\pi(\boldsymbol{\beta}|D_n)d\boldsymbol{\beta}, \quad i = 1, 2, \ldots, K_n, \tag{6}$$

where $e_{i|\nu(\boldsymbol{\gamma},\boldsymbol{\beta})}$ is the indicator for the existence of connection c_i in the network $\nu(\boldsymbol{\gamma}, \boldsymbol{\beta})$.
Let $A(\epsilon_n) = \{\boldsymbol{\beta} : d(p_{\boldsymbol{\beta}}, p_{\mu^*}) \geq \epsilon_n\}$. Define

$$\rho(\epsilon_n) = \max_{1 \leq i \leq K_n} \int_{A(\epsilon_n)^c} \sum_{\boldsymbol{\gamma}} |e_{i|\nu(\boldsymbol{\gamma},\boldsymbol{\beta})} - e_{i|\nu(\boldsymbol{\gamma}^*,\boldsymbol{\beta}^*)}| \pi(\gamma|\boldsymbol{\beta}, D_n)\pi(\boldsymbol{\beta}|D_n)d\boldsymbol{\beta}.$$

The identifiability condition can be stated as follows:

B.1 $\rho(\epsilon_n) \to 0$, as $n \to \infty$ and $\epsilon_n \to 0$.

Consistency of DNN structure selection

Theorem 2

Assume that the conditions of Theorem 1 and the identifiability condition B.1 hold. Then

(i) $\max_{1\le i\le K_n}\{|q_i - e_{i|\nu(\gamma^*,\beta^*)}|\} \xrightarrow{p} 0$, *where* $\xrightarrow{p}$ *denotes convergence in probability;*

(ii) *(sure screening)* $P(\gamma_* \subset \hat{\gamma}_{\hat{q}}) \xrightarrow{p} 1$ *for any pre-specified* $\hat{q} \in (0, 1)$.

(iii) *(Consistency)* $P(\gamma_* = \hat{\gamma}_{0.5}) \xrightarrow{p} 1$.

Laplace approximation of marginal inclusion probability

$$\text{Define:} \quad h_n(\boldsymbol{\beta}) = \frac{1}{n}\sum_{i=1}^{n} \log(p(y_i, \boldsymbol{x}_i|\boldsymbol{\beta})) + \frac{1}{n}\log(\pi(\boldsymbol{\beta})). \tag{7}$$

Let $\hat{\boldsymbol{\beta}}$ denote a strict local maximum of $h_n(\boldsymbol{\beta})$.

Theorem 3

Assume some regularity conditions hold for $h_n(\boldsymbol{\beta})$. For any bounded function $b(\boldsymbol{\beta})$, if $|b_{i_1,\ldots,i_d}(\boldsymbol{\beta})| = |\frac{\partial^d b(\boldsymbol{\beta})}{\partial\beta_{i_1}\partial\boldsymbol{\beta}_{i_2}\cdots\partial\boldsymbol{\beta}_{i_d}}| < M$ holds for any $1 \le d \le 2$ and any $1 \le i_1, \ldots, i_d \le K_n$, then for the posterior mean of $b(\boldsymbol{\beta})$, we have

$$\frac{\int b(\boldsymbol{\beta}) e^{nh_n(\boldsymbol{\beta})} d\boldsymbol{\beta}}{\int e^{nh_n(\boldsymbol{\beta})} d\boldsymbol{\beta}} = b(\hat{\boldsymbol{\beta}}) + O\left(\frac{r_n^4}{n}\right).$$

Laplace approximation of marginal inclusion probability

With an appropriate choice of prior hyperparameters, we can show that $\pi(\gamma_i = 1|\boldsymbol{\beta})$ satisfies all the requirements of $b(\boldsymbol{\beta})$ in Theorem 3 with a probability tending to 1 as $n \to \infty$. Then, by Theorem 3, q_k and $\pi(\gamma_i = 1|\hat{\boldsymbol{\beta}})$ are approximately the same as $n \to \infty$. Combining with Theorem 2, we have that $\pi(\gamma_i = 1|\hat{\boldsymbol{\beta}})$ is a consistent estimator of $e_{i|\nu(\boldsymbol{\gamma}^*, \boldsymbol{\beta}^*)}$.

Asymptotic Normality of Connection Weights

Theorem 4

Assume the conditions of Theorem 2 hold with $\rho(\epsilon_n) = o(\frac{1}{K_n})$ *and* $C_1 > \frac{2}{3}$ *in Condition A.2.2. Under some further conditions,*

$$\pi[\sqrt{n}(\tilde{\nu}(\boldsymbol{\beta}) - \boldsymbol{\beta}^*) \mid D_n] \rightsquigarrow N(0, \boldsymbol{V}),$$

as $n \to \infty$, *where* $\boldsymbol{V} = (v_{ij})$, *and* $v_{i,j} = E(h^{i,j}(\boldsymbol{\beta}^*))$ *if* $i, j \in \boldsymbol{\gamma}^*$ *and* 0 *otherwise.*

Asymptotic Normality of Prediction

Theorem 5

Assume the conditions of Theorem 4 and the following condition hold: $|\mu_i(\boldsymbol{\beta}^*, \boldsymbol{x}_0)| < M$, $|\mu_{i,j}(\boldsymbol{\beta}, \boldsymbol{x}_0)| < M$, $|\mu_k(\boldsymbol{\beta}, \boldsymbol{x}_0)| < M$ *hold for any* $i, j \in \boldsymbol{\gamma}^*, k \notin \boldsymbol{\gamma}^*$ *and* $\boldsymbol{\beta} \in B_{2\delta_n}(\boldsymbol{\beta}^*)$, *where* M *denotes a constant. Then*

$$\pi[\sqrt{n}(\mu(\boldsymbol{\beta}, \boldsymbol{x}_0) - \mu(\boldsymbol{\beta}^*, \boldsymbol{x}_0)) \mid D_n] \rightsquigarrow N(0, \Sigma),$$

where $\Sigma = \nabla_{\boldsymbol{\gamma}^*} \mu(\boldsymbol{\beta}^*, \boldsymbol{x}_0)^T H^{-1} \nabla_{\boldsymbol{\gamma}^*} \mu(\boldsymbol{\beta}^*, \boldsymbol{x}_0)$ *and* $H = E(-\nabla^2_{\boldsymbol{\gamma}^*} l_n(\boldsymbol{\beta}^*))$ *is the Fisher information matrix.*

Training Algorithm I: Sparse DNN Elicitation with Bayesian Evidence

Repeat the following steps for $t = 1, 2, \ldots, T$, output the model with the largest evidence.

(i) *Initialization*: Randomly initialize the weights and biases of a fully connection DNN.

(ii) *Optimization*: Run SGD to maximize $h_n(\boldsymbol{\beta})$ as defined in (7). Denote the estimate of $\boldsymbol{\beta}$ by $\hat{\boldsymbol{\beta}}$.

(iii) *Sparsification*: For each $i \in \{1, 2, \ldots, K_n\}$, set $\gamma_i = 1$ if $|\hat{\beta}_i| > \frac{\sqrt{2}\sigma_{0,n}\sigma_{1,n}}{\sqrt{\sigma_{1,n}^2 - \sigma_{0,n}^2}} \sqrt{\log\left(\frac{1-\lambda_n}{\lambda_n}\frac{\sigma_{1,n}}{\sigma_{0,n}}\right)}$ and 0 otherwise. Denote the yielded sparse DNN structure by $\boldsymbol{\gamma}^t$.

(iv) *Nonzero-weights refining*: Refine the nonzero weights of the sparsified DNN, and denote the resulting DNN model by $\tilde{\boldsymbol{\beta}}_{\boldsymbol{\gamma}^t}$.

(v) *Model evaluation*: Calculate the Bayesian Evidence: $\mathit{Evidence}^t = \det(-\frac{n}{2\pi} H_n(\tilde{\boldsymbol{\beta}}_{\boldsymbol{\gamma}^t}))^{-\frac{1}{2}} e^{n h_n(\tilde{\boldsymbol{\beta}}_{\boldsymbol{\gamma}^t})}$.

Algorithm I: Remark

For a large-scale neural network, even if it is sparse, the number of nonzero elements can easily exceed a few thousands or millions. In this case, we suggest to approximate the log(Bayesian evidence) by $nh_n(\hat{\boldsymbol{\beta}}_{\boldsymbol{\gamma}}) - \frac{1}{2}|\boldsymbol{\gamma}| \log(n)$.

If the prior information imposed on the sparse DNNs is further ignored, then the sparse DNNs can be elicited by BIC.

Sparse Deep Learning under the Framework of Statistical Modeling

Repeat steps (i)-(v) for $t = 1, 2, \ldots, T$, output the minimum BIC sparse network.

(i) *Initialization*: Randomly initialize the weights of a fully connection DNN.

(ii) *Optimization*: Run SGD to find the MAP network (with an appropriate mixture Gaussian prior).

(iii) *Sparsification*: Truncate small weights to zero.

(iv) *Nonzero-weights refining*: Refine the nonzero weights of the sparsified DNN.

(v) *Model evaluation*: Calculate BIC of the sparsified DNN.

Code is available at https://github.com/sylydya/Consistent-Sparse-Deep-Learning-Theory-and-Computation.

Sparse Deep Learning Algorithm II: Prior Annealing

It has been shown in Nguyen and Hein (2017) and Gori and Tesi (1992) that the loss of an over-parameterized DNN exhibits good properties:

(S^*) For a fully connected DNN with an analytic activation function and a convex loss function at the output layer, if the number of hidden units of one layer is larger than the number of training points and the network structure from this layer on is pyramidal, then almost all local minima are globally optimal.

Training Algorithm II: Prior Annealing

(i) (*Initial training*) Train a DNN satisfying condition (S*) by SGD or Adam such that an optimum $\boldsymbol{\beta}_0 = \arg\max_{\boldsymbol{\beta}} l_n(\boldsymbol{\beta})$ is reached.

(ii) (*Prior annealing*) Initialize $\boldsymbol{\beta}$ at $\boldsymbol{\beta}_0$ and simulate from a sequence of distributions

$$\pi(\boldsymbol{\beta}|D_n, \tau, \eta^{(k)}, \sigma_{0,n}^{(k)}) \propto e^{n l_n(\boldsymbol{\beta})/\tau} \pi_k^{\eta^{(k)}/\tau}(\boldsymbol{\beta})$$

for $k = 1, 2, \ldots, m$, where $0 < \eta^{(1)} \leq \eta^{(2)} \leq \cdots \leq \eta^{(m)} = 1$.

(iii) (*Structure sparsification*) For each connection $i \in \{1, 2, \ldots, K_n\}$, set $\tilde{\gamma}_i = 1$ if $|\hat{\beta}_i| > \frac{\sqrt{2}\sigma_{0,n}\sigma_{1,n}}{\sqrt{\sigma_{1,n}^2 - \sigma_{0,n}^2}} \sqrt{\log\left(\frac{1-\lambda_n}{\lambda_n}\frac{\sigma_{1,n}}{\sigma_{0,n}}\right)}$ and 0 otherwise. Denote the yielded sparse DNN structure by $\tilde{\boldsymbol{\gamma}}$.

(iv) (*Nonzero-weights refining*) Refine the nonzero weights of the sparsified DNN by maximizing $l_n(\boldsymbol{\beta})$. Denote the resulting estimate by $\tilde{\boldsymbol{\beta}}_{\tilde{\boldsymbol{\gamma}}}$, which represents the MLE of $\boldsymbol{\beta}^*$.

The code is available at https://github.com/sylydya/Sparse-Deep-Learning-A-New-Framework-Immuneto-Local-Traps-and-Miscalibration.

Training Algorithm III: Bayesian Computation

For certain problems the size (or #nonzero elements) of γ^* is large, calculation of the Fisher information matrix is difficult. In this case, the prediction uncertainty can be quantified via posterior simulations. The simulation can be started with a DNN satisfying condition (S*) and performed using a SGMCMC algorithm with an annealed prior. The over-parameterized structure and annealed prior make the simulations immune to local traps.

Sparse Deep Learning Algorithm III: Bayesian Computation

Theorem 6

Suppose some regularity conditions hold. Then

$$\begin{aligned} &\mathbb{E}\left(\frac{1}{T}\sum_{t=1}^{T-1}\phi(\boldsymbol{\beta}^{(t)}) - \int \phi(\boldsymbol{\beta})\pi(\boldsymbol{\beta}|D_n, \eta^*, \sigma^*_{0,n})d\boldsymbol{\beta}\right) \\ =&O\left(\frac{1}{T\epsilon} + \frac{\sum_{t=0}^{T-1}(|\eta^{(t)} - \eta^*| + |\sigma^{(t)}_{0,n} - \sigma^*_{0,n}|)}{T} + \epsilon\right), \end{aligned} \tag{8}$$

where η^* and $\sigma^*_{0,n}$ are treated as fixed constants.

Theorem 6 shows that with prior annealing simulations, the path averaging estimator can still be used for estimating the mean and variance of the prediction and uncertainty quantification.

Simulation: Structure Selection

We generated 10 datasets from the following neural network model:

$$y = \tanh(2\tanh(2x_1 - x_2)) + 2\tanh(\tanh(x_3 - 2x_4) - \tanh(2x_5)) + 0x_6 + \cdots + 0x_{1000} + \varepsilon,$$

where $\varepsilon \sim N(0, 1)$ and is independent of x_i's.

Over ten datasets, in terms of variable selection, we got perfect results with $FSR = 0$ and $NSR = 0$; in terms of structure selection, we got $FSR = 0.377$ and $NSR = 0$.

Structure Selection

Figure 1: The left and right panels show the network structures selected for one dataset after the stages of training and retraining, respectively, where the black lines show the connections that are selected and exist in the true model, and the red lines show the connections that are selected but do not exist in the true model.

Simulation: Nonlinear Regression

We generated 10 datasets from the following model:

$$y = \frac{5x_2}{1+x_1^2} + 5\sin(x_3 x_4) + 2x_5 + 0x_6 + \cdots + 0x_{2000} + \varepsilon, \qquad (9)$$

where $\varepsilon \sim N(0,1)$ and is independent of x_i's. Each dataset consist of 10,000 samples for training and 1000 samples for testing. We modeled the data by a 3-hidden layer neural network with structure 2000-10000-100-10-1.

Simulation: Nonlinear regression

Table 1: Simulation Result: MSFE and MSPE were calculated by averaging over 10 datasets, and their standard deviations were given in the parentheses.

Method	$\lvert\hat{S}\rvert$	FSR	NSR	MSFE	MSPE
BNN_anneal	5(0)	0	0	2.353(0.296)	2.428(0.297)
BNN_Evidence	5(0)	0	0	2.372(0.093)	2.439(0.132)
Spinn	10.7(3.874)	0.462	0	4.157(0.219)	4.488(0.350)
DNN	-	-	-	1.17e-5(1.15e-6)	16.923(0.323)
Dropout	-	-	-	1.104(0.068)	13.183(0.716)
BART50	16.5(1.222)	0.727	0.1	11.182(0.334)	12.097(0.366)
LASSO	566.8(4.844)	0.993	0.26	8.542(0.022)	9.496(0.148)

Nonlinear regression: Uncertainty Quantification for Prediction

We conducted experiments over 100 different training sets, based on which we constructed 95% prediction intervals over 1000 test points. The average coverage rate over the 1000 test points is 94.72%(0.61%).

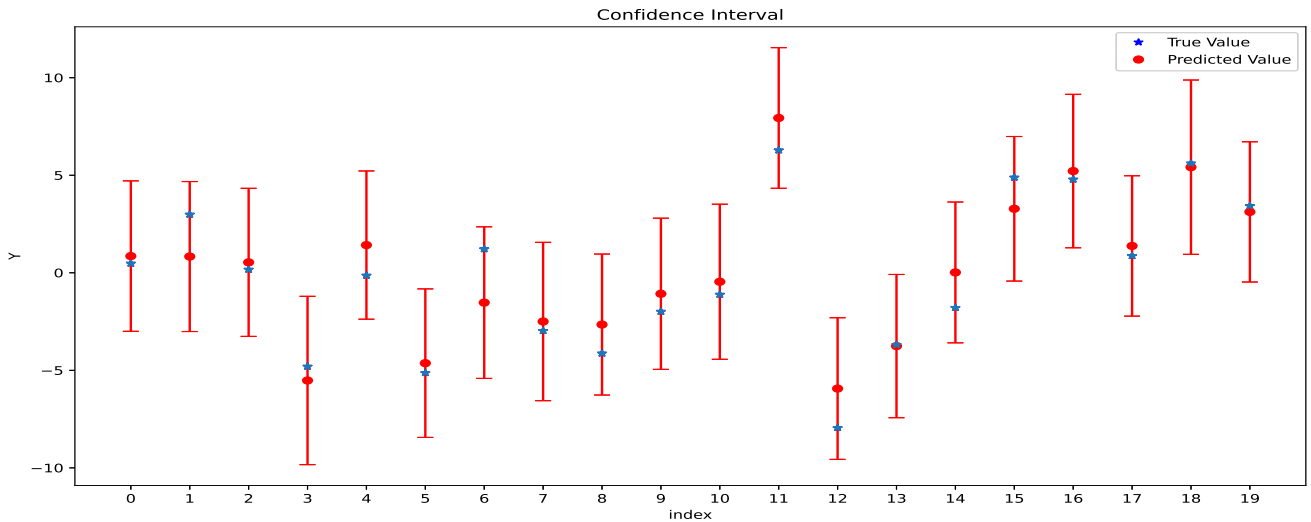

Figure 2: Prediction intervals of 20 testing points, where y-axis is for response, x-axis is for index of test points, and the blue point represents the true observation.

Residual Network Compression

Table 2: ResNet network pruning results for CIFAR-10 data.

Method	ResNet-20		ResNet-32	
	Pruning Ratio	Test Accuracy	Pruning Ratio	Test Accuracy
DNN_dense	100%	92.93(0.04)	100%	93.76(0.02)
BNN_average	19.85%(0.18%)	92.53(0.08)	9.99%(0.08%)	93.12(0.09)
BNN_anneal	19.80%(0.01%)	92.30(0.16)	9.97%(0.03%)	92.63(0.09)
BNN_BIC	19.67%(0.05%)	92.27(0.03)	9.53%(0.04%)	92.74(0.07)
SM[1]	20%	91.54(0.16)	10%	91.54(0.18)
DSR[1]	20%	91.78(0.28)	10%	91.41(0.23)
DPF[1]	20%	92.17(0.21)	10%	92.42(0.18)
BNN_average	9.88%(0.02%)	91.65(0.08)	4.77%(0.08%)	91.30(0.16)
BNN_anneal	9.95%(0.03%)	91.28(0.11)	4.88%(0.02%)	91.17(0.08)
BNN_BIC	9.55%(0.03%)	91.27(0.05)	4.78%(0.01%)	91.21(0.01)
SM	10%	89.76(0.40)	5%	88.68(0.22)
DSR	10%	87.88(0.04)	5%	84.12(0.32)
DPF	10%	90.88(0.07)	5%	90.94(0.35)

[1]Sparse momentum (SM) by Dettmers and Zettlemoyer (2019);
[1]Dynamic sparse representation (DSR) by Mostafa and Wang (2019)
[1]Dynamic pruning with feedback (DPF) by Lin et al. (2020)

Residual Network Compression

Table 3: ResNet network pruning results for CIFAR-10 data, which were calculated by averaging over 3 independent runs with the standard deviation reported in the parentheses.

Method	Model	Pruning Ratio	NLL	JS-Distance	ECE
DNN_dense	ResNet20	100%	0.2276(0.0021)	7.9118(0.9316)	0.02627(0.0005)
BNN_average	ResNet20	9.88%(0.02%)	0.2528(0.0029)	9.9641(0.3069)	0.0113(0.0010)
BNN_anneal	ResNet20	9.95%(0.03%)	0.2618(0.0037)	10.1251(0.1797)	0.0175(0.0011)
DPF	ResNet20	10%	0.2833(0.0004)	7.5712(0.4466)	0.0294(0.0009)
BNN_average	ResNet20	19.85%(0.18%)	0.2323(0.0033)	7.7007(0.5374)	0.0173(0.0014)
BNN_anneal	ResNet20	19.80%(0.01%)	0.2441(0.0042)	6.4435(0.2029)	0.0233(0.0020)
DPF	ResNet20	20%	0.2874(0.0029)	7.7329(0.1400)	0.0391(0.0001)
DNN_dense	ResNet32	100%	0.2042(0.0017)	6.7699(0.5253)	0.02613(0.00029)
BNN_average	ResNet32	9.99%(0.08%)	0.2116(0.0012)	9.4549(0.5456)	0.0132(0.0001)
BNN_anneal	ResNet32	9.97%(0.03%)	0.2218(0.0013)	8.5447(0.1393)	0.0192(0.0009)
DPF	ResNet32	10%	0.2677(0.0041)	7.8693(0.1840)	0.0364(0.0015)
BNN_average	ResNet32	4.77%(0.08%)	0.2587(0.0022)	7.0117(0.2222)	0.0100(0.0002)
BNN_anneal	ResNet32	4.88%(0.02%)	0.2676(0.0014)	6.8440(0.4850)	0.0149(0.0006)
DPF	ResNet32	5%	0.2921(0.0067)	6.3990(0.8384)	0.0276(0.0019)

Advantage of Sparse Deep Learning

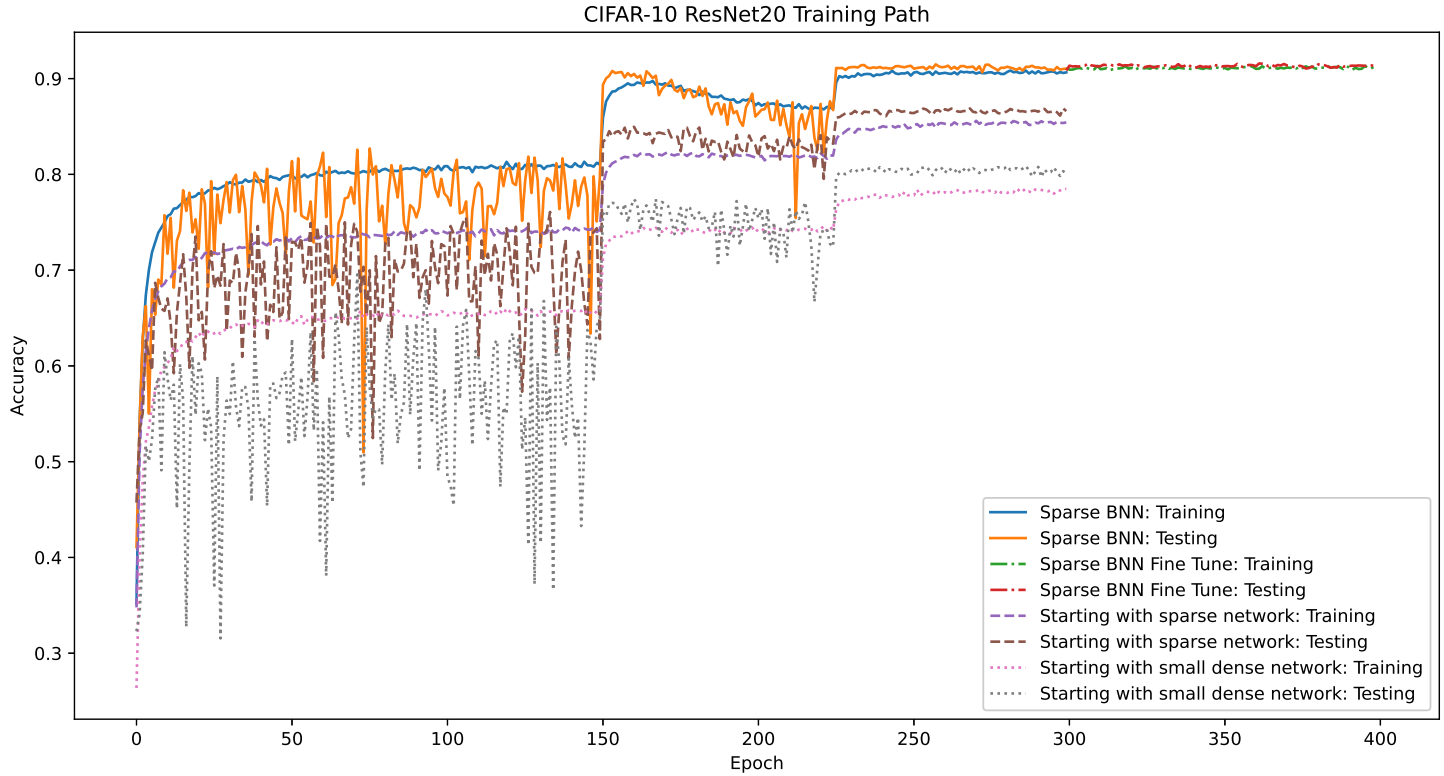

Figure 3: Training and testing paths of a ResNet20 model (with 10% sparsity level) on the CIFAR-10 dataset.

Conclusion: Theory

We have tamed deep neural networks into the framework of statistical modeling, and the resulting sparse DNN possesses nice theoretical properties such as posterior consistency, structure selection consistency, and asymptotic normality of prediction.

- The proposed method works like a frequentist method, but is justified under the Bayesian framework. With the proposed method, a sparse neural network with at most $O(n/\log(n))$ connections can be learned via sparsifying an over-parameterized one.

Conclusion: Computation

- In computation, we proposed three methods:
  - Use Bayesian evidence or BIC for eliciting sparse DNN models learned by an optimization method in multiple runs with different initializations.
  - Use a prior annealing method.
  - Bayesian computation with annealed priors.

  Since conventional optimization methods such as SGD and Adam can be used for training the DNN, the proposed method is computationally more efficient than the standard Bayesian method.
- Our numerical results show that the proposed method can perform very well in large-scale network compression, high-dimensional nonlinear variable selection, and prediction uncertainty quantification. The sparse DNN learned by the proposed method tends to predict better than those by the existing methods.

Reference

- Liang, F., Li, Q. and Zhou, L. (2018) Bayesian Neural Networks for Selection of Drug Sensitive Genes. *Journal of the American Statistical Association*, **113**, 955-972.
- 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. (* equal contribution)
- Sun, Y., Song, Q., and Liang, F. (2022) Learning sparse deep neural networks with a spike-and-slab prior. *Statistics and Probability Letters*, **180**, 109246.
- Sun, Y., Xiong, W. and Liang, F. (2021). Sparse deep learning: A new framework immune to local traps and miscalibration. *NeurIPS 2021*.
- Zhang, M., Sun, Y., and Liang, F. (2023). Sparse Deep Learning for Time Series Data: Theory and Applications. *NeurIPS 2023*.