Journal of the American Statistical Association

ISSN: (Print) (Online) Journal homepage: https://www.tandfonline.com/loi/uasa20

Consistent Sparse Deep Learning: Theory and Computation

Yan Sun, Qifan Song & Faming Liang

To cite this article: Yan Sun, Qifan Song & Faming Liang (2021): Consistent Sparse Deep Learning: Theory and Computation, Journal of the American Statistical Association, DOI: 10.1080/01621459.2021.1895175

To link to this article: https://doi.org/10.1080/01621459.2021.1895175

JOURNAL OF THE AMERICAN STATISTICAL ASSOCIATION
2021, VOL. 00, NO. 0, 1–15: Theory and Methods
https://doi.org/10.1080/01621459.2021.1895175

Consistent Sparse Deep Learning: Theory and Computation

Yan Sun*, Qifan Song*, and Faming Liang

Department of Statistics, Purdue University, West Lafayette, IN

ABSTRACT

Deep learning has been the engine powering many successes of data science. However, the deep neural network (DNN), as the basic model of deep learning, is often excessively over-parameterized, causing many difficulties in training, prediction and interpretation. We propose a frequentist-like method for learning sparse DNNs and justify its consistency under the Bayesian framework: the proposed method could learn a sparse DNN with at most $O(n/\log(n))$ connections and nice theoretical guarantees such as posterior consistency, variable selection consistency and asymptotically optimal generalization bounds. In particular, we establish posterior consistency for the sparse DNN with a mixture Gaussian prior, show that the structure of the sparse DNN can be consistently determined using a Laplace approximation-based marginal posterior inclusion probability approach, and use Bayesian evidence to elicit sparse DNNs learned by an optimization method such as stochastic gradient descent in multiple runs with different initializations. The proposed method is computationally more efficient than standard Bayesian methods for large-scale sparse DNNs. The numerical results indicate that the proposed method can perform very well for large-scale network compression and high-dimensional nonlinear variable selection, both advancing interpretable machine learning.

ARTICLE HISTORY
Received October 2019
Accepted February 2021

KEYWORDS
Bayesian evidence; Laplace approximation; Network compression; Nonlinear feature selection; Posterior consistency

1. Introduction

During the past decade, the deep neural network (DNN) has achieved great successes in solving many complex machine learning tasks such as pattern recognition and natural language processing. A key factor to the successes is its superior approximation power over the shallow one (Montufar et al. 2014; Mhaskar, Liao, and Poggio 2017; Telgarsky 2017; Yarotsky 2017). The DNNs used in practice may consist of hundreds of layers and millions of parameters, see, for example, He et al. (2016) on image classification. Training and operation of DNNs of this scale entail formidable computational challenges. Moreover, the DNN models with massive parameters are more easily overfitted when the training samples are insufficient. DNNs are known to have many redundant parameters (Glorot, Bordes, and Bengio 2011; Denil et al. 2013; Cheng et al. 2015; Scardapane et al. 2017; Yoon and Hwang 2017; Mocanu et al. 2018). For example, Denil et al. (2013) showed that in some networks, only 5% of the parameters are enough to achieve acceptable models; and Glorot, Bordes, and Bengio (2011) showed that sparsity (via employing a ReLU activation function) can generally improve the training and prediction performance of the DNN. Over-parameterization often makes the DNN model less interpretable and miscalibrated (Guo et al. 2017), which can cause serious issues in human-machine trust and thus hinder applications of artificial intelligence (AI) in human life.

The desire to reduce the complexity of DNNs naturally leads to two questions: (i) Is a sparsely connected DNN, also known as sparse DNN, able to approximate the target mapping with a desired accuracy? and (ii) how to train and determine the structure of a sparse DNN? This article answers these two questions in a coherent way. The proposed method is essentially a regularization method, but justified under the Bayesian framework.

The approximation power of sparse DNNs has been studied in the literature from both frequentist and Bayesian perspectives. From the frequentist perspective, Bölcskei et al. (2019) quantified the minimum network connectivity that guarantees uniform approximation rates for a class of affine functions; and Schmidt-Hieber (2017) and Bauler and Kohler (2019) characterized the approximation error of a sparsely connected neural network for Hölder smooth functions. From the Bayesian perspective, Liang, Li, and Zhou (2018) established posterior consistency for Bayesian shallow neural networks under mild conditions; and Polson and Ročková (2018) established posterior consistency for Bayesian DNNs but under some restrictive conditions such as a spike-and-slab prior is used for connection weights, the activation function is ReLU, and the number of input variables keeps at an order of $O(1)$ while the sample size grows to infinity.

The existing methods for learning sparse DNNs are usually developed separately from the approximation theory. For example, Alvarez and Salzmann (2016), Scardapane et al. (2017), and Ma et al. (2019) developed some regularization methods for learning sparse DNNs; Wager, Wang, and Liang (2013) showed that dropout training is approximately equivalent to an L_2-regularization; Han, Mao, and Dally (2015) introduced a deep compression pipeline, where pruning, trained quantization and

CONTACT Faming Liang fmliang@purdue.edu Department of Statistics, Purdue University, West Lafayette, IN 47907.
*Sun and Song contribute equally and are co-first authors.
Supplementary materials for this article are available online. Please go to www.tandfonline.com/r/JASA.

Huffman coding work together to reduce the storage requirement of DNNs; Liu et al. (2015) proposed a sparse decomposition method to sparsify convolutional neural networks (CNNs); Frankle and Carbin (2018) considered a lottery ticket hypothesis for selecting a sparse subnetwork; and Ghosh and Doshi-Velez (2017) proposed to learn Bayesian sparse neural networks via node selection with a horseshoe prior under the framework of variational inference. For these methods, it is generally unclear if the resulting sparse DNN is able to provide a desired approximation accuracy to the true mapping and how close in structure the sparse DNN is to the underlying true DNN.

On the other hand, there are some work which developed the approximation theory for sparse DNNs but not the associated learning algorithms (see, e.g., Polson and Ročková 2018; Bölcskei et al. 2019). An exception is Liang, Li, and Zhou (2018), where the population stochastic approximation Monte Carlo (pop-SAMC) algorithm (Song, Wu, and Liang 2014) was employed to learn sparse neural networks. However, since the pop-SAMC algorithm belongs to the class of traditional MCMC algorithms, where the full data likelihood needs to be evaluated at each iteration, it is not scalable for big data problems. Moreover, it needs to run for a large number of iterations for ensuring convergence. As an alternative to overcome the convergence issue of MCMC simulations, the variational Bayesian method (Jordan et al. 1999) has been widely used in the machine learning community. Recently, it has been applied to learn Bayesian neural networks (BNNs) (see, e.g., Mnih and Gregor 2014; Blundell et al. 2015). However, theoretical properties of the variational posterior of the BNN are still not well understood due to its approximation nature.

This article provides a frequentist-like method for learning sparse DNNs, which, with theoretical guarantee, converges to the underlying true DNN model in probability. The proposed method is to first train a dense DNN using an optimization method such as stochastic gradient descent (SGD) by maximizing its posterior distribution with a mixture Gaussian prior, and then sparsify its structure according to the Laplace approximation of the marginal posterior inclusion probabilities. Finally, Bayesian evidence is used as the criterion for eliciting sparse DNNs learned by the optimization method in multiple runs with different initializations. To justify consistency of the sparsified DNN, we first establish posterior consistency for Bayesian DNNs with mixture Gaussian priors and consistency of structure selection for Bayesian DNNs based on the marginal posterior inclusion probabilities, and then establish consistency of the sparsified DNN via Laplace approximation to the marginal posterior inclusion probabilities. In addition, we show that the Bayesian sparse DNN has asymptotically an optimal generalization bound.

The proposed method works with various activation functions such as sigmoid, tanh, and ReLU, and our theory allows the number of input variables to increase with the training sample size in an exponential rate. Under regularity conditions, the proposed method learns a sparse DNN of size $O(n/\log(n))$ with nice theoretical guarantees such as posterior consistency, variable selection consistency, and asymptotically optimal generalization bound. Since, for the proposed method, the DNN only needs to be trained using an optimization method, it is computationally much more efficient than standard Bayesian methods. As a by-product, this work also provides an effective method for high-dimensional nonlinear variable selection. Our numerical results indicate that the proposed method can work very well for large-scale DNN compression and high-dimensional nonlinear variable selection. For some benchmark DNN compression examples, the proposed method produced the state-of-the-art prediction accuracy using about the same amounts of parameters as the existing methods. In summary, this article provides a complete treatment for sparse DNNs in both theory and computation.

The remaining part of the article is organized as follows. Section 2 studies the consistency theory of Bayesian sparse DNNs. Section 3 proposes a computational method for training sparse DNNs. Section 4 presents some numerical examples. Section 5 concludes the article with a brief discussion.

2. Consistent Sparse DNNs: Theory

2.1. Bayesian Sparse DNNs With Mixture Gaussian Prior

Let $D_n = (\boldsymbol{x}^{(i)}, y^{(i)})_{i=1,\ldots,n}$ denote a training dataset of n iid observations, where $\boldsymbol{x}^{(i)} \in R^{p_n}$, $y^{(i)} \in R$, and p_n denotes the dimension of input variables and is assumed to grow with the training sample size n. We first study the posterior approximation theory of Bayesian sparse DNNs under the framework of generalized linear models, for which the distribution of y given $\boldsymbol{x}$ is given by

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

where $\mu^*(\boldsymbol{x})$ denotes a nonlinear function of $\boldsymbol{x}$, and $A(\cdot)$, $B(\cdot)$ and $C(\cdot)$ are appropriately defined functions. The theoretical results presented in this work mainly focus on logistic regression models and normal linear regression models. For logistic regression, we have $A(\mu^*) = \mu^*$, $B(\mu^*) = -\log(1 + e^{\mu^*})$, and $C(y) = 1$. For normal regression, by introducing an extra dispersion parameter σ^2, we have $A(\mu^*) = \mu^*/\sigma^2$, $B(\mu^*) = -\mu^{*2}/2\sigma^2$ and $C(y) = -y^2/2\sigma^2 - \log(2\pi\sigma^2)/2$. For simplicity, $\sigma^2 = 1$ is assumed to be known in this article. How to extend our results to the case that σ^2 is unknown will be discussed in Remark 2.3.

We approximate $\mu^*(\boldsymbol{x})$ using a DNN. Consider a DNN with $H_n - 1$ hidden layers and L_h hidden units at layer h, where $L_{H_n} = 1$ for the output layer and $L_0 = p_n$ for the input layer. Let $\boldsymbol{w}^h \in \mathbb{R}^{L_h \times L_{h-1}}$ and $\boldsymbol{b}^h \in \mathbb{R}^{L_h \times 1}$, $h \in \{1, 2, \ldots, H_n\}$ denote the weights and bias of layer h, and let $\psi^h : R^{L_h \times 1} \to \mathbb{R}^{L_h \times 1}$ denote a coordinate-wise and piecewise differentiable activation function of layer h. The DNN forms a nonlinear mapping

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

where $\boldsymbol{\beta} = (\boldsymbol{w}, \boldsymbol{b}) = \left\{ w_{ij}^h, b_k^h : h \in \{1, 2, \ldots, H_n\}, i, k \in \{1, \ldots, L_h\}, j \in \{1, \ldots, L_{h-1}\} \right\}$ denotes the collection of all weights and biases, consisting of $K_n = \sum_{h=1}^{H_n} \left(L_{h-1} \times L_h + L_h \right)$ elements in total. To facilitate representation of the sparse DNN, we introduce an indicator variable for each weight and bias of the DNN, which indicates the existence of the connection in the network. 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. Further, we let $\boldsymbol{\gamma} = \{\boldsymbol{\gamma}_{ij}^{\boldsymbol{w}^h}, \boldsymbol{\gamma}_k^{\boldsymbol{b}^h} : h \in \{1, 2, \ldots, H_n\},$

$i,k \in \{1,\ldots,L_h\}, j \in \{1,\ldots,L_{h-1}\}$ and $\boldsymbol{\beta}_{\boldsymbol{\gamma}} = \{w_{ij}^h, b_k^h : \boldsymbol{\gamma}_{ij}^{w^h} = 1, \boldsymbol{\gamma}_k^{b^h} = 1, h \in \{1,2,\ldots,H_n\}, i,k \in \{1,\ldots,L_h\}, j \in \{1,\ldots,L_{h-1}\}\}$, which specify, respectively, the structure and associated parameters for a sparse DNN.

To conduct Bayesian analysis for the sparse DNN, we consider a mixture Gaussian prior specified as follows:

$$\boldsymbol{\gamma}_{ij}^{w^h} \sim \text{Bernoulli}(\lambda_n), \quad \boldsymbol{\gamma}_k^{b^h} \sim \text{Bernoulli}(\lambda_n), \tag{2}$$

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

where $h \in \{1,2,\ldots,H_N\}, i \in \{1,\ldots,L_{h-1}\}, j,k \in \{1,\ldots,L_h\}$, 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. When $\sigma_{0,n}^2 \to 0$, the prior is reduced to the spike-and-slab prior (Ishwaran and Rao 2005). Therefore, this prior can be viewed as a continuous relaxation of the spike-and-slab prior. Such a prior has been used by many authors in Bayesian variable selection (see, e.g., George and McCulloch 1993; Song and Liang 2017).

2.2. Posterior Consistency

Posterior consistency plays a major role in validating Bayesian methods especially for high-dimensional models (see, e.g., Jiang 2007; Liang, Song, and Yu 2013). For DNNs, since the total number of parameters K_n is often much larger than the sample size n, posterior consistency provides a general guideline in prior setting or choosing prior hyperparameters for a class of prior distributions. Otherwise, the prior information may dominate data information, rendering a biased inference for the underlying true model. In what follows, we prove the posterior consistency of the DNN model with the mixture Gaussian prior (4).

With slight abuse of notation, we rewrite $\mu(\boldsymbol{\beta},\boldsymbol{x})$ in (1) as $\mu(\boldsymbol{\beta},\boldsymbol{\gamma},\boldsymbol{x})$ for a sparse network by including its network structure information. We assume $\mu^*(\boldsymbol{x})$ can be well approximated by a *sparse DNN* with relevant variables, and call this sparse DNN as the *true DNN* in this article. More precisely, we define the *true DNN* as

$$(\boldsymbol{\beta}^*,\boldsymbol{\gamma}^*) = \underset{(\boldsymbol{\beta},\boldsymbol{\gamma})\in\mathcal{G}_n, ||\mu(\boldsymbol{\beta},\boldsymbol{\gamma},\boldsymbol{x})-\mu^*(\boldsymbol{x})||_{L^2(\Omega)}\le\varpi_n}{\arg\min} |\boldsymbol{\gamma}|, \tag{5}$$

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 $n \to \infty$. For any given DNN $(\boldsymbol{\beta},\boldsymbol{\gamma})$, the error $\mu(\boldsymbol{\beta},\boldsymbol{\gamma},\boldsymbol{x}) - \mu^*(\boldsymbol{x})$ can be generally decomposed as the network approximation error $\mu(\boldsymbol{\beta}^*,\boldsymbol{\gamma}^*,\boldsymbol{x}) - \mu^*(\boldsymbol{x})$ and the network estimation error $\mu(\boldsymbol{\beta},\boldsymbol{\gamma},\boldsymbol{x}) - \mu(\boldsymbol{\beta}^*,\boldsymbol{\gamma}^*,\boldsymbol{x})$. The L_2 norm of the former one is bounded by ϖ_n, and the order of the latter will be given in Theorem 2.1. In what follows, we will treat ϖ_n as the network approximation error. In addition, we make the following assumptions:

A.1 The input $\boldsymbol{x}$ is bounded by 1 entry-wisely, that is, $\boldsymbol{x} \in \Omega = [-1,1]^{p_n}$, 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: $r_n H_n \log n + r_n \log \bar{L} + s_n \log p_n \le C_0 n^{1-\varepsilon}$, where $0 < \varepsilon < 1$ is a small constant, $r_n = |\boldsymbol{\gamma}^*|$ denotes the connectivity of $\boldsymbol{\gamma}^*$, $\bar{L} = \max_{1\le j\le H_n-1} L_j$ denotes the maximum hidden layer width, s_n denotes the input dimension of $\boldsymbol{\gamma}^*$.
- A.2.2 The network weights are polynomially bounded: $||\boldsymbol{\beta}^*||_\infty \le E_n$, 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.

Assumption A.1 is a typical assumption for posterior consistency (see, e.g., Jiang 2007; Polson and Ročková 2018). In practice, all bounded data can be normalized to satisfy this assumption, for example, image data are bounded and usually normalized before training. Assumption A.3 is satisfied by many conventional activation functions such as sigmoid, tanh and ReLU.

Assumption A.2 specifies the class of DNN models that we are considering in this article. They are sparse, while still being able to approximate many types of functions arbitrarily well as the training sample size becomes large, that is, $\lim_{n\to\infty} \varpi_n = 0$. The approximation power of sparse DNNs has been studied in several existing work. For example, for the functions that can be represented by an affine system, Bölcskei et al. (2019) proved that if the network parameters are bounded in absolute value by some polynomial $g(r_n)$, that is, $||\boldsymbol{\beta}^*||_\infty \le g(r_n)$, then the approximation error $\varpi_n = O(r_n^{-\alpha^*})$ for some constant α^*. To fit this this result into our framework, we can let $r_n \asymp n^{(1-\epsilon)/2}$ for some $0 < \epsilon < 1$, $p_n = d$ for some constant d, $H_n < r_n + d$ and $\bar{L} < r_n$ (i.e., the setting given in Proposition 3.6 of Bölcskei et al. (2019)). Suppose that the degree of $g(\cdot)$ is c_2, that is, $g(r_n) \asymp r_n^{c_2}$, then $||\boldsymbol{\beta}^*||_\infty \asymp n^{c_2(1-\epsilon)/2} \prec n^{C_1} = E_n$ for some constant $C_1 > c_2(1-\epsilon)/2$. Therefore, Assumption A.2 is satisfied with the approximation error $\varpi_n = O(r_n^{-\alpha^*}) = O(n^{-\alpha^*(1-\epsilon)/2}) \triangleq O(n^{-\varsigma})$ (by defining $\varsigma = \alpha^*(1-\epsilon)/2$), which goes to 0 as $n \to \infty$. In summary, the minimax rate in $\sup_{\mu^*(\boldsymbol{x})\in\mathcal{C}} \inf_{(\boldsymbol{\beta},\boldsymbol{\gamma})\in\mathcal{G}} ||\mu(\boldsymbol{\beta},\boldsymbol{\gamma},\boldsymbol{x}) - \mu^*(\boldsymbol{x})||_{L^2(\Omega)} \in \mathcal{O}(n^{-\varsigma})$ can be achieved by sparse DNNs under our assumptions, where $\mathcal{C}$ denotes the class of functions represented by an affine system.

Other than affine functions, our setup for the sparse DNN also matches the approximation theory for many other types of functions. For example, Corollary 3.7 of 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 $\varpi_n = O(n^{-\varsigma})$ for some constant $\varsigma > 0$. As another example, Theorem 3 of Schmidt-Hieber (2017) (see also Lemma 5.1 of Polson and Ročková (2018)) proved that any bounded α-Hölder smooth function $\mu^*(\boldsymbol{x})$ 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 also satisfies condition A.2.2 as long as $p_n^2 \ll \log n$.

It is important to note that there is a 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. In contrast, 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. In the practice of modern neural networks, the depth and width have been increased without much scruple. These increases reduce the training error, improve the generalization performance under certain regimes (Nakkiran et al. 2020), but negatively affect model calibration (Guo et al. 2017). We expect that our theory can tame the powerful neural networks into the framework of statistical modeling; that is, 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 model while controlling the training error to a reasonable level. The calibration of the sparse DNN will be explored elsewhere.

Let P^* and E^* denote the respective probability measure and expectation for data D_n. Let $d(p_1, p_2) = \left(\int \left[p_1^{\frac{1}{2}}(\boldsymbol{x}, y) - p_2^{\frac{1}{2}}(\boldsymbol{x}, y)\right]^2 dy d\boldsymbol{x}\right)^{\frac{1}{2}}$ denote the Hellinger distance between two density functions $p_1(\boldsymbol{x}, y)$ and $p_2(\boldsymbol{x}, y)$. Let $\pi(A|D_n)$ be the posterior probability of an event A. The following theorem establishes posterior consistency for sparse DNNs under the mixture Gaussian prior (4).

Theorem 2.1. Suppose Assumptions A.1–A.3 hold. If the mixture Gaussian prior (4) satisfies the conditions: $\lambda_n = O(1/\{K_n[n^{H_n}(\bar{L}p_n)]^\tau\})$ for some constant $\tau > 0$, $E_n/\{H_n \log n + \log \bar{L}\}^{1/2} \lesssim \sigma_{1,n} \lesssim n^\alpha$ for some constant $\alpha > 0$, and $\sigma_{0,n} \lesssim \min\left\{1/\{\sqrt{n}K_n(n^{3/2}\sigma_{1,0}/H_n)^{H_n}\}, \quad 1/\{\sqrt{n}K_n(nE_n/H_n)^{H_n}\}\right\}$, 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

$$\begin{aligned} &P^*\left\{\pi[d(p_{\boldsymbol{\beta}}, p_{\mu^*}) > 4\epsilon_n | D_n] \geq 2e^{-cn\epsilon_n^2}\right\} \leq 2e^{-cn\epsilon_n^2}, \\ &E_{D_n}^* \pi[d(p_{\boldsymbol{\beta}}, p_{\mu^*}) > 4\epsilon_n | D_n] \leq 4e^{-2cn\epsilon_n^2}, \end{aligned} \tag{6}$$

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

The proof of Theorem 2.1 can be found in the supplementary materials. Regarding this theorem, we have a few remarks:

Remark 2.1. Theorem 2.1 provides a posterior contraction rate ϵ_n for the sparse BNN. The contraction rate contains two components, ϖ_n and ζ_n, where ϖ_n, as defined previously, represents the network approximation error, and ζ_n represents the network estimation error measured in Hellinger distance. Since the estimation error ζ_n grows with the network connectivity r_n, there is a trade-off between the network approximation error and the network estimation error. A larger network has a lower approximation error and a higher estimation error, and vice versa.

Remark 2.2. Theorem 2.1 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. Compared to the fully connected DNN, the sparsity of the proposed BNN enables some theoretical guarantees for its performance. The sparse BNN has nice theoretical properties, such as posterior consistency, variable selection consistency, and asymptotically optimal generalization bounds, which are beyond the ability of general neural networks. The latter two properties will be established in Sections 2.3 and 2.4, respectively.

Remark 2.3. Although Theorem 2.1 is proved by assuming σ^2 is known, it can be easily extended to the case that σ^2 is unknown by assuming an inverse gamma prior $\sigma^2 \sim \text{IG}(a_0, b_0)$ for some constants $a_0, b_0 > 0$. If a relatively uninformative prior is desired, one can choose $a_0 \in (0, 1)$ such that the inverse gamma prior is very diffuse with a nonexisting mean value. However, if $a_0 = b_0 = 0$, that is, the Jeffreys' prior $\pi(\sigma^2) \propto 1/\sigma^2$, the posterior consistency theory established Theorem 2.1 might not hold any more. In general, to achieve posterior consistency, the prior is required, at least in the framework adopted by the article, to satisfy two conditions (Ghosal, Ghosh, and Van Der Vaart 2000; Jiang 2007): (i) a not too little prior probability is placed over the neighborhood of the true density, and (ii) a very little prior probability is placed outside of a region that is not too complex. Obviously, the Jeffreys' prior and thus the joint prior of σ^2 and the regression coefficients do not satisfy neither of the two conditions. We note that the inverse gamma prior $\sigma^2 \sim \text{IG}(a_0, b_0)$ has long been used in Bayesian inference for many different statistical models, such as linear regression (George and McCulloch 1997), nonparametric regression (Kohn, Smith, and Chan 2001), and Gaussian graphical models (Dobra et al. 2004).

2.3. Consistency of DNN Structure Selection

This section establishes consistency of DNN structure selection under posterior consistency. It is known that the DNN model is generally nonidentifiable due to the symmetry of the network structure. For example, the approximation $\mu(\boldsymbol{\beta}, \boldsymbol{\gamma}, \boldsymbol{x})$ can be invariant if one permutes the orders of certain hidden nodes, simultaneously changes the signs of certain weights and biases if *tanh* is used as the activation function, or rescales certain weights and bias if Relu is used as the activation function. However, by introducing appropriate constraints (see, e.g., Pourzanjani, Jiang, and Petzold 2017; Liang, Li, and Zhou 2018), we can define a set of neural networks such that any possible neural networks can be represented by one and only one neural network in the set via nodes permutation, sign changes, weight rescaling, etc. Let Θ denote such set of DNNs, where each

element in Θ can be viewed as an equivalent class of DNN models. Let $\nu(\boldsymbol{\gamma}, \boldsymbol{\beta}) \in \Theta$ be an operator that maps any neural network to Θ via appropriate transformations such as nodes permutation, sign changes, weight rescaling, etc. To serve the purpose of structure selection in the space Θ, we consider the marginal posterior inclusion probability approach proposed in Liang, Song, and Yu (2013) for high-dimensional variable selection.

For a better description of this approach, we reparameterize $\boldsymbol{\beta}$ and $\boldsymbol{\gamma}$ as $\boldsymbol{\beta} = (\boldsymbol{\beta}_1, \boldsymbol{\beta}_2, \ldots, \boldsymbol{\beta}_{K_n})$ and $\boldsymbol{\gamma} = (\boldsymbol{\gamma}_1, \boldsymbol{\gamma}_2, \ldots, \boldsymbol{\gamma}_{K_n})$, respectively, according to their elements. Without possible confusions, we will often use the indicator vector $\boldsymbol{\gamma}$ and the active set $\{i : \boldsymbol{\gamma}_i = 1, i = 1, 2, \ldots, K_n\}$ exchangeably; that is, $i \in \boldsymbol{\gamma}$ and $\boldsymbol{\gamma}_i = 1$ are equivalent. In addition, we will treat the connection weights $\boldsymbol{w}$ and the hidden unit biases $\boldsymbol{b}$ equally; that is, they will not be distinguished in $\boldsymbol{\beta}$ and $\boldsymbol{\gamma}$. For convenience, we will call each element of $\boldsymbol{\beta}$ and $\boldsymbol{\gamma}$ a "connection" in what follows.

2.3.1. Marginal Posterior Inclusion Probability Approach

For each connection c_i, we define its marginal posterior inclusion probability by

$$q_i = \int \sum_{\boldsymbol{\gamma}} e_{i|\nu(\boldsymbol{\gamma},\boldsymbol{\beta})} \pi(\boldsymbol{\gamma}|\boldsymbol{\beta}, D_n) \pi(\boldsymbol{\beta}|D_n) d\boldsymbol{\beta}, \quad i = 1, 2, \ldots, K_n, \tag{7}$$

where $e_{i|\nu(\boldsymbol{\gamma},\boldsymbol{\beta})}$ is the indicator for the existence of connection c_i in the network $\nu(\boldsymbol{\gamma}, \boldsymbol{\beta})$. Similarly, we define $e_{i|\nu(\boldsymbol{\gamma}^*,\boldsymbol{\beta}^*)}$ as the indicator for the existence of connection c_i in the true model $\nu(\boldsymbol{\gamma}^*, \boldsymbol{\beta}^*)$. The proposed approach is to choose the connections whose marginal posterior inclusion probabilities are greater than a threshold value $\hat{q}$; that is, setting $\hat{\boldsymbol{\gamma}}_{\hat{q}} = \{i : q_i > \hat{q}, i = 1, 2, \ldots, K_n\}$ as an estimator of $\boldsymbol{\gamma}_* = \{i : e_{i|\nu(\boldsymbol{\gamma}^*,\boldsymbol{\beta}^*)} = 1, i = 1, \ldots, K_n\}$, where $\boldsymbol{\gamma}_*$ can be viewed as the uniquenized true model. To establish the consistency of $\hat{\boldsymbol{\gamma}}_{\hat{q}}$, an identifiability condition for the true model is needed. Let $A(\epsilon_n) = \{\boldsymbol{\beta} : d(p_{\boldsymbol{\beta}}, p_{\mu^*}) \geq \epsilon_n\}$. Define

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

which measures the structure difference between the true model and the sampled models on the set $A(\epsilon_n)^c$. Then the identifiability condition can be stated as follows:

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

That is, when n is sufficiently large, if a DNN has approximately the same probability distribution as the true DNN, then the structure of the DNN, after mapping into the parameter space Θ, must coincide with that of the true DNN. Note that this identifiability is different from the one mentioned at the beginning of the section. The earlier one is only with respect to structure and parameter rearrangement of the DNN. Theorem 2.2 concerns consistency of $\hat{\boldsymbol{\gamma}}_{\hat{q}}$ and its sure screening property, whose proof is given in the supplementary materials.

Theorem 2.2. Assume that the conditions of Theorem 2.1 and the identifiability condition B.1 hold. Then

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

(ii) (Sure screening) $P(\boldsymbol{\gamma}_* \subset \hat{\boldsymbol{\gamma}}_{\hat{q}}) \xrightarrow{p} 1$ for any prespecified $\hat{q} \in (0, 1)$.

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

For a network $\boldsymbol{\gamma}$, it is easy to identify the relevant variables. Recall that $\boldsymbol{\gamma}^{\boldsymbol{w}^h} \in \mathbb{R}^{L_h \times L_{h-1}}$ denotes the connection indicator matrix of layer h. Let

$$\boldsymbol{\gamma}^{\boldsymbol{x}} = \boldsymbol{\gamma}^{\boldsymbol{w}^{H_n}} \boldsymbol{\gamma}^{\boldsymbol{w}^{H_n-1}} \cdots \boldsymbol{\gamma}^{\boldsymbol{w}^1} \in \mathbb{R}^{1 \times p_n}, \tag{8}$$

and let $\boldsymbol{\gamma}_i^{\boldsymbol{x}}$ denote the ith element of $\boldsymbol{\gamma}^{\boldsymbol{x}}$. It is easy to see that if $\boldsymbol{\gamma}_i^{\boldsymbol{x}} > 0$ then the variable $\boldsymbol{x}_i$ is effective in the network $\boldsymbol{\gamma}$, and $\boldsymbol{\gamma}_i^{\boldsymbol{x}} = 0$ otherwise. Let $e_{\boldsymbol{x}_i|\nu(\boldsymbol{\gamma}^*,\boldsymbol{\beta}^*)}$ be the indicator for the effectiveness of variable $\boldsymbol{x}_i$ in the network $\nu(\boldsymbol{\gamma}^*, \boldsymbol{\beta}^*)$, and let $\boldsymbol{\gamma}_*^{\boldsymbol{x}} = \{i : e_{\boldsymbol{x}_i|\nu(\boldsymbol{\gamma}^*,\boldsymbol{\beta}^*)} = 1, i = 1, \ldots, p_n\}$ denote the set of true variables. Similar to (7), we can define the marginal inclusion probability for each variable:

$$q_i^{\boldsymbol{x}} = \int \sum_{\boldsymbol{\gamma}} e_{\boldsymbol{x}_i|\nu(\boldsymbol{\gamma},\boldsymbol{\beta})} \pi(\boldsymbol{\gamma}|\boldsymbol{\beta}, D_n) \pi(\boldsymbol{\beta}|D_n) d\boldsymbol{\beta}, \quad i = 1, 2, \ldots, p_n. \tag{9}$$

Then we can select the variables whose marginal posterior inclusion probabilities greater than a threshold $\hat{q}^{\boldsymbol{x}}$, for example, setting $\hat{q}^{\boldsymbol{x}} = 0.5$. As implied by (8), the consistency of structure selection implies consistency of variable selection.

It is worth noting that the above variable selection consistency result is with respect to the relevant variables defined by the true network $\boldsymbol{\gamma}^*$. To achieve the variable selection consistency with respect to the relevant variables of $\mu^*(\boldsymbol{x})$, some extra assumptions are needed in defining $(\boldsymbol{\beta}^*, \boldsymbol{\gamma}^*)$. How to specify these assumptions is an open problem and we would leave it to readers. However, as shown by our simulation example, the sparse model $(\boldsymbol{\beta}^*, \boldsymbol{\gamma}^*)$ defined in (5) works well, which correctly identifies all the relevant variables of the underlying nonlinear system.

2.3.2. Laplace Approximation of Marginal Posterior Inclusion Probabilities

Theorem 2.2 establishes the consistency of DNN structure selection based on the marginal posterior inclusion probabilities. To obtain Bayesian estimates of the marginal posterior inclusion probabilities, intensive Markov chain Monte Carlo (MCMC) simulations are usually required. Instead of performing MCMC simulations, we propose to approximate the marginal posterior inclusion probabilities using the Laplace method based on the DNN model trained by an optimization method such as SGD. Traditionally, such approximation is required to be performed at the maximum a posteriori (MAP) estimate of the DNN. However, finding the MAP for a large DNN is not computationally guaranteed, as there can be many local minima on its energy landscape. To tackle this issue, we proposed a Bayesian evidence method, see Section 3 for the detail, for eliciting sparse DNN models learned by an optimization method in multiple runs with different initializations. Since conventional optimization methods such as SGD can be used to train the DNN here, the proposed method is computationally much more efficient than

the standard Bayesian method. More importantly, as explained in Section 3, consistent estimates of the marginal posterior inclusion probabilities might be obtained at a local maximizer of the log-posterior instead of the MAP estimate. In what follows, we justify the validity of Laplace approximation for marginal posterior inclusion probabilities.

Based on the marginal posterior distribution $\pi(\boldsymbol{\beta}|D_n)$, the marginal posterior inclusion probability q_i of connection c_i can be re-expressed as

$$q_i = \int \pi(\boldsymbol{\gamma}_i = 1|\boldsymbol{\beta})\pi(\boldsymbol{\beta}|D_n)d\boldsymbol{\beta}, \quad i = 1, 2, \ldots, K_n.$$

Under the mixture Gaussian prior, it is easy to derive that

$$\pi(\boldsymbol{\gamma}_i = 1|\boldsymbol{\beta}) = \tilde{b}_i/(\tilde{a}_i + \tilde{b}_i), \tag{10}$$

where

$$\tilde{a}_i = \frac{1-\lambda_n}{\sigma_{0,n}} \exp\left\{-\frac{\boldsymbol{\beta}_i^2}{2\sigma_{0,n}^2}\right\}, \quad \tilde{b}_i = \frac{\lambda_n}{\sigma_{1,n}} \exp\left\{-\frac{\boldsymbol{\beta}_i^2}{2\sigma_{1,n}^2}\right\}.$$

Let's define

$$h_n(\boldsymbol{\beta}) = \frac{1}{n}\sum_{i=1}^{n} \log(p(y_i, \boldsymbol{x}_i|\boldsymbol{\beta})) + \frac{1}{n}\log(\pi(\boldsymbol{\beta})), \tag{11}$$

where $p(y_i, \boldsymbol{x}_i|\boldsymbol{\beta})$ denotes the likelihood function of the observation $(y_i, \boldsymbol{x}_i)$ and $\pi(\boldsymbol{\beta})$ denotes the prior as specified in (4). Then $\pi(\boldsymbol{\beta}|D_n) = \frac{e^{nh_n(\boldsymbol{\beta})}}{\int e^{nh_n(\boldsymbol{\beta})}d\boldsymbol{\beta}}$ and, for a function $b(\boldsymbol{\beta})$, the posterior expectation is given by $\frac{\int b(\boldsymbol{\beta})e^{nh_n(\boldsymbol{\beta})}d\boldsymbol{\beta}}{\int e^{nh_n(\boldsymbol{\beta})}d\boldsymbol{\beta}}$. Let $\hat{\boldsymbol{\beta}}$ denote a strict local maximum of $\pi(\boldsymbol{\beta}|D_n)$. Then $\hat{\boldsymbol{\beta}}$ is also a local maximum of $h_n(\boldsymbol{\beta})$. Let $B_\delta(\boldsymbol{\beta})$ denote an Euclidean ball of radius δ centered at $\boldsymbol{\beta}$. Let $h_{i_1,i_2,\ldots,i_d}(\boldsymbol{\beta})$ denote the dth order partial derivative $\frac{\partial^d h(\boldsymbol{\beta})}{\partial\boldsymbol{\beta}_{i_1}\partial\boldsymbol{\beta}_{i_2}\cdots\partial\boldsymbol{\beta}_{i_d}}$, let $H_n(\boldsymbol{\beta})$ denote the Hessian matrix of $h_n(\boldsymbol{\beta})$, let h_{ij} denote the (i,j)th component of the Hessian matrix, and let h^{ij} denote the (i,j)-component of the inverse of the Hessian matrix. Recall that $\boldsymbol{\gamma}^*$ denotes the set of indicators for the connections of the true sparse DNN, r_n denotes the size of the true sparse DNN, and K_n denotes the size of the fully connected DNN. The following theorem justifies the Laplace approximation of the posterior mean for a bounded function $b(\boldsymbol{\beta})$.

Theorem 2.3. Assume that there exist positive numbers ϵ, M, η, and n_0 such that for any $n > n_0$, the function $h_n(\boldsymbol{\beta})$ in (11) satisfies the following conditions:

C.1 $|h_{i_1,\ldots,i_d}(\hat{\boldsymbol{\beta}})| < M$ hold for any $\boldsymbol{\beta} \in B_\epsilon(\hat{\boldsymbol{\beta}})$ and any $1 \le i_1, \ldots, i_d \le K_n$, where $3 \le d \le 4$.

C.2 $|h^{ij}(\hat{\boldsymbol{\beta}})| < M$ if $\boldsymbol{\gamma}_i^* = \boldsymbol{\gamma}_j^* = 1$ and $|h^{ij}(\hat{\boldsymbol{\beta}})| = O(\frac{1}{K_n^2})$ otherwise.

C.3 $\det(-\frac{n}{2\pi}H_n(\hat{\boldsymbol{\beta}}))^{\frac{1}{2}} \int_{\mathbb{R}^{K_n}\setminus B_\delta(\hat{\boldsymbol{\beta}})} e^{n(h_n(\boldsymbol{\beta})-h_n(\hat{\boldsymbol{\beta}}))}d\boldsymbol{\beta} = O(\frac{r_n^4}{n}) = o(1)$ for any $0 < \delta < \epsilon$.

For any bounded function $b(\boldsymbol{\beta})$, if $|b_{i_1,\ldots,i_d}(\boldsymbol{\beta})| = |\frac{\partial^d b(\boldsymbol{\beta})}{\partial\boldsymbol{\beta}_{i_1}\partial\boldsymbol{\beta}_{i_2}\cdots\partial\boldsymbol{\beta}_{i_d}}| < M$ 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

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

Conditions C.1 and C.3 are typical conditions for Laplace approximation (see, e.g., Kass, Tierney, and Kadane 1990). Condition C.2 requires the inverse Hessian to have very small values for the elements corresponding to the false connections. To justify condition C.2, we note that for a multivariate normal distribution, the inverse Hessian is its covariance matrix. Thus, we expect that for the weights with small variance, their corresponding elements in the inverse Hessian matrix would be small as well. The following lemma quantifies the variance of the weights for the false connections.

Lemma 2.1. Assume that $\sup_n \int |\boldsymbol{\beta}_i|^{2+\delta}\pi(\beta_i|D_n)d\boldsymbol{\beta}_i \le C < \infty$ a.s. for some constants $\delta > 0$ and $C > 0$ and $\rho(\epsilon_n) \asymp \pi(d(p_{\boldsymbol{\beta}}, p_{\mu^*}) \ge \epsilon_n|D_n)$, where $\rho(\epsilon_n)$ is defined in Condition B.1. Then with an appropriate choice of prior hyperparameters and ϵ_n, $P^*\{E(\boldsymbol{\beta}_i^2|D_n) \prec \frac{1}{K_n^{2H_n-1}}\} \ge 1 - 2e^{-n\epsilon_n^2/4}$ holds for any false connection c_i in $\boldsymbol{\gamma}^*$ (i.e., $\boldsymbol{\gamma}_i^* = 0$).

In addition, with an appropriate choice of prior hyperparameters, we can also show that $\pi(\boldsymbol{\gamma}_i = 1|\boldsymbol{\beta})$ satisfies all the requirements of $b(\boldsymbol{\beta})$ in Theorem 2.3 with a probability tending to 1 as $n \to \infty$ (refer to Section 2.4 of the supplementary materials for the detail). Then, by Theorem 2.3, q_k and $\pi(\boldsymbol{\gamma}_i = 1|\hat{\boldsymbol{\beta}})$ are approximately the same as $n \to \infty$, where $\pi(\boldsymbol{\gamma}_i = 1|\hat{\boldsymbol{\beta}})$ is as defined in (10) but with $\boldsymbol{\beta}$ replaced by $\hat{\boldsymbol{\beta}}$. Combining with Theorem 2.2, we have that $\pi(\boldsymbol{\gamma}_i = 1|\hat{\boldsymbol{\beta}})$ is a consistent estimator of $e_{i|\nu(\boldsymbol{\gamma}^*,\boldsymbol{\beta}^*)}$.

2.4. Asymptotically Optimal Generalization Bound

This section shows the sparse BNN has asymptotically an optimal generalization bound. First, we introduce a PAC Bayesian bound due to McAllester (1999a, 1999b), where the acronym PAC stands for probably approximately correct. It states that with an arbitrarily high probability, the performance (as provided by a loss function) of a learning algorithm is upper-bounded by a term decaying to an optimal value as more data is collected (hence, "approximately correct"). PAC-Bayes has proven over the past two decades to be a powerful tool to derive theoretical guarantees for many machine learning algorithms.

Lemma 2.2 (PAC Bayesian bound). Let P be any data independent distribution on the machine parameters $\boldsymbol{\beta}$, and Q be any distribution that is potentially data-dependent and absolutely continuous with respective to P. If the loss function $l(\boldsymbol{\beta}, \boldsymbol{x}, y) \in [0, 1]$, then the following inequality holds with probability $1-\delta$,

$$\int E_{\boldsymbol{x},y} l(\boldsymbol{\beta}, \boldsymbol{x}, y)dQ \le \int \frac{1}{n}\sum_{i=1}^{n} l(\boldsymbol{\beta}, \boldsymbol{x}^{(i)}, y^{(i)})dQ + \sqrt{\frac{d_0(Q,P) + \log\frac{2\sqrt{n}}{\delta}}{2n}},$$

where $d_0(Q,P)$ denotes the Kullback–Leibler divergence between Q and P, and $(\boldsymbol{x}^{(i)}, y^{(i)})$ denotes the ith observation of the dataset.

For the binary classification problem, the DNN model fits a predictive distribution as $\hat{p}_1(\boldsymbol{x}; \boldsymbol{\beta}) := \hat{Pr}(y = 1|\boldsymbol{x}) =$

$\text{logit}^{-1}(\mu(\boldsymbol{\beta},\boldsymbol{x}))$ and $\hat{p}_0(\boldsymbol{x};\boldsymbol{\beta}) := \hat{Pr}(y = 0|\boldsymbol{x}) = 1 - \text{logit}^{-1}(\mu(\boldsymbol{\beta},\boldsymbol{x}))$. Given an observation $(\boldsymbol{x}, y)$, we define the loss with margin $\nu > 0$ as

$$l_\nu(\boldsymbol{\beta},\boldsymbol{x},y) = 1(\hat{p}_y(\boldsymbol{x};\boldsymbol{\beta}) - \hat{p}_{1-y}(\boldsymbol{x};\boldsymbol{\beta}) < \nu).$$

Therefore, the empirical loss for the whole dataset $\{\boldsymbol{x}^{(i)}, y^{(i)}\}_{i=1}^n$ is defined as $L_{\text{emp},\nu}(\boldsymbol{\beta}) = \sum l_\nu(\boldsymbol{\beta},\boldsymbol{x}^{(i)},y^{(i)})/n$, and the population loss is defined as $L_\nu(\boldsymbol{\beta}) = E_{\boldsymbol{x},y} l_\nu(\boldsymbol{\beta},\boldsymbol{x},y)$.

Theorem 2.4 (Bayesian Generalization error for classification). Suppose the conditions of Theorem 2.1 hold. For any $\nu > 0$, when n is sufficiently large, the following inequality holds with probability greater than $1 - \exp\{c_0 n\epsilon_n^2\}$,

$$\int L_0(\boldsymbol{\beta})d\pi(\boldsymbol{\beta}|D_n) \le \frac{1}{1 - 2\exp\{-c_1 n\epsilon_n^2\}} \times \int L_{\text{emp},\nu}(\boldsymbol{\beta})d\pi(\boldsymbol{\beta}|D_n) + O(\epsilon_n + \sqrt{\log n/n} + \exp\{-c_1 n\epsilon_n^2\}),$$

for some $c_0, c_1 > 0$, where ϵ_n is as defined in Theorem 2.1.

Theorem 2.4 characterizes the relationship between Bayesian population risk $\int L_0(\boldsymbol{\beta})d\pi(\boldsymbol{\beta}|D_n)$ and Bayesian empirical risk $\int L_{\text{emp},\nu}(\boldsymbol{\beta})d\pi(\boldsymbol{\beta}|D_n)$, and implies that the difference between them is $O(\epsilon_n)$. Furthermore, this generalization performance extends to any point estimator $\hat{\boldsymbol{\beta}}$, as long as $\hat{\boldsymbol{\beta}}$ belongs to the dominating posterior mode.

Theorem 2.5. Suppose that the conditions of Theorem 2.1 hold and estimation $\hat{\boldsymbol{\beta}}$ belongs to the dominating posterior mode under Theorem 2.1, then for any $\nu > 0$, the following inequality holds with probability greater than $1 - \exp\{c_0 n\epsilon_n^2\}$,

$$L_0(\hat{\boldsymbol{\beta}}) \le L_{\text{emp},\nu}(\hat{\boldsymbol{\beta}}) + O(\epsilon_n),$$

for some $c_0 > 0$.

It is worth to clarify that the statement "$\hat{\boldsymbol{\beta}}$ belongs to the dominating posterior mode" means $\hat{\boldsymbol{\beta}} \in B_n$ where B_n is defined in the proof of Theorem 2.1 and its posterior is greater than $1-\exp\{-cn\epsilon_n^2\}$ for some $c > 0$. Therefore, if $\hat{\boldsymbol{\beta}} \sim \pi(\boldsymbol{\beta}|D_n)$, that is, $\hat{\boldsymbol{\beta}}$ is one valid posterior sample, then with high probability, it belongs to the dominating posterior mode. The proof of the above two theorems can be found in the supplementary materials.

Now we consider the generalization error for regression models. Assume the following additional assumptions:

D.1 The activation function $\psi \in [-1, 1]$.
D.2 The last layer weights and bias in $\boldsymbol{\beta}^*$ are restricted to the interval $[-F_n, F_n]$ for some $F_n \le E_n$, while $F_n \to \infty$ is still allowed as $n \to \infty$.
D.3 $\max_{\boldsymbol{x}\in\Omega} |\mu^*(\boldsymbol{x})| \le F$ for some constant F.

Correspondingly, the priors of the last layer weights and bias are truncated on $[-F_n, F_n]$, that is, the two normal mixture prior (4) truncated on $[-F_n, F_n]$. By the same argument of Theorem S1 (in the supplementary materials), Theorem 2.1 still holds.

Note that the Hellinger distance for regression problem is defined as

$$d^2(p_{\boldsymbol{\beta}}, p_{\mu^*}) = \mathbb{E}_{\boldsymbol{x}}\left(1 - \exp\left\{-\frac{[\mu(\boldsymbol{\beta},\boldsymbol{x}) - \mu^*(\boldsymbol{x})]^2}{8\sigma^2}\right\}\right).$$

By our assumption, for any $\boldsymbol{\beta}$ on the prior support, $|\mu(\boldsymbol{\beta},\boldsymbol{x}) - \mu^*(\boldsymbol{x})|^2 \le (F + \bar{L}F_n)^2 := \bar{F}^2$, thus,

$$d^2(p_{\boldsymbol{\beta}}, p_{\mu^*}) \ge C_{\bar{F}} E_{\boldsymbol{x}}|\mu(\boldsymbol{\beta},\boldsymbol{x}) - \mu^*(\boldsymbol{x})|^2, \tag{12}$$

where $C_F = [1 - \exp(-4\bar{F}^2/8\sigma^2)]/4\bar{F}^2$. Furthermore, (6) implies that with probability at least $1 - 2\exp\{-cn\epsilon_n^2\}$,

$$\int d^2(p_{\boldsymbol{\beta}}, p_{\mu^*})d\pi(\boldsymbol{\beta}|D_n) \le 16\epsilon_n^2 + 2e^{-cn\epsilon_n^2}. \tag{13}$$

By combining (12) and (13), we obtain the following Bayesian generalization error result:

Theorem 2.6 (Bayesian generalization error for regression). Suppose the conditions of Theorem 2.1 hold. When n is sufficiently large, the following inequality holds with probability at least $1 - 2\exp\{-cn\epsilon_n^2\}$,

$$\int E_{\boldsymbol{x}}|\mu(\boldsymbol{\beta},\boldsymbol{x}) - \mu^*(\boldsymbol{x})|^2 d\pi(\boldsymbol{\beta}|D_n) \le [16\epsilon_n^2 + 2e^{-cn\epsilon_n^2}]/C_F \asymp [\epsilon_n^2 + e^{-cn\epsilon_n^2}]\bar{L}^2F_n^2. \tag{14}$$

Similarly, if an estimator $\hat{\boldsymbol{\beta}}$ belongs to the dominating posterior mode (refer to the discussion of Theorem 2.5 for more details), then $\hat{\boldsymbol{\beta}} \in \{\boldsymbol{\beta} : d(p_{\boldsymbol{\beta}}, p_{\mu^*}) \le 4\epsilon_n\}$ and the following result hold:

Theorem 2.7. Suppose the conditions of Theorem 2.1 hold, then

$$E_{\boldsymbol{x}}|\mu(\hat{\boldsymbol{\beta}},\boldsymbol{x}) - \mu^*(\boldsymbol{x})|^2 \le [16\epsilon_n^2]/C_F \asymp \epsilon_n^2\bar{L}^2F_n^2. \tag{15}$$

3. Consistent Sparse DNN: Computation

The theoretical results established in previous sections show that the Bayesian sparse DNN can be learned with a mixture Gaussian prior and, more importantly, the posterior inference is not necessarily directly drawn based on posterior samples, which avoids the convergence issue of the MCMC implementation for large complex models. As shown in Theorems 2.3, 2.5, and 2.7, for the sparse BNN, a good local maximizer of the log-posterior distribution also guarantees consistency of the network structure selection and asymptotic optimality of the network generalization performance. This local maximizer, in the spirit of condition C.3 and the conditions of Theorems 2.5 and 2.7, is not necessarily a MAP estimate, as the factor $\det(-\frac{n}{2\pi}H_n(\hat{\boldsymbol{\beta}}))^{\frac{1}{2}}$ can play an important role. In other words, an estimate of $\boldsymbol{\beta}$ lies in a wide valley of the energy landscape is generally preferred. This is consistent with the view of many other authors (see, e.g., Chaudhari et al. 2016; Izmailov et al. 2018), where different techniques have been developed to enhance convergence of SGD to a wide valley of the energy landscape.

Condition C.3 can be re-expressed as $\int_{\mathbb{R}^{K_n}\setminus B_\delta(\hat{\boldsymbol{\beta}})} e^{nh_n(\boldsymbol{\beta})}d\boldsymbol{\beta} = o(\det(-\frac{n}{2\pi}H_n(\hat{\boldsymbol{\beta}}))^{-\frac{1}{2}}e^{nh_n(\hat{\boldsymbol{\beta}})})$, which requires that $\hat{\boldsymbol{\beta}}$ is a dominating mode of the posterior. Based on this observation,

Algorithm 1 Sparse DNN elicitation with Bayesian evidence

Input: T—the number of independent tries in training the DNN, and the prior hyperparameters $\sigma_{0,n}$, $\sigma_{1,n}$, and λ_n.

for $t = 1, 2, \ldots, T$ **do**

(i) *Initialization*: Randomly initialize the weights and biases, set $\boldsymbol{\gamma}_i$=1 for $i = 1, 2, \ldots, K_n$.

(ii) *Optimization*: Run SGD to maximize $h_n(\boldsymbol{\beta})$ as defined in (11). Denote the estimate of $\boldsymbol{\beta}$ by $\hat{\boldsymbol{\beta}}$.

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

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

$$h_n(\boldsymbol{\beta}_{\boldsymbol{\gamma}^t}) = \frac{1}{n}\sum_{i=1}^{n}\log(p(y_i, \boldsymbol{x}_i|\boldsymbol{\beta}_{\boldsymbol{\gamma}^t})) + \frac{1}{n}\log(\pi(\boldsymbol{\beta}_{\boldsymbol{\gamma}^t})), \quad (16)$$

which can be accomplished by running SGD for a few epochs with the initial value $\hat{\boldsymbol{\beta}}_{\boldsymbol{\gamma}^t}$. Denote the resulting DNN model by $\tilde{\boldsymbol{\beta}}_{\boldsymbol{\gamma}^t}$.

(v) *Model evaluation*: Calculate the Bayesian evidence: $Evidence^t = \det(-\frac{n}{2\pi}H_n(\tilde{\boldsymbol{\beta}}_{\boldsymbol{\gamma}^t}))^{-\frac{1}{2}}e^{nh_n(\tilde{\boldsymbol{\beta}}_{\boldsymbol{\gamma}^t})}$, where $H_n(\boldsymbol{\beta}_{\boldsymbol{\gamma}}) = \frac{\partial^2 h_n(\boldsymbol{\beta}_{\boldsymbol{\gamma}})}{\partial\boldsymbol{\beta}_{\boldsymbol{\gamma}}\partial^T\boldsymbol{\beta}_{\boldsymbol{\gamma}}}$ is the Hessian matrix.

end for

Output $\tilde{\boldsymbol{\beta}}_{\boldsymbol{\gamma}^t}$ with the largest Bayesian evidence.

we suggest to use the Bayesian evidence (MacKay 1992; Liang 2005) as the criterion for eliciting estimates of $\boldsymbol{\beta}$ produced by an optimization method in multiple runs with different initializations. The Bayesian evidence is calculated as $\det(-\frac{n}{2\pi}H_n(\hat{\boldsymbol{\beta}}))^{-\frac{1}{2}}e^{nh_n(\hat{\boldsymbol{\beta}})}$. Since Theorem 2.2 ensures only consistency of structure selection but not consistency of parameter estimation, we suggest to refine its nonzero weights by a short optimization process after structure selection. The complete algorithm is summarized in Algorithm 1.

For a large-scale neural network, even if it is sparse, the number of nonzero elements can easily exceed a few thousands or millions, see, for example, the networks considered in Section 4.2. In this case, evaluation of the determinant of the Hessian matrix can be very time consuming. For this reason, we suggest to approximate the log(Bayesian evidence) by $nh_n(\hat{\boldsymbol{\beta}}_{\boldsymbol{\gamma}}) - \frac{1}{2}|\boldsymbol{\gamma}|\log(n)$ with the detailed arguments given in Section 2.5 of the supplementary materials. As explained there, if the prior information imposed on the sparse DNNs is further ignored, then the sparse DNNs can be elicited by BIC.

The main parameters for Algorithm 1 are the prior hyperparameters $\sigma_{0,n}$, $\sigma_{1,n}$, and λ_n. Theorem 2.1 provides theoretical suggestions for the choice of the prior-hyperparameters, see also the proof of Lemma 2.1 for a specific setting for them. Our theory allows $\sigma_{1,n}$ to grow with n from the perspective of data fitting, but in our experience, the magnitude of weights tend to adversely affect the generalization ability of the network. For this reason, we usually set $\sigma_{1,n}$ to a relatively small number such as 0.01 or 0.02, and then tune the values of $\sigma_{0,n}$ and λ_n for the network sparsity as well as the network approximation error. As a trade-off, the resulting network might be a little denser than the ideal one. If it is too dense to satisfy the sparse constraint given in Assumption A.2.2, one might increase the value of $\sigma_{0,n}$ and/or decrease the value of λ_n, and rerun the algorithm to get a sparser structure. This process can be repeated until the constraint is satisfied.

Algorithm 1 employs SGD to optimize the log-posterior of the BNN. Since SGD generally converges to a local optimal solution, the multiple initialization method is used to find a local optimum close to the global one. It is interesting to note that SGD has some nice properties in nonconvex optimization: It works on the convolved (thus smoothed) version of the loss function (Kleinberg, Li, and Yuan 2018) and tends to converge to flat local minimizers which are with very high probability also global minimizers (Zhang et al. 2018). In this article, we set the number of initializations to $T = 10$ as default unless otherwise stated. We note that Algorithm 1 is not very sensitive to the value of T, although a large value of T can generally improve its performance.

For network weight initialization, we adopted the standard method, see Glorot and Bengio (2010) for tanh activation and He et al. (2015) for ReLU activation, which ensures that the variance of the gradient of each layer is of the same order at the beginning of the training process.

4. Numerical Examples

This section demonstrates the performance of the proposed algorithm on synthetic datasets and real datasets.

4.1. Simulated Examples

For all simulated examples, the covariates $x_1, x_2, \ldots, x_p$ were generated in the following procedure: (i) simulate $e, z_1, \ldots, z_{p_n}$ independently from the truncated standard normal distribution on the interval $[-10, 10]$; and (ii) set $x_i = \frac{e+z_i}{\sqrt{2}}$ for $i = 1, 2, \ldots, p_n$. In this way, all the covariates fall into a compact set and are mutually correlated with a correlation coefficient of about 0.5. We generated 10 datasets for each example. Each dataset consisted of $n = 10{,}000$ training samples, 1000 validation samples and 1000 test samples. For comparison, the sparse input neural network (Spinn) (Feng and Simon 2017), dropout (Srivastava et al. 2014) , and dynamic pruning with feedback (DPF) (Lin et al. 2020) methods were also applied to these examples. Note that the validation samples were only used by Spinn, but not by the other methods. The performances of these methods in variable selection or connection selection were measured using the false selection rate (FSR) and the negative selection rate (NSR):

$$\text{FSR} = \frac{\sum_{i=1}^{10}|\hat{S}_i \backslash S|}{\sum_{i=1}^{10}|\hat{S}_i|}, \qquad \text{NSR} = \frac{\sum_{i=1}^{10}|S \backslash \hat{S}_i|}{\sum_{i=1}^{10}|S|},$$

where S is the set of true variables/connections, $\hat{S}_i$ is the set of selected variables/connections for dataset i, and $|\hat{S}_i|$ is the

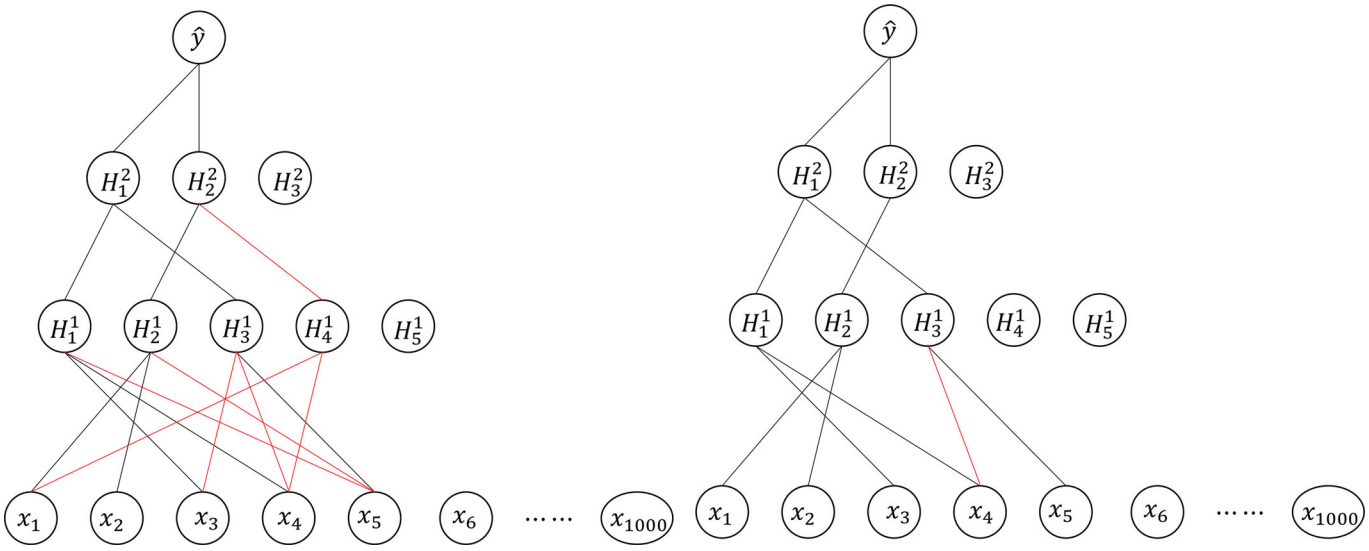

Figure 1. The left and right panels show the network structures selected for one dataset after the stages of training and retraining, respectively, where the black lines show the connections that are selected and exist in the true model, and the red lines show the connections that are selected but do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

size of $\hat{S}_i$. For the regression examples, the prediction and fitting performances of each method were measured by mean square prediction error (MSPE) and mean square fitting error (MSFE), respectively; and for the classification examples, they were measured by prediction accuracy (PA) and fitting accuracy (FA), respectively. To make each method to achieve the best or nearly best performance, we intentionally set the training and nonzero-weight refining process excessively long. In general, this is unnecessary. For example, for the read data example reported in Section 4.2, the nonzero-weight refining process consisted of only one epoch.

4.1.1. Network 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. We fit the data using a neural network with structure 1000-5-3-1 and tanh as the activation function. For each dataset, we ran SGD for 80,000 iterations to train the neural network with a learning rate of $\epsilon_t = 0.01$. The subsample size was set to 500. For the mixture Gaussian prior, we set $\sigma_{1,n} = 0.01$, $\sigma_{0,n} = 0.0005$, and $\lambda_n = 0.00001$. The number of independent tries was set to $T = 10$. After structure selection, the DNN was retrained using SGD for 40,000 iterations. Over the 10 datasets, we got MSFE = 1.030 (0.004) and MSPE = 1.041 (0.014), where the numbers in the parentheses denote the standard deviation of the estimates. This indicates a good approximation of the neural network to the underlying true function. In terms of variable selection, we got perfect results with FSR = 0 and NSR = 0. In terms of structure selection, under the above setting, our method selected a little more connections with FSR = 0.377 and NSR = 0. The left panel of Figure 1 shows a network structure selected for one dataset, which includes a few more connections than the true network. This "redundant" connection selection phenomenon is due to that $\sigma_{1,n} = 0.01$ was too small, which enforces more connections to be included in the network to compensate the effect of the shrunk true connection weights. However, in practice, such an under-biased setting of $\sigma_{1,n}$ is usually preferred from the perspective of neural network training and prediction, which effectively prevents the neural network to include large connection weights. It is known that including large connection weights in the neural network is likely to cause vanishing gradients in training as well as large error in prediction especially when the future observations are beyond the range of training samples. We also note that such a "redundant" connection selection phenomenon can be alleviated by performing another round of structure selection after retraining. In this case, the number of false connections included in the network and their effect on the objective function (16) are small, the true connections can be easily identified even with an under-biased value of $\sigma_{1,n}$. The right panel of Figure 1 shows the network structure selected after retraining, which indicates that the true network structure can be identified almost exactly. Summarizing over the networks selected for the 10 datasets, we had FSR = 0.152 and NSR = 0 after retraining; that is, on average, there were only about 1.5 more connections selected than the true network for each dataset. This result is remarkable!

For comparison, we have applied the sparse input neural network (Spinn) method (Feng and Simon 2017) to this example, where Spinn was run with a LASSO penalty and a regularization parameter of $\lambda = 0.05$. We have tried $\lambda \in \{0.01, 0.02, \ldots, 0.1\}$ and found that $\lambda = 0.05$ generally led to a better structure selection result. In terms of structure selection, Spinn got FSR = 0.221 and NSR = 0.26. Even with another round of structure selection after retraining, Spinn only got FSR = 0.149 and NSR = 0.26. It indicates that Spinn missed some true connections, which is inferior to the proposed method.

4.1.2. Nonlinear Regression

We generated 10 datasets from the following model:

$$y = \frac{5x_2}{1 + x_1^2} + 5\sin(x_3 x_4) + 2x_5 + 0x_6 + \cdots + 0x_{2000} + \varepsilon, \quad (17)$$

Table 1. Comparison of different methods for the simulated nonlinear regression example, where MSFE and MSPE were calculated by averaging over 10 datasets with the standard deviation given in the parentheses.

Activation	Method	$\lvert\hat{S}\rvert$	FSR	NSR	MSFE	MSPE
Tanh	BNN	5(0)	0	0	2.372(0.093)	2.439(0.132)
	Spinn	36.1(15.816)	0.861	0	3.090(0.194)	3.250(0.196)
	DPF	55.6(1.002)	0.910	0	2.934(0.132)	3.225(0.524)
	dropout	–	–	–	10.491(0.078)	13.565(0.214)
ReLU	BNN	5(0)	0	0	2.659(0.098)	2.778(0.111)
	Spinn	136.3(46.102)	0.963	0	3.858(0.243)	4.352(0.171)
	DPF	67.8(1.606)	0.934	0	5.893(0.619)	6.252(0.480)
	dropout	–	–	–	17.279(0.571)	18.630(0.559)

where $\varepsilon \sim N(0,1)$ and is independent of x_i's. We modeled the data by a 3-hidden layer neural network, which has 6, 4, and 3 hidden units on the first, second, and third hidden layers, respectively. The tanh was used as the activation function. For each dataset, we ran SGD for 80,000 iterations to train the neural network with a learning rate of $\epsilon_t = 0.005$. The subsample size was set to 500. For the mixture Gaussian prior, we set $\sigma_{1,n} = 0.01$, $\sigma_{0,n} = 0.0001$, and $\lambda_n = 0.00001$. The number of independent tries was set to $T = 10$. After structure selection, the DNN was retrained using SGD for 40,000 iterations.

For comparison, the Spinn, dropout and DPF methods were also applied to this example with the same DNN structure and the same activation function. For Spinn, the regularization parameter for the weights in the first layer was tuned from the set $\{0.01, 0.02, \ldots, 0.1\}$. For a fair comparison, it was also retrained for 40,000 iterations after structure selection as for the proposed method. For dropout, we set the dropout rate to be 0.2 for the first layer and 0.5 for the other layers. For DPF, we set the target pruning ratio to the ideal value 0.688%, which is the ratio of the number of connections related to true variables and the total number of connections, that is, $(12{,}053 - 1995 \times 6)/12{,}053$ with 12,053 being the total number of connections including the biases. The results were summarized in Table 1.

Table 1 indicates that the proposed BNN method significantly outperforms the Spinn and DPF methods in both prediction and variable selection. For this example, BNN can correctly identify the 5 true variables of the nonlinear regression (17), while Spinn and DPF identified too many false variables. Figure 2 shows the structure of a selected neural network by the BNN method. In terms of prediction, BNN, Spinn and DPF all significantly outperform the dropout method. In our experience, when irrelevant features are present in the data, learning a sparse DNN is always rewarded in prediction.

Figure 3 explores the relationship between Bayesian evidence and prediction accuracy. Since we set the number of tries $T = 10$ for each of the 10 datasets, there are a total of 100 pairs of (Bayesian evidence, prediction error) shown in the plot. The plot shows a strong linear pattern that the prediction error of the sparse neural network decreases as Bayesian evidence increases. This justifies the rationale of Algorithm 1, where Bayesian evidence is employed for eliciting sparse neural network models learned by an optimization method in multiple runs with different initializations.

For this example, we have also compared different methods with the ReLU activation. The same network structure and the same hyperparameter setting were used as in the experiments with the tanh activation. The results are also summarized in Table 1, which indicate that the proposed BNN method still significantly outperforms the competing ones.

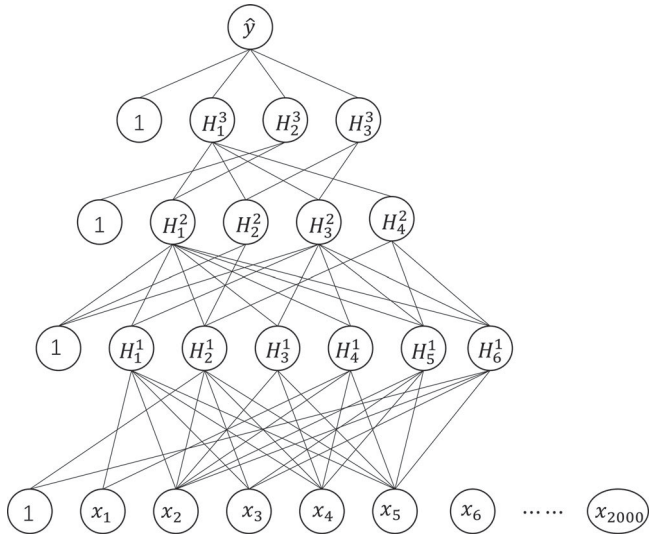

Figure 2. A network structure selected by the Bayesian evidence method for a simulated dataset from model (17), where the node "1" denotes the bias term.

4.1.3. Nonlinear Classification

We generated 10 datasets from the following nonlinear system:

$$y = \begin{cases} 1 & e^{x_1} + x_2^2 + 5\sin(x_3 x_4) \\ & \quad -3 + 0x_5 + \cdots + 0x_{1000} > 0, \\ 0 & \text{otherwise.} \end{cases}$$

Each dataset consisted of half of the observations with the response $y = 1$. We modeled the data by a 3-hidden layer logistic regression neural network, which had 6,4,3 hidden units on the first, second, and third hidden layers, respectively. The tanh was used as the activation function. The proposed method was compared with Spinn, dropout and DPF. For all the methods, the same hyperparameter values were used as in the nonlinear regression example of Section 4.1.2. The results are summarized in Table 2, which shows that the proposed method can identify the true variable for the nonlinear system and make more accurate prediction than Spinn, dropout and DPF. Compared to BNN and Spinn, DPF missed some true variables and produced a larger value of NSR.

4.2. Residual Network Compression

This section assessed the performance of the proposed BNN method on network compression with CIFAR-10 (Krizhevsky and Hinton 2009) used as the illustrative dataset. The CIFAR-10 dataset is a benchmark dataset for computer vision, which consists of 10 classes, 50,000 training images, and 10,000 testing images. We modeled the data using both ResNet20 and ResNet32 (He et al. 2016) and then pruned them to different sparsity levels. We compared the proposed BNN method with DPF (Lin et al. 2020), dynamic sparse reparameterization (DSR) (Mostafa and Wang 2019), sparse momentum (SM) (Dettmers

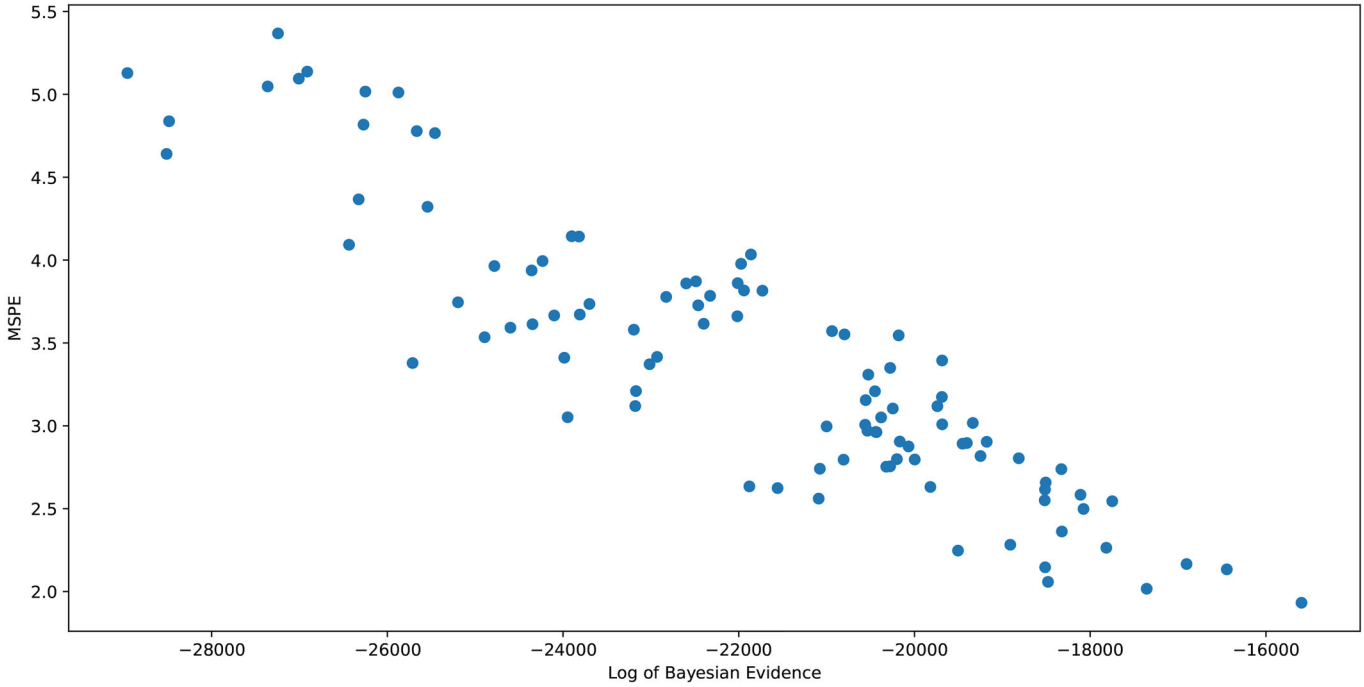

Figure 3. Scatterplot of MSPE versus the logarithm of Bayesian evidence, for which the fitted OLS regression line is $y = -2.459 \times 10^{-4}x - 1.993$ with $R^2 = 0.8037$ and p-value $= 2.2 \times 10^{-16}$.

Table 2. Comparison of different methods for the simulated classification example, where FA and PA were calculated by averaging over 10 datasets with the standard deviation given in the parentheses.

Method	$\lvert\hat{S}\rvert$	FSR	NSR	FA	PA
BNN	4(0)	0	0	0.8999(0.0023)	0.8958(0.0039)
Spinn	4.1(0.09)	0.024	0	0.8628(0.0009)	0.8606(0.0036)
DPF	61.9(0.81)	0.935	0.333	0.8920(0.0081)	0.8697(0.0010)
dropout	–	–	–	0.4898(0.0076)	0.4906(0.0071)

and Zettlemoyer 2019), and variational Bayes (VB) (Blundell et al. 2015). All experiments were implemented using PyTorch (Paszke et al. 2017).

In all of our experiments, we followed the same training setup as used in Lin et al. (2020), that is, the model was trained using SGD with momentum for 300 epochs, the data augmentation strategy (Zhong et al. 2017) was employed, the mini-batch size was set to 128, the momentum parameter was set to 0.9, and the initial learning rate was set to 0.1. We divided the learning rate by 10 at epoch 150 and 225. For the proposed method, we set the number of independent trials $T = 10$, and used BIC to elicit sparse networks. In each trial, the mixture normal prior was imposed on the network weights after 150 epochs. After pruning, the model was retrained for one epoch for refining the nonzero weights. For the mixture normal prior, we set $\sigma_{1,n}^2 = 0.02$ and tried different values for $\sigma_{0,n}$ and λ_n to achieve different sparsity levels. For ResNet-20, to achieve 10% target sparsity, we set $\sigma_{0,n}^2 = 4e-5$ and $\lambda_n = 1e-6$; to achieve 20% target sparsity, we set $\sigma_{0,n}^2 = 6e-6$ and $\lambda_n = 1e-7$. For ResNet-32, to achieve 5% target sparsity, we set $\sigma_{0,n}^2 = 6e-5$ and $\lambda_n = 1e-7$; to achieve 10% target sparsity, we set $\sigma_{0,n}^2 = 2e-5$ and $\lambda_n = 1e-5$.

Following the experimental setup in Lin et al. (2020), all experiments were run for 3 times and the averaged test accuracy and standard deviation were reported. For the VB method, we followed Blundell et al. (2015) to impose a mixture Gaussian prior (the same prior as used in our method) on the connection weights, and employed a diagonal multivariate Gaussian distribution to approximate the posterior. As in Blundell et al. (2015), we ordered the connection weights in the signal-to-noise ratio $\frac{|\mu|}{\sigma}$, and identified a sparse structure by removing the weights with a low signal-to-noise ratio. The results were summarized in Table 3, where the results of other baseline methods were taken from Lin et al. (2020). The comparison indicates that the proposed method is able to produce better prediction accuracy than the existing methods at about the same level of sparsity, and that the VB method is not very competitive in statistical inference although it is very attractive in computation. Note that the proposed method provides a one-shot pruning strategy. As discussed in Han et al. (2015), it is expected that these results can be further improved with appropriately tuned hyperparameters and iterative pruning and retraining.

The CIFAR-10 has been used as a benchmark example in many DNN compression experiments. Other than the competing methods considered above, the targeted dropout method (Gomez et al. 2019) reported a Resnet32 model with 47K parameters (90% sparsity) and the prediction accuracy 91.48%. The Bayesian compression method with a group normal-Jeffreys' prior (BC-GNJ) (Louizos, Ullrich, and Welling 2017) reported a VGG16 model, a very deep convolutional neural network model proposed by Simonyan and Zisserman (2014), with 9.2M parameters (93.3% sparsity) and the prediction accuracy 91.4%. A comparison with our results reported in Table 3 indicates again the superiority of the proposed BNN method.

Finally, we note that the proposed BNN method belongs to the class of pruning methods and it provides an effective way for learning sparse DNNs. Contemporary experience shows that

Table 3. Network compression for CIFAR-10 data, where the number in the parentheses denotes the standard deviation of the respective estimate.

	ResNet-20		ResNet-32	
Method	Pruning ratio	Test accuracy	Pruning ratio	Test accuracy
BNN	19.673%(0.054%)	92.27(0.03)	9.531%(0.043%)	92.74(0.07)
SM	20%	91.54(0.16)	10%	91.54(0.18)
DSR	20%	91.78(0.28)	10%	91.41(0.23)
DPF	20%	92.17(0.21)	10%	92.42(0.18)
VB	20%	90.20(0.04)	10%	90.11(0.06)
BNN	9.546%(0.029%)	91.27(0.05)	4.783%(0.013%)	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)
VB	10%	89.33(0.16)	5%	88.14(0.04)

directly training a sparse or small dense network from the start typically converges slower than training with a pruning method (see, e.g., Frankle and Carbin 2018). This issue can be illustrated using a network compression example. Three experiments were conducted for a ResNe20 (with 10% sparsity level) on the CIFAR 10 dataset: (a) sparse BNN, that is, running the proposed BNN method with randomly initialized weights; (b) starting with sparse network, that is, training the sparse network learned by the proposed BNN method but with the weights randomly reinitialized; and (c) starting with small dense network, that is, training a network whose number of parameters in each layer is about the same as that of the sparse network in experiment (b) and whose weights are randomly initialized. The experiment (a) consisted of 400 epochs, where the last 100 epochs were used for refining the nonzero-weights of the sparse network obtained at epoch 300 via *connection sparsification*. Both the experiments (b) and (c) consisted of 300 epochs. In each of the experiments, the learning rate was set in the standard scheme (Lin et al. 2020), that is, started with 0.1 and then decreased by a factor of 10 at epochs 150 and 225, respectively. For random initialization, we used the default method in PyTorch (He et al. 2015). For example, for a two-dimensional convolutional layer with n_{in} input feature map channels, n_{out} output feature map channels, and a convolutional kernel of size $w \times h$, the weights and bias of the layer were initialized by independent draws from the uniform distribution $\text{Unif}(-\frac{1}{\sqrt{n_{\text{in}} \times w \times h}}, \frac{1}{\sqrt{n_{\text{in}} \times w \times h}})$.

Figure 4 shows the training and testing paths obtained in the three experiments. It indicates that the sparse neural network learned by the proposed method significantly outperforms the other two networks trained with randomly initialized weights. This result is consistent with the finding of Frankle and Carbin (2018) that the architectures uncovered by pruning are harder to train from the start and they often reach lower accuracy than the original neural networks.

Regarding this experiment, we have further two remarks. First, the nonzero weights refining step in Algorithm 1 can consist of a few epochs only. This step is mainly designed for a theoretical purpose, ensuring the followed evidence evaluation to be done on a local mode of the posterior. In terms of sparse neural network learning, the mixture Gaussian prior plays a key role in sparsifying the neural network, while the nonzero weights refining step is not essential. As illustrated by Figure 4, this step did not significantly improve the training and testing errors of the sparse neural network. Second, as mentioned previously, the proposed BNN method falls into the class of pruning methods suggested by Frankle and Carbin (2018) for learning

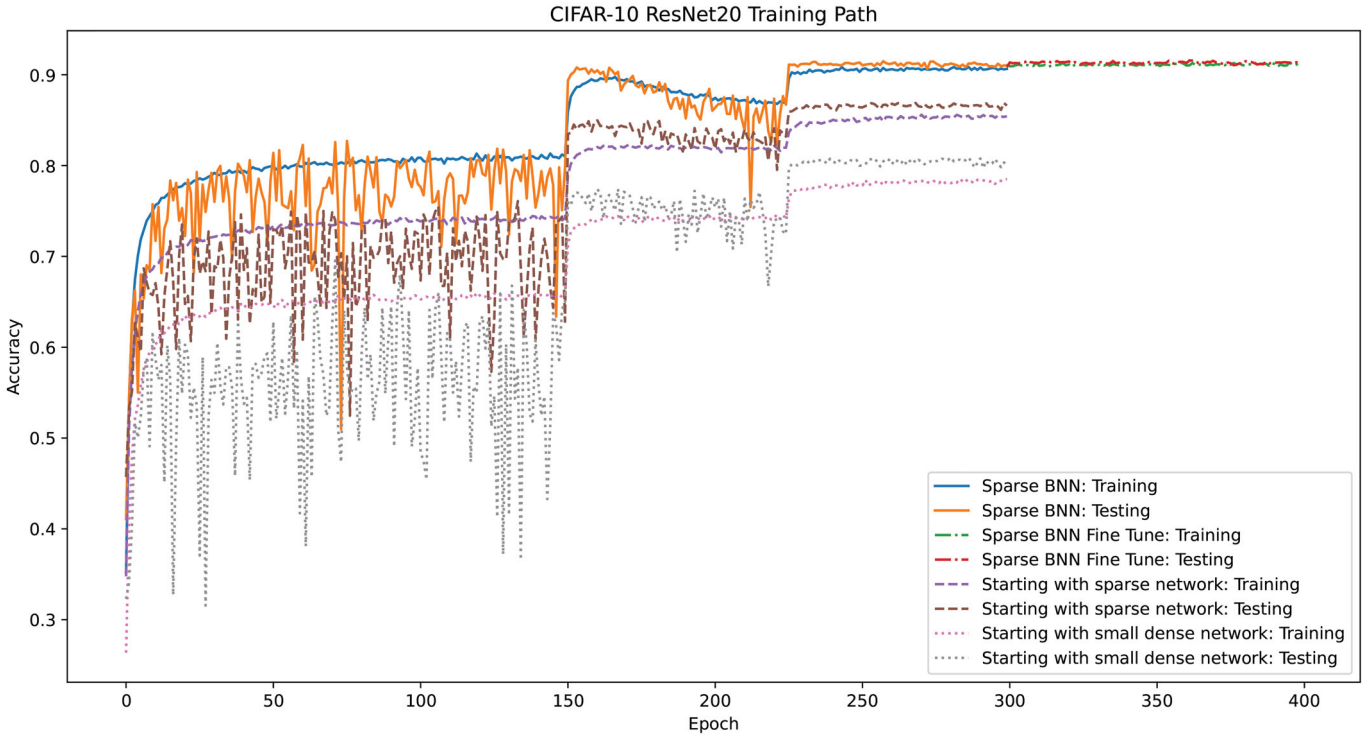

Figure 4. Training and testing paths of a ResNet20 model (with 10% sparsity level) on the CIFAR-10 dataset: "Sparse BNN: Training" is the path for fitting the full neural network in experiment (a) (for epochs 1–300); "Sparse BNN Fine Tune: Training" is the path for refining the nonzero-weights of the sparse network obtained in experiment (a) via *connection sparsification* (for epochs 301–400); "Starting with sparse network: Training" is the path for fitting a sparse neural network in experiment (b); and "Starting with small dense network: Training" is the path for fitting a small dense neural network in experiment (c). The curves for testing can be interpreted similarly.

sparse DNNs. Compared to the existing pruning methods, the proposed method is more theoretically sound, which ensures the resulting sparse network to possess nice theoretical properties such as posterior consistency, variable selection consistency and asymptotically optimal generalization bounds.

5. Discussion

This article provides a complete treatment for sparse DNNs in both theory and computation. 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 could be learned via sparsifying an over-parameterized one. Such a sparse neural network has nice theoretical properties, such as posterior consistency, variable selection consistency, and asymptotically optimal generalization bound.

In computation, we proposed to use Bayesian evidence or BIC for eliciting sparse DNN models learned by an optimization method in multiple runs with different initializations. Since conventional optimization methods such as SGD and Adam (Kingma and Ba 2014) can be used to train the DNNs, 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 and high-dimensional nonlinear variable selection. The networks learned by the proposed method tend to predict better than the existing methods.

Regarding the number of runs of the optimization method, that is, the value of T, in Algorithm 1, we would note again that Algorithm 1 is not very sensitive to it. For example, for the nonlinear regression example, Algorithm 1 correctly identified the true variables in each of the 10 runs. For the CIFAR-10 example with ResNet20 and 10% target sparsity level, Algorithm 1 achieved the test accuracies in 10 runs: 91.09%, 91.16%, 91.17%, 91.18%, 91.18%, 91.24%, 91.24%, 91.24%, 91.27%, 91.31% (in ascending order), where the worst one is still better than those achieved by the baseline methods.

In this work, we choose the two-mixture Gaussian prior for the weights and biases of the DNN, mainly for the sake of computational convenience. Other choices, such as two-mixture Laplace prior (Ročková 2018), which will lead to the same posterior contraction with an appropriate choice for the prior hyperparameters. To be more specific, Theorem S1 (in the supplementary materials) establishes sufficient conditions that guarantee the posterior consistency, and any prior distribution satisfying the sufficient conditions can yield consistent posterior inferences for the DNN.

Beyond the absolutely continuous prior, the hierarchical prior used in Liang, Li, and Zhou (2018) and Polson and Ročková (2018) can be adopted for DNNs. To be more precise, one can assume that

$$\boldsymbol{\beta}_{\boldsymbol{\gamma}}|\boldsymbol{\gamma} \sim N(0, \sigma_{1,n}^2 \boldsymbol{I}_{|\boldsymbol{\gamma}|\times|\boldsymbol{\gamma}|}), \quad \boldsymbol{\beta}_{\boldsymbol{\gamma}^c} = 0; \tag{18}$$

$$\pi(\boldsymbol{\gamma}) \propto \lambda_n^{|\boldsymbol{\gamma}|} (1-\lambda_n)^{K_n - |\boldsymbol{\gamma}|} \mathbf{1}\{1 \le |\boldsymbol{\gamma}| \le \bar{r}_n, \boldsymbol{\gamma} \in \mathcal{G}\}, \tag{19}$$

where $\boldsymbol{\beta}_{\boldsymbol{\gamma}^c}$ is the complement of $\boldsymbol{\beta}_{\boldsymbol{\gamma}}$, $|\boldsymbol{\gamma}|$ is the number of nonzero elements of $\boldsymbol{\gamma}$, $\boldsymbol{I}_{|\boldsymbol{\gamma}|\times|\boldsymbol{\gamma}|}$ is a $|\boldsymbol{\gamma}| \times |\boldsymbol{\gamma}|$ identity matrix, $\bar{r}_n$ is the maximally allowed size of candidate networks, $\mathcal{G}$ is the set of valid DNNs, and the hyperparameter λ_n, as in (4), can be read as an approximate prior probability for each connection or bias to be included in the DNN. Under this prior, the product of the weight or bias and its indicator follows a discrete spike-and-slab prior distribution, that is,

$$\begin{aligned} \boldsymbol{w}_{ij}^h \boldsymbol{\gamma}_{ij}^{\boldsymbol{w}^h} | \boldsymbol{\gamma}_{ij}^{\boldsymbol{w}^h} &\sim \boldsymbol{\gamma}_{ij}^{\boldsymbol{w}^h} N(0, \sigma_{1,n}^2) + (1 - \boldsymbol{\gamma}_{ij}^{\boldsymbol{w}^h})\delta_0, \\ \boldsymbol{b}_k^h \boldsymbol{\gamma}_k^{\boldsymbol{b}^h} | \boldsymbol{\gamma}_k^{\boldsymbol{b}^h} &\sim \boldsymbol{\gamma}_k^{\boldsymbol{b}^h} N(0, \sigma_{1,n}^2) + (1 - \boldsymbol{\gamma}_k^{\boldsymbol{b}^h})\delta_0, \end{aligned}$$

where δ_0 denotes the Dirac delta function. Under this hierarchical prior, it is not difficult to show that the posterior consistency and structure selection consistency theory developed in this article still hold. However, from the computational perspective, the hierarchical prior might be inferior to the mixture Gaussian prior adopted in the article, as the posterior $\pi(\boldsymbol{\beta}_{\boldsymbol{\gamma}}, \boldsymbol{\gamma}|D_n)$ is hard to be optimized or simulated from. It is known that directly simulating from $\pi(\boldsymbol{\beta}_{\boldsymbol{\gamma}}, \boldsymbol{\gamma}|D_n)$ using an acceptance-rejection based MCMC algorithm can be time consuming. A feasible way is to formulate the prior of $\boldsymbol{\beta}_{\boldsymbol{\gamma}}$ as $\boldsymbol{\beta}_{\boldsymbol{\gamma}} = \boldsymbol{\theta} \otimes \boldsymbol{\gamma}$, where $\boldsymbol{\theta} \sim N(0, \sigma_{1,n}^2 I_{H_n \times H_n})$ can be viewed as a latent variable and $\otimes$ denotes entry-wise product. Then one can first simulate from the marginal posterior $\pi(\boldsymbol{\theta}|D_n)$ using a stochastic gradient MCMC algorithm and then make inference of the network structure based on the conditional posterior $\pi(\boldsymbol{\gamma}|\boldsymbol{\theta}, D_n)$. We note that the gradient $\nabla_{\boldsymbol{\theta}} \log \pi(\boldsymbol{\theta}|D^n)$ can be approximated based on the following identity developed in Song et al. (2020),

$$\nabla_{\boldsymbol{\theta}} \log \pi(\boldsymbol{\theta}|D^n) = \sum_{\boldsymbol{\gamma}} \pi(\boldsymbol{\gamma}|\boldsymbol{\theta}, D^n) \nabla_{\boldsymbol{\theta}} \log \pi(\boldsymbol{\theta}|\boldsymbol{\gamma}, D^n),$$

where D_n can be replaced by a dataset duplicated with mini-batch samples if the subsampling strategy is used to accelerate the simulation. This identity greatly facilitates the simulations for the dimension jumping problems, which requires only some samples to be drawn from the conditional posterior $\pi(\boldsymbol{\gamma}|\boldsymbol{\theta}, D^n)$ for approximating the gradient $\nabla_{\boldsymbol{\theta}} \log \pi(\boldsymbol{\theta}|D^n)$ at each iteration. A further exploration of this discrete prior for its use in deep learning is of great interest, although there are some difficulties needing to be addressed in computation.

Supplementary Materials

Supplement description: (i) proofs of Theorems 2.1–2.7 and Lemma 2.1; (ii) verification of the bounded gradient in Theorem 2.3; (iii) approximation of Bayesian evidence; and (iv) some mathematical facts of the sparse DNN.

Acknowledgments

The authors thank the editor, associate editors and two referees for their constructive comments which have led to significant improvement of this article.

Funding

Liang's research was supported in part by the grants DMS-2015498, NIH R01-GM117597 and NIH R01-GM126089. Song's research was supported in part by the grant DMS-1811812.

References

Alvarez, J. M., and Salzmann, M. (2016), "Learning the Number of Neurons in Deep Networks," in *Advances in Neural Information Processing Systems*, pp. 2270–2278. [1]

Bauler, B., and Kohler, M. (2019), "On Deep Learning as a Remedy for the Curse of Dimensionality in Nonparametric Regression," *The Annals of Statistics*, 47, 2261–2285. [1]

Blundell, C., Cornebise, J., Kavukcuoglu, K., and Wierstra, D. (2015), "Weight Uncertainty in Neural Networks," in *Proceedings of the 32nd International Conference on Machine Learning, ICML'15* (Vol. 37), JMLR.org, pp. 1613–1622. [2,11]

Bölcskei, H., Grohs, P., Kutyniok, G., and Petersen, P. (2019), "Optimal Approximation With Sparsely Connected Deep Neural Networks," *SIAM Journal on Mathematics of Data Science*, 1, 8–45. [1,2,3]

Chaudhari, P., Choromanska, A., Soatto, S., LeCun, Y., Baldassi, C., Borgs, C., Chayes, J., Sagun, L., and Zecchina, R. (2016), "Entropy-SGD: Biasing Gradient Descent Into Wide Valleys," arXiv no. 1611.01838. [7]

Cheng, Y., Yu, F. X., Feris, R. S., Kumar, S., Choudhary, A. N., and Chang, S.-F. (2015), "An Exploration of Parameter Redundancy in Deep Networks With Circulant Projections," in *2015 IEEE International Conference on Computer Vision (ICCV)*, pp. 2857–2865. [1]

Denil, M., Shakibi, B., Dinh, L., Ranzato, M., and de Freitas, N. (2013), "Predicting Parameters in Deep Learning," in *NIPS*. [1]

Dettmers, T., and Zettlemoyer, L. (2019), "Sparse Networks From Scratch: Faster Training Without Losing Performance," arXiv no. 1907.04840. [11]

Dobra, A., Hans, C., Jones, B., Nevins, J. R., Yao, G., and West, M. (2004), "Sparse Graphical Models for Exploring Gene Expression Data," *Journal of Multivariate Analysis*, 90, 196–212. [4]

Feng, J., and Simon, N. (2017), "Sparse-Input Neural Networks for High-Dimensional Nonparametric Regression and Classification," arXiv no. 1711.07592. [8,9]

Frankle, J., and Carbin, M. (2018), "The Lottery Ticket Hypothesis: Finding Sparse, Trainable Neural Networks," arXiv no. 1803.03635. [2,12]

George, E. I., and McCulloch, R. E. (1993), "Variable Selection via Gibbs Sampling," *Journal of the American Statistical Association*, 88, 881–889. [3]

——— (1997), "Approaches for Bayesian Variable Selection," *Statistica Sinica*, 7, 339–373. [4]

Ghosal, S., Ghosh, J. K., and Van Der Vaart, A. W. (2000), "Convergence Rates of Posterior Distributions," *The Annals of Statistics*, 28, 500–531. [4]

Ghosh, S., and Doshi-Velez, F. (2017), "Model Selection in Bayesian Neural Networks via Horseshoe Priors," arXiv no. 1705.10388. [2]

Glorot, X., and Bengio, Y. (2010), "Understanding the Difficulty of Training Deep Feedforward Neural Networks," in *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, pp. 249–256. [8]

Glorot, X., Bordes, A., and Bengio, Y. (2011), "Deep Sparse Rectifier Neural Networks," in *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, pp. 315–323. [1]

Gomez, A. N., Zhang, I., Kamalakara, S. R., Madaan, D., Swersky, K., Gal, Y., and Hinton, G. E. (2019), "Learning Sparse Networks Using Targeted Dropout," arXiv no. 1905.13678. [11]

Guo, C., Pleiss, G., Sun, Y., and Weinberger, K. Q. (2017), "On Calibration of Modern Neural Networks," in *Proceedings of the 34th International Conference on Machine Learning, ICML'17* (Vol. 70), JMLR.org, pp. 1321–1330. [1,4]

Han, S., Mao, H., and Dally, W. J. (2015), "Deep Compression: Compressing Deep Neural Networks With Pruning, Trained Quantization and Huffman Coding," arXiv no. 1510.00149. [1]

Han, S., Pool, J., Tran, J., and Dally, W. (2015), "Learning Both Weights and Connections for Efficient Neural Network," in *Advances in Neural Information Processing Systems*, pp. 1135–1143. [11]

He, K., Zhang, X., Ren, S., and Sun, J. (2015), "Delving Deep Into Rectifiers: Surpassing Human-Level Performance on imagenet Classification," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1026–1034. [8,12]

——— (2016), "Deep Residual Learning for Image Recognition," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778. [1,10]

Ishwaran, H., and Rao, J. S. (2005), "Spike and Slab Variable Selection: Frequentist and Bayesian Strategies," *The Annals of Statistics*, 33, 730–773. [3]

Izmailov, P., Podoprikhin, D., Garipov, T., Vetrov, D., and Wilson, A. G. (2018), "Averaging Weights Leads to Wider Optima and Better Generalization," arXiv no. 1803.05407. [7]

Jiang, W. (2007), "Bayesian Variable Selection for High Dimensional Generalized Linear Models: Convergence Rate of the Fitted Densities," *The Annals of Statistics*, 35, 1487–1511. [3,4]

Jordan, M., Ghahramani, Z., Jaakkola, T., and Saul, L. (1999), "Introduction to Variational Methods for Graphical Models," *Machine Learning*, 37, 183–233. [2]

Kass, R., Tierney, L., and Kadane, J. (1990), "The Validity of Posterior Expansions Based on Laplace's Method," in *Bayesian and Likelihood Methods in Statistics and Econometrics*, eds. S. Geisser, J. Hodges, S. Press, and A. Zellner, Amsterdam: North-Holland (Elsevier Science Publisher B.V.), pp. 473–488. [6]

Kingma, D. P., and Ba, J. (2014), "Adam: A Method for Stochastic Optimization," arXiv no. 1412.6980. [13]

Kleinberg, R., Li, Y., and Yuan, Y. (2018), "An Alternative View: When Does SGD Escape Local Minima?" in *Proceedings of the 35th International Conference on Machine Learning, ICML'18* (Vol. 70), JMLR.org. [8]

Kohn, R., Smith, M., and Chan, D. (2001), "Nonparametric Regression Using Linear Combinations of Basis Functions," *Statistics and Computing*, 11, 313–322. [4]

Krizhevsky, A. and Hinton, G. (2009), "Learning Multiple Layers of Features From Tiny Images," Tech. Rep., Citeseer. [10]

Liang, F. (2005), "Evidence Evaluation for Bayesian Neural Networks Using Contour Monte Carlo," *Neural Computation*, 17, 1385–1410. [8]

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. [1,2,4,13]

Liang, F., Song, Q., and Yu, K. (2013), "Bayesian Subset Modeling for High Dimensional Generalized Linear Models," *Journal of the American Statistical Association*, 108, 589–606. [3,5]

Lin, T., Stich, S. U., Barba, L., Dmitriev, D., and Jaggi, M. (2020), "Dynamic Model Pruning With Feedback," in *International Conference on Learning Representations (ICLR)*. [8,10,11,12]

Liu, B., Wang, M., Foroosh, H., Tappen, M., and Pensky, M. (2015), "Sparse Convolutional Neural Networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 806–814. [2]

Louizos, C., Ullrich, K., and Welling, M. (2017), "Bayesian Compression for Deep Learning," in *Advances in Neural Information Processing Systems*, pp. 3288–3298. [11]

Ma, R., Miao, J., Niu, L., and Zhang, P. (2019), "Transformed l_1 Regularization for Learning Sparse Deep Neural Networks," arXiv no. 1901.01021v1. [1]

MacKay, D. J. (1992), "The Evidence Framework Applied to Classification Networks," *Neural Computation*, 4, 720–736. [8]

McAllester, D. (1999a), "PAC-Bayesian Model Averaging," in *Proceedings of the 12th Annual Conference on Computational Learning Theory*, pp. 164–170. [6]

——— (1999b), "Some PAC-Bayesian Theorems," *Machine Learning*, 37, 335–363. [6]

Mhaskar, H., Liao, Q., and Poggio, T. (2017), "When and Why Are Deep Networks Better Than Shallow Ones?" in *Thirty-First AAAI Conference on Artificial Intelligence*. [1]

Mnih, A., and Gregor, K. (2014), "Neural Variational Inference and Learning in Belief Networks," in *Proceedings of the 31st International Conference on International Conference on Machine Learning, ICML'14* (Vol. 32), JMLR.org, pp. II-1791–II-1799. [2]

Mocanu, D. C., Mocanu, E., Stone, P., Nguyen, P. H., Gibescu, M., and Liotta, A. (2018), "Scalable Training of Artificial Neural Networks With Adaptive Sparse Connectivity Inspired by Network Science," *Nature Communications*, 9, 2383. [1]

Montufar, G. F., Pascanu, R., Cho, K., and Bengio, Y. (2014), "On the Number of Linear Regions of Deep Neural Networks," in *Advances in Neural Information Processing Systems*, pp. 2924–2932. [1]

Mostafa, H., and Wang, X. (2019), "Parameter Efficient Training of Deep Convolutional Neural Networks by Dynamic Sparse Reparameterization," arXiv no. 1902.05967. [10]

Nakkiran, P., Kaplun, G., Bansal, Y., Yang, T., Barak, B., and Sutskever, I. (2020), "Deep Double Descent: Where Bigger Models and More Data Hurt," in *International Conference on Learning Representations*. [4]

Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L., and Lerer, A. (2017), "Automatic Differentiation in PyTorch," in *NIPS 2017 Autodiff Workshop: The Future of Gradient-based Machine Learning Software and Techniques*, Long Beach, CA. [11]

Petersen, P., and Voigtlaender, F. (2018), "Optimal Approximation of Piecewise Smooth Functions Using Deep ReLU Neural Networks," *Neural Networks*, 108, 296–330. [3]

Polson, N. G., and Ročková, V. (2018), "Posterior Concentration for Sparse Deep Learning," in *Proceedings of the 32nd International Conferences on Neural Information Processing Systems (NeurIPS)*. [1,2,3,13]

Pourzanjani, A. A., Jiang, R. M., and Petzold, L. R. (2017), "Improving the Identifiability of Neural Networks for Bayesian Inference," in *NIPS Workshop on Bayesian Deep Learning*. [4]

Ročková, V. (2018), "Bayesian Estimation of Sparse Signals With a Continuous Spike-and-Slab Prior," *The Annals of Statistics*, 46, 401–437. [13]

Scardapane, S., Comminiello, D., Hussain, A., and Uncini, A. (2017), "Group Sparse Regularization for Deep Neural Networks," *Neurocomputing*, 241, 81–89. [1]

Schmidt-Hieber, J. (2017), "Nonparametric Regression Using Deep Neural Networks With ReLU Activation Function," arXiv no. 1708.06633v2. [1,3]

Simonyan, K., and Zisserman, A. (2014), "Very Deep Convolutional Networks for Large-Scale Image Recognition," arXiv no. 1409.1556. [11]

Song, Q., and Liang, F. (2017), "Nearly Optimal Bayesian Shrinkage for High Dimensional Regression," arXiv no. 1712.08964. [3]

Song, Q., Sun, Y., Ye, M., and Liang, F. (2020), "Extended Stochastic Gradient MCMC Algorithms for Large-Scale Bayesian Computing," arXiv no. 2002.02919v1. [13]

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. [2]

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. (2014), "Dropout: A Simple Way to Prevent Neural Networks From Overfitting," *The Journal of Machine Learning Research*, 15, 1929–1958. [8]

Telgarsky, M. (2017), "Neural Networks and Rational Functions," arXiv no. 1706.03301. [1]

Wager, S., Wang, S., and Liang, P. S. (2013), "Dropout Training as Adaptive Regularization," in *Advances in Neural Information Processing Systems (NIPS)*, pp. 351–359. [1]

Yarotsky, D. (2017), "Error Bounds for Approximations With Deep ReLU Networks," *Neural Networks*, 94, 103–114. [1]

Yoon, J., and Hwang, S. J. (2017), "Combined Group and Exclusive Sparsity for Deep Neural Networks," in *Proceedings of the 34th International Conference on Machine Learning*, Proceedings of Machine Learning Research (Vol. 70), eds. D. Precup and Y. W. Teh, International Convention Centre, Sydney, Australia, pp. 3958–3966. [1]

Zhang, C., Liao, Q., Rakhlin, A., Miranda, B., Golowich, N., and Poggio, T. (2018), "Theory of Deep Learning IIb: Optimization Properties of SGD," arXiv no. 1801.02254. [8]

Zhong, Z., Zheng, L., Kang, G., Li, S., and Yang, Y. (2017), "Random Erasing Data Augmentation," arXiv no. 1708.04896. [11]