Masked Bayesian Neural Networks : Theoretical Guarantee and its Posterior Inference

Insung Kong [1] **Dongyoon Yang** [1] **Jongjin Lee** [1] **Ilsang Ohn** [2] **Gyuseung Baek** [3] **Yongdai Kim** [1]

Abstract

Bayesian approaches for learning deep neural networks (BNN) have been received much attention and successfully applied to various applications. Particularly, BNNs have the merit of having better generalization ability as well as better uncertainty quantification. For the success of BNN, search an appropriate architecture of the neural networks is an important task, and various algorithms to find good sparse neural networks have been proposed. In this paper, we propose a new node-sparse BNN model which has good theoretical properties and is computationally feasible. We prove that the posterior concentration rate to the true model is near minimax optimal and adaptive to the smoothness of the true model. In particular the adaptiveness is the first of its kind for node-sparse BNNs. In addition, we develop a novel MCMC algorithm which makes the Bayesian inference of the node-sparse BNN model feasible in practice.

1. Introduction

Bayesian approaches for learning deep neural networks (DNN), which is called Bayesian Neural Networks (BNN) (MacKay, 1992; Neal, 2012), have been received much attention and successfully applied to various applications. Particularly, BNNs have the merit of having better generalization ability as well as better uncertainty quantification (Wilson & Izmailov, 2020; Izmailov et al., 2021). Applications of BNNs range from recommender systems (Wang et al., 2015) to topic modeling (Gan et al., 2015), medical diagnosis (Filos et al., 2019), and astrophysics (Cranmer et al., 2021), to name just a few.

Various BNN models to have desirable theoretical properties have been proposed. In particular, edge-sparse BNNs are the main focus of research of BNNs. Fast posterior concentrate rates of edge-sparse BNNs have been studied by (Polson & Ročková, 2018; Chérief-Abdellatif, 2020; Bai et al., 2020; Lee & Lee, 2022). Particularly, Polson & Ročková (2018) uses Spike-and-Slab prior on each edge and prove the near minimax concentrate rate of the posterior, but posterior computation is difficult. To ease computation, Molchanov et al. (2017) uses variational dropout (Kingma et al., 2015) to search good edge-sparse BNNs and Deng et al. (2019) and Wang et al. (2021) develop edge-sparse learning algorithms via adaptive empirical Bayesian methods. Nalisnick et al. (2019) shows that multiplicative noise induces structured shrinkage priors on a network's weights. However, theoretical justifications of these algorithms lack.

Node-sparse BNNs are useful alternatives to edge-sparse BNNs because their inferential costs are lighter than edge-sparse BNNs. Using scale-mixture priors, Louizos et al. (2017) and Ghosh et al. (2019) develop node-sparse BNNs based on the VI, but theoretical justifications of these algorithms are not available. Jantre et al. (2021) derives the posterior concentration rate of a node-sparse VI approach using Spike-and-Slab prior. However, their result does not meet the minimax optimal rate and is not adaptive to the smoothness of the true model.

In this paper, we propose a new node-sparse BNN model called the masked BNN (mBNN) which has good theoretical properties and is computationally feasible. We prove that the posterior concentration rate to the true model is near minimax optimal and adaptive to the smoothness of the true model. The adaptiveness, which is the first of its kind for node-sparse BNNs, means that the mBNN selects optimal sparse architectures without knowing the complexity of the true model. Moreover, we develop a novel MCMC algorithm which makes the Bayesian inference of the node-sparse BNN model possible.

Our proposed node-sparse BNN can be also used for compressing complex DNNs. Mots recent DNNs are based on complex network architectures consisting of multiple nonlinear hidden layers, which results in expensive computation costs and requirements of excessive storage capacities when they are deployed to application systems. Various methodologies to alleviate memory and computation costs have

[1]Department of Statistics, Seoul National University [2]Department of Statistics, Inha University [3]Obzen. Correspondence to: Yongdai Kim <ydkim0903@gmail.com>.

Proceedings of the 40^{th} International Conference on Machine Learning, Honolulu, Hawaii, USA. PMLR 202, 2023.

been suggested. A popular approach is to prune iteratively unnecessary edges (Han et al., 2015; Frankle & Carbin, 2018) or unnecessary nodes (Wen et al., 2016; He et al., 2017; Wang et al., 2019; Chin et al., 2020). However, such algorithms do not provide proper uncertainty quantification. By analyzing image data with CNNs, we illustrate that our node-sparse BNN is good at compressing DNNs without hampering the ability of uncertainty quantification.

Our contributions are summarized as follows:

- We develop a node-sparse prior for DNNs such that the posterior concentration rate to the true model is near minimax optimal adaptively to the smoothness of the true model and Bayesian inference with a specially designed MCMC algorithm is possible.
- We implement an efficient MCMC algorithm for searching good node-sparse BNNs. In particular, we develop a local informed proposal distribution in the Metropolis-Hastings (MH) algorithm to search good node-sparse architectures efficiently.
- By numerical experiments, we illustrate that the mBNN outperforms other Bayesian approaches including nonsparse BNN and VI algorithms in terms of generalization and uncertainty quantification.

2. Preliminaries

2.1. Notation

Let $\mathbb{R}$ and $\mathbb{N}$ be the sets of real numbers and natural numbers, respectively. For an integer $n \in \mathbb{N}$, we denote $[n] := \{1, \dots, n\}$. A capital letter denotes a random variable or matrix interchangeably whenever its meaning is clear, and a vector is denoted by a bold letter, e.g. $\boldsymbol{x} := (x_1, \dots, x_d)^\top$. For a d-dimensional vector $\boldsymbol{x} \in \mathbb{R}^d$, we denote $|\boldsymbol{x}|_p := (\sum_{j=1}^d |x_j|^p)^{1/p}$ for $1 \le p < \infty$, $|\boldsymbol{x}|_0 := \sum_{j=1}^d \mathbb{I}(x_j \neq 0)$ and $|\boldsymbol{x}|_\infty := \max_{j\in[d]} |x_j|$. For a real-valued function $f : \mathcal{X} \to \mathbb{R}$ and $1 \le p < \infty$, we denote $||f||_{p,n} := (\sum_{i=1}^n f(\boldsymbol{x}_i)^p/n)^{1/p}$ and $||f||_{p,\mathrm{P}_{\boldsymbol{X}}} := \left(\int_{\boldsymbol{X}\in\mathcal{X}} f(\boldsymbol{X})^p d\mathrm{P}_{\boldsymbol{X}}\right)^{1/p}$ where $\mathrm{P}_{\boldsymbol{X}}$ is a probability measure defined on input space $\mathcal{X}$. Moreover, we define $||f||_\infty = \sup_{\boldsymbol{x}\in\mathcal{X}} |f(\boldsymbol{x})|$. For $b_1 \le b_2$, we define $f_{[b_1,b_2]}(\cdot) := \min(\max(f(\cdot), b_1), b_2)$ which is a truncated version of f on b_1 and b_2. We denote $\circ$ and $\odot$ as the composition of functions and element-wise product of vectors or matrices, respectively. For a probability vector $\boldsymbol{q} \in \mathbb{R}^r$, we denote $\mathrm{Cat}(\boldsymbol{q})$ as the categorical distribution with the probabilities of each category being $\boldsymbol{q}$ and $\mathrm{Multi}_{(N,\boldsymbol{q})}$ as the distribution of N many selected balls without replacement from the jar containing r many balls whose selection probabilities are $\boldsymbol{q}$. For technical simplicity, we assume $\mathcal{X} = [-1, 1]^d$.

2.2. Data generating process

We consider two supervised learning problems : regression and classification. In regression problems, the input vector $\boldsymbol{X}$ and the response variable $Y \in \mathbb{R}$ are generated from the model

$$\begin{aligned} \boldsymbol{X} &\sim \mathrm{P}_{\boldsymbol{X}}, \\ Y|\boldsymbol{X} &\sim N(f_0(\boldsymbol{X}), \sigma_0^2), \end{aligned} \tag{1}$$

where $\mathrm{P}_{\boldsymbol{X}}$ is the probability measure defined on $\mathcal{X}$. Here, $f_0 : \mathcal{X} \to \mathbb{R}$ and $\sigma_0^2 > 0$ are the unknown true regression function and unknown variance of the noise, respectively.

For K-class classification problems, the input vector $\boldsymbol{X}$ and the response variable $Y \in [K]$ are generated from the model

$$\begin{aligned} \boldsymbol{X} &\sim \mathrm{P}_{\boldsymbol{X}}, \\ Y|\boldsymbol{X} &\sim \mathrm{Cat}\left(\mathrm{softmax}(\boldsymbol{f}_0(\boldsymbol{X}))\right), \end{aligned} \tag{2}$$

where $\mathrm{P}_{\boldsymbol{X}}$ is the probability measure defined on $\mathcal{X}$ and $\boldsymbol{f}_0 : \mathcal{X} \to \mathbb{R}^K$ is the logit of the unknown true conditional class probability function.

3. Masked Bayesian Neural Network

To construct a node-sparse BNN, we propose to use masking vectors which screen some nodes of the hidden layers. We first define the masked Deep Neural Network (mDNN) model, and then propose the mBNN on the top of the mDNN by specifying a prior appropriately.

3.1. Deep Neural Network

For $L \in \mathbb{N}$ and $\boldsymbol{p} = (p^{(0)}, p^{(1)}, ..., p^{(L)}, p^{(L+1)})^\top \in \mathbb{N}^{L+2}$, DNN with the $(L, \boldsymbol{p})$ architecture is a DNN model which has L hidden layers and $p^{(l)}$ many nodes at the l-th hidden layer for $l \in [L]$. The input and output dimensions are $p^{(0)}$ and $p^{(L+1)}$, respectively. The output of the DNN model can be written as

$$f_{\boldsymbol{\theta}}^{\mathrm{DNN}}(\cdot) := A_{L+1} \circ \rho \circ A_L \cdots \circ \rho \circ A_1(\cdot), \tag{3}$$

where $A_l : \mathbb{R}^{p^{(l-1)}} \mapsto \mathbb{R}^{p^{(l)}}$ for $l \in [L+1]$ is an affine map defined as $A_l(\boldsymbol{x}) := W_l\boldsymbol{x} + \boldsymbol{b}_l$ with $W_l \in \mathbb{R}^{p^{(l)}\times p^{(l-1)}}$ and $\boldsymbol{b}_l \in \mathbb{R}^{p^{(l)}}$ and ρ is the RELU activation function. The DNN model is parameterized by $\boldsymbol{\theta}$ which is the concatenation of the weight matrices and bias vectors, that is

$$\boldsymbol{\theta} := (\mathrm{vec}(W_1)^\top, b_1^\top, \dots, \mathrm{vec}(W_{L+1})^\top, b_{L+1}^\top)^\top.$$

3.2. Masked Deep Neural Network

For a given standard DNN, the corresponding masked DNN (mDNN) is constructed by simply adding masking parameters to the standard DNN model. For $l \in [L]$, the mDNN

model screens the nodes at the l-th hidden layer using the binary masking vector $\boldsymbol{m}^{(l)} \in \{0,1\}^{p^{(l)}}$. When $\left(\boldsymbol{m}^{(l)}\right)_j = 0$, the j-th node at the l-th hidden layer becomes inactive. The output of the mDNN model with the $(L, \boldsymbol{p})$ architecture can be written as

$$f^{\text{mDNN}}_{\boldsymbol{M},\boldsymbol{\theta}}(\cdot) := A_{L+1} \circ \rho_{\boldsymbol{m}^{(L)}} \circ A_L \cdots \circ \rho_{\boldsymbol{m}^{(1)}} \circ A_1(\cdot),$$

where A_l for $l \in [L+1]$ is the affine map defined in (3) and $\rho_{\boldsymbol{m}^{(l)}} : \mathbb{R}^{p^{(l)}} \to \mathbb{R}^{p^{(l)}}$ for $l \in [L]$ is the masked-RELU activation function defined as

$$\rho_{\boldsymbol{m}^{(l)}} \begin{pmatrix} x_1 \\ \vdots \\ x_{p^{(l)}} \end{pmatrix} := \boldsymbol{m}^{(l)} \odot \begin{pmatrix} \max(x_1, 0) \\ \vdots \\ \max(x_{p^{(l)}}, 0) \end{pmatrix}.$$

The model is parameterized by $\boldsymbol{M}$ and $\boldsymbol{\theta}$, where $\boldsymbol{M} \in \{0,1\}^{\sum_{l=1}^{L} p^{(l)}}$ is the concatenate of the all masking vectors

$$\boldsymbol{M} := \left({\boldsymbol{m}^{(1)}}^\top, \ldots, {\boldsymbol{m}^{(L)}}^\top\right)^\top$$

and $\boldsymbol{\theta}$ is the concatenation of the weight matrices and the bias vectors in the standard DNN model.

Note that the mDNN is nothing but a standard DNN with the architecture $(L, \tilde{\boldsymbol{p}})$ where $(\tilde{\boldsymbol{p}})_l = \left|\boldsymbol{m}^{(l)}\right|_0$. That is, the mDNN is a reparameterization of the standard DNN using the masking vectors. This reparameterization, however, allows us to develop an efficient MCMC algorithm, in particular for searching good architectures (i.e. good masking vectors).

3.3. Prior and posterior distribution

We adopt a data-dependent prior Π_n on the parameters $\boldsymbol{M}$ and $\boldsymbol{\theta}$ (as well as σ^2 for regression problems). We consider a data-dependent prior to ensure the optimal posterior concentration rate and adaptiveness. We assume that a priori $\boldsymbol{M}$ and $\boldsymbol{\theta}$ (as well as σ^2 for regression problems) are independent.

For $\boldsymbol{M}$, we use the following hierarchical prior. For each $\boldsymbol{m}^{(l)}$ with $l \in [L]$, let $s^{(l)} := |\boldsymbol{m}^{(l)}|_0$ be the sparsity of the masking vector $\boldsymbol{m}^{(l)}$. We put a prior mass on $s^{(l)}$ by

$$\Pi_n(s^{(l)}) \propto e^{-(\lambda \log n)^5 {s^{(l)}}^2} \qquad \text{for } s^{(l)} \in [p^{(l)}], \quad (4)$$

where $\lambda > 0$ is a hyper-parameter. Note that the prior on $s^{(l)}$ regularizes the width of the network, and more strong regularization is enforced as more data are accumulated. Given the sparsity level $s^{(l)}$, the masking vector $\boldsymbol{m}^{(l)}$ is sampled from the set $\{\boldsymbol{m}^{(l)} \in \{0,1\}^{p^{(l)}} : |\boldsymbol{m}^{(l)}|_0 = s^{(l)}\}$ uniformly. In other words,

$$\Pi_n(\boldsymbol{m}^{(l)} | s^{(l)}) \overset{iid}{\propto} \frac{1}{\binom{p^{(l)}}{s^{(l)}}} \mathbb{I}(|\boldsymbol{m}^{(l)}|_0 = s^{(l)}) \quad (5)$$

and the prior of $\boldsymbol{m}^{(l)}$ is the product of (4) and (5)

For the prior of $\boldsymbol{\theta}$, we assume that

$$\theta_i \overset{iid}{\sim} \mathfrak{p}(\theta_i) \qquad i \in [T], \quad (6)$$

where $T := \sum_{l=0}^{L} (p^{(l)} + 1) p^{(l+1)}$ is the length of the vector $\boldsymbol{\theta}$, and choose $\mathfrak{p}$ carefully to ensure desirable theoretical properties. For high-dimensional linear regression problems, Castillo & van der Vaart (2012) and Castillo et al. (2015) notice that using a heavy-tailed distribution for the prior of the regression coefficients is essential for theoretical optimality. Motivated by these observations, we consider a heavy-tailed distribution for $\mathfrak{p}$. Let $\mathfrak{P}$ be the class of polynomial tail distributions on $\mathbb{R}$ defined as

$$\mathfrak{P} := \left\{ \mathfrak{p} : \lim_{x\to\infty} \frac{x^{-\log x}}{\mathfrak{p}(x)} \to 0 \text{ and } \lim_{x\to\infty} \frac{x^{-\log x}}{\mathfrak{p}(-x)} \to 0 \right\}.$$

Examples of polynomial tail distributions are the Cauchy distribution and Student's t-distribution. On the other hand, the Gaussian and Laplace distributions do not belong to $\mathfrak{P}$. We assume that $\mathfrak{p}$ belongs to $\mathfrak{P}$. We will show in Section 4 that any prior in $\mathfrak{P}$ yields the optimal posterior concentration rate.

For the prior of σ^2 in regression problems, a standard distribution such as the inverse-gamma distribution can be used. Any distribution whose density is positive at the true σ_0^2 works for theoretical optimality.

3.4. Comparison with MC-dropout

The idea of masking nodes in DNN has been already used in various algorithms. Dropout (Srivastava et al., 2014) and MC-dropout (Gal & Ghahramani, 2016) are two representative examples, where they randomly mask the nodes of DNN during the training phase. While Dropout abolishes the masking vectors and uses the scaled-down version of the trained weights in the prediction phase, MC-dropout uses the trained weights obtained in the training phase multiplied by a random masking vectors. Since the masking vectors is treated as a random vectors following its posterior distribution at the prediction phase, our mBNN is similar to MC-dropout. A key difference between the mBNN and MC-dropout, however, is that the mBNN learns the distribution of the masking vectors from data via the posterior distribution but MC-dropout does not. That is, the mBNN learns the architecture of DNN from data, which makes the mBNN have good theoretical and empirical properties.

4. Theoretical optimalities

In this section, we derive the posterior concentration rates of the mBNNs for regression and classification problems, which are minimax optimal up to a logarithmic factor. In addition, we show that the mBNN achieves the optimal sparsity

asymptotically. We assume that $\mathcal{D}^{(n)} := \{(\boldsymbol{X}_i, Y_i)\}_{i\in[n]}$ are independent copies following the true distribution $\mathbb{P}_0$ specified by either the model (1) or model (2).

4.1. Posterior concentration rate for nonparametric regression

We consider the nonparametric regression model (1). We assume the true regression function f_0 belongs to the β-Hölder class $\mathcal{H}_d^\beta$.[1] Here, the β-Hölder class $\mathcal{H}_d^\beta$ is given as

$$\mathcal{H}_d^\beta := \{f : [-1,1]^d \to \mathbb{R}; ||f||_{\mathcal{H}^\beta} < \infty\},$$

where $||f||_{\mathcal{H}^\beta}$ denotes the Hölder norm defined by

$$||f||_{\mathcal{H}^\beta} := \sum_{\boldsymbol{\alpha}:|\boldsymbol{\alpha}|_1<\beta} \|\partial^{\boldsymbol{\alpha}} f\|_\infty + \sum_{\boldsymbol{\alpha}:|\boldsymbol{\alpha}|_1=\lfloor\beta\rfloor} \sup_{\substack{\boldsymbol{x}_1,\boldsymbol{x}_2\in[-1,1]^d \\ \boldsymbol{x}_1\neq\boldsymbol{x}_2}} \frac{|\partial^{\boldsymbol{\alpha}} f(\boldsymbol{x}_1) - \partial^{\boldsymbol{\alpha}} f(\boldsymbol{x}_2)|}{|\boldsymbol{x}_1 - \boldsymbol{x}_2|_\infty^{\beta-\lfloor\beta\rfloor}}.$$

For inference, we consider the probabilistic model

$$Y_i \overset{ind.}{\sim} N(f^{\mathrm{mDNN}}_{\boldsymbol{M},\boldsymbol{\theta}[-F,F]}(\boldsymbol{X}_i), \sigma^2),$$

where $f^{\mathrm{mDNN}}_{\boldsymbol{M},\boldsymbol{\theta}}(\boldsymbol{X}_i)$ has the $(L_n, \boldsymbol{p}_n)$ architecture with L_n and $\boldsymbol{p}_n$ given as

$$L_n := \lceil C_L \log n \rceil, \tag{7}$$
$$p_n := \lceil C_p \sqrt{n} \rceil,$$
$$\boldsymbol{p_n} := (d, p_n, \ldots, p_n, 1)^\top \in \mathbb{N}^{L_n+2} \tag{8}$$

for positive constants C_L and C_p that are defined in Lemma A.1. Then, the likelihood $\mathcal{L}(w|\mathcal{D}^{(n)})$ of $w := (\boldsymbol{M}, \boldsymbol{\theta}, \sigma^2)$ is expressed as

$$(2\pi\sigma^2)^{-\frac{n}{2}} \exp\left(-\frac{\sum_{i=1}^n (Y_i - f^{\mathrm{mDNN}}_{\boldsymbol{M},\boldsymbol{\theta}\,[-F,F]}(\boldsymbol{X}_i))^2}{2\sigma^2}\right),$$

and the corresponding posterior distribution is given as

$$\Pi_n(w \mid \mathcal{D}^{(n)}) \propto \Pi_n(w)\mathcal{L}(w|\mathcal{D}^{(n)}),$$

where Π_n is the prior defined on Section 3.3.

In the following theorem, we show that the mBNN model achieves the optimal (up to a logarithmic factor) posterior concentration rate to the true regression function.

Theorem 4.1 (Posterior Concentration of the mBNN for regression problems)**.** *Assume $f_0 \in \mathcal{H}_d^\beta$, $\beta < d$, and there exist $F > 0$ and $\sigma_{max}^2 > 0$ such that $\|f_0\|_\infty \le F$ and $\sigma_0^2 \le \sigma_{max}^2$. Consider the mDNN model with the $(L_n, \boldsymbol{p}_n)$ architecture, where L_n and $\boldsymbol{p}_n$ are given in (7) and (8). If we put the prior given as (4), (5) and (6) over $\boldsymbol{M}$, $\boldsymbol{\theta}$ and any prior on σ^2 whose density (with respect to Lebesgue measure) is positive on its support $(0, \sigma_{max}^2]$, the posterior distribution concentrates to the f_0 and σ_0^2 at the rate $\varepsilon_n = n^{-\beta/(2\beta+d)} \log^\gamma(n)$ for $\gamma > \frac{5}{2}$ in the sense that*

$$\Pi_n\Big((f,\sigma^2) : ||f - f_0||_{2,\mathrm{P}_X} + |\sigma^2 - \sigma_0^2| > M_n\varepsilon_n \;\Big|\; \mathcal{D}^{(n)}\Big) \overset{\mathbb{P}_0^n}{\to} 0$$

as $n \to \infty$ for any $M_n \to \infty$, where $\mathbb{P}_0^n$ is the probability measure of the training data $\mathcal{D}^{(n)}$.

The convergence rate $n^{-\beta/(2\beta+d)}$ is known to be minimax lower bound when estimating the β-Hölder smooth function (Tsybakov, 2009). Our concentration rate is near optimal up to a logarithmic factor and adaptive to the smoothness of the true model.

Comparison with other works Similar convergence rates are derived in the non-bayesian theoretical deep learning literature (Schmidt-Hieber, 2020; Kohler & Langer, 2021). However, the architectures considered in Schmidt-Hieber (2020) and Kohler & Langer (2021) depend on the smoothness β of the true regression function, which is rarely known in practice. In contrast, architecture and prior of the mBNN do not depend on the smoothness β, which makes the mBNN very attractive.

Polson & Ročková (2018), Chérief-Abdellatif (2020) and Bai et al. (2020) also derive near optimal concentration rates for edge-sparse BNNs. Posterior computations for their smoothness-adaptive models, however, are almost impossible and inferential cost of edge-sparse BNNs could be large. Jantre et al. (2021) provides the posterior concentration rate for node-sparse BNN, but their result does not guarantee minimax optimality and is not adaptive to the smoothness of the true model. Theorem 4.1 is the first result for theoretical optimality of the Bayesian analysis for node-sparse DNNs.

4.2. Posterior concentration rate for binary classification

Theoretical results for regression problem can be extended to classification problem. We consider the classification problem (2) with $K = 2$, and denote $f_0 := (\boldsymbol{f}_0)_2 - (\boldsymbol{f}_0)_1$. We assume that f_0 belongs to the β-Hölder class $\mathcal{H}_d^\beta$. For inference, we consider the probabilistic model

$$Y_i \overset{ind.}{\sim} \mathrm{Bernoulli}(\phi \circ f^{\mathrm{mDNN}}_{\boldsymbol{M},\boldsymbol{\theta}[-F,F]}(\boldsymbol{X}_i)),$$

where ϕ is the sigmoid function and $f^{\mathrm{mDNN}}_{\boldsymbol{M},\boldsymbol{\theta}}$ has the $(L_n, \boldsymbol{p}_n)$ architecture with L_n and $\boldsymbol{p}_n$ given as (7) and (8). Then, the likelihood $\mathcal{L}(w|\mathcal{D}^{(n)})$ of $w := (\boldsymbol{M}, \boldsymbol{\theta})$ is ex-

[1] Theoretical results for hierarchical composition functions are provided in Appendix B.

pressed as

$$\prod_{i=1}^{n}(\phi \circ f^{\mathrm{mDNN}}_{\boldsymbol{M},\boldsymbol{\theta}[-F,F]}(\boldsymbol{X}_i))^{Y_i}(1-\phi \circ f^{\mathrm{mDNN}}_{\boldsymbol{M},\boldsymbol{\theta}[-F,F]}(\boldsymbol{X}_i))^{1-Y_i}.$$

In the following theorem, we prove that the mBNN model achieves the optimal (up to a logarithmic factor) posterior concentration rate to the true conditional class probability adaptive to the smoothness of the true model.

Theorem 4.2 (Posterior Concentration of the mBNN for classification problems)**.** *Assume $f_0 \in \mathcal{H}^{\beta}_d$, $\beta < d$, and there exists $F > 0$ such that $\|f_0\|_\infty \leq F$. Consider the mDNN model with the $(L_n, \boldsymbol{p}_n)$ architecture where L_n and $\boldsymbol{p}_n$ are given in (7) and (8). If we put the prior given as (4), (5) and (6) over $\boldsymbol{M}$ and $\boldsymbol{\theta}$, the posterior distribution concentrates to the true conditional class probability at the rate $\varepsilon_n = n^{-\beta/(2\beta+d)}\log^{\gamma}(n)$ for $\gamma > \frac{5}{2}$ in the sense that*

$$\Pi_n\Big(f : ||\phi \circ f - \phi \circ f_0||_{2,\mathrm{P}_X} > M_n\varepsilon_n \;\Big|\; \mathcal{D}^{(n)}\Big) \overset{\mathbb{P}^n_0}{\to} 0$$

as $n \to \infty$ for any $M_n \to \infty$, where $\mathbb{P}^n_0$ is the probability measure of the training data $\mathcal{D}^{(n)}$.

4.3. Guaranteed sparsity of the mBNN

Not only the fast concentration rate, the mBNN achieves the optimal sparsity too. Theorem 4.3, which is a by-product of the proofs for Theorems 4.1 and 4.2, gives the level of sparsity of the mBNN.

Theorem 4.3 (Guaranteed sparsity of the mBNN)**.** *Under the assumptions in Theorem 4.1 or Theorem 4.2, we have*

$$\Pi_n\Big(f : f = f^{\mathrm{mDNN}}_{\boldsymbol{M},\boldsymbol{\theta}[-F,F]},\ \max_{l\in[L]}|\boldsymbol{m}^{(l)}|_0 > \mathfrak{s}_n \;\Big|\; \mathcal{D}^{(n)}\Big) \overset{\mathbb{P}^n_0}{\to} 0$$

as $n \to \infty$, where $\mathfrak{s}_n$ is defined as

$$\mathfrak{s}_n := \left\lceil C_p\left(n^{\frac{d}{2\beta+d}}(\log n)\right)^{1/2}\right\rceil$$

and $\mathbb{P}^n_0$ is the probability measure of the training data $\mathcal{D}^{(n)}$.

Yarotsky (2017) proves that the lower bound of the sparsity of DNNs to approximate functions in $\mathcal{H}^{\beta}_d$ is equal to $\mathfrak{s}_n$ up to a logarithmic factor. That is, the mBNN automatically learns the optimal sparsity from data.

5. Posterior inference

Let $\mathcal{D}^{(n)} := \{(\boldsymbol{X}_i, Y_i)\}_{i\in[n]}$ be training data. Let w denote all of the parameters in the mBNN, that is, $w := (\boldsymbol{M}, \boldsymbol{\theta}, \sigma^2)$ for the regression problem (1) and $w := (\boldsymbol{M}, \boldsymbol{\theta})$ for the classification problem (2). For a new test example $\boldsymbol{x} \in \mathcal{X}$, the prediction with the mBNN is done by the predictive distribution:

$$p(y \mid \boldsymbol{x}, \mathcal{D}^{(n)}) = \int_w p(y \mid \boldsymbol{x}, w)\Pi_n(w \mid \mathcal{D}^{(n)})dw,$$

where $\Pi_n(w \mid \mathcal{D}^{(n)})$ is the posterior posterior distribution of w. When the integral is difficult to be evaluated, it is common to approximate it by the Monte Carlo method

$$p(y \mid \boldsymbol{x}, \mathcal{D}^{(n)}) \approx \frac{1}{T}\sum_{t=1}^{T} p(y \mid \boldsymbol{x}, w^{(t)}),$$

where $w^{(t)} \sim \Pi_n(w \mid \mathcal{D}^{(n)})$. In this section, we develop a MCMC algorithm to sample w efficiently from $\Pi_n(w \mid \mathcal{D}^{(n)})$.

5.1. MCMC algorithm

The proposed MCMC algorithm samples $\boldsymbol{\theta}$ (and σ^2) given $\boldsymbol{M}$ and $\mathcal{D}^{(n)}$ and then samples $\boldsymbol{M}$ given $\boldsymbol{\theta}$ (and σ^2) and $\mathcal{D}^{(n)}$, and iterates these two samplings until convergence.

There are various efficient sampling algorithms for $\boldsymbol{\theta}$ (and σ^2) given $\boldsymbol{M}$ and $\mathcal{D}^{(n)}$ such as Hamiltonian Monte Carlo (HMC) (Neal et al., 2011), Stochastic Gradient Langevin Dynamics (SGLD) (Welling & Teh, 2011) and Stochastic Gradient HMC (SGHMC) (Chen et al., 2014). In practice, we select a sampling algorithm among those depending on the sizes of data and model.

For generating $\boldsymbol{M}$ from its conditional posterior, we consider the Metropolis-Hastings (MH) algorithm. The hardest part is to design a good proposal distribution since the dimension of $\boldsymbol{M}$ is quite large and all entries are binary. In the next subsection, we propose an efficient proposal distribution for $\boldsymbol{M}$.

5.2. Proposal for the MH algorithm

Essentially, sampling $\boldsymbol{M}$ is equivalent to sampling a large dimensional binary vector. A well known strategy for sampling a large dimensional binary vector is to use the MH algorithm with the locally informed proposal (Umrigar, 1993; Zanella, 2020) given as

$$q\left(\boldsymbol{M}^{\star} \mid \boldsymbol{M}\right) \propto e^{\frac{1}{2}(l(\boldsymbol{M}^{\star})-l(\boldsymbol{M}))}\mathbb{I}\left(\boldsymbol{M}^{\star} \in H(\boldsymbol{M})\right), \quad (9)$$

where $l(\boldsymbol{M})$ is the log-posterior of $\boldsymbol{M}$ and $H(\boldsymbol{M})$ is the Hamming ball of a certain size around $\boldsymbol{M}$. The key point of (9) is to give more probability to $\boldsymbol{M}^{\star}$ whose posterior probability is high. While powerful, this locally informed proposal requires to compute $l(\boldsymbol{M}^{\star})$ for every $\boldsymbol{M}^{\star} \in H(\boldsymbol{M})$, which is time consuming. To resolve this problem, we propose a proposal distribution which only uses the information of the current $\boldsymbol{M}$.

First, we select either *birth* or *death*, where *birth* makes some inactive nodes become active and *death* makes some active nodes become inactive. When *death* is selected, motivated by the pruning algorithms (Lee et al., 2018; Tanaka et al., 2020), our strategy is to prune less sensitive nodes. That is, we delete nodes which do not affect much to the

Algorithm 1 The algorithm of the proposed MH algorithm

1: Sample $u \sim \text{Bernoulli}(0.5)$, $N \sim \text{Uniform}([N_{\max}])$.
2: Calculate the selection probability $\boldsymbol{Q}_u$ using (10) or (11).
3: Sample N many nodes $\{i_1, \ldots, i_N\}$ by

$$\{i_1, \ldots, i_N\} \sim \text{Multi}_{(N, \boldsymbol{Q}_u)}\,.$$

4: $\boldsymbol{M}^\star = \text{flip}(\boldsymbol{M}^{\text{curr}}, \{i_1, \ldots, i_N\})$, where $\boldsymbol{M}^{\text{curr}}$ is the current masking vectors.
5: Accept or reject $\boldsymbol{M}^\star$ with the acceptance probability (12).

current DNN model when their values are changed. On the other hand, when *birth* is selected, we choose some of inactive nodes with equal probabilities and make them active. We use equal probabilities because the posterior of the edges connected to the inactive nodes is the same as the prior and thus there is no reason to prefer certain inactive nodes more. Moreover, the proposal with equal probabilities for *birth* is helpful for increasing the acceptance rate of *death* to result in fast mixing.

To be more specific, we first select the move $u \in \{0, 1\}$ with probability 1/2, where $u = 0$ and $u = 1$ indicates *birth* and *death*, respectively. In addition, we select an integer N from $[N_{\max}]$ uniformly, where $N_{\max}$ is a prespecified positive integer. Then, we select randomly N nodes among the nodes whose masking values are u following $\text{Multi}_{(N, \boldsymbol{Q}_u)}$, where

$$\boldsymbol{Q}_0 \propto \mathbb{I}(\boldsymbol{m}_j^{(l)} = 0), \qquad (\textit{birth}) \quad (10)$$

$$\boldsymbol{Q}_1 \propto \mathbb{I}(\boldsymbol{m}_j^{(l)} = 1) \exp\left(-|\nabla l(\boldsymbol{M})|/2\right). \qquad (\textit{death}) \quad (11)$$

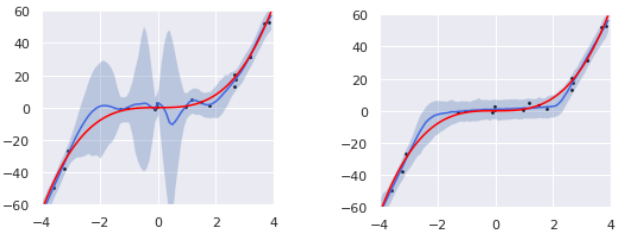
Even though $l(\boldsymbol{M})$ is defined only on binary vectors, the gradient $\nabla l(\boldsymbol{M}) = \partial l(\boldsymbol{M})/\partial \boldsymbol{M}$ of $l(\boldsymbol{M})$ can be defined by extending the domain of $l(\boldsymbol{M})$ appropriately. Finally, we flip the masking values of the selected nodes to have a new proposal $\boldsymbol{M}^*$. To sum up, the proposal distribution first selects a set of nodes $\{i_1, \ldots, i_N\}$ following $\text{Multi}_{(N, \boldsymbol{Q}_u)}$ and changes their mask values to $(1 - \boldsymbol{M}_{i_1}, \ldots, 1 - \boldsymbol{M}_{i_N})$. Then, we accept the new proposal $\boldsymbol{M}^*$ with probability

$$\min\left(1, \frac{\Pi_n(\boldsymbol{M}^\star \mid X^{(n)}, Y^{(n)}, \boldsymbol{\theta})}{\Pi_n(\boldsymbol{M} \mid X^{(n)}, Y^{(n)}, \boldsymbol{\theta})} \frac{q\left(\boldsymbol{M} \mid \boldsymbol{M}^\star\right)}{q\left(\boldsymbol{M}^\star \mid \boldsymbol{M}\right)}\right), \quad (12)$$

where

$$\frac{q\left(\boldsymbol{M} \mid \boldsymbol{M}^\star\right)}{q\left(\boldsymbol{M}^\star \mid \boldsymbol{M}\right)} = \frac{\text{Multi}_{(N, \boldsymbol{Q}^\star_{1-u})}(\{i_1, \ldots, i_N\})}{\text{Multi}_{(N, \boldsymbol{Q}_u)}(\{i_1, \ldots, i_N\})}$$

and $\boldsymbol{Q}^\star_{1-u}$ is the selection probability vector defined by (10) or (11) with the mask vectors $\boldsymbol{M}^\star$. Note that some inactive nodes become active when $u = 0$ (i.e. *birth* of nodes) and some active nodes become inactive when $u = 1$ (i.e. *death* of nodes). The proposed MH algorithm is summarized in Algorithm 1.



(a) BNN (b) mBNN

Figure 1. **Simulated data.** Predictive distributions of BNN and the mBNN on simulated data. The true polynomial function and the noisy observations are plotted by the red line and black dots, respectively. For each predictive distribution, the mean is drawn by the blue line and the 95% predictive interval is shown as the shaded areas.

One may consider a linear approximation of l in (9) using the gradient information at $\boldsymbol{M}$, as is suggested by Grathwohl et al. (2021) and Zhang et al. (2022). However, we found that the linear approximation is not accurate for the mBNN, which is partly because the corresponding DNN is not locally smooth enough. In section 6.5, we provide the results of the experiment for comparing several proposal distributions which support the choice of (10) and (11).

6. Experiment

In this section, we perform experiments to empirically justify the usefulness of the mBNN. In Section 6.1 and 6.2, we conduct an experiment to demonstrate the necessity of masking variables using simulation and real datasets. In Section 6.3, we apply the mBNN to a Bayesian structural time series model to illustrate its practical usefulness. In Section 6.4, we extend the mBNN to CNN and experimentally show it is also useful for compressing large complex DNNs. In Section 6.5, we investigate the efficiency of the proposal distribution (10) and (11) in the MH algorithm. All the experimental details as well as the results of additional numerical experiments are given in Appendix D and E. The code is available at https://github.com/ggong369/mBNN.

6.1. Simulation

We obtain the predictive distribution when the true regression function is the noisy polynomial regression problem considered in Hernández-Lobato & Adams (2015). Inputs x_i are sampled from $x_i \sim \text{Uniform}(-4, 4)$ and the corresponding outputs y_i are obtained by $y_i = x_i^3 + \epsilon_i$, where $\epsilon_i \sim N(0, 9)$. We generate 20 training examples and compare the predictive distribution of the mBNN with that of BNN by generating 1000 MCMC samples from each model. For each model, the two hidden layer MLP with the layer sizes (1000,1000) is used.

The mean and 95% predictive intervals of the predictive

Table 1. **Real dataset.** Performance comparison of BNN, NS-VI and the mBNN on UCI datasets. The boldface numbers are the best one among the three methods.

Dataset	Method	Coverage	RMSE	NLL	CRPS
Boston	BNN	0.912(0.006)	3.411(0.145)	2.726(0.087)	1.738(0.052)
	NS-VI	0.746(0.010)	3.079(0.179)	4.242(0.568)	1.661(0.069)
	mBNN	**0.933**(0.007)	**2.902**(0.143)	**2.472**(0.085)	**1.462**(0.055)
Concrete	BNN	0.908(0.008)	5.080(0.178)	3.091(0.055)	2.658(0.072)
	NS-VI	0.568(0.011)	5.046(0.149)	9.713(0.686)	2.965(0.088)
	mBNN	**0.912**(0.009)	**4.913**(0.180)	**3.027**(0.046)	**2.628**(0.087)
Energy	BNN	**0.945**(0.005)	0.591(0.017)	0.902(0.025)	0.322(0.007)
	NS-VI	0.913(0.006)	1.322(0.117)	1.792(0.121)	0.720(0.055)
	mBNN	**0.945**(0.004)	**0.474**(0.015)	**0.670**(0.034)	**0.256**(0.006)
Yacht	BNN	0.977(0.006)	0.675(0.048)	1.011(0.056)	0.332(0.013)
	NS-VI	0.932(0.011)	1.842(0.096)	1.915(0.063)	0.969(0.042)
	mBNN	**0.953**(0.008)	**0.664**(0.044)	**0.932**(0.062)	**0.318**(0.015)

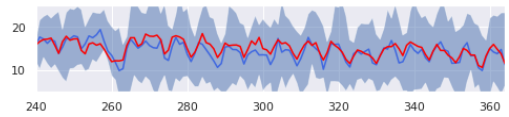
distributions of BNN and the mBNN are presented in Figure 1. The figure illustrates that BNN quantifies uncertainty on a data sparse region overly to have a too wide predictive interval. Note that the true regression function is assumed to smooth and hence it is possible to transfer information on data dense regions to data sparse regions. Thus, proper uncertainty quantification even on data sparse regions could be possible. The coverage probability of the predictive interval of the mBNN is 95.3%, which is obtained by generating additional 1000 test sample, while that of BNN is 98.6%. The results indicate that deleting unnecessary nodes is important not only for fast convergence rates but also proper uncertainty quantification.

6.2. Real dataset

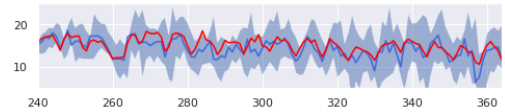
We evaluate BNN, node-sparse VI (NS-VI) (Louizos et al., 2017) and the mBNN on four UCI regression datasets (*Boston, Concrete, Energy, Yacht*). For each dataset, we construct 20 random 90-to-10 train-test splits to provide the standard errors. For each method, the two hidden layer MLP with the layer sizes (1000,1000) is used. We select 20 models from several thousands MCMC samples and compare the predictive distributions obtained by the selected 20 models in terms of generalization and uncertainty quantification.

In Table 1, we report the means and standard errors of the performance measures on the 20 repeated experiments. For the performance measures, the coverage probability of the 95% predictive interval (Coverage), the root mean square error (RMSE) of the Bayes estimator, the negative log-likelihood (NLL) and continuous ranked probability score (CRPS) (Gneiting & Raftery, 2007) on test data are considered.

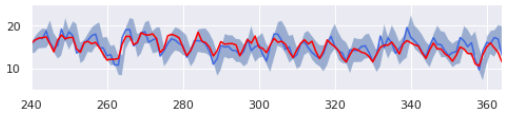
It is obvious that the mBNN outperform the other two competitors with respect to all of the four measures. That is, the mBNN is good at not only estimating the regression function and but also uncertainty quantification. It is noticeable that NS-VI performs too badly, which suggests that full Bayesian analysis of DNN is must. More detailed results such as the sparsity and the sensitivity to the hyper-parameter selection are provided in Appendix E.



(a) Linear ; (RMSE, NLL) = (1.34, 2.36)



(b) BNN ; (RMSE, NLL) = (1.59, 2.10)



(c) mBNN ; (RMSE, NLL) = (1.21, 1.64)

Figure 2. **Daily search volume dataset.** Predictive distribution of BSTS model using linear, DNN and mDNN. X-axis and y-axis represent time (in day) and search volumes (in thousand), respectively. For each method, the means of predictive distributions and 95% predictive intervals are shown as the blue line and shaded area, respectively. The observed values are plotted by the red line.

6.3. Application to the Bayesian structural time series model

Bayesian structural time series (BSTS) models (Scott & Varian, 2014; Qiu et al., 2018) provide a useful tool for time series forecasting, nowcasting, inferring causal relationships and anomaly detection (Brodersen et al., 2015; Feng & Tian, 2021). The key of BSTS is the state space model, which is given as

$$\begin{aligned} y_t &= \mu_t + \boldsymbol{\beta}^\top \boldsymbol{x}_t + \epsilon_t, & \epsilon_t &\sim \mathcal{N}(0, \sigma_\epsilon^2) \\ \mu_{t+1} &= r\mu_t + \eta_t, & \eta_t &\sim \mathcal{N}(0, \sigma_\eta^2) \\ \mu_0 &\sim N(a_0, \sigma_0^2) \end{aligned}$$

where $\boldsymbol{x}_t \in \mathbb{R}^d$, $y_t \in \mathbb{R}$ and $\mu_t \in \mathbb{R}$ denote observed input variables, output variable and unobserved local trend at time t, respectively. For inference and prediction, MCMC algorithm using Kalman filter (Welch et al., 1995; Durbin & Koopman, 2002) is mainly used.

In many cases, the linear component $\boldsymbol{\beta}^\top \boldsymbol{x}_t$ may not be insufficient to explain complicate relations between inputs and outputs, and DNN can be considered instead. In turn, the mBNN is a useful inferential tool for this nonlinear BSTS. To illustrate that the mBNN works well for BSTS, we analyze a real dataset consisting of daily search volumes of several keywords collected by a Korea search platform company. The dataset consists of daily search volumes of keywords in year 2021 associated with a pre-specified product (eg. shampoo). The aim is to predict the search volume of the pre-specified product based on the search

Table 2. **Image datasets.** Performance comparison of NS-VI, NS-Ens, NS-MC and mBCNN on image datasets. When inferring the posterior of mBCNN, we use SGLD (Welling & Teh, 2011).

	Measure	NS-VI	NS-Ens	NS-MC	mBCNN✓
CIFAR10	ACC	0.906(0.002)	0.926(0.003)	0.918(0.002)	**0.932**(0.001)
	NLL	0.298(0.004)	0.255(0.011)	0.525(0.006)	**0.220**(0.005)
	ECE	0.009(0.001)	0.012(0.002)	0.020(0.001)	**0.008**(0.001)
	FLOPs	44.85(2.67)%	22.45(1.64)%	41.16(0.00)%	**12.26**(0.06)%
	Capacity	23.24(1.56)%	12.78(0.75)%	41.07(0.00)%	**3.47**(0.03)%
CIFAR100	ACC	0.600(0.004)	0.735(0.003)	0.679(0.002)	**0.737**(0.001)
	NLL	1.977(0.039)	1.076(0.014)	2.792(0.061)	**1.004**(0.008)
	ECE	0.006(0.000)	**0.002**(0.000)	0.006(0.000)	**0.002**(0.000)
	FLOPs	37.38(0.64)%	27.02(2.30)%	53.50(0.00)%	**18.34**(0.27)%
	Capacity	41.61(4.04)%	20.08(1.15)%	53.45(0.00)%	**12.21**(0.07)%

volumes of other related keywords.

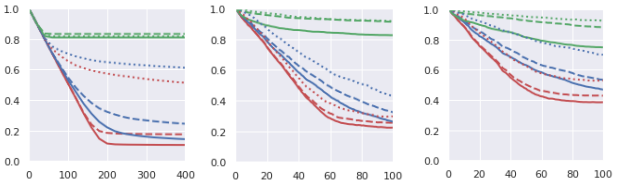
We apply the three BSTS models corresponding to the three regression components - linear, BNN and the mBNN. For BNN and the mBNN, the two hidden layer MLP with the layer sizes (100,100) is used. The predictive intervals with the NLL and RMSE values on the data from $t = 241$ to $t = 365$ obtained by the posterior distribution inferred on the data from $t = 1$ to $t = 240$ are presented in figure 2. It is clearly observed that the mBNN is superior in both nowcasting ability and uncertainty quantification. It is interesting that the predictive intervals of the linear and BNN models are much wider than those of the mBNN, which amply indicates that the choice of an appropriate architecture of DNN is crucial for desirable uncertainty quantification. More details about the dataset, methodology and additional experimental results are provided in Appendix D.3.

6.4. Extension to convolution neural network

We also extend the mBNN to masked Bayesian convolution neural network (mBCNN) for image dataset. We construct a masked CNN (mCNN) by adding masking vectors to the CNN model. See Appendix C.2 for details of the mBCNN.

We evaluate compression ability of the mBCNN on image datasets. For comparison, we use node-sparse versions of approximated Bayesian methodologies : NS-VI, node-sparse Deep ensemble (NS-Ens) and node-sparse MC-dropout (NS-MC). Detail description of competitors are provided in Appendix D.4. For each methods, ResNet18 (He et al., 2016) architecture is used. For fair comparison, we use five networks for inference in all the methods. We repeat the experiments 3 times with different seeds.

In Table 2, we report the means and standard errors of the performance measures on the repeated experiments. For the performance measures, accuracy of the Bayes estimator, NLL, expected calibration error (ECE) (Kumar et al., 2019) of test data are considered. In addition, we report inferential costs (FLOPs) and the numbers of nonzero parameters (Capacity) relative to the non-sparse model. The results in Table 2 show that mBCNN provides better generalization and uncertainty quantification with less inferential cost and model capacity compared to the other competitors. That is, the mBNN is a useful tool for compressing complex DNNs without hampering uncertainty quantification much.



(a) Yacht (b) CIFAR10 (c) CIFAR100

Figure 3. **Efficiency of the proposal.** The ratios of activated nodes (y-axis) relative to the largest DNN are drawn as the MCMC algorithms iterate in the burn-in period (x-axis). The solid, dashed and dotted lines correspond to (13), (14) and (15) for *birth*, respectively while blue, red and green lines correspond to (13), (14) and (15) for *death*, respectively. It is obvious that the number of activated nodes with our proposal distribution (the solid red line) decreases much faster than the other proposals.

6.5. Ablation study: Efficiency of the proposal

To illustrate the efficiency of our proposal distribution in Algorithm 1, we conduct a comparative experiment. As an alternative to the proposal distribution with selection probability (10) for *birth* and (11) for *death*, we consider the following candidates for the selection probability of either *birth* or *death*:

$$\boldsymbol{Q}_u \propto \mathbb{I}(\boldsymbol{m}_j^{(l)} = u), \tag{13}$$

$$\boldsymbol{Q}_u \propto \mathbb{I}(\boldsymbol{m}_j^{(l)} = u) \exp\left(-|\nabla l(\boldsymbol{M})|/2\right), \tag{14}$$

$$\boldsymbol{Q}_u \propto \mathbb{I}(\boldsymbol{m}_j^{(l)} = u) \exp\left((1-2u)\nabla l(\boldsymbol{M})/2\right). \tag{15}$$

While (13) gives the same selection probability on each active (or inactive) node, (14) and (15) put proposals depending on the role of each node. The proposal of (14) is devised to select less sensitive nodes more by giving proposal probabilities reciprocally proportional to $|\nabla l(\boldsymbol{M})|$, and (15) uses a linear approximation of l in (9) (Grathwohl et al., 2021).

We compare the speeds of convergence of the MCMC algorithm with the 9 proposal distributions which are all 3^2 combinations of the three candidates (13), (14) and (15) and the two moves for *birth* and *death*. Figure 3 presents how the ratios of the activated nodes relative to the largest DNN decrease in the burn-in phase of the MCMC algorithm. It is obvious that our proposal (the red solid lines) is most fast to eliminate unnecessary nodes. Note that the proposals involving the linear approximation of $l(\boldsymbol{M})$ do not work well, which suggests that DNNs are not smooth enough to be approximated linearly.

7. Discussion

We have proposed the mBNN which searches for a DNN with an appropriate complexity. We prove theoretical optimalities of the mBNN and develop an efficient MCMC algorithm. By extensive numerical studies, we illustrate that the proposed BNN discovers well condensed DNN architectures with better prediction accuracy and uncertainty quantification compared to large DNNs.

A node-sparse network can be considered as a dense network with a smaller width because all edges connected to survived nodes are active. This property is sharply contrast with edge-sparse networks. A key difference of the node-sparse network and a dense network with a smaller width is that the widths of each layer can vary for node-sparse networks while they should be fixed in advance for dense network. In practice, it would be difficult to find the optimal width in advance before analyzing data. mBNN is a kind of tools to find the optimal width data adaptively.

We do not insist that the proposal distribution in our MCMC algorithm is optimal. There would be more efficient proposals. More data adaptive proposals for *birth* would be possible which we leave as a future work.

Condensing the posterior distribution is also interesting. We have to employ multiple DNNs in the prediction phase, which would be expensive. Summarizing multiple posterior samples into a single random DNN would be useful. For bootstrapping, Shin et al. (2021) proposes a similar method, which can be modified for the mBNN.

Acknowledgements

This work was supported by National Research Foundation of Korea(NRF) grant funded by the Korea government (MSIT) (No. 2020R1A2C3A0100355014), Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government(MSIT) [NO.2022-0-00184, Development and Study of AI Technologies to Inexpensively Conform to Evolving Policy on Ethics] and INHA UNIVERSITY Research Grant.

References

Bai, J., Song, Q., and Cheng, G. Efficient variational inference for sparse deep learning with theoretical guarantee. *Advances in Neural Information Processing Systems*, 33: 466–476, 2020.

Banbura, M., Giannone, D., and Reichlin, L. Nowcasting. 2010.

Brodersen, K. H., Gallusser, F., Koehler, J., Remy, N., and Scott, S. L. Inferring causal impact using bayesian structural time-series models. *The Annals of Applied Statistics*, pp. 247–274, 2015.

Castillo, I. and van der Vaart, A. Needles and straw in a haystack: Posterior concentration for possibly sparse sequences. *The Annals of Statistics*, 40(4):2069–2101, 2012.

Castillo, I., Schmidt-Hieber, J., and Van der Vaart, A. Bayesian linear regression with sparse priors. *The Annals of Statistics*, 43(5):1986–2018, 2015.

Chen, T., Fox, E., and Guestrin, C. Stochastic gradient hamiltonian monte carlo. In *International conference on machine learning*, pp. 1683–1691. PMLR, 2014.

Chérief-Abdellatif, B.-E. Convergence rates of variational inference in sparse deep learning. In *International Conference on Machine Learning*, pp. 1831–1842. PMLR, 2020.

Chin, T.-W., Ding, R., Zhang, C., and Marculescu, D. Towards efficient model compression via learned global ranking. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 1518–1528, 2020.

Cranmer, M., Tamayo, D., Rein, H., Battaglia, P., Hadden, S., Armitage, P. J., Ho, S., and Spergel, D. N. A bayesian neural network predicts the dissolution of compact planetary systems. *Proceedings of the National Academy of Sciences*, 118(40), 2021.

Deng, W., Zhang, X., Liang, F., and Lin, G. An adaptive empirical bayesian method for sparse deep learning. *Advances in neural information processing systems*, 2019: 5563, 2019.

Durbin, J. and Koopman, S. J. A simple and efficient simulation smoother for state space time series analysis. *Biometrika*, 89(3):603–616, 2002.

Feng, C. and Tian, P. Time series anomaly detection for cyber-physical systems via neural system identification and bayesian filtering. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pp. 2858–2867, 2021.

Filos, A., Farquhar, S., Gomez, A. N., Rudner, T. G., Kenton, Z., Smith, L., Alizadeh, M., De Kroon, A., and Gal, Y. A systematic comparison of bayesian deep learning robustness in diabetic retinopathy tasks. *arXiv preprint arXiv:1912.10481*, 2019.

Frankle, J. and Carbin, M. The lottery ticket hypothesis: Finding sparse, trainable neural networks. In *International Conference on Learning Representations*, 2018.

Gal, Y. and Ghahramani, Z. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*, pp. 1050–1059. PMLR, 2016.

Gan, Z., Chen, C., Henao, R., Carlson, D., and Carin, L. Scalable deep poisson factor analysis for topic modeling. In *International Conference on Machine Learning*, pp. 1823–1832. PMLR, 2015.

Ghosal, S. and Van Der Vaart, A. Convergence rates of posterior distributions for noniid observations. *The Annals of Statistics*, 35(1):192–223, 2007.

Ghosh, S., Yao, J., and Doshi-Velez, F. Model selection in bayesian neural networks via horseshoe priors. *J. Mach. Learn. Res.*, 20(182):1–46, 2019.

Giannone, D., Reichlin, L., and Small, D. Nowcasting: The real-time informational content of macroeconomic data. *Journal of monetary economics*, 55(4):665–676, 2008.

Gneiting, T. and Raftery, A. E. Strictly proper scoring rules, prediction, and estimation. *Journal of the American statistical Association*, 102(477):359–378, 2007.

Gomez, A. N., Zhang, I., Kamalakara, S. R., Madaan, D., Swersky, K., Gal, Y., and Hinton, G. E. Learning sparse networks using targeted dropout. *arXiv preprint arXiv:1905.13678*, 2019.

Grathwohl, W., Swersky, K., Hashemi, M., Duvenaud, D., and Maddison, C. Oops i took a gradient: Scalable sampling for discrete distributions. In *International Conference on Machine Learning*, pp. 3831–3841. PMLR, 2021.

Györfi, L., Kohler, M., Krzyżak, A., and Walk, H. *A distribution-free theory of nonparametric regression*, volume 1. Springer, 2002.

Han, S., Pool, J., Tran, J., and Dally, W. Learning both weights and connections for efficient neural network. *Advances in neural information processing systems*, 28, 2015.

Harvey, N., Liaw, C., and Mehrabian, A. Nearly-tight vc-dimension bounds for piecewise linear neural networks. In *Conference on learning theory*, pp. 1064–1068. PMLR, 2017.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

He, Y., Zhang, X., and Sun, J. Channel pruning for accelerating very deep neural networks. In *Proceedings of the IEEE international conference on computer vision*, pp. 1389–1397, 2017.

Hernández-Lobato, J. M. and Adams, R. Probabilistic backpropagation for scalable learning of bayesian neural networks. In *International conference on machine learning*, pp. 1861–1869. PMLR, 2015.

Hoffman, M. D., Gelman, A., et al. The no-u-turn sampler: adaptively setting path lengths in hamiltonian monte carlo. *J. Mach. Learn. Res.*, 15(1):1593–1623, 2014.

Izmailov, P., Vikram, S., Hoffman, M. D., and Wilson, A. G. G. What are bayesian neural network posteriors really like? In *International Conference on Machine Learning*, pp. 4629–4640. PMLR, 2021.

Jantre, S., Bhattacharya, S., and Maiti, T. Layer adaptive node selection in bayesian neural networks: Statistical guarantees and implementation details. *arXiv preprint arXiv:2108.11000*, 2021.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Kingma, D. P., Salimans, T., and Welling, M. Variational dropout and the local reparameterization trick. *Advances in neural information processing systems*, 28:2575–2583, 2015.

Kohler, M. and Langer, S. On the rate of convergence of fully connected deep neural network regression estimates. *The Annals of Statistics*, 49(4):2231–2249, 2021.

Kumar, A., Liang, P. S., and Ma, T. Verified uncertainty calibration. *Advances in Neural Information Processing Systems*, 32, 2019.

Lakshminarayanan, B., Pritzel, A., and Blundell, C. Simple and scalable predictive uncertainty estimation using deep ensembles. *Advances in neural information processing systems*, 30, 2017.

Lee, K. and Lee, J. Asymptotic properties for bayesian neural network in besov space. In *Advances in Neural Information Processing Systems*, 2022. URL `https://openreview.net/forum?id=y5ziOXtKybL`.

Lee, N., Ajanthan, T., and Torr, P. Snip: Single-shot network pruning based on connection sensitivity. In *International Conference on Learning Representations*, 2018.

Louizos, C., Ullrich, K., and Welling, M. Bayesian compression for deep learning. *Advances in neural information processing systems*, 30, 2017.

MacKay, D. J. A practical bayesian framework for backpropagation networks. *Neural computation*, 4(3):448–472, 1992.

Molchanov, D., Ashukha, A., and Vetrov, D. Variational dropout sparsifies deep neural networks. In *International Conference on Machine Learning*, pp. 2498–2507. PMLR, 2017.

Nalisnick, E., Hernández-Lobato, J. M., and Smyth, P. Dropout as a structured shrinkage prior. In *International Conference on Machine Learning*, pp. 4712–4722. PMLR, 2019.

Neal, R. M. *Bayesian learning for neural networks*, volume 118. Springer Science & Business Media, 2012.

Neal, R. M. et al. Mcmc using hamiltonian dynamics. *Handbook of markov chain monte carlo*, 2(11):2, 2011.

Polson, N. G. and Ročková, V. Posterior concentration for sparse deep learning. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pp. 938–949, 2018.

Qiu, J., Jammalamadaka, S. R., and Ning, N. Multivariate bayesian structural time series model. *J. Mach. Learn. Res.*, 19(1):2744–2776, 2018.

Schmidt-Hieber, J. Nonparametric regression using deep neural networks with relu activation function. *The Annals of Statistics*, 48(4):1875–1897, 2020.

Scott, S. L. and Varian, H. R. Predicting the present with bayesian structural time series. *International Journal of Mathematical Modelling and Numerical Optimisation*, 5 (1-2):4–23, 2014.

Shin, M., Cho, H., Min, H.-s., and Lim, S. Neural bootstrapper. *Advances in Neural Information Processing Systems*, 34:16596–16609, 2021.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958, 2014.

Tanaka, H., Kunin, D., Yamins, D. L., and Ganguli, S. Pruning neural networks without any data by iteratively conserving synaptic flow. *Advances in Neural Information Processing Systems*, 33:6377–6389, 2020.

Tsybakov, A. B. Introduction to nonparametric estimation, 2009.

Umrigar, C. Accelerated metropolis method. *Physical review letters*, 71(3):408, 1993.

van der Vaart, A. W. and van Zanten, J. H. Rates of contraction of posterior distributions based on gaussian process priors. *The Annals of Statistics*, 36(3):1435–1463, 2008.

Wang, C., Grosse, R., Fidler, S., and Zhang, G. Eigendamage: Structured pruning in the kronecker-factored eigenbasis. In *International Conference on Machine Learning*, pp. 6566–6575. PMLR, 2019.

Wang, H., Wang, N., and Yeung, D.-Y. Collaborative deep learning for recommender systems. In *Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining*, pp. 1235–1244, 2015.

Wang, Y., Deng, W., and Lin, G. Bayesian sparse learning with preconditioned stochastic gradient mcmc and its applications. *Journal of Computational Physics*, 432: 110134, 2021.

Welch, G., Bishop, G., et al. An introduction to the kalman filter. 1995.

Welling, M. and Teh, Y. W. Bayesian learning via stochastic gradient langevin dynamics. In *Proceedings of the 28th international conference on machine learning (ICML-11)*, pp. 681–688. Citeseer, 2011.

Wen, W., Wu, C., Wang, Y., Chen, Y., and Li, H. Learning structured sparsity in deep neural networks. *Advances in neural information processing systems*, 29:2074–2082, 2016.

Wenzel, F., Roth, K., Veeling, B., Swiatkowski, J., Tran, L., Mandt, S., Snoek, J., Salimans, T., Jenatton, R., and Nowozin, S. How good is the bayes posterior in deep neural networks really? In *International Conference on Machine Learning*, pp. 10248–10259. PMLR, 2020.

Wilson, A. G. and Izmailov, P. Bayesian deep learning and a probabilistic perspective of generalization. *Advances in neural information processing systems*, 33:4697–4708, 2020.

Wu, T.-Y., Rachel Wang, Y., and Wong, W. H. Mini-batch metropolis–hastings with reversible sgld proposal. *Journal of the American Statistical Association*, 117(537): 386–394, 2022.

Xie, F. and Xu, Y. Adaptive bayesian nonparametric regression using a kernel mixture of polynomials with application to partial linear models. *Bayesian Analysis*, 15(1): 159–186, 2020.

Yarotsky, D. Error bounds for approximations with deep relu networks. *Neural Networks*, 94:103–114, 2017.

Zanella, G. Informed proposals for local mcmc in discrete spaces. *Journal of the American Statistical Association*, 115(530):852–865, 2020.

Zhang, R., Li, C., Zhang, J., Chen, C., and Wilson, A. G. Cyclical stochastic gradient mcmc for bayesian deep

learning. In *International Conference on Learning Representations*, 2019.

Zhang, R., Cooper, A. F., and De Sa, C. M. Asymptotically optimal exact minibatch metropolis-hastings. *Advances in Neural Information Processing Systems*, 33:19500–19510, 2020.

Zhang, R., Liu, X., and Liu, Q. A langevin-like sampler for discrete distributions. In *International Conference on Machine Learning*, pp. 26375–26396. PMLR, 2022.

A. Proofs for Main Theorems

A.1. Additional notations

In this section, we describe additional notations not mentioned earlier.

We define $\mathcal{F}^{\mathrm{DNN}}(L, \boldsymbol{p}, F)$ as the function class of truncated DNNs with $(L, \boldsymbol{p})$ architecture :

$$\mathcal{F}^{\mathrm{DNN}}(L, \boldsymbol{p}, F) := \big\{ f : f = f^{\mathrm{DNN}}_{\boldsymbol{\theta}[-F,F]} \text{ is a DNN with } (L, \boldsymbol{p}) \\ \text{architecture truncated on } [-F, F] \big\}.$$

In a similar fashion, we define $\mathcal{F}^{\mathrm{mDNN}}(L, \boldsymbol{p}, F)$ as the function class of truncated mDNNs with $(L, \boldsymbol{p})$ architecture :

$$\mathcal{F}^{\mathrm{mDNN}}(L, \boldsymbol{p}, F) := \big\{ f : f = f^{\mathrm{mDNN}}_{\boldsymbol{M},\boldsymbol{\theta}[-F,F]} \text{ is a mDNN with } (L, \boldsymbol{p}) \\ \text{architecture truncated on } [-F, F] \big\}.$$

and $\mathcal{F}^{\mathrm{mDNN}}(L, \boldsymbol{p}, F, s)$ as

$$\mathcal{F}^{\mathrm{mDNN}}(L, \boldsymbol{p}, F, s) := \big\{ f : f = f^{\mathrm{mDNN}}_{\boldsymbol{M},\boldsymbol{\theta}[-F,F]} \in \mathcal{F}^{\mathrm{mDNN}}(L, \boldsymbol{p}, F),\ \max_{l\in[L]} |\boldsymbol{m}^{(l)}|_0 \le s \big\}.$$

For a real number $x \in \mathbb{R}$, we denote $\lceil x \rceil := \min\{z \in \mathbb{Z} : z \ge x\}$. For a vector $\boldsymbol{x} \in \mathbb{R}^d$ and m dimensional index vector $\boldsymbol{j} \subset [d]$, we denote $(\boldsymbol{x})_{\boldsymbol{j}} \in \mathbb{R}^m$ as the sub-vector whose elements are consist of $\boldsymbol{j}$th index of $\boldsymbol{x}$. Let $\boldsymbol{x}^{(n)} := (\boldsymbol{x}_i)_{i=1}^n$, $\boldsymbol{X}^{(n)} := (\boldsymbol{X}_i)_{i=1}^n$. For a real-valued function $f : \mathcal{X} \to \mathbb{R}$ and $1 \le p < \infty$, we denote $||f||_{p,n} := (\sum_{i=1}^n f(\boldsymbol{x}_i)^p/n)^{1/p}$ and $||f||_{p,\mathrm{P}_{\boldsymbol{X}}} := \big(\int_{\boldsymbol{X}\in\mathcal{X}} f(\boldsymbol{X})^p d\mathrm{P}_{\boldsymbol{X}}\big)^{1/p}$ where $\mathrm{P}_{\boldsymbol{X}}$ is a probability measure defined on input space $\mathcal{X}$.

For two positive sequences $\{a_n\}$ and $\{b_n\}$, we denote $a_n \lesssim b_n$ if there exists a positive sequence $C > 0$ such that $a_n \le C b_n$ for all $n \in \mathbb{N}$. We denote $a_n \asymp b_n$ if $a_n \lesssim b_n$ and $a_n \gtrsim b_n$ hold. We use the little o notation, that is, we write $a_n = o(b_n)$ if $\lim_{n\to\infty} a_n/b_n = 0$.

Let $\mathcal{F}$ be a set of functions $\mathcal{X} \to \mathbb{R}$ and d_n be a semimetric defined on $\mathcal{F}$. We denote $\mathcal{N}(\varepsilon, \mathcal{F}, d_n)$ and $\mathcal{M}(\varepsilon, \mathcal{F}, d_n)$ as the ε-covering number and ε-packing number of $\mathcal{F}$ w.r.t. d_n, respectively. Also, we denote $V^+_{\mathcal{F}}$ as the VC dimension of the set $\mathcal{F}^+ := \{\{(x,t) \in \mathcal{X}\times\mathbb{R}; t \le f(x)\}; f \in \mathcal{F}\}$.

A.2. Auxiliary lemmas

First, we describe a lemma that approximates Hölder smooth functions as DNN functions.

Lemma A.1 (Theorem 2 of Kohler & Langer (2021))**.** *There exists $C_L > 0$ and $C_p > 0$ only depending on d such that for every $f_0 \in \mathcal{H}^\beta_d$ with $\|f_0\|_\infty \le F$, there exist $f^{\mathrm{DNN}}_{\hat{\boldsymbol{\theta}}\ [-F,F]} \in \mathcal{F}^{\mathrm{DNN}}(L_n, \boldsymbol{v}_n, F)$ with*

$$\begin{aligned} L_n &:= \lceil C_L \log n \rceil, \\ v_n &:= \left\lceil C_p \left(n^{\frac{d}{2\beta+d}} (\log n)^{-1} \right)^{1/2} \right\rceil, \\ \boldsymbol{v}_n &:= (d, v_n, ..., v_n, 1)^\top \in \mathbb{N}^{L_n+2}, \end{aligned}$$

such that

$$\left\| f^{\mathrm{DNN}}_{\hat{\boldsymbol{\theta}}\ [-F,F]} - f_0 \right\|_\infty \lesssim n^{-\frac{\beta}{2\beta+d}} \log n$$

and

$$|\hat{\boldsymbol{\theta}}|_\infty \le n \tag{A.1}$$

hold.

Note that the upper bound (A.1) is not mentioned in statement of Kohler & Langer (2021), but it can be easily confirmed by following their proof. Next, we describe a standard tool for establishing concentration rates.

Lemma A.2 (Theorem 4 of Ghosal & Van Der Vaart (2007)). *Let $\left(\mathfrak{Y}^{(n)}, \mathcal{A}^{(n)}, P_\eta^{(n)} : \eta \in \mathcal{F}_n\right)$ be a sequence statistical experiments with observations $Y^{(n)}$. We consider the case where the observation $Y^{(n)}$ is a vector $Y^{(n)} = (Y_1, \ldots, Y_n)^\top$ of independent observations Y_i. We assume that the distribution $P_{\eta,i}$ of the ith component Y_i possesses a density $p_{\eta,i}$ relative to Lebesgue measure for $i \in [n]$. We define*

$$K_i(\eta_0, \eta) = \int \log(p_{\eta_0,i}/p_{\eta,i}) dP_{\eta_0,i},$$
$$V_{2,0;i}(\eta_0, \eta) = \int \left(\log(p_{\eta_0,i}/p_{\eta,i}) - K_i(\eta_0, \eta)\right)^2 dP_{\eta_0,i},$$

and

$$B_n^*(\eta_0, \varepsilon_n; 2) = \left\{\eta \in \mathcal{F}_n : \frac{1}{n}\sum_{i=1}^n K_i(\eta_0, \eta) \le \varepsilon_n^2,\ \frac{1}{n}\sum_{i=1}^n V_{2,0;i}(\eta_0, \eta) \le \varepsilon_n^2\right\}.$$

Let h_n be a semimetric on $\mathcal{F}_n$ with the property that there exist universal constants $\xi > 0$ and $K > 0$ such that for every $\varepsilon > 0$ and for each $\eta_1 \in \mathcal{F}_n$ with $h_n(\eta_1, \eta_0) > \varepsilon$, there exists a test ϕ_n such that

$$P_{\eta_0}^{(n)}\phi_n \le e^{-Kn\varepsilon^2}, \quad \sup_{\eta_2 \in \mathcal{F}_n : h_n(\eta_2, \eta_1) < \varepsilon\xi} P_{\eta_2}^{(n)}(1 - \phi_n) \le e^{-Kn\varepsilon^2}.$$

Let $\varepsilon_n > 0, \varepsilon_n \to 0$ and $(n\varepsilon_n^2)^{-1} = O(1)$. If for every sufficiently large $j \in \mathbb{N}$,

$$\sup_{\varepsilon > \varepsilon_n} \log N\left(\frac{1}{2}\varepsilon\xi, \{\eta \in \mathcal{F}_n : h_n(\eta, \eta_0) < \varepsilon\}, h_n\right) \le n\varepsilon_n^2,$$
$$\frac{\Pi_n(\eta \in \mathcal{F}_n : j\varepsilon_n < h_n(\eta, \eta_0) \le 2j\varepsilon_n)}{\Pi_n(B_n(\eta_0, \varepsilon_n; 2))} \le e^{Kn\varepsilon_n^2 j^2/2}$$

for all but finite many n, then we have that

$$P_{\eta_0}^{(n)}\Pi_n\left(\eta \in \mathcal{F}_n : h_n(\eta, \eta_0) \ge M_n\varepsilon_n \mid X^{(n)}\right) \to 0$$

for every $M_n \to \infty$.

Next, we state the lemma which describes an upper bound of supremum norm distance of two DNN functions whose parameters are similar.

Lemma A.3. *Consider two DNN models $f_{\boldsymbol{\theta}_1}^{\mathrm{DNN}} : [-1,1]^d \to \mathbb{R}$, $f_{\boldsymbol{\theta}_2}^{\mathrm{DNN}} : [-1,1]^d \to \mathbb{R}$ with $(L, \boldsymbol{p})$ architecture, where $L \in \mathbb{N}$ and $\boldsymbol{p} = (d, p, p, ..., p, 1)^\top \in \mathbb{N}^{L+2}$ for some $p \in \mathbb{N}$. If $|\boldsymbol{\theta}_1|_\infty \le B$, $|\boldsymbol{\theta}_2|_\infty \le B$ and $|\boldsymbol{\theta}_1 - \boldsymbol{\theta}_2|_\infty \le \delta$ holds for some $B > 0$ and $\delta > 0$, then*

$$\|f_{\boldsymbol{\theta}_1}^{\mathrm{DNN}} - f_{\boldsymbol{\theta}_2}^{\mathrm{DNN}}\|_\infty \le dp^L B^{L+1}(L+1)\delta$$

holds.

proof of Lemma A.3. Define

$$f_{\boldsymbol{\theta}_1}^{\mathrm{DNN}}(\cdot) = A_{L+1,1} \circ \rho \circ A_{L,1} \cdots \circ \rho \circ A_{1,1}(\cdot),$$
$$f_{\boldsymbol{\theta}_2}^{\mathrm{DNN}}(\cdot) = A_{L+1,2} \circ \rho \circ A_{L,2} \cdots \circ \rho \circ A_{1,2}(\cdot).$$

Also, we define $\boldsymbol{h}_{\boldsymbol{\theta},L'} : [-1,1]^d \to \mathbb{R}^{p_{L'}}$ for $L' \in [L]$ as the DNN model whose output is L'-th hidden layer of $f_{\boldsymbol{\theta}}^{\mathrm{DNN}}$. In other words,

$$\boldsymbol{h}_{\boldsymbol{\theta}_1,L'}(\cdot) = A_{L',1} \circ \rho \circ A_{L'-1,1} \cdots \circ \rho \circ A_{1,1}(\cdot),$$
$$\boldsymbol{h}_{\boldsymbol{\theta}_2,L'}(\cdot) = A_{L',2} \circ \rho \circ A_{L'-1,2} \cdots \circ \rho \circ A_{1,2}(\cdot).$$

We let $\boldsymbol{h}_{\boldsymbol{\theta},L+1}(\cdot) = f_{\boldsymbol{\theta}}^{\mathrm{DNN}}(\cdot)$. Since

$$\left\||\boldsymbol{h}_{\boldsymbol{\theta}_1,L'+1} - \boldsymbol{h}_{\boldsymbol{\theta}_2,L'+1}|_\infty\right\|_\infty \le p|\boldsymbol{\theta}_1 - \boldsymbol{\theta}_2|_\infty \left\||\boldsymbol{h}_{\boldsymbol{\theta}_1,L'}|_\infty\right\|_\infty + p|\boldsymbol{\theta}_2|_\infty \left\||\boldsymbol{h}_{\boldsymbol{\theta}_1,L'} - \boldsymbol{h}_{\boldsymbol{\theta}_2,L'}|_\infty\right\|_\infty$$

and

$$\left\| |\boldsymbol{h}_{\boldsymbol{\theta}_1, L'}|_\infty \right\|_\infty \leq dp^{L'-1}B^{L'}$$

hold, we can show that $|\boldsymbol{\theta}_1 - \boldsymbol{\theta}_2|_\infty \leq \delta$ implies

$$\left\| |\boldsymbol{h}_{\boldsymbol{\theta}_1, L'}(x) - \boldsymbol{h}_{\boldsymbol{\theta}_2, L'}(x)|_\infty \right\|_\infty \leq dp^{L'-1}B^{L'}L'\delta$$

for every $L' \in [L+1]$ by recursion. □

Lastly, we state the lemma about empirical process theory.

Lemma A.4 (Theorem 19.3 of Györfi et al. (2002)). *Let $\boldsymbol{X}, \boldsymbol{X}_1, \ldots, \boldsymbol{X}_n$ be independent and identically distributed random vectors with values in $\mathbb{R}^d$. Let $K_1, K_2 \geq 1$ be constants and let $\mathcal{G}$ be a class of functions $g : \mathbb{R}^d \to \mathbb{R}$ with*

$$|g(\boldsymbol{x})| \leq K_1, \quad \mathbb{E}(g(\boldsymbol{X})^2) \leq K_2 \mathbb{E}(g(\boldsymbol{X})).$$

Let $0 < \kappa < 1$ and $\alpha > 0$. Assume that

$$\sqrt{n}\kappa\sqrt{1-\kappa}\sqrt{\alpha} \geq 288 \max\left\{2K_1, \sqrt{2K_2}\right\}$$

and that, for all $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n \in \mathbb{R}^d$ and for all $t \geq \frac{\alpha}{8}$,

$$\frac{\sqrt{n}\kappa(1-\kappa)t}{96\sqrt{2}\max\{K_1, 2K_2\}} \geq \int_{\frac{\kappa(1-\kappa)t}{16\max\{K_1, 2K_2\}}}^{\sqrt{t}} \sqrt{\log \mathcal{N}\left(u, \left\{g \in \mathcal{G} : \frac{1}{n}\sum_{i=1}^n g\left(\boldsymbol{x}_i\right)^2 \leq 16t\right\}, ||\cdot||_{1,n}\right)} du.$$

Then,

$$\mathbf{P}\left\{\sup_{g \in \mathcal{G}} \frac{\left|\mathbb{E}\{g(\boldsymbol{X})\} - \frac{1}{n}\sum_{i=1}^n g\left(\boldsymbol{X}_i\right)\right|}{\alpha + \mathbb{E}\{g(\boldsymbol{X})\}} > \kappa\right\} \leq 60 \exp\left(-\frac{n\alpha\kappa^2(1-\kappa)}{128 \cdot 2304 \max\{K_1^2, K_2\}}\right).$$

A.3. Proof of Theorem 4.1

Let $\tau := \gamma - \frac{5}{2}$. It is enough to show the main statement for $0 < \tau < 1$. Note that $\varepsilon_n = n^{-\frac{\beta}{2\beta+d}}(\log n)^\gamma = n^{-\frac{\beta}{2\beta+d}}(\log n)^{\tau+\frac{5}{2}}$. For C_p defined in Lemma A.1, we define s_n as

$$s_n := \left\lceil C_p \left(n^{\frac{d}{2\beta+d}}(\log n)^{3\tau}\right)^{1/2} \right\rceil$$

and $\boldsymbol{s}_n = (d, s_n, \ldots, s_n, 1)^\top \in \mathbb{N}^{L_n+2}$. We define

$$T_n := (d+1)p_n + (L_n - 1)p_n(p_n+1) + (p_n+1)$$

and

$$S_n := (d+1)s_n + (L_n - 1)s_n(s_n+1) + (s_n+1)$$

as the numbers of parameters in the DNNs with $(L_n, \boldsymbol{p}_n)$ and $(L_n, \boldsymbol{s}_n)$ architectures, respectively. Let $\mathcal{F}_n$ be the set of pairs of truncated mDNN with $(L_n, \boldsymbol{p}_n)$ architecture and variances of the Gaussian noise,

$$\mathcal{F}_n := \left\{(f, \sigma^2)^\top \ : \ f \in \mathcal{F}^{\mathrm{mDNN}}(L_n, \boldsymbol{p}_n, F),\ 0 < \sigma^2 \leq \sigma_{\max}^2\right\}.$$

Also, we let $\mathcal{F}_n' \subset \mathcal{F}_n$ by

$$\mathcal{F}_n' := \left\{(f, \sigma^2)^\top \ : \ f \in \mathcal{F}^{\mathrm{mDNN}}(L_n, \boldsymbol{p}_n, F, s_n),\ 0 < \sigma^2 \leq \sigma_{\max}^2\right\}.$$

In the first step of the proof, we fix $\{\boldsymbol{x}^{(n)}\}_{n=1}^{\infty}$ and show

$$\mathbb{E}_0\left[\Pi_n\left((f,\sigma^2)^\top \in \mathcal{F}_n' : ||f-f_0||_{2,n} + |\sigma^2-\sigma_0^2| > M_n\varepsilon_n \middle| \mathcal{D}^{(n)}\right)\middle| \boldsymbol{X}^{(n)} = \boldsymbol{x}^{(n)}\right] \to 0 \tag{A.2}$$

as $n \to \infty$ for any $M_n \to \infty$.

In the second step of proof, we extend empirical L_2 error to expected L_2 error. In other words, we show

$$\mathbb{E}_0\left[\Pi_n\left((f,\sigma^2)^\top \in \mathcal{F}_n' : ||f-f_0||_{2,\mathrm{P}_X} + |\sigma^2-\sigma_0^2| > M_n\varepsilon_n \middle| \mathcal{D}^{(n)}\right)\right] \to 0 \tag{A.3}$$

as $n \to \infty$ for any $M_n \to \infty$.

In the last step of proof, we show

$$\mathbb{E}_0\left[\Pi_n\left((f,\sigma^2)^\top \in (\mathcal{F}_n \setminus \mathcal{F}_n') \middle| \mathcal{D}^{(n)}\right)\right] \to 0. \tag{A.4}$$

and the proof of Theorem 4.1 is done by (A.3) and (A.4).

Step 1 For fixed $\{\boldsymbol{x}^{(n)}\}_{n=1}^{\infty}$, let $P_{(f,\sigma^2),i}$ and $p_{(f,\sigma^2),i}$ be the probability measure and density corresponding to Gaussian distribution $N(f(\boldsymbol{x}_i),\sigma^2)$, respectively. We define the semimetric h_n^2 on $\mathcal{F}_n'$ as the average of the squares of the Hellinger distances for the distributions of the n individual observations. In other words, for $(f_1,\sigma_1^2),(f_2,\sigma_2^2) \in \mathcal{F}_n'$,

$$h_n^2\left((f_1,\sigma_1^2),(f_2,\sigma_2^2)\right) := \frac{1}{n}\sum_{i=1}^{n}\int\left(\sqrt{p_{(f_1,\sigma_1^2),i}} - \sqrt{p_{(f_2,\sigma_2^2),i}}\right)^2 dP_{(f_1,\sigma_1^2),i}.$$

Note that h_n^2 satisfies

$$\begin{aligned}(||f_1-f_2||_{2,n} + |\sigma_1^2-\sigma_2^2|)^2 &\le 2\left(||f_1-f_2||_{2,n}^2 + |\sigma_1^2-\sigma_2^2|^2\right)\\ &\lesssim h_n^2\left((f_1,\sigma_1^2),(f_2,\sigma_2^2)\right).\end{aligned}$$

by Lemma B.1 of Xie & Xu (2020). Hence, to prove (A.2), it is suffices to show

$$\mathbb{E}_0\left[\Pi_n\left((f,\sigma^2)^\top \in \mathcal{F}_n' : h_n\left((f,\sigma^2),(f_0,\sigma_0^2)\right) > M_n\varepsilon_n \middle| \mathcal{D}^{(n)}\right)\middle| \boldsymbol{X}^{(n)} = \boldsymbol{x}^{(n)}\right] \to 0. \tag{A.5}$$

Since Hellinger distance possesses an exponentially powerful local test with respect to both the type-I and type-II errors (Lemma 2 of Ghosal & Van Der Vaart (2007)), we can use the standard tool to establish concentration rates that we restate in Lemma A.2 for the convenience of the reader.

If we define semimetric d_n on $\mathcal{F}_n'$ as

$$d_n^2\left((f_1,\sigma_1^2),(f_2,\sigma_2^2)\right) := ||f_1-f_2||_{1,n} + |\sigma_1^2-\sigma_2^2|^2,$$

then $h_n^2(\cdot) \lesssim d_n^2(\cdot)$ holds by by Lemma B.1 of Xie & Xu (2020) and hence $\mathcal{N}\left(\varepsilon,\mathcal{F}_n',h_n\right) \le \mathcal{N}\left(\varepsilon^2,\mathcal{F}_n',d_n^2\right)$. Also, by the fact that $||f_1-f_2||_{1,n} \le \frac{\varepsilon^2}{2}$ and $|\sigma_1^2-\sigma_2^2|^2 \le \frac{\varepsilon^2}{2}$ implies $||f_1-f_2||_{1,n} + |\sigma_1^2-\sigma_2^2|^2 \le \varepsilon^2$ and for every $f_{\boldsymbol{M},\boldsymbol{\theta}\,[-F,F]}^{\mathrm{mDNN}} \in \mathcal{F}^{\mathrm{mDNN}}(L_n,\boldsymbol{p}_n,F,s_n)$ there exist $\boldsymbol{\psi}_{\boldsymbol{M},\boldsymbol{\theta}} \in \mathbb{R}^{S_n}$ and $f_{\boldsymbol{\psi}_{\boldsymbol{M},\boldsymbol{\theta}}\,[-F,F]}^{\mathrm{DNN}} \in \mathcal{F}^{\mathrm{DNN}}(L_n,\boldsymbol{s}_n,F)$ such that $f_{\boldsymbol{M},\boldsymbol{\theta}\,[-F,F]}^{\mathrm{mDNN}} = f_{\boldsymbol{\psi}_{\boldsymbol{M},\boldsymbol{\theta}}\,[-F,F]}^{\mathrm{DNN}}$ holds, we get

$$\begin{aligned}\mathcal{N}\left(\varepsilon,\mathcal{F}_n',h_n\right) \le& \mathcal{N}\left(\varepsilon^2,\mathcal{F}_n',d_n^2\right)\\ \le& \mathcal{N}\left(\frac{\varepsilon^2}{2},\mathcal{F}^{\mathrm{mDNN}}(L_n,\boldsymbol{p}_n,F,s_n),||\cdot||_{1,n}\right)\frac{\sqrt{2}\sigma_{max}^2}{\varepsilon}\\ \le& \mathcal{N}\left(\frac{\varepsilon^2}{2},\mathcal{F}^{\mathrm{DNN}}(L_n,\boldsymbol{s}_n,F),||\cdot||_{1,n}\right)\frac{\sqrt{2}\sigma_{max}^2}{\varepsilon}.\end{aligned}$$

Since functions in $\mathcal{F}^{DNN}(L_n, \boldsymbol{s}_n, F)$ are bounded by $[-F, F]$, there exists $c_1 > 0$ such that

$$\begin{aligned}\mathcal{N}\left(\frac{\varepsilon^2}{2}, \mathcal{F}^{\mathrm{DNN}}(L_n, \boldsymbol{s}_n, F), ||\cdot||_{1,n}\right) \leq & \mathcal{M}\left(\frac{\varepsilon^2}{2}, \mathcal{F}^{DNN}(L_n, \boldsymbol{s}_n, F), ||\cdot||_{1,n}\right) \\ \leq & 3\left(\frac{8eF}{\epsilon^2}\log\frac{12eF}{\epsilon^2}\right)^{V^+_{\mathcal{F}^{\mathrm{DNN}}(L_n, \boldsymbol{s}_n, F)}} \\ \leq & 3\left(\frac{8eF}{\epsilon^2}\log\frac{12eF}{\epsilon^2}\right)^{c_1 L_n S_n \log S_n}\end{aligned}$$

holds for every $\varepsilon > 0$ by Theorem 9.4 of Györfi et al. (2002) and Theorem 6 of Harvey et al. (2017). Hence,

$$\begin{aligned}\sup_{\varepsilon > \varepsilon_n} \log \mathcal{N}\Big(\varepsilon, \mathcal{F}'_n, h_n\Big) \lesssim & L_n S_n \log S_n \log n \\ \asymp & n^{\frac{d}{2\beta+d}} (\log n)^{4+3\tau} \\ \leq & n\varepsilon_n^2 \end{aligned} \tag{A.6}$$

holds.

Now, we define

$$\begin{aligned}K_i((f_0, \sigma_0^2), (f, \sigma^2)) &= \int \log(p_{(f_0,\sigma_0^2),i}/p_{(f,\sigma^2),i}) dP_{(f_0,\sigma_0^2),i}, \\ V_{2,0;i}((f_0, \sigma_0^2), (f, \sigma^2)) &= \int \Big(\log(p_{(f_0,\sigma_0^2),i}/p_{(f,\sigma^2),i} - K_i((f_0, \sigma_0^2), (f, \sigma^2))\Big)^2 dP_{(f_0,\sigma_0^2),i}\end{aligned}$$

and

$$\begin{aligned}B_n^*\left((f_0, \sigma_0^2), \varepsilon; 2\right) = \big\{(f, \sigma^2) \in \mathcal{F}'_n : & \frac{1}{n}\sum_{i=1}^n K_i\left((f_0, \sigma_0^2), (f, \sigma^2)\right) \leq \varepsilon^2, \\ & \frac{1}{n}\sum_{i=1}^n V_{2,0;i}\left((f_0, \sigma_0^2), (f, \sigma^2)\right) \leq \varepsilon^2\big\}.\end{aligned}$$

with notations on Lemma A.2. In addition, for $\varepsilon > 0$, define

$$\begin{aligned}A_n^*\left((f_0, \sigma_0^2), \varepsilon; 2\right) := \Big\{(f, \sigma^2) \in \mathcal{F}'_n : & \max_i |f(\boldsymbol{x}_i) - f_0(\boldsymbol{x}_i)| \leq \frac{\sigma_0 \varepsilon}{2}, \\ & \sigma^2 \in [\sigma_0^2, (1+\varepsilon^2)\sigma_0^2]\Big\}.\end{aligned}$$

Then for every $f \in A_n^*\left((f_0, \sigma_0^2), \varepsilon; 2\right)$ and $i \in [n]$,

$$K_i((f_0, \sigma_0^2), (f, \sigma^2)) = \frac{1}{2}\log\frac{\sigma^2}{\sigma_0^2} + \frac{\sigma_0^2 + (f_0(\boldsymbol{x}_i) - f(\boldsymbol{x}_i))^2}{2\sigma^2} - \frac{1}{2} \leq \varepsilon^2$$

and

$$\begin{aligned}V_{2,0;i}((f_0, \sigma_0^2), (f, \sigma^2)) &= \mathrm{Var}_{f_0,\sigma_0^2}\left(-\frac{(Y_i - f_0(\boldsymbol{x}_i))^2}{2\sigma_0^2} + \frac{(Y_i - f(\boldsymbol{x}_i))^2}{2\sigma^2}\right) \\ &= \mathrm{Var}_{f_0,\sigma_0^2}\left(-\frac{1}{2}(1 - \frac{\sigma_0^2}{\sigma^2})Z_i^2 + \frac{\sigma_0(f_0(\boldsymbol{x}_i) - f(\boldsymbol{x}_i))Z_i}{\sigma^2}\right) \leq \varepsilon^2\end{aligned}$$

where $Y_i \sim N(f_0(\boldsymbol{x}_i), \sigma_0^2)$ and $Z_i := \frac{Y_i - f_0(\boldsymbol{x}_i)}{\sigma_0} \sim N(0, 1)$. Hence, we can conclude that

$$A_n^*\left((f_0, \sigma_0^2), \varepsilon_n; 2\right) \subset B_n^*\left((f_0, \sigma_0^2), \varepsilon_n; 2\right). \tag{A.7}$$

Now we define v_n as

$$v_n := \left\lceil C_p \left(n^{\frac{d}{2\beta+d}} (\log n)^{-1} \right)^{1/2} \right\rceil,$$

which is the width of the network in Lemma A.1. Let $\boldsymbol{v}_n = (d, v_n, \ldots, v_n, 1)^\top \in \mathbb{N}^{L_n+2}$, and let

$$V_n := (d+1)v_n + (L_n - 1)v_n(v_n+1) + (v_n+1),$$

which is the number of parameters in DNN with $(L_n, \boldsymbol{v}_n)$ architecture. For any $\boldsymbol{\theta} \in \mathbb{R}^{T_n}$ and any $\boldsymbol{M}$ with $|\boldsymbol{m}^{(l)}|_0 = v_n$ for $l \in [L]$, there exists V_n dimension index vector $\boldsymbol{j}_{\boldsymbol{M}} \subset [T_n]$ such that $f^{\mathrm{DNN}}_{\boldsymbol{\psi}_{\boldsymbol{M},\boldsymbol{\theta}}} = f^{\mathrm{mDNN}}_{\boldsymbol{M},\boldsymbol{\theta}}$ holds for $\boldsymbol{\psi}_{\boldsymbol{M},\boldsymbol{\theta}} := (\boldsymbol{\theta})_{\boldsymbol{j}_{\boldsymbol{M}}} \in \mathbb{R}^{V_n}$. In other words, $f^{\mathrm{DNN}}_{\boldsymbol{\psi}_{\boldsymbol{M},\boldsymbol{\theta}}}$ is the sub-network of $f^{\mathrm{DNN}}_{\boldsymbol{M},\boldsymbol{\theta}}$ consisting of the unmasked nodes. Also, there exists $\hat{\boldsymbol{\psi}} \in [-n, n]^{V_n}$ such that

$$\left\| f^{\mathrm{DNN}}_{\hat{\boldsymbol{\psi}}\ [-F,F]} - f_0 \right\|_\infty < \frac{\sigma_0 \varepsilon_n}{4} \tag{A.8}$$

satisfies for large n by Lemma A.1.

With (A.7), (A.8) and Lemma A.3, we can obtain the lower bound of $\Pi_n\left(B_n^*\left((f_0, \sigma_0^2), \varepsilon_n; 2\right)\right)$ by

$$\begin{aligned}
&\Pi_n\left(B_n^*\left((f_0, \sigma_0^2), \varepsilon_n; 2\right)\right) \\
\geq &\Pi_n\left(A_n^*\left((f_0, \sigma_0^2), \varepsilon_n; 2\right)\right) \\
= &\Pi_n\Big(\Big\{ (\boldsymbol{M}, \boldsymbol{\theta}) : \max_i |f^{\mathrm{mDNN}}_{\boldsymbol{M},\boldsymbol{\theta}\ [-F,F]}(\boldsymbol{x}_i) - f_0(\boldsymbol{x}_i)| \leq \frac{\sigma_0 \varepsilon_n}{2} \Big\} \Big) \Pi_n\Big(\sigma^2 \in [\sigma_0^2, (1+\varepsilon_n^2)\sigma_0^2] \Big) \\
\geq &\Pi_n\left(\left\{ (\boldsymbol{M}, \boldsymbol{\theta}) : |\boldsymbol{m}^{(1)}|_0 = \cdots = |\boldsymbol{m}^{(L)}|_0 = v_n,\ \max_i \left| f^{\mathrm{DNN}}_{\boldsymbol{\psi}_{\boldsymbol{M},\boldsymbol{\theta}}\ [-F,F]}(\boldsymbol{x}_i) - f_0(\boldsymbol{x}_i) \right| \leq \frac{\sigma_0 \varepsilon_n}{2} \right\} \right) \\
&\quad \times \Pi_n\Big(\sigma^2 \in [\sigma_0^2, (1+\varepsilon_n^2)\sigma_0^2] \Big) \\
\geq &\Pi_n\left(\left\{ (\boldsymbol{M}, \boldsymbol{\theta}) : |\boldsymbol{m}^{(1)}|_0 = \cdots = |\boldsymbol{m}^{(L)}|_0 = v_n,\ \max_i \left| f^{\mathrm{DNN}}_{\boldsymbol{\psi}_{\boldsymbol{M},\boldsymbol{\theta}}\ [-F,F]}(\boldsymbol{x}_i) - f^{\mathrm{DNN}}_{\hat{\boldsymbol{\psi}}\ [-F,F]}(\boldsymbol{x}_i) \right| \leq \frac{\sigma_0 \varepsilon_n}{4} \right\} \right) \\
&\quad \times \Pi_n\Big(\sigma^2 \in [\sigma_0^2, (1+\varepsilon_n^2)\sigma_0^2] \Big) \\
\geq &\Pi_n\left(\left\{ (\boldsymbol{M}, \boldsymbol{\theta}) : |\boldsymbol{m}^{(1)}|_0 = \cdots = |\boldsymbol{m}^{(L)}|_0 = v_n,\ \left| \boldsymbol{\psi}_{\boldsymbol{M},\boldsymbol{\theta}} - \hat{\boldsymbol{\psi}} \right|_\infty \leq \frac{\sigma_0 \varepsilon_n}{4 d v_n^{L_n} n^{L_n+1} (L_n+1)} \right\} \right) \\
&\quad \times \Pi_n\Big(\sigma^2 \in [\sigma_0^2, (1+\varepsilon_n^2)\sigma_0^2] \Big) \\
\gtrsim &\exp\left(-(\lambda \log n)^5 v_n^2 L_n \right) \exp\left(-V_n (\log n)^2 \right) \varepsilon_n^2 \\
\gtrsim &\exp\left(-\lambda^5 C_p^2 C_L (\log n)^5 n^{\frac{d}{2\beta+d}} \right) \exp\left(-C_p^2 C_L (\log n)^2 n^{\frac{d}{2\beta+d}} \right) n^{-1}.
\end{aligned} \tag{A.9}$$

Hence we have that

$$\Pi_n\left(B_n^*\left(\eta_0, \varepsilon_n; 2\right)\right) \geq e^{-n\varepsilon_n^2} \tag{A.10}$$

for all but finite many n. Hence by (A.6), (A.10) and Lemma A.2, the proof of (A.5) is done. □

Step 2. Since (A.2) holds for arbitrary $\{\boldsymbol{x}^{(n)}\}_{n=1}^\infty$,

$$\mathbb{E}_0\left[\Pi_n\left((f, \sigma^2)^\top \in \mathcal{F}_n' : ||f - f_0||_{2,n} + |\sigma^2 - \sigma_0^2| > M_n \varepsilon_n \Big| \mathcal{D}^{(n)} \right) \right] \to 0 \tag{A.11}$$

also holds. Next, we will check the conditions in Lemma A.4 for

$$\begin{aligned}
\mathcal{G} :=& \left\{ g \ :\ g = (f^{\mathrm{mDNN}}_{\boldsymbol{M},\boldsymbol{\theta}\ [-F,F]} - f_0)^2, f^{\mathrm{mDNN}}_{\boldsymbol{M},\boldsymbol{\theta}\ [-F,F]} \in \mathcal{F}^{\mathrm{mDNN}}(L_n, \boldsymbol{p}_n, F, s_n) \right\}, \\
\kappa :=& \frac{1}{2},\ \alpha := \varepsilon_n^2,\ K_1 = K_2 = 4F^2.
\end{aligned}$$

First, it is easy to check $||g(\boldsymbol{x})||_\infty \leq 4F^2$ and $\mathbb{E}(g(\boldsymbol{X})^2) \leq 4F^2\mathbb{E}(g(\boldsymbol{X}))$ for $g \in \mathcal{G}$. Also, for every $f^{\mathrm{mDNN}}_{\boldsymbol{M},\boldsymbol{\theta}\,[-F,F]} \in \mathcal{F}^{\mathrm{mDNN}}(L_n, \boldsymbol{p}_n, F, s_n)$, there exist $\boldsymbol{\psi}_{\boldsymbol{M},\boldsymbol{\theta}} \in \mathbb{R}^{S_n}$ and $f^{\mathrm{DNN}}_{\boldsymbol{\psi}_{\boldsymbol{M},\boldsymbol{\theta}}\,[-F,F]} \in \mathcal{F}^{\mathrm{DNN}}(L_n, \boldsymbol{s}_n, F)$ such that $f^{\mathrm{mDNN}}_{\boldsymbol{M},\boldsymbol{\theta}\,[-F,F]} = f^{\mathrm{DNN}}_{\boldsymbol{\psi}_{\boldsymbol{M},\boldsymbol{\theta}}\,[-F,F]}$ holds. Since

$$\left\|(f^{\mathrm{DNN}}_{\boldsymbol{\psi}_1\,[-F,F]} - f_0)^2 - (f^{\mathrm{DNN}}_{\boldsymbol{\psi}_2\,[-F,F]} - f_0)^2\right\|_{n,1} \leq 4F\left\|f^{\mathrm{DNN}}_{\boldsymbol{\psi}_1\,[-F,F]} - f^{\mathrm{DNN}}_{\boldsymbol{\psi}_2\,[-F,F]}\right\|_{n,1}$$

holds for $\boldsymbol{\psi}_1, \boldsymbol{\psi}_2 \in \mathbb{R}^{S_n}$, there exists $c_2 > 0$ such that

$$\begin{aligned}
\mathcal{N}\left(u, \mathcal{G}, ||\cdot||_{n,1}\right) \leq & \mathcal{N}\left(\frac{u}{4F}, \mathcal{F}^{\mathrm{DNN}}(L_n, \boldsymbol{s}_n, F), ||\cdot||_{n,1}\right) \\
\leq & \mathcal{M}\left(\frac{u}{4F}, \mathcal{F}^{\mathrm{DNN}}(L_n, \boldsymbol{s}_n, F), ||\cdot||_{n,1}\right) \\
\leq & 3\left(\frac{16eF^2}{u}\log\frac{24eF^2}{u}\right)^{V^+_{\mathcal{F}^{\mathrm{DNN}}(L_n,\boldsymbol{s}_n,F)}} \\
\lesssim & n^{c_2 S_n L_n \log S_n}
\end{aligned}$$

for $u \geq n^{-1}$ by Theorem 9.4 of Györfi et al. (2002) and Theorem 6 of Harvey et al. (2017). Hence for all $t \geq \frac{\varepsilon_n^2}{8}$,

$$\begin{aligned}
\int_{\frac{\kappa(1-\kappa)t}{16\max\{K_1,2K_2\}}}^{\sqrt{t}} \sqrt{\log\mathcal{N}\left(u, \mathcal{G}, ||\cdot||_{n,1}\right)}du \lesssim & \sqrt{t}\left(n^{\frac{d}{2\beta+d}}(\log n)^{4+3\tau}\right)^{\frac{1}{2}} \\
= & o\left(\frac{\sqrt{n}t/4}{96\sqrt{2}\max\{K_1, 2K_2\}}\right)
\end{aligned}$$

holds. To sum up, we conclude that

$$\mathbf{P}\left\{\sup_{f\in\mathcal{F}^{\mathrm{DNN}}(L_n,\boldsymbol{p}'_n)} \frac{\left|||f-f_0||^2_{2,\mathrm{P}_X} - ||f-f_0||^2_{2,n}\right|}{\varepsilon_n^2 + ||f-f_0||^2_{2,\mathrm{P}_X}} > \frac{1}{2}\right\} \leq 60\exp\left(-\frac{n\varepsilon_n^2/8}{128\cdot 2304\cdot 16F^4}\right) \tag{A.12}$$

holds for all but finite many n by Lemma A.4. Hence by (A.11) and (A.12), the proof of (A.3) is done. □

Step 3. Since

$$\left(\frac{1}{2ks} - \frac{1}{4k^2s^3}\right)e^{-ks^2} \leq \int_s^\infty e^{-kt^2}dt \leq \frac{1}{2ks}e^{-ks^2}$$

for any $k > 0$ and $s > 0$,

$$\begin{aligned}
\Pi_n(|\boldsymbol{m}^{(l)}|_0 > s_n) \leq & \frac{\sum_{s=s_n+1}^{p_n} e^{-(\lambda\log n)^5 s^2}}{e^{-(\lambda\log n)^5}} \\
\lesssim & e^{-(\lambda\log n)^5 {s_n}^2} e^{(\lambda\log n)^5}(\log n)^{-5}
\end{aligned}$$

holds for every $l \in [L]$. Then we have that

$$\begin{aligned}
\Pi_n\left(\mathcal{F}_n \setminus \mathcal{F}'_n\right) = & \Pi_n\left(\bigcup_{l\in[L]}\{|\boldsymbol{m}^{(l)}|_0 > s_n\}\right) \\
\leq & L_n \cdot \Pi_n\left(\{|\boldsymbol{m}^{(1)}|_0 > s_n\}\right) \\
\lesssim & \exp\left(-\lambda^5 C_p^2(\log n)^{5+3\tau} n^{\frac{d}{2\beta+d}}\right)\exp\left((\lambda\log n)^5\right)(\log n)^{-4}.
\end{aligned} \tag{A.13}$$

By (A.9) and (A.13), we obtain

$$\begin{aligned}
&\frac{\Pi_n\left(\mathcal{F}_n \setminus \mathcal{F}'_n\right)}{\Pi_n\left(B_n^\star(\eta_0, \varepsilon_n; 2)\right)e^{-2n\varepsilon_n^2}} \\
&\lesssim \frac{\exp\left(-\lambda^5 C_p^2(\log n)^{5+3\tau} n^{\frac{d}{2\beta+d}}\right)\exp\left((\lambda\log n)^5\right)}{\exp(-\lambda^5 C_p^2 C_L(\log n)^5 n^{\frac{d}{2\beta+d}})\exp(-C_p^2 C_L(\log n)^2 n^{\frac{d}{2\beta+d}})n^{-1}\exp(-2(\log n)^{5+2\tau}n^{\frac{d}{2\beta+d}})} \\
&= o(1),
\end{aligned}$$

and thus we have

$$\mathbb{E}_0\left[\Pi_n\left((f,\sigma^2)^\top \in (\mathcal{F}_n \setminus \mathcal{F}_n') \middle| \mathcal{D}^{(n)}\right)\right] \to 0.$$

by apply Lemma 1 of Ghosal & Van Der Vaart (2007), which completes the proof of (A.4). □

A.4. Proof of Theorem 4.2

The proof is a slight modification of the proof of Theorem 4.1. Let $\tau := \gamma - \frac{5}{2}$. It suffices to show the main statement for $0 < \tau < 1$. Note that $\varepsilon_n = n^{-\frac{\beta}{2\beta+d}}(\log n)^\gamma = n^{-\frac{\beta}{2\beta+d}}(\log n)^{\tau+\frac{5}{2}}$. For C_p defined in Lemma A.1, we define s_n as

$$s_n := \left\lceil C_p\left(n^{\frac{d}{2\beta+d}}(\log n)^{3\tau}\right)^{1/2}\right\rceil$$

and $\boldsymbol{s}_n = (d, s_n, \ldots, s_n, 1)^\top \in \mathbb{N}^{L_n+2}$. We define

$$T_n := (d+1)p_n + (L_n-1)p_n(p_n+1) + (p_n+1)$$

and

$$S_n := (d+1)s_n + (L_n-1)s_n(s_n+1) + (s_n+1)$$

as the numbers of parameters in the DNNs with $(L_n, \boldsymbol{p}_n)$ and $(L_n, \boldsymbol{s}_n)$ architectures, respectively. Let $\mathcal{F}_n$ be the set of truncated mDNN with the $(L_n, \boldsymbol{p}_n)$ architecture,

$$\mathcal{F}_n := \mathcal{F}^{\mathrm{mDNN}}(L_n, \boldsymbol{p}_n, F).$$

Also, we let $\mathcal{F}_n' \subset \mathcal{F}_n$ by

$$\mathcal{F}_n' := \mathcal{F}^{\mathrm{mDNN}}(L_n, \boldsymbol{p}_n, F, s_n)$$

In the first step of proof, we fix $\{\boldsymbol{x}^{(n)}\}_{n=1}^\infty$ and show

$$\mathbb{E}_0\left[\Pi_n\left(f \in \mathcal{F}_n' : ||\phi\circ f - \phi\circ f_0||_{2,n} > M_n\varepsilon_n \middle| \mathcal{D}^{(n)}\right) \middle| \boldsymbol{X}^{(n)} = \boldsymbol{x}^{(n)}\right] \to 0 \tag{A.14}$$

as $n \to \infty$ for any $M_n \to \infty$.

In the second step of proof, we extend empirical L_2 error to expected L_2 error. In other words, we show

$$\mathbb{E}_0\left[\Pi_n\left(f \in \mathcal{F}_n' : ||\phi\circ f - \phi\circ f_0||_{2,\mathrm{P}_X} > M_n\varepsilon_n \middle| \mathcal{D}^{(n)}\right)\right] \to 0 \tag{A.15}$$

as $n \to \infty$ for any $M_n \to \infty$.

Then, since we already showed

$$\mathbb{E}_0\left[\Pi_n\left(f \in (\mathcal{F}_n \setminus \mathcal{F}_n') \middle| \mathcal{D}^{(n)}\right)\right] \to 0 \tag{A.16}$$

in the last step of the proof of Theorem 4.1, the proof of Theorem 4.2 is done by (A.15) and (A.16).

Step 1 For fixed $\{\boldsymbol{x}^{(n)}\}_{n=1}^\infty$, let $P_{f,i}$ and $p_{f,i}$ be the probability measure and density corresponding to the Bernoulli distribution $\mathrm{Ber}(\phi\circ f(\boldsymbol{x}_i))$, respectively. We define the semimetric h_n^2 on $\mathcal{F}_n'$ as the average of the squares of the Hellinger distances for the distributions of the n individual observations. In other words, for $f_1, f_2 \in \mathcal{F}_n'$,

$$h_n^2(f_1, f_2) := \frac{1}{n}\sum_{i=1}^n \int \left(\sqrt{p_{f_1,i}} - \sqrt{p_{f_2,i}}\right)^2 dP_{f_1,i}.$$

Also, we define semimetric d_n on $\mathcal{F}_n'$ as

$$d_n(f_1, f_2) := ||\phi\circ f_1 - \phi\circ f_2||_{2,n}.$$

Note that since $f_1, f_2 \in \mathcal{F}'_n$ are bounded,

$$\begin{aligned} d_n^2(f_1, f_2) =& \frac{1}{n}\sum_{i=1}^{n}(\phi \circ f_1(\boldsymbol{x}_i) - \phi \circ f_2(\boldsymbol{x}_i))^2 \\ =& \frac{1}{n}\sum_{i=1}^{n}\left(\sqrt{\phi \circ f_1(\boldsymbol{x}_i)} - \sqrt{\phi \circ f_2(\boldsymbol{x}_i)}\right)^2\left(\sqrt{\phi \circ f_1(\boldsymbol{x}_i)} + \sqrt{\phi \circ f_2(\boldsymbol{x}_i)}\right)^2 \\ \lesssim& h_n^2(f_1, f_2) \end{aligned}$$

and

$$\begin{aligned} d_n^2(f_1, f_2) =& \frac{1}{2n}\sum_{i=1}^{n}\Bigg\{\left(\sqrt{\phi \circ f_1(\boldsymbol{x}_i)} - \sqrt{\phi \circ f_2(\boldsymbol{x}_i)}\right)^2\left(\sqrt{\phi \circ f_1(\boldsymbol{x}_i)} + \sqrt{\phi \circ f_2(\boldsymbol{x}_i)}\right)^2 \\ &+ \left(\sqrt{1-\phi \circ f_1(\boldsymbol{x}_i)} - \sqrt{1-\phi \circ f_2(\boldsymbol{x}_i)}\right)^2\left(\sqrt{1-\phi \circ f_1(\boldsymbol{x}_i)} + \sqrt{1-\phi \circ f_2(\boldsymbol{x}_i)}\right)^2\Bigg\} \\ \gtrsim& h_n^2(f_1, f_2) \end{aligned}$$

holds. Hence, to prove (A.14), it suffices to show

$$\mathbb{E}_0\left[\Pi_n\left((f, \sigma^2)^\top \in \mathcal{F}'_n : h_n(f, f_0) > M_n\varepsilon_n \middle| \mathcal{D}^{(n)}\right) \middle| \boldsymbol{X}^{(n)} = \boldsymbol{x}^{(n)}\right] \to 0. \tag{A.17}$$

Since Hellinger distance possesses an exponentially powerful local test with respect to both the type-I and type-II errors (Lemma 2 of Ghosal & Van Der Vaart (2007)), we can use standard tools to establish concentration rates that we restate in Lemma A.2 for the convenience of the reader.

Since ϕ is L1-Lipschitz function,

$$\begin{aligned} \mathcal{N}(\varepsilon, \mathcal{F}'_n, h_n) \lesssim& \mathcal{N}(\varepsilon, \mathcal{F}'_n, d_n) \\ \leq& \mathcal{N}(\varepsilon, \mathcal{F}'_n, ||\cdot||_{2,n}). \end{aligned}$$

For every $f^{\mathrm{mDNN}}_{\boldsymbol{M},\boldsymbol{\theta}\ [-F,F]} \in \mathcal{F}^{\mathrm{mDNN}}(L_n, \boldsymbol{p}_n, F, s_n)$ there exist $\boldsymbol{\psi}_{\boldsymbol{M},\boldsymbol{\theta}} \in \mathbb{R}^{S_n}$ and $f^{\mathrm{DNN}}_{\boldsymbol{\psi}_{\boldsymbol{M},\boldsymbol{\theta}}\ [-F,F]} \in \mathcal{F}^{\mathrm{DNN}}(L_n, \boldsymbol{s}_n, F)$ such that $f^{\mathrm{mDNN}}_{\boldsymbol{M},\boldsymbol{\theta}\ [-F,F]} = f^{\mathrm{DNN}}_{\boldsymbol{\psi}_{\boldsymbol{M},\boldsymbol{\theta}}\ [-F,F]}$ holds. So we get

$$\begin{aligned} \mathcal{N}(\varepsilon, \mathcal{F}'_n, ||\cdot||_{2,n}) =& \mathcal{N}\left(\varepsilon, \mathcal{F}^{\mathrm{DNN}}(L_n, \boldsymbol{s}_n, F), ||\cdot||_{2,n}\right) \\ \leq& \mathcal{M}\left(\varepsilon, \mathcal{F}^{\mathrm{DNN}}(L_n, \boldsymbol{s}_n, F), ||\cdot||_{2,n}\right). \end{aligned}$$

Since functions in $\mathcal{F}^{DNN}(L_n, \boldsymbol{s}_n, F)$ are bounded by $[-F, F]$, there exists $c_3 > 0$ such that

$$\begin{aligned} \mathcal{M}\left(\varepsilon, \mathcal{F}^{\mathrm{DNN}}(L_n, \boldsymbol{s}_n, F), ||\cdot||_{2,n}\right) \leq& 3\left(\frac{8eF^2}{\epsilon^2}\log\frac{12eF^2}{\epsilon^2}\right)^{V^+_{\mathcal{F}^{\mathrm{DNN}}(L_n, \boldsymbol{s}_n, F)}} \\ \leq& 3\left(\frac{8eF^2}{\epsilon^2}\log\frac{12eF^2}{\epsilon^2}\right)^{c_3 L_n S_n \log S_n} \end{aligned}$$

holds for every $\varepsilon > 0$ by Theorem 9.4 of Györfi et al. (2002) and Theorem 6 of Harvey et al. (2017). To sum up,

$$\begin{aligned} \sup_{\varepsilon > \varepsilon_n} \log \mathcal{N}\left(\varepsilon, \mathcal{F}'_n, h_n\right) \lesssim& L_n S_n \log S_n \log n \\ \asymp& n^{\frac{d}{2\beta+d}}(\log n)^{4+3\tau} \\ \leq& n\varepsilon_n^2 \end{aligned} \tag{A.18}$$

holds.

Now, we define

$$K_i(f_0, f) = \int \log(p_{f_0,i}/p_{f,i}) dP_{f_0,i},$$
$$V_{2,0;i}(f_0, f) = \int \left(\log(p_{f_0,i}/p_{f,i}) - K_i(f_0, f)\right)^2 dP_{f_0,i},$$

and

$$B_n^*(f_0, \varepsilon; 2) = \big\{ f \in \mathcal{F}_n' : \frac{1}{n}\sum_{i=1}^n K_i(f_0, f) \le \varepsilon^2, \\ \frac{1}{n}\sum_{i=1}^n V_{2,0;i}(f_0, f) \le \varepsilon^2 \big\}.$$

For $\varepsilon > 0$, define

$$A_n^*(f_0, \varepsilon; 2) := \Big\{ f \in \mathcal{F}_n' : \max_i |f(\boldsymbol{x}_i) - f_0(\boldsymbol{x}_i)| \le \varepsilon \Big\}.$$

Then by Lemma 3.2 of van der Vaart & van Zanten (2008), we can get

$$A_n^*(f_0, \varepsilon_n; 2) \subset B_n^*(f_0, \varepsilon_n; 2).$$

Now by following the proof of (A.9), we obtain

$$\Pi_n(B_n^*(\eta_0, \varepsilon_n; 2)) \ge e^{-n\varepsilon_n^2} \tag{A.19}$$

for all but finite many n. Hence by (A.18), (A.19) and Lemma A.2, the proof of (A.17) is done. □

Step 2. Since (A.14) holds for arbitrary $\{\boldsymbol{x}^{(n)}\}_{n=1}^\infty$,

$$\mathbb{E}_0\left[\Pi_n\left(f \in \mathcal{F}_n' : ||\phi \circ f - \phi \circ f_0||_{2,n} > M_n\varepsilon_n \Big| \mathcal{D}^{(n)}\right)\right] \to 0 \tag{A.20}$$

also holds. Next, we will check the conditions in Lemma A.4 for

$$\mathcal{G} := \Big\{ g \ : \ g = (\phi \circ f^{\mathrm{mDNN}}_{\boldsymbol{M},\boldsymbol{\theta}\ [-F,F]} - \phi \circ f_0)^2, f^{\mathrm{mDNN}}_{\boldsymbol{M},\boldsymbol{\theta}\ [-F,F]} \in \mathcal{F}^{\mathrm{mDNN}}(L_n, \boldsymbol{p}_n, F, s_n) \Big\}$$
$$\kappa := \frac{1}{2},\ \alpha := \varepsilon_n^2,\ K_1 = K_2 = 1.$$

First, it is easy to check $||g(\boldsymbol{x})||_\infty \le 1$ and $E(g(\boldsymbol{X})^2) \le E(g(\boldsymbol{X}))$ for $g \in \mathcal{G}$. Also, for every $f^{\mathrm{mDNN}}_{\boldsymbol{M},\boldsymbol{\theta}\ [-F,F]} \in \mathcal{F}^{\mathrm{mDNN}}(L_n, \boldsymbol{p}_n, F, s_n)$, there exist $\boldsymbol{\psi}_{\boldsymbol{M},\boldsymbol{\theta}} \in \mathbb{R}^{S_n}$ and $f^{\mathrm{DNN}}_{\boldsymbol{\psi}_{\boldsymbol{M},\boldsymbol{\theta}}\ [-F,F]} \in \mathcal{F}^{\mathrm{DNN}}(L_n, \boldsymbol{s}_n, F)$ such that $f^{\mathrm{mDNN}}_{\boldsymbol{M},\boldsymbol{\theta}\ [-F,F]} = f^{\mathrm{DNN}}_{\boldsymbol{\psi}_{\boldsymbol{M},\boldsymbol{\theta}}\ [-F,F]}$ holds. Since

$$\begin{aligned}
\left\| (\phi \circ f^{\mathrm{DNN}}_{\boldsymbol{\psi}_1\ [-F,F]} - \phi \circ f_0)^2 - (\phi \circ f^{\mathrm{DNN}}_{\boldsymbol{\psi}_2\ [-F,F]} - \phi \circ f_0)^2 \right\|_{n,1} \\
&\le 4 \left\| \phi \circ f^{\mathrm{DNN}}_{\boldsymbol{\psi}_1\ [-F,F]} - \phi \circ f^{\mathrm{DNN}}_{\boldsymbol{\psi}_2\ [-F,F]} \right\|_{n,1} \\
&\le 4 \left\| f^{\mathrm{DNN}}_{\boldsymbol{\psi}_1\ [-F,F]} - f^{\mathrm{DNN}}_{\boldsymbol{\psi}_2\ [-F,F]} \right\|_{n,1}
\end{aligned}$$

holds for $\boldsymbol{\psi}_1, \boldsymbol{\psi}_2 \in \mathbb{R}^{S_n}$, there exists $c_4 > 0$ such that

$$\begin{aligned}
\mathcal{N}(u, \mathcal{G}, ||\cdot||_{n,1}) &\le \mathcal{N}\left(\frac{u}{4}, \mathcal{F}^{\mathrm{DNN}}(L_n, \boldsymbol{s}_n, F), ||\cdot||_{n,1}\right) \\
&\le \mathcal{M}\left(\frac{u}{4}, \mathcal{F}^{\mathrm{DNN}}(L_n, \boldsymbol{s}_n, F), ||\cdot||_{n,1}\right) \\
&\le 3\left(\frac{16eF}{u}\log\frac{24eF}{u}\right)^{V^+_{\mathcal{F}^{\mathrm{DNN}}(L_n, \boldsymbol{s}_n, F)}} \\
&\lesssim n^{c_4 S_n L_n \log S_n}
\end{aligned}$$

for $u \geq n^{-1}$ by Theorem 9.4 in Györfi et al. (2002) and Theorem 6 of Harvey et al. (2017). Hence for all $t \geq \frac{\varepsilon_n^2}{8}$,

$$\begin{aligned}\int_{\frac{\kappa(1-\kappa)t}{16\max\{K_1,2K_2\}}}^{\sqrt{t}} \sqrt{\log \mathcal{N}\left(u, \mathcal{G}, ||\cdot||_{n,1}\right)} du \lesssim & \sqrt{t}\left(n^{\frac{d}{2\beta+d}}(\log n)^{4+3\tau}\right)^{\frac{1}{2}} \\ = & o\left(\frac{\sqrt{n}t/4}{96\sqrt{2}\max\{K_1, 2K_2\}}\right)\end{aligned}$$

holds. To sum up, we conclude that

$$\mathbf{P}\left\{\sup_{f\in\mathcal{F}^{\mathrm{DNN}}(L_n,\boldsymbol{p}_n')} \frac{\left|||f-f_0||_{2,\mathrm{P}_X}^2 - ||f-f_0||_{2,n}^2\right|}{\varepsilon_n^2 + ||f-f_0||_{2,\mathrm{P}_X}^2} > \frac{1}{2}\right\} \leq 60\exp\left(-\frac{n\varepsilon_n/8}{128\cdot 2304}\right). \tag{A.21}$$

holds for all but finite many n by Lemma A.4. Hence by (A.20) and (A.21), the proof of (A.15) is done. □

A.5. Proof of Theorem 4.3

Theorem holds by (A.4) for nonparametric regression (1) and by (A.16) for binary classification (2). □

B. Theoretical results for hierarchical composition functions

In Section 4, we only describe the results of Holder continuous functions for the sake of simplicity. However, theoretical results for mBNN can be easily extended from the Holder continuous functions to the hierarchical compositional structured function considered in Kohler & Langer (2021). First, we define the function class of hierarchical composition functions.

Definition B.1 (hierarchical composition function)**.** Let $f : \mathbb{R}^d \to \mathbb{R}$ and $\mathcal{P}$ be a subset of $(0, \infty) \times \mathbb{N}$.

a) We say that f satisfies a hierarchical composition model of level 0 with order and smoothness constraint $\mathcal{P}$, if there exists a $K \in [d]$ such that

$$f(\boldsymbol{x}) = x^{(K)} \quad \text{for all } \boldsymbol{x} = \left(x^{(1)}, \ldots, x^{(d)}\right)^\top \in \mathbb{R}^d.$$

b) We say that f satisfies a hierarchical composition model of level $q+1$ with order and smoothness constraint $\mathcal{P}$, if there exist $(\beta, K) \in \mathcal{P}$, $g : \mathbb{R}^K \to \mathbb{R}$ and $f_1, \ldots, f_K : \mathbb{R}^d \to \mathbb{R}$ such that $g \in \mathcal{H}_K^\beta$, $f_1, \ldots, f_K$ satisfy a hierarchical composition model of level q with order and smoothness constraint $\mathcal{P}$ and

$$f(\boldsymbol{x}) = g\left(f_1(\boldsymbol{x}), \ldots, f_K(\boldsymbol{x})\right) \quad \text{for all } \boldsymbol{x} \in \mathbb{R}^d.$$

c) For $q \in \mathbb{N}$ and $\mathcal{P} \subset (0, \infty) \times \mathbb{N}$, consider the hierarchical composition function of level q with constraint $\mathcal{P}$, where for each function g in the definition can be of different smoothness $p_g = q_g + s_g$ ($q_g \in \mathbb{N}_0$ and $s_g \in (0, 1]$) and of different input dimension K_g, where $(p_g, K_g) \in \mathcal{P}$. Assume the maximal input dimension and the maximal smoothness of g are bounded. Assume that each g is Lipschitz continuous and all partial derivatives of order less than or equal to q_g are bounded. We define the set of functions that satisfy these conditions as $\mathcal{H}(q, \mathcal{P})$.

We describe a lemma that approximates hierarchical composition functions as DNN functions.

Lemma B.2 (Theorem 3 of Kohler & Langer (2021))**.** *There exists $C_L > 0$ and $C_p > 0$ only depending on d such that for every $f_0 \in \mathcal{H}(q, \mathcal{P})$ with $\|f_0\|_\infty \leq F$, there exist $f^{\mathrm{DNN}}_{\hat{\boldsymbol{\theta}}\,[-F,F]} \in \mathcal{F}^{\mathrm{DNN}}(L_n, \boldsymbol{v}_n, F)$ with*

$$\begin{aligned} L_n &:= \lceil C_L \log n \rceil, \\ v_n &:= \left\lceil C_p (\log n)^{-1/2} \max_{(\beta,K)\in\mathcal{P}} n^{\frac{K}{2(2\beta+K)}} \right\rceil, \\ \boldsymbol{v}_n &:= (d, v_n, ..., v_n, 1)^\top \in \mathbb{N}^{L_n+2}, \end{aligned}$$

such that

$$\left\| f^{\mathrm{DNN}}_{\hat{\boldsymbol{\theta}}\,[-F,F]} - f_0 \right\|_\infty \lesssim \max_{(\beta,K)\in\mathcal{P}} n^{-\frac{\beta}{2\beta+K}} \log n$$

and

$$|\hat{\boldsymbol{\theta}}|_\infty \leq n \tag{A.1}$$

hold.

In Section 4, we assume the true regression function (or the logit of the true conditional probability) f_0 belongs to the β-Holder class. If d is relatively large compared to β, concentration rate $n^{-\frac{\beta}{(2\beta+d)}}$ can be extremely slow. However, mBNN can avoid curse of dimensionality for hierarchical composition function, which is demonstrated in the following theorem.

Theorem B.3 (Theoretical results for hierarchical composition functions)**.** *For $f_0 \in \mathcal{H}(q, \mathcal{P})$, Theorem 4.1 and 4.2 hold with the concentration rate*

$$\varepsilon_n = \max_{(\beta,K)\in\mathcal{P}} n^{-\beta/(2\beta+K)} \log^\gamma(n)$$

for $\gamma > \frac{5}{2}$. Furthermore, Theorem 4.3 holds with

$$\mathfrak{s}_n := \left\lceil C_p (\log n)^{1/2} \max_{(\beta,K)\in\mathcal{P}} n^{\frac{K}{2(2\beta+K)}} \right\rceil$$

Proof. We can follow every step in Appendix A.3, A.4 and A.5 by simply changing the definition of ε_n to

$$\varepsilon_n = \max_{(\beta,K)\in\mathcal{P}} n^{-\beta/(2\beta+K)}(\log n)^{\gamma} = \max_{(\beta,K)\in\mathcal{P}} n^{-\beta/(2\beta+K)}(\log n)^{\tau+\frac{5}{2}}$$

and definition of s_n to

$$s_n := \left\lceil C_p \left(\max_{(\beta,K)\in\mathcal{P}} n^{\frac{K}{2\beta+K}} (\log n)^{3\tau} \right)^{1/2} \right\rceil .$$

□

C. More detail for the mBNN

C.1. MCMC algorithm

First we keep $\boldsymbol{M}^{(t)}$ fixed and update $\boldsymbol{\theta}^{(t)}$ (and ${\sigma^2}^{(t)}$) using existing MCMC algorithm, then we update $\boldsymbol{M}^{(t)}$ using MH algorithm. In practical, we can update $\boldsymbol{M}^{(t)}$ using only the data in single mini-batch for large scale dataset. Algorithm 2 is a brief summary of our algorithm.

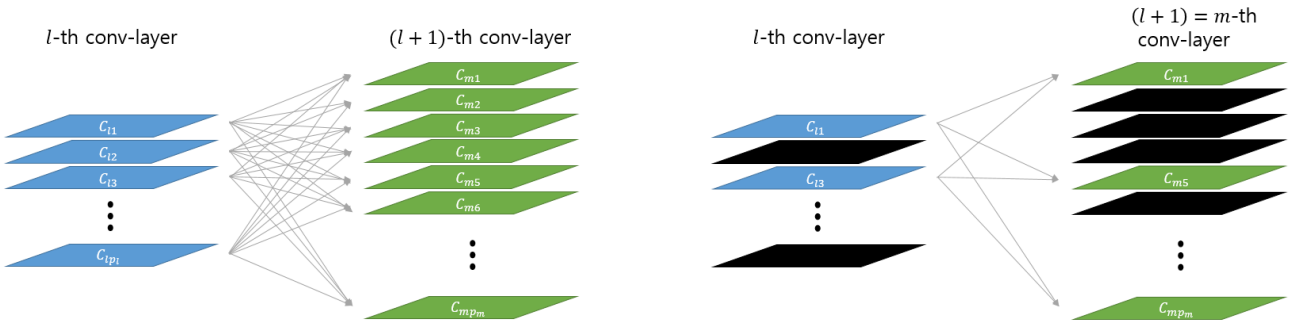
Algorithm 2 Proposed MCMC algorithm

INPUT: $T_{\text{total}}, T_{\text{MH}}, n_{\text{MH}} \in \mathbb{N}$

for $t = 1$ to T_{total} **do**
  Update $\boldsymbol{\theta}^{(t)}$ (and ${\sigma^2}^{(t)}$) using existing MCMC algorithm (e.g. HMC, SGLD).
  if $t \,\%\, T_{\text{MH}} = 0$ **then**
    Update $\boldsymbol{M}^{(t)}$ using Algorithm 1, n_{MH} times.
  end if
end for

C.2. Masked Bayesian CNN

First, we introduce how to apply masking variables for masked Bayesian CNN. Most of CNN architectures consist of a mixture of sequences of convolution layer and RELU activation function. For a given CNN, the corresponding masked CNN is constructed by simply adding masking parameters to the CNN model. For the l-th convolution layer, the masked CNN screens the channels using binary masking vector whose dimension is equal to the number of channels in l-th layer. Figure 4 is a illustration of the masked convolution layer.



(a) Original convolution layer

(b) Convolution layer with masking vector

Figure 4. **Convolution layer and masked convolution layer.** Masking vector $\boldsymbol{m}^{(l)}$ screens channels of the l-th layer. When $\left(\boldsymbol{m}^{(l)}\right)_j = 0$, the j-th channel of the l-th layer becomes inactive. Blue and green rectangles represent input and output channels, and black rectangles represent masked channels.

As we did in Section 3.2, we can simply implement this by converting RELU activation functions into masked-RELU activation functions. For instance, since Resnet18 structure has 17 RELU activation functions, we use 17 masking vectors whose dimensions are equal to the number of channels of the corresponding hidden layer.

Then, the prior Π_n given as (4), (5) and (6) is adopted on parameters $\boldsymbol{M}$ and $\boldsymbol{\theta}$, where $\boldsymbol{M}$ is the concatenate of the all masking vectors and $\boldsymbol{\theta}$ is the concatenation of the filters, weight matrices and bias vectors in the CNN model.

D. Detailed settings for the experiments

D.1. Noisy polynomial regression

Hernández-Lobato & Adams (2015) considers a noisy polynomial regression problem, where the input X and output Y are sampled from

$$\begin{aligned} X &\sim \text{Uniform}(-4, 4), \\ Y &= X^3 + \epsilon, \qquad\qquad \epsilon \sim N(0, 9). \end{aligned} \tag{C.1}$$

Following their setting, We first generate 20 training examples from (C.1), and compare the predictive distribution of the mBNN with that of BNN. For the both methods, the two hidden layer MLP with the layer sizes (1000,1000) and Cauchy prior with scale 0.3 are used. The prediction distributions are approximated by 1000 MCMC samples which are obtained by HMC with the step size 10^{-2} and NUTs (Hoffman et al., 2014), 300 burn-in samples and thinning interval 10. For the mBNN, we use $N_{\max} = 3$, $\lambda = 0.1$, $T_{MH} = 1$ and $n_{MH} = 2$.

After we obtain the predictive distributions, we generate 1000 test examples from (C.1). For each method and test examples, we obtain the 95% predictive interval of the predictive distribution. That is, for the i-th test example $x^{(i)}$, we sample $y_1^{(i)}, \ldots, y_N^{(i)}$ from the predictive distribution (which is expressed as a Gaussian mixture distribution where the mixing distribution is the empirical distribution on the MCMC samples), and then obtain the 95% predictive interval by the 2.5% and 97.5% quantiles among $y_1^{(i)}, \ldots, y_N^{(i)}$.

D.2. UCI dataset

For BNN and the mBNN, two hidden layer MLP with the layer sizes (1000,1000) and Cauchy prior with scale 1.0 are used. The prediction distributions are approximated by 20 MCMC samples which are obtained by HMC with step size 10^{-2} and NUTs, 2 burn-in samples and thinning interval 200. For the mBNN, we use $N_{\max} = 3$, $\lambda = 0.1$, $T_{MH} = 1$ and $n_{MH} = 10$.

For the node-sparse VI (NS-VI), we adopt Bayesian compression using group normal-Jeffreys prior (Louizos et al., 2017). Note that most of existing node-sparse VI have the same spirit with Louizos et al. (2017) in the sense of using hierarchical scale-mixture priors to prune nodes. Among those algorithms, this method is chosen in consideration of versatility and reproducibility. For UCI datasets, we use Adam (Kingma & Ba, 2014) optimizer with learning rate 0.01, 500 epoch and batch size 100.

D.3. Bayesian structural time series model

The state space model with a general regression function $f_{\boldsymbol{\psi}}(\cdot)$, which is parameterized by $\boldsymbol{\psi}$, can be expressed as

$$\begin{aligned} y_t &= \mu_t + f_{\boldsymbol{\psi}}(\boldsymbol{x}_t) + \epsilon_t, & \epsilon_t &\sim \mathcal{N}(0, \sigma_\epsilon^2) \\ \mu_{t+1} &= r\mu_t + \eta_t, & \eta_t &\sim \mathcal{N}(0, \sigma_\eta^2) \\ \mu_0 &\sim N(a_0, \sigma_0^2) \end{aligned}$$

where $0 < r < 1$ and $\boldsymbol{x}_t \in \mathbb{R}^d$, $y_t \in \mathbb{R}$ and $\mu_t \in \mathbb{R}$ denote observed input vector, output variable and unobserved local trend at time t, respectively. Based on the state space model, BSTS model aims to forecast the current value of y using the current values of other variables. This problem is called "nowcasting"(Banbura et al., 2010), which is widely used for economics (Giannone et al., 2008), inferring causal relationships (Brodersen et al., 2015), anomaly detection (Feng & Tian, 2021), to name just a few.

The dataset consists of daily search volumes of keywords in year 2021 associated with a pre-specified product (eg. shampoo). We aim to predict the level of interest of the pre-specified product based on the past and current search volumes of related keywords. To be more specific, y_t is the daily search volumes of the pre-specified product and $\boldsymbol{x}_t$ is the vector of the daily search volumes of the related keywords.

For regression function $f_{\boldsymbol{\psi}}(\boldsymbol{x})$, we consider the linear, BNN and the mBNN models. For the linear model, we adopt i.i.d standard Gaussian prior on $\boldsymbol{\psi}$, while for BNN and the mBNN we adopt the priors considered in the paper. We use inverse-gamma prior on σ_ϵ^2 and standard Gaussian prior on μ_0. We use $r = 0.95$ and $\sigma_\eta^2 = 0.1$. Using the past dataset $\{\boldsymbol{x}_t, y_t\}_{t=1}^T$ where $T = 240$, we generate MCMC samples of $\boldsymbol{\psi}$, σ_ϵ^2 and $\{\mu_t\}_{t=1}^T$ from the posterior distribution using an

MCMC algorithm. To be more specific, we first sample $\boldsymbol{\psi}$ from its conditional posterior distribution. For the linear model, $\boldsymbol{\psi}$ can be sampled easily since its conditional posterior distribution is also Gaussian. For BNN and the mBNN, we use the MCMC algorithm developed in this paper. Once $\boldsymbol{\psi}$ is sampled, we sample σ_ϵ^2 and sample μ_t using the Kalman filter (Welch et al., 1995; Durbin & Koopman, 2002).

For every method, the prediction distributions are approximated by 20 MCMC samples with 10 burn-in samples and thinning interval 10. We use standard Gaussian prior for the linear model, and For BNN and the mBNN, two hidden layer MLP with the layer sizes (100,100) and Cauchy prior with scale 1.0 are used. For the mBNN, we use $N_{\max} = 3$, $\lambda = 0.1$, $T_{MH} = 1$ and $n_{MH} = 2$.

D.4. Image dataset

For NS-VI, starting from a pretrained network, we use the Adam optimizer with the learning rate 0.0001, 200 epoch and batch size 100 for optimization.

For node-sparse Deep ensemble (NS-Ens), we first train each member of the ensemble to be node-wise sparse DNNs using LeGR (Chin et al., 2020), and then combine them as Lakshminarayanan et al. (2017) does. We use the default hyper-parameters provided in (Chin et al., 2020).

As MC-dropout (Gal & Ghahramani, 2016) uses the weights obtained by Dropout (Srivastava et al., 2014) multiplied by a random masking vectors, we use the node-wise sparse weights obtained by Targeted Dropout (Gomez et al., 2019) multiplied by a random masking vectors for node-sparse MC-Dropout (NS-MC). Specifically, let $\gamma \in [0, 1]$ and $\alpha \in [0, 1]$ be the pre-specified proportions for pruning and drop probabilities, respectively. Before each gradient step, we select nodes having the $\gamma \times 100\%$ lowest magnitude (L^2-norm of the connected weights) on each layer, and then randomly mask the selected nodes with probability α. This implies that the expected ratio of nodes to survive during each step is $(1 - \alpha\gamma)$. For inference, we multiply a random masking vector to the nodes. Through careful tuning, we choose $(\alpha, \gamma) = (0.9, 0.4)$ for CIFAR10 and $(\alpha, \gamma) = (0.9, 0.3)$ for CIFAR100. For optimization, we use the Adam optimizer with the learning rate 0.001, 200 epoch and batch size 100.

For the mBNN, we use SGLD (Welling & Teh, 2011) with the batch-size 100, step size 10^{-3} with the Cosine scheduler and temperature $1/\sqrt{n}$, which are commonly used in SG-MCMC literature (Zhang et al., 2019; Wenzel et al., 2020). MCMC samples are obtained after 5 burn-in samples and thinning interval 20. We use $N_{\max} = 3$, $\lambda = 0.05$, $T_{MH} = 10$ (batch) and $n_{MH} = 1$.

E. Additional experimental results

Additional results on various λ In Section 6.2, we conduct an experiment to demonstrate the necessity of masking variables on real dataset, where we report the result of the mBNN with $\lambda = 0.1$. Asymptotically the mBNN automatically finds an appropriate network architecture for given data and complexity of the true regression model as long as λ remains bounded, but finite sample performance varies according to the choice of λ.

Table 3. **Additional results on various λ.** Performance of BNN, NS-VI and the mBNN (with various λ) on UCI datasets.

Dataset	Method	λ	R in 95% C.I.	RMSE	NLL	CRPS	# of activated nodes
Boston	BNN	·	0.912(0.006)	3.411(0.145)	2.726(0.087)	1.738(0.052)	(1000,1000)
	NS-VI	·	0.746(0.010)	3.079(0.179)	4.242(0.568)	1.661(0.069)	(13, 21)
	mBNN	0.075	0.924(0.142)	2.961(0.142)	2.547(0.079)	1.535(0.057)	(102, 44)
	mBNN	0.1	0.933(0.007)	2.902(0.143)	2.472(0.085)	1.462(0.055)	(29, 12)
	mBNN	0.125	0.939(0.061)	2.790(0.170)	2.381(0.063)	1.431(0.061)	(11, 5)
Concrete	BNN	·	0.908(0.008)	5.080(0.178)	3.091(0.055)	2.658(0.072)	(1000,1000)
	NS-VI	·	0.568(0.011)	5.046(0.149)	9.713(0.686)	2.965(0.088)	(30, 19)
	mBNN	0.075	0.916(0.127)	4.908(0.127)	3.023(0.034)	2.626(0.062)	(108,54)
	mBNN	0.1	0.912(0.009)	4.913(0.180)	3.027(0.046)	2.628(0.087)	(35, 17)
	mBNN	0.125	0.924(0.006)	5.002(0.139)	3.035(0.035)	2.683(0.068)	(16, 7)
Energy	BNN	·	0.945(0.005)	0.591(0.017)	0.902(0.025)	0.322(0.007)	(1000,1000)
	NS-VI	·	0.913(0.006)	1.322(0.117)	1.792(0.121)	0.720(0.055)	(7, 10)
	mBNN	0.075	0.943(0.005)	0.473(0.012)	0.676(0.030)	0.254(0.005)	(211, 210)
	mBNN	0.1	0.945(0.004)	0.474(0.015)	0.670(0.034)	0.256(0.006)	(35, 17)
	mBNN	0.125	0.955(0.004)	0.510(0.026)	0.736(0.047)	0.278(0.013)	(19, 13)
Yacht	BNN	·	0.977(0.006)	0.675(0.048)	1.011(0.056)	0.332(0.013)	(1000,1000)
	NS-VI	·	0.932(0.011)	1.842(0.096)	1.915(0.063)	0.969(0.042)	(102, 54)
	mBNN	0.075	0.976(0.006)	0.604(0.054)	0.920(0.084)	0.295(0.015)	(317, 193)
	mBNN	0.1	0.953(0.008)	0.664(0.044)	0.932(0.062)	0.318(0.015)	(120, 76)
	mBNN	0.125	0.954(0.011)	0.621(0.054)	0.966(0.102)	0.301(0.023)	(42,26)

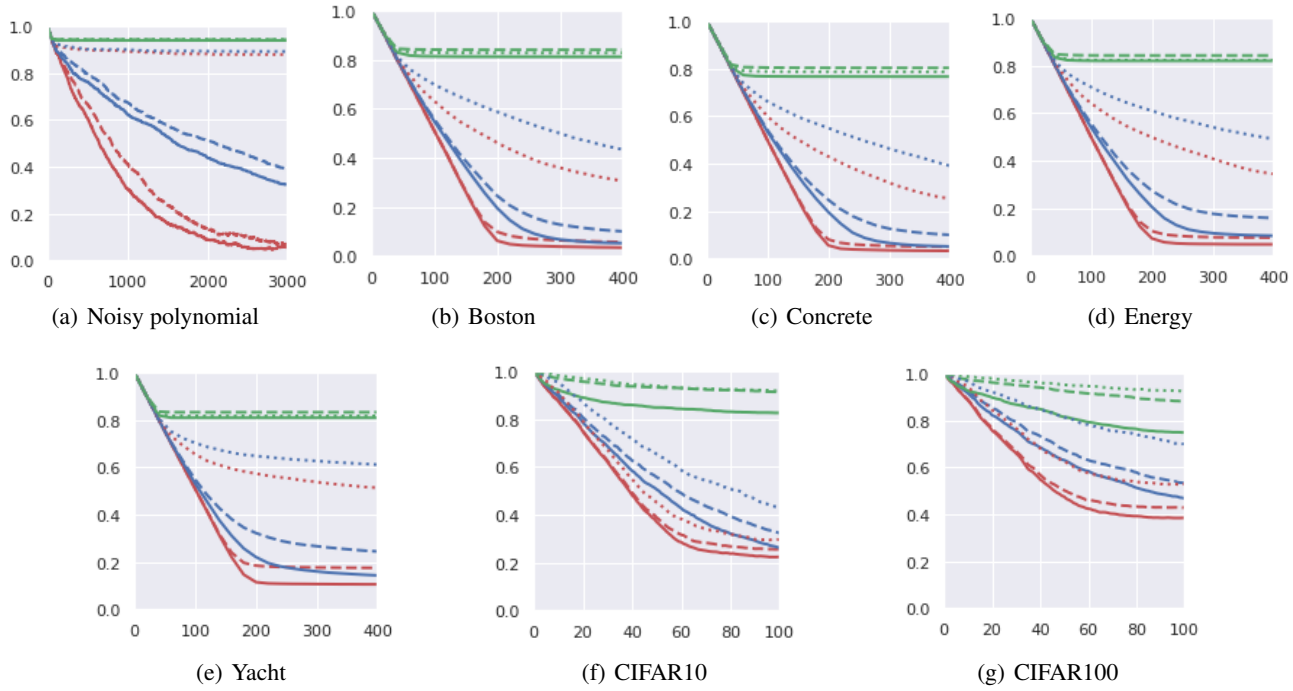
In Table 3, we report the means and standard errors of the performance measures of the mBNN with various values of λ. The results indicate that the mBNN performs stably with respect to the choice of λ even though the level of sparsity is proportional to λ. In practice, λ can be selected based on either validation data or putting a prior on λ.

Compatibility of proposed MCMC algorithm with mini-batching The bias of MH with mini-batch has been discussed, and recently there have been many works on this topic. Specifically, TunaMH (Zhang et al., 2020) and MHBT (Wu et al., 2022) are representative examples, and both algorithms ensure the convergence of mini-batch MH. Both algorithms propose unbiased mini-batch MH through appropriate modifications of batch-sampling method and acceptance rate. That is, their methods can be applied to our MH algorithm without much modification. We conduct experiments with mBNN utilizing MHBT on CIFAR10 and CIFAR100 to obtain the following results in Table E. The results are similar to the original MH algorithm (without caring biases due to mini-batches).

Table 4. **mBNN utilizing MHBT** Performance of mBNN utilizing MHBT on image datasets.

Dataset	ACC	NLL	ECE	FLOPs	Capacity
CIFAR10	0.933	0.206	0.006	12.92%	3.82%
CIFAR100	0.740	0.982	0.002	21.52%	14.27%

Additional results for Section 6.5 In Figure 3, we report the results only on Yacht, CIFAR10 and CIFAR100 datasets to illustrate the efficiency of our proposal distribution in Algorithm 1. Here, we report the results for the other datasets which are presented in Figure 5. The patterns are similar in that our proposal distribution works most efficiently.



(a) Noisy polynomial (b) Boston (c) Concrete (d) Energy

(e) Yacht (f) CIFAR10 (g) CIFAR100

Figure 5. **Efficiency of the proposal.** The ratios of activated nodes (y-axis) relative to the largest DNN are presented as the MCMC algorithms iterate in the burn-in period (x-axis). The solid, dashed and dotted lines correspond to (13), (14) and (15) for *birth*, respectively while blue, red and green lines correspond to (13), (14) and (15) for *death*, respectively.