AIMS Mathematics, 10(9): 21929–21952.
DOI:10.3934/math.2025977
Received: 18 June 2025
Revised: 20 August 2025
Accepted: 27 August 2025
Published: 22 September 2025

https://www.aimspress.com/journal/Math

Research article

Variational sparse Bayesian neural networks with regularized horseshoe priors and scale mixture priors

Xu Chen[1,*], Lilong Sima[1], Zhen Wei[1], Xingde Duan[2], Ping Feng[1] and Fuhong Song[1]

[1] Guizhou Provincial Key Laboratory of Computing and Network Convergence, School of Information, Guizhou University of Finance and Economics, Guiyang 550025, China

[2] School of Mathematics and Statistics, Guizhou University of Finance and Economics, Guiyang 550025, China

* **Correspondence:** Email: chenxu@mail.gufe.edu.cn.

Abstract: Choosing an appropriate prior distribution for the weights of Bayesian neural networks (BNNs) remains an open challenge. In most cases, a Gaussian prior is adopted, but its typically high variance can lead to overestimation of the predictive uncertainty. Recently, horseshoe priors have been proposed for model selection and compression, as they effectively deactivate units that do not contribute to explaining the data and yield well-calibrated structural weight uncertainty estimates. However, the horseshoe prior has been found to underestimate predictive uncertainty, especially in regions lacking data. In this paper, we proposed an efficient variational sparse BNN that integrates both a regularized horseshoe prior and a Gaussian scale mixture prior. Both priors can induce sparsity, thereby mitigating overfitting and improving the model's generalization ability. Our approach enables computationally efficient optimization via variational inference while providing more reliable predictive uncertainty. Experimental results demonstrate that the proposed model delivers competitive predictive performance and reasonable posterior weight uncertainty estimates in non-linear regression, image classification, and anomaly detection tasks compared with recent methods.

Keywords: Bayesian neural networks; regularized horseshoe prior; variational inference; uncertainty estimation
Mathematics Subject Classification: 62F15

1. Introduction

Deep neural networks (DNNs) have achieved state-of-the-art performance in a wide range of complex tasks, such as image classification [44], medical image analysis [41], and artificial intelligence (AI) agents [50]. However, their decisions are based solely on deterministic estimates, which may

result in overconfident yet incorrect predictions [7], undermining reliability. In high-risk domains—such as medical diagnosis and autonomous driving—system reliability is critical, as incorrect decisions in life-threatening situations can have fatal consequences [12]. Moreover, DNN predictions often lack interpretability, leading many to regard these models as "black boxes" [39]. Additionally, their deep architectures are prone to overfitting [8], requiring large datasets and strong regularization to maintain performance.

In contrast, BNNs can potentially address these limitations [11]. They treat weights and biases as random variables, learning their posterior distributions during training while also estimating predictive uncertainty. This Bayesian framework offers a principled way to control overfitting, quantify confidence, and improve reliability in safety-critical applications [1, 28, 32]. As a result, research interest in BNNs and uncertainty estimation has grown substantially [6, 33].

A commonly used learning technique for BNNs is variational inference (VI) [31, 49], which approximates the posterior distribution with a tractable distribution and optimizes it to be close to the true posterior. VI-based BNNs are relatively easy to implement and robust against overfitting. Several approximate inferences—such as black-box variational inference [18] and alpha-divergence minimization [27]—make Bayesian inference computationally efficient and scalable on large datasets [49]. These advances have led to BNN adoption in various applications, including active learning [46], out-of-distribution (OOD) detection [11], adversarial attack detection [21], reinforcement learning [32], and disease diagnosis [8].

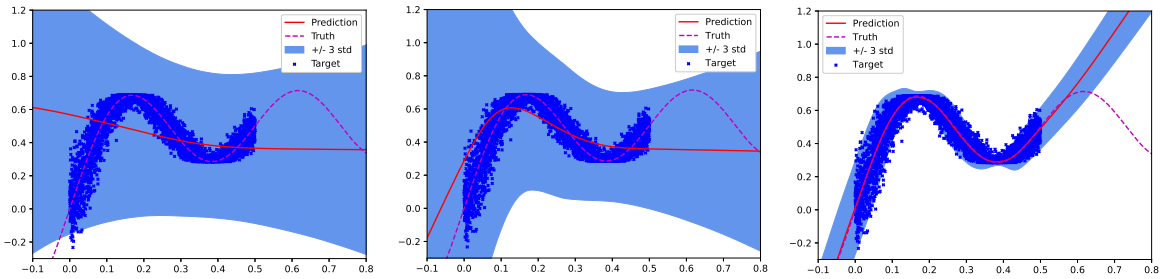
Choosing an effective prior is crucial for successful Bayesian inference. Unfortunately, the predictive performance and uncertainty estimation of BNNs can be highly sensitive to prior selection, and meaningful priors are difficult to specify due to the complex, non-intuitive relationship between network weights and their functional role [30, 36, 38, 43, 48].

In practice, Gaussian priors are the most widely used in modern BNN applications. Blundell et al. [4] proposed a Gaussian scale mixture prior—similar to a spike-and-slab prior—and employed the reparameterization trick [25] to obtain unbiased gradient estimates. This prior consists of two zero-mean Gaussian distributions with different variances. Despite progress in variational inference methods, BNNs with Gaussian priors still tend to underperform compared with networks trained via stochastic gradient descent and incur high computational costs [45].

Sparsity-inducing priors, particularly the horseshoe prior, have gained attention for creating sparse neural networks when combined with VI. The horseshoe prior can promote network weight sparsity [29], enable model compression [15], and support interpretable feature selection [39]. Louizo et al. [29] applied a group horseshoe prior in DNNs to prune units, achieving state-of-the-art compression rates. Ghosh et al. [14] applied a horseshoe prior to the incoming weights of each unit in a BNN for model selection. This method effectively deactivates units that do not contribute to explaining the data. However, the horseshoe prior can introduce minor perturbations when training data is limited, adversely affecting the generalization performance of BNNs. Ghosh et al. [15] proposed applying a regularized horseshoe prior to unit-specific weights for structured variational learning, which improved the compactness of the learned model. Nalisnick et al. [37] used the horseshoe prior to regularize both the number of layers and the number of hidden units per layer in DNNs, showing it to be more competitive than traditional regularization mechanisms such as dropout. Dropout randomly zeroes out certain weight parameters with a given probability to encourage sparsity, and the two approaches can be made equivalent via reparameterization [37]. However, these methods generally place priors over

hidden units.

Popkes et al. [39] proposed a two-layer network in which the input-layer weights were assigned a horseshoe prior for feature selection, while the second-layer weights used a zero-mean Gaussian prior. This model was applied to medical outcome prediction. Bai et al. [2] developed computationally efficient variational inference methods for sparse DNNs using spike-and-slab priors, providing theoretical guarantees for their approach. Sun [47] proposed a frequentist-like method for learning sparse DNNs, demonstrating its consistency within a Bayesian framework. Jantre et al. [22] introduced a Gaussian spike-and-slab node-selection model that achieved strong predictive performance. Dabiran et al. [9] proposed a sparse BNN using a hybrid prior that combined automatic relevance determination and informative priors, addressing both overfitting and computational challenges in posterior estimation. Skaaret-Lund et al. [45] developed a sparse BNN with latent binary variables and normalizing flows, improving predictive performance while producing more compact networks. Jantre et al. [23] proposed two structurally sparse BNNs that prune redundant nodes using spike-and-slab group lasso and spike-and-slab group horseshoe priors, with computationally tractable variational inference. Experimental results showed these methods to be competitive in prediction accuracy, model compression, and inference latency.



(a) Gaussian prior (b) Gaussian scale mixture prior (c) Horseshoe prior

Figure 1. Predictive performance and uncertainty estimations of BNNs with different priors in regression for a toy dataset.

Figure 1 shows the predictive performance and uncertainty estimation of BNNs with different priors for a toy dataset [4]. Blue crosses represent 5000 training samples, solid red lines indicate median predictions, and shaded regions correspond to ±3 standard deviations around the mean, illustrating posterior predictive uncertainty. Gaussian and Gaussian scale mixture priors tend to overinflate predictive uncertainty and underfit when data is limited. Under the same conditions, BNNs with horseshoe priors perform better in the interval [0.5, 0.6], but still underestimate predictive uncertainty—especially in regions without data. Additionally, because the number of output units is fixed by the task, sparsity-inducing priors are not appropriate for the output layer [15].

Motivated by these observations, we propose a variational sparse BNN that combines regularized horseshoe priors and Gaussian scale mixture priors. The proposed model is a multilayer network in which the weights of the first H layers follow a regularized horseshoe prior, and the remaining layers use a Gaussian scale mixture prior. The regularized horseshoe prior improves both generalization [15, 40] and feature selection [39], while the Gaussian scale mixture prior—resembling the classic spike-and-slab prior—has only hyperparameters and does not require updating during optimization, improving computational efficiency and reducing overfitting [4, 39]. Both priors promote sparsity.

We evaluated our model on regression, image classification, and anomaly detection tasks. Compared with recent BNN models using different priors, our approach achieved superior predictive performance, demonstrated stability in regression and image classification, and delivered competitive results in anomaly detection—where high-quality uncertainty estimates are especially critical.

Our main contributions are as follows:

- We propose a variational sparse BNN with regularized horseshoe priors and Gaussian scale mixture priors to improve uncertainty estimation and underfitting problems.
- Both priors induce sparsity, preventing overfitting and improving generalization, while enabling more reliable predictive uncertainty estimates.
- We demonstrate the effectiveness of our model in regression, image classification, and anomaly detection, showing competitive performance compared with strong baselines.

Paper organization: Section 2 outlines the key components of the proposed model. Section 3 describes the regularized horseshoe prior for sparsity and feature selection, the Gaussian scale mixture prior and its regularization, and the computational methods used. Section 4 presents the experimental evaluation. Section 5 concludes the paper.

2. Preliminary

In this section, we first briefly provide a general description of BNNs and then introduce stochastic variational inference frameworks. Finally, we describe the uncertainty measures of BNNs.

2.1. Bayesian neural networks

Consider a deep fully connected neural network with L layers ($L-1$ hidden layers), which is parameterized by a set of weight matrices $\theta = \{W_l\}_{l-1}^{L}$, where W_l is the weight matrix between layer $l-1$ and layer l with size $K_l \times K_{l-1}$, where K_l is the number of hidden units in layer l. We denote the inputs of the layers by $\{z_l\}_{l=0}^{L}$, where $z_l \in \mathbb{R}^{K_l \times 1}$ is the input into layer l, and z_0 is the input layer.

Given N observation data pairs $\mathcal{D} = \{(x_n, y_n)\}_{n=1}^{N}$, where $x_n \in \mathbb{R}^D$ is a D-dimensional feature vector and $y_n \in \mathbb{R}$ is the corresponding scalar target variable, a network maps an input x_n to a response $f(\theta, x_n)$ by recursively applying the nonlinear transformation $a\left(W_l^T \left[z_l^T, 1\right]^T\right)$, where $a(\cdot)$ is a nonlinearity activation function for hidden layers, such as rectified linear units (ReLUs), $a(x) = max(x, 0)$. Each target y_n can be obtained as $y_n = f(\theta, x_n) + \varepsilon_n$, where $\varepsilon_n \sim \mathcal{N}\left(0, \sigma^2\right)$ is an input-dependent Gaussian noise and σ^2 is the variance of noise.

The parameters, θ, of the BNNs with $\theta \sim p(\theta)$ prior distributions represent the prior belief of their configuration. Assuming that the observed data are independent and identically distributed, we observe the training data $\mathcal{D}$, and obtain the posterior distribution over the space of parameters by applying Bayes' theorem as follows:

$$p(\theta \mid \mathcal{D}) = \frac{p(\mathcal{D} \mid \theta)\, p(\theta)}{p(\mathcal{D})}, \tag{2.1}$$

where $p(\mathcal{D}) = \int p(\mathcal{D} \mid \theta)\, p(\theta)\, d\theta$ is a normalization constant (or the model evidence) and $p(\mathcal{D} \mid \theta)$ is the likelihood function. The posterior distribution captures the most likely network parameters as per the observed data. For the regression tasks, we have the following Gaussian likelihood:

$$p(\mathcal{D} \mid \theta) = \prod_{n=1}^{N} \mathcal{N}\left(y_n; f(\theta, x_n), \sigma^2 I\right). \tag{2.2}$$

For a multi-class classification task with K classes, given class $k \in \{1, 2, \cdots, K\}$, a softmax likelihood can be assumed, as follows:

$$p(y_n = k \mid x_n, \theta) = softmax(f(\theta, x_n)) = \frac{exp(f_k(\theta, x_n))}{\sum_{i=1}^{K} exp(f_i(\theta, x_n))}. \tag{2.3}$$

Using Eq (2.1), we can predict the response y^* of a new input x^* by integrating, as follows:

$$p(y^* \mid x^*, \mathcal{D}) = \int p(y^* \mid f(\theta, x^*)) \, p(\theta \mid \mathcal{D}) \, d\theta. \tag{2.4}$$

2.2. Variational inference

In most cases, the true posterior distribution $p(\theta \mid \mathcal{D})$ of BNNs cannot be evaluated analytically, making an approximation method necessary—variational inference being one of the more widely used methods. In this approach, we define a simpler approximate posterior distribution $q_\phi(\theta) \approx p(\theta \mid \mathcal{D})$, with free variational parameters, ϕ, and then minimize the Kullback-Leibler (KL) divergence between the approximating distribution and the posterior—that is, $KL\left[q_\phi(\theta) \parallel p(\theta \mid \mathcal{D})\right]$ is minimized. This optimization problem can be solved by stochastic gradient ascent with respect to variational parameters, ϕ.

The minimization of $KL\left[q_\phi(\theta) \parallel p(\theta \mid \mathcal{D})\right]$ is equivalent to the maximization of the evidence lower bound (ELBO), as follows:

$$\mathcal{L}_{VI}(\phi) = -\underbrace{\mathbb{E}_{q_\phi(\theta)}\left[logp(\mathcal{D} \mid \theta)\right]}_{model\ fit} + \underbrace{KL\left[q_\phi(\theta) \parallel p(\theta)\right]}_{regularization}. \tag{2.5}$$

As we can see, the $\mathcal{L}_{VI}(\phi)$ in Eq (2.5) consists of two terms: 1) The first term referred to as the expected log-likelihood, which is a data-dependent model fit term, encourages $q_\phi(\theta)$ to fit the observed data well by choosing the variational parameter ϕ. 2) The second term referred to as prior KL acts as regularization, encouraging $q_\phi(\theta)$ to be similar to the prior $p(\theta)$ and preventing the model from overfitting.

Variational inference thus performs approximate Bayesian inference by maximizing the ELBO. It has been successfully applied to estimate weight distributions in neural networks. However, because neural network weights have highly complex functional forms, the expected log-likelihood term in the ELBO cannot generally be computed analytically.

To address this, the reparameterization trick is often employed. This technique enables the computation of unbiased stochastic gradients of the ELBO with respect to the variational parameters, ϕ, converting the problem into a standard stochastic gradient ascent optimization.

2.3. Uncertainty estimation and calibration metrics

When we find an approximate posterior distribution, $q_\phi(\theta)$, by minimizing the KL divergence to the true model posterior, $p(\theta \mid \mathcal{D})$, we can replace the true posterior with the approximating distribution in

Eq (2.4) as follows:

$$p\left(y^* \mid x^*, \mathcal{D}\right) \approx \int p\left(y^* \mid f\left(\theta, x^*\right)\right) q_\phi\left(\theta\right) d\theta. \tag{2.6}$$

Equation (2.6) is difficult to compute because of integration; therefore, we approximate the integral using Monte Carlo (MC) sampling as follows:

$$p\left(y^* \mid x^*, \mathcal{D}\right) = \mathbb{E}_{p(\theta|\mathcal{D})}\left[p\left(y^* \mid x^*, \theta\right)\right] \approx \frac{1}{T}\sum_{t=1}^{T} p\left(y^* \mid f\left(\hat{\theta}_t, x^*\right)\right), \tag{2.7}$$

where $\left\{\hat{\theta}_t\right\}_{t=1}^{T}$ is randomly drawn from the approximate distribution, $q_\phi\left(\theta\right)$, using sampling T. Uncertainty in the weight induces prediction uncertainty [24]. Typically, uncertainty measures are used. A simple measure is the probability of the predicted class k or the maximum probability [34,35]:

$$\underset{k}{max} p\left(y^* = y_k \mid x^*\right). \tag{2.8}$$

This is the maximum value of the softmax output and is a measure of confidence in the prediction. We also compute the mean standard deviation (mean STD) as a proxy for uncertainty:

$$\sigma_k = \sqrt{\mathbb{E}_{q(\theta)}\left[p\left(y^* = k \mid x^*, \theta\right)^2\right] - \mathbb{E}_{q(\theta)}\left[p\left(y^* = k \mid x^*, \theta\right)\right]^2}, \tag{2.9}$$

where $\sigma\left(x\right) = \frac{1}{K}\sum\limits_{k}\sigma_k$, or compute the predictive entropy (max entropy) [13]:

$$\mathbb{H}\left[y^* \mid x^*, \mathcal{D}\right] = -\sum_{k}\left(\frac{1}{T}\sum_{t=1}^{T} p\left(y^* = k \mid f\left(\hat{\theta}_t, x^*\right)\right)\right) log\left(\frac{1}{T}\sum_{t=1}^{T} p\left(y^* = k \mid f\left(\hat{\theta}_t, x^*\right)\right)\right). \tag{2.10}$$

BNNs can naturally produce well-calibrated uncertainty estimates. However, assessing the quality and reliability of these estimates within a Bayesian framework is challenging, as there is no ground truth for uncertainty [8]. To address this, various calibration metrics have been proposed. In this work, we use the negative log-likelihood (NLL) and the expected calibration error (ECE) [3] to evaluate the performance of different uncertainty estimation approaches.

3. Methodology

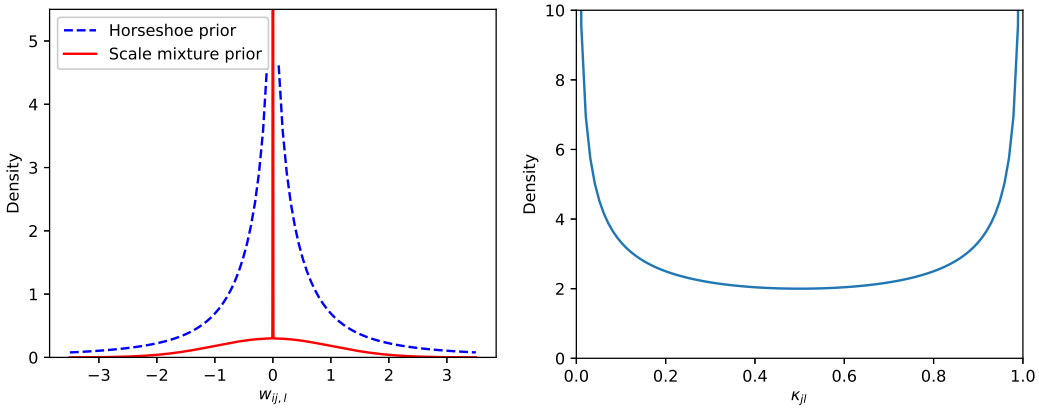
In this section, we describe the specific prior distributions used in our model. We first introduce the principle of parameter shrinkage in the horseshoe prior and then describe the Gaussian scale mixture prior along with its regularization mechanism. Next, we explain how hierarchical variational inference is applied to learn the parameters of the proposed model. Finally, we present two practical parameterization methods necessary for using a regularized horseshoe prior—namely, half-Cauchy reparameterization and weight non-centered parameterization—followed by our variational approximation choices and a summary of the entire model process.

3.1. Weight shrinkage with a horseshoe prior

Let $w_{ij,l}$ denote the element of the i-th row and j-th column of the weight matrix W_l and assume that each weight is conditionally independent. The horseshoe prior [5] can be described as follows:

$$w_{ij,l} \mid \tau_{jl}, \upsilon_l \sim \mathcal{N}\left(0, \tau_{jl}^2 \upsilon_l^2\right), \tau_{jl} \sim C^+(0, b_0), \upsilon_l \sim C^+\left(0, b_g\right), \tag{3.1}$$

where $a \sim C^+(0, b)$ is the half-Cauchy distribution with density $p(a \mid b) = \frac{2}{\pi b\left(1+\left(\frac{a}{b}\right)^2\right)}$, $(a > 0)$, υ_l is the global shrinkage parameter, and τ_{jl} is the local shrinkage parameter. The plots of the heavy-tailed priors are shown in Figure 2. The density function of the horseshoe prior (Figure 2(a), dashed lines) exhibits Cauchy-like, flat, heavy tails while maintaining an infinitely tall spike at zero. The heavy tails allow important weights to remain large, while the tall spike strongly shrinks to small weights toward zero. These two properties make the horseshoe prior particularly effective as a shrinkage prior for sparse modeling problems.



(a) Density of priors (b) Horseshoe prior on shrinkage coefficient

Figure 2. Plots of heavy-tailed priors.

To better understand the model, we assume $\upsilon_l^2 = b_0^2 = b_g^2 = 1$, and define $\kappa_{jl} = \frac{1}{1+\tau_{jl}^2} \in [0, 1]$. κ_{jl} is the shrinkage coefficient and is the amount of weight that the posterior mean for $w_{ij,l}$ places on zero once the data y_i have been observed:

$$\mathbb{E}\left(w_{ij,l} \mid y_i, \tau_{jl}^2\right) = \left(\frac{\tau_{jl}^2}{1+\tau_{jl}^2}\right) y_i + \left(\frac{1}{1+\tau_{jl}^2}\right) 0 = \left(1-\kappa_{jl}\right) y_i, \tag{3.2}$$

where $\tau_{jl} \sim C^+(0, 1)$, and it yields a beta prior to the shrinkage coefficient—that is, $\kappa_{jl} \sim Be\left(\frac{1}{2}, \frac{1}{2}\right)$. Figure 2(b) plots the density of κ_{jl}. Based on the above assumptions, Eq (3.1) can be written as: $w_{ij,l} \mid \kappa_{jl} \sim \mathcal{N}\left(0, \frac{1-\kappa_{jl}}{\kappa_{jl}}\right)$. When $\kappa_{jl} = 0$, we have $p\left(w_{ij,l} \mid \kappa_{jl} = 0\right) = U(-\infty, \infty)$—that is, no shrinkage or regularization for $w_{ij,l}$—and when $\kappa_{jl} = 1$, we have $p\left(w_{ij,l} \mid \kappa_{jl} = 1\right) = \delta\left(w_{ij,l} = 0\right)$ ($\delta(\cdot)$ as the Dirac delta function)—that is, shrinkage for all $w_{ij,l}$.

The global shrinkage parameter, υ_l, tends to shrink all weights in the layer, while τ_{jl} reduces the shrinkage of the weight associated with the input feature, z_{jl}. Consequently, the horseshoe prior can select certain features of the input vector z_l, while others are ignored.

3.2. Regularized horseshoe prior

Theoretically, this property of the horseshoe prior is desirable. In practice, when the training data are limited, some weights with no shrinkage produce larger values, adversely affecting the generalization performance of BNNs. To address this problem, we used a regularized horseshoe prior [40] that can be expressed as follows:

$$\begin{aligned} &w_{i,j,l} \mid \tau_{jl}, \upsilon_l, c \sim \mathcal{N}\left(0, \tilde{\tau}_{jl}^2 \upsilon_l^2\right), \tilde{\tau}_{jl}^2 = \frac{c^2 \tau_{jl}^2}{c^2 + \tau_{jl}^2 \upsilon_l^2}, \\ &c^2 \sim Inv - Gamma\left(c_a, c_b\right), \tau_{jl} \sim C^+\left(0, b_0\right), \upsilon_l \sim C^+\left(0, b_g\right), \end{aligned} \tag{3.3}$$

where $Inv - Gamma\left(c_a, c_b\right)$ is the inverse gamma distribution with density $p\left(c\right) \propto c^{-c_a-1} exp\left(-c_b/c\right)$ for $c > 0$ and c acts as a weight decay hyperparameter [15]. When the weight shrinks, we have $\tau_{jl}^2 \upsilon_l^2 \ll c^2, \tilde{\tau}_{jl}^2 \to \tau_{jl}^2 \upsilon_l^2$. Subsequently, Eq (3.3) is equivalent to Eq (3.1). When the weight unshrinks, $\tau_{jl}^2 \upsilon_l^2 \gg c^2, \tilde{\tau}_{jl}^2 \to c^2$, $w_{i,j,l} \mid \tau_{jl}, \upsilon_l, c \sim \mathcal{N}\left(0, c^2\right)$. We can adjust the weight by placing a $Inv - Gamma\left(c_a, c_b\right)$ prior to c^2, where c_a and c_b are hyperparameters. Therefore, the regularized horseshoe prior has better generalization performance than the horseshoe prior [15].

3.3. Gaussian scale mixture prior

The Gaussian scale mixture prior [4] is composed of two Gaussian distributions:

$$w_{ij,\ell} \sim p_1 \mathcal{N}\left(0, \sigma_1^2\right) + \left(1 - p_1\right) \mathcal{N}\left(0, \sigma_2^2\right), \tag{3.4}$$

where $w_{ij,\ell}$ is the element of the i-th row and j-th column of the weight matrix, W_ℓ, and σ_1^2 and σ_2^2 are the variances in the mixture components. In practice, we need to make $\sigma_1 > \sigma_2$ such that this prior has a heavier tail than a plain Gaussian prior, and $\sigma_2 \ll 1$ to ensure this prior has a tall spike at zero. The Gaussian scale mixture prior resembles a spike-and-slab prior, and its density function is shown in Figure 2(a) (solid lines). This prior shrinks the weights by probability. $p_1 \in [0, 1]$, σ_1, and σ_2 are hyperparameters, so we can set the values or use a hyperparameter optimization technique to adjust them to make it easier for the prior to use stochastic gradient descent and effectively prevent overfitting.

3.4. Half-Cauchy re-parameterization and weight non-centered parameterization for hierarchical variational learning

Direct parameterization of the half-Cauchy prior in Eq (3.3) leads to a high-variance gradient in the VI for BNNs. To avoid this problem, we used the auxiliary variable distribution parameterization method [14, 15, 29] to parameterize the half-Cauchy distribution, as follows:

$$a \sim C^+\left(0, b\right) \Leftrightarrow a^2 \mid \lambda \sim Inv - Gamma\left(\frac{1}{2}, \frac{1}{\lambda}\right); \lambda \sim Inv - Gamma\left(\frac{1}{2}, \frac{1}{b^2}\right). \tag{3.5}$$

This avoids the challenge that standard exponential family variational approximations struggle to capture thick Cauchy tails, whereas a Cauchy approximating family leads to high variance gradients [15].

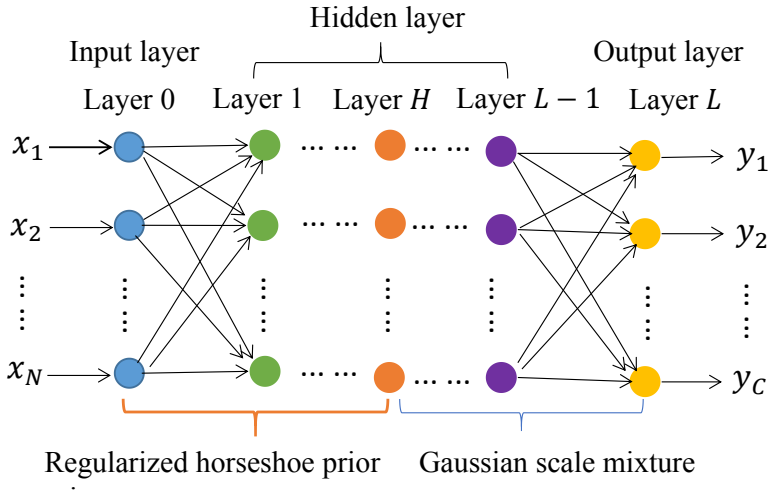


Figure 3. Overall framework of the proposed method. The prior of the weights of the first $H\,(1 \le H < L)$ layers is given by a regularized horseshoe prior, and the other layers are given by a Gaussian scale mixture prior.

Considering generality, a BNN with L layers, the prior of the weights in the first $H\,(1 \le H < L)$ layers is given by a regularized horseshoe prior distribution, and the remaining layers are given by a Gaussian scale mixture prior. The overall framework of the proposed method is illustrated in Figure 3. We use a parameter set, θ, which constitutes all the parameters, as follows:

$$\theta = \left\{\mathcal{W}, c, \left\{\tau_{jl}\right\}_{l=1}^{H}, \{\upsilon_l\}_{l=1}^{H}, \left\{\lambda_{jl}\right\}_{l=1}^{H}, \{\vartheta_l\}_{l-1}^{H}\right\},$$

where $\mathcal{W} = [W_l, W_\ell]$, $1 \le l \le H$, and $H < \ell \le L$. Combined with Eqs (3.3)–(3.5), the prior of θ is then given by:

$$p(\theta) = p(c \mid c_a, c_b) \prod_{l=1}^{H} p\left(W_l, \tau_{jl}, \upsilon_l, \lambda_{jl}, \vartheta_l\right) \prod_{\ell=H+1}^{L} p(W_\ell), \tag{3.6}$$

where

$$p\left(W_l, \tau_{jl}, \upsilon_l, \lambda_{jl}, \vartheta_l\right) = p(\upsilon_l \mid \vartheta_l)\, p(\vartheta_l) \prod_j p\left(\tau_{jl} \mid \lambda_{jl}\right) p\left(\lambda_{jl}\right) \prod_i \mathcal{N}\left(w_{i,j,l} \mid 0, \tilde{\tau}_{ji}^2 \upsilon_l^2\right),$$

$$p\left(\tau_{jl} \mid \lambda_{jl}\right) = Inv - Gamma\left(\frac{1}{2}, \frac{1}{\lambda_{jl}}\right), p\left(\lambda_{jl}\right) = Inv - Gamma\left(\frac{1}{2}, \frac{1}{b_0^2}\right),$$

$$p(\upsilon_l \mid \vartheta_l) = Inv - Gamma\left(\frac{1}{2}, \frac{1}{\vartheta_l}\right), p(\vartheta_l) = Inv - Gamma\left(\frac{1}{2}, \frac{1}{b_g^2}\right),$$

$$p(W_\ell) = \prod_{i,j} p_1 \mathcal{N}\left(w_{ij,\ell} \mid 0, \sigma_1^2\right) + (1 - p_1)\, \mathcal{N}\left(w_{ij,\ell} \mid 0, \sigma_2^2\right).$$

In the regularized horseshoe prior, the weight, $w_{i,j,l}$, and scale, $\tau_{jl}\upsilon_l$, have a strong correlation. This correlation produces good sparsity and gives the posterior distribution of the parameters a pathological funnel-shaped geometry [20], which is difficult to sample or approximate reliably. The use of non-centered parameterization [15, 20] helps alleviate this issue. For Eq (3.3), the re-parameterization weight is

$$\beta_{ij,l} \sim \mathcal{N}(0,1), w_{ij,l} = \tilde{\tau}_{jl}\upsilon_l\beta_{ij,l}. \tag{3.7}$$

Hernandez et al. [17] showed that adopting a non-centered parameterization could significantly improve the posterior approximation quality of BNNs, and that non-centered parameterization leads to a simpler posterior geometry.

3.5. Variational approximation choices

The choice of an approximate posterior distribution significantly affects the inference quality. We employed a fully factorized approximation, which is the simplest and most effective approximation method:

$$q_\phi(\theta) = \prod_{i,j,l} q_\phi\left(w_{ij,l} \mid \tilde{\tau}_{jl}, \upsilon_l\right) \prod_{j,l} q_\phi\left(\tau_{jl}\right) q_\phi\left(\lambda_{jl}\right) \prod_{l} q_\phi(\upsilon_l)\, q_\phi(\vartheta_l)\, q_\phi(W_\ell). \tag{3.8}$$

For the non-negative shrinkage parameters, τ_{jl} and υ_l, we use a log-normal distribution approximation to the posterior—that is, $q_\phi\left(log\tau_{jl}\right) = \mathcal{N}\left(\mu_{\tau_{jl}}, \sigma^2_{\tau_{jl}}\right)$ and $q_\phi(log\upsilon_l) = \mathcal{N}\left(\mu_{\upsilon_l}, \sigma^2_{\upsilon_l}\right)$—which allows us to compute the KL divergence in closed form [15]. The log-likelihood, $logp(\mathcal{D} \mid \theta)$, does not depend on variables λ_{jl} and ϑ_l. Thus, they do not need to impose distributional constraints on the variational approximation. The optimal approximations, $q_\phi\left(\lambda_{jl}\right)$, follow the inverse gamma distributions. We can calculate the necessary fixed-point updates for λ_{jl} [14, 15], as follows:

$$\begin{aligned} q\left(\lambda_{jl}\right) &= Inv - Gamma\left(\lambda_{jl} \mid r, s\right), \\ r = 1, s &= \mathbb{E}\left[\frac{1}{\tau_{jl}}\right] + \frac{1}{b_0^2}. \end{aligned} \tag{3.9}$$

Since $q_\phi\left(log\tau_{jl}\right) = \mathcal{N}\left(\mu_{\tau_{jl}}, \sigma^2_{\tau_{jl}}\right)$, it follows that $\mathbb{E}\left[\frac{1}{\tau_{jl}}\right] = exp\left\{-\mu_{\tau_{jl}} + 0.5\sigma^2_{\tau_{jl}}\right\}$. Then, λ_{jl} can be calculated by the fixed-point updates conditioned on $\mu_{\tau_{jl}}$ and $\sigma^2_{\tau_{jl}}$. The estimates of $\mu_{\tau_{jl}}$ and $\sigma^2_{\tau_{jl}}$ are given by the fixed point updates after each Adam step. We can use the same method to update parameter ϑ_l.

Algorithm 1 A variational sparse Bayesian neural network with regularized horseshoe priors and scale mixture priors

Input: Training set $\mathcal{D}$, model $p(\theta)$, variational approximation $q_\phi(\theta)$, hyperparameters.
Output: Variational parameters ϕ.
1: Initialize variational parameters ϕ.
2: Sample $\varepsilon \sim \mathcal{N}(0,1)$.
3: Calculate $\mathcal{L}_{VI}(\phi)$ using Eq (2.5), and combine with Eq (3.6) and Eq (3.8).
4: **for** iteration T **do**
5: Update $\phi_{\tau_{jl}}, \phi_{\upsilon_l}, \mu_{\beta_{ij,l}}, \sigma_{\beta_{ij,l}} \leftarrow Adam - Optimizer\left(q_\phi(\theta)\right)$ for layer l.
6: Update $\phi_{\mu_{ij,L}}, \phi_{\rho_{w_{ij,\ell}}} \leftarrow Adam - Optimizer\left(q_\phi(\theta)\right)$ for layer ℓ.
7: **for** layer l **do**
8: Conditioned on $\phi_{\tau_{jl}}, \phi_{\upsilon_l}$, update $\phi_{\lambda_{jl}}, \phi_{\vartheta_l}$ using fixed-point updates Eq (3.9)
9: **end for**
10: **end for**

Based on Eq (3.7), parameterize the approximate distribution $q_\phi\left(w_{i,j,l} \mid \tilde{\tau}_{jl}, \upsilon_l\right) = \mathcal{N}\left(\tilde{\tau}_{jl}\upsilon_l\mu_{\beta_{i,j,l}}, \tilde{\tau}_{jl}\upsilon_l\sigma^2_{\beta_{i,j,l}}\right)$, where $\beta_{i,j,l} \sim \mathcal{N}\left(\mu_{\beta_{i,j,l}}, \sigma^2_{\beta_{i,j,l}}\right)$. The approximate posterior distribution of W_ℓ is $q_\phi(W_\ell) = \prod_{i,j}\mathcal{N}\left(\mu_{w_{i,j,\ell}}, \sigma^2_{w_{i,j,\ell}}\right)$ in layer ℓ. The probability distribution of the weight, $w_{i,j,\ell}$, can be learned using the Bayes by Backprop algorithm [4]. $\mu_{w_{i,j,\ell}}$ and $\rho_{w_{i,j,\ell}}$ are variational parameters, where $\sigma_{w_{i,j,\ell}} = log\left(1 + exp\left(\rho_{w_{i,j,\ell}}\right)\right)$. Combined with the techniques of VI, we fit the variational posterior $q_\phi(\theta)$. This procedure is summarized in Algorithm 1. There are six parameters $\phi_{\tau_{jl}}, \phi_{\upsilon_l}, \mu_{\beta_{i,j,l}}, \sigma_{\beta_{i,j,l}}, \phi_{\lambda_{jl}}$, and ϕ_{ϑ_l} to be updated for layer l, and two parameters $\phi_{\mu_{i,j,L}}$ and $\phi_{\rho_{w_{i,j,\ell}}}$ that need to be updated for layer ℓ.

4. Experimental results

In this section, we demonstrate the effectiveness and superiority of the proposed model through a series of experiments. Regression and classification experiments were performed on synthetic toy datasets and widely used UCI benchmarks. We then performed ablation studies to validate the contributions of each component in our model. Finally, we evaluated the performance of our model on anomaly detection tasks. We ran the experiments on a system equipped with a 16 GB GPU and an AMD Ryzen 5900HX CPU. All the experiments were implemented using the PyTorch framework.

4.1. Experimental settings

Training settings. Following previous studies [4, 19], we used the regularized horseshoe prior in Eq (3.3) with $-logc_a = 0$, $-logc_b = 6$, and $b_0 = b_g = 1$. The Gaussian scale mixture prior in Eq (3.4) used $p_1 = 0.25$, $-log\sigma_1 = 0$, and $-log\sigma_2 = 6$. We used ReLU activation and an Adam optimizer with a weight decay of 0.01. The learning rate was set to 10^{-2} for the UCI dataset regression, and 10^{-3} for the UCI dataset classification. In the other experiments, we used a learning rate of 10^{-3} and $p_1 = 0.5$. For the classification and anomaly detection tasks, the training and test batch sizes were set to 128 and 100, respectively. We scaled to initialize weights, according to [27].

Moreover, the prediction and uncertainty estimation performances of the proposed method are affected by the combination of priors. We found that the $H = L - 1$ in Eq (3.6) obtained better results than the other combinations for these simple architectures. See the ablation study section for a relevant discussion.

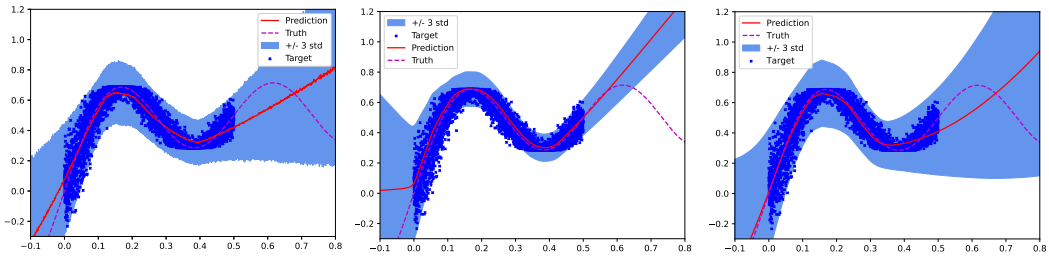
Baseline methods. Our model is inspired by a BNN with a Gaussian scale mixture prior (GSM-BNN) [4], and a BNN with a horseshoe prior (HS-BNN), which establishes these two approaches as natural baselines. We also compared MC-Dropout, a widely adopted technique in computer vision owing to its simple implementation. The subnetwork-linearized Laplace BNN (SLL-BNN [10]), a universal framework for scalable Bayesian deep learning, has demonstrated well-calibrated prediction uncertainty. Adversarial α-divergence minimization (AADM [42]) is a novel approximate Bayesian inference method that achieves competitive performance in both regression and classification tasks. The Riemannian BNN (Riem-BNN [3]) addresses the practical limitations of existing methods by adapting them to posterior geometry through a Riemannian metric derived from the log-posterior gradient. Automatic relevance determination (ARD-BNN [9]) employs a sparsity-inducing prior to automatically prune redundant parameters. In addition, the spike-and-slab group horseshoe (SS-GHS [23]) prior is used to model compression in BNNs and exhibits strong predictive performance. We compared the proposed method to these methods. All baseline methods and the proposed method

were trained under the same experimental settings with the same dataset split.

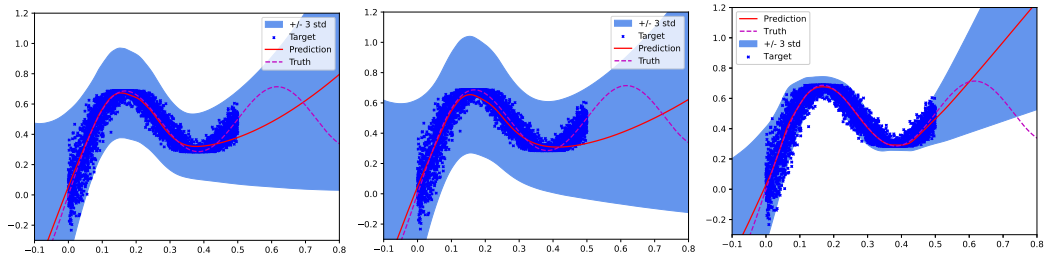
4.2. Regression task

4.2.1. Toy regression dataset

First, we performed a synthetic data experiment [4, 14, 15] on the regression tasks. Specifically, we used a toy dataset. Based on the work of [4], we consider a noisy regression problem: $y = x + 0.3sin(2\pi(x+\epsilon)) + 0.3sin(4\pi(x+\epsilon)) + \epsilon$, where $\epsilon \sim \mathcal{N}\left(0, 0.02^2\right)$. We sampled 5000 inputs, x, from uniform distribution, $x \in U[0, 0.5]$. In this experiment, we used an MLP model with two hidden layers of 800 units each (inputs-800-800-outputs). We compared our model with baseline methods. The resulting predictive distributions made by each method on a toy dataset are shown in Figure 4.



(a) MC-dropout (b) SS-GHS (c) AADM-BNN



(d) Riem-BNN (e) ARD-BNN (f) Ours

Figure 4. Predictive distributions made by each method.

From Figure 4, we can see that all methods have good predictive performance (red solid lines) on the training data in $U[0, 0.5]$, but poor predictive performance in the interval $[0.6, 0.8]$ without data. The proposed model is closer to the true model for x in [0.5, 0.6]. BNNs tend to be uncertain in the absence of data. In uncertainty estimations, the baselines lead to an over-inflated predictive uncertainty in the intervals $[-0.1, 0]$ and $[0.6, 0.8]$, and our method appears to be more reasonable than the baseline methods. Moreover, the proposed method combines the advantages of the regularized horseshoe prior and Gaussian scale mixture prior to bring the weight posterior closer to the prior and provides a realistic predictive distribution that is better than other methods in both fitting and predictive uncertainty. These results indicate that the proposed method is effective and exhibits better performance.

4.2.2. Experiments on UCI regression datasets

In this subsection, we evaluate the predictive performance of our model on benchmark UCI datasets that are widely used in real-valued regression tasks.

Following the experimental protocols of [14, 15], for smaller datasets, we created a training set using 90% of the data, and the remaining data were used as the test set. This process was repeated 20 times. Because the data magnitudes differed significantly, each dataset was normalized such that the input features and targets had zero mean and unit variance. The prior for the noise precision, γ, in Eq (2.2) is the gamma distribution—that is, $p(\gamma) = Gamma(6, 6)$. In this experiment, we used a neural network with one hidden layer of 50 units and the batch size for training was set to 64.

Table 1. Average test RMSE and standard errors for the UCI regression datasets. Bold indicates the best results.

Dataset	MC-Dropout	GSM-BNN [4]	HS-BNN [14]	SS-GHS [23]	SLL-BNN [10]	AADM-BNN [42]	Riem-BNN [3]	ARD-BNN [9]	Ours
Boston	2.91 ± 0.27	4.28 ± 0.88	3.20 ± 0.60	2.92 ± 0.62	2.97 ± 0.19	2.94 ± 0.32	2.91 ± 0.18	2.90 ± 0.26	**2.90 ± 0.20**
Concrete	4.92 ± 0.31	8.72 ± 0.66	5.46 ± 0.55	5.47 ± 0.44	5.23 ±0.12	4.98 ± 0.13	4.86 ± 0.38	4.82 ± 0.35	**4.81 ± 0.46**
Energy	1.41 ± 0.06	3.04 ± 0.36	1.90 ± 0.24	1.69 ± 0.10	1.66 ± 0.04	1.52 ± 0.11	1.63 ± 0.18	1.59 ± 0.24	**1.37 ± 0.38**
Kin8nm	0.06 ± 0.00	0.09 ± 0.00	0.07 ± 0.00	0.07 ± 0.00	0.10 ± 0.00	0.06 ± 0.00	0.07 ± 0.00	0.07 ± 0.00	**0.06 ± 0.00**
Naval	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00	**0.00 ± 0.00**
Power	3.70 ± 0.10	4.39 ± 0.18	4.03 ± 0.16	3.93 ± 0.02	4.02 ± 0.04	3.97 ± 0.05	3.93 ± 0.14	3.92 ± 0.12	**3.69 ± 0.15**
Wine	0.61 ± 0.01	0.65 ± 0.04	0.68 ± 0.04	0.64 ± 0.03	**0.62 ± 0.01**	0.63 ± 0.11	0.64 ± 0.03	0.63 ± 0.03	0.81 ± 0.05
Yacht	0.97 ± 0.33	6.21 ± 2.43	1.63 ± 1.12	1.19 ± 0.41	1.01 ± 0.09	1.18 ± 0.14	1.16 ± 0.15	1.12 ± 0.14	**0.94 ± 0.39**

Table 2. Average test log-likelihood and standard errors for the UCI regression datasets. Bold indicates the best results.

Dataset	MC-Dropout	GSM-BNN [4]	HS-BNN [14]	SS-GHS [23]	SLL-BNN [10]	AADM-BNN [42]	Riem-BNN [3]	ARD-BNN [9]	Ours
Boston	-2.41 ± 0.15	-3.49 ± 0.03	-2.51 ± 0.12	-2.51 ± 0.21	**-2.46 ± 0.06**	-2.51 ± 0.24	-2.50 ± 0.03	-2.49 ± 0.16	-2.47 ± 0.19
Concrete	-3.12 ± 0.04	-3.92 ± 0.02	-3.91 ± 0.02	-3.12 ± 0.05	-3.04 ±0.02	-3.08 ± 0.07	-3.14 ± 0.02	-3.10 ± 0.09	**-3.00 ± 0.16**
Energy	-2.12 ± 0.00	-3.37 ± 0.00	-2.14 ± 0.05	-2.22 ± 0.01	-1.99 ± 0.02	-2.08 ± 0.01	-2.23 ± 0.00	-2.12 ± 0.03	**-1.99 ± 0.01**
Kin8nm	1.25 ± 0.01	0.34 ± 0.00	1.24 ± 0.03	1.22 ± 0.03	0.95 ± 0.01	1.29 ± 0.02	1.34 ± 0.00	**1.35 ± 0.03**	1.23 ± 0.01
Naval	5.91 ± 0.00	3.96 ± 0.00	5.21 ± 0.00	5.48 ± 0.01	3.80 ± 0.01	**6.07 ± 0.08**	3.96 ± 0.00	4.58 ± 0.18	5.22 ± 0.00
Power	-2.83 ± 0.02	-3.80 ± 0.00	-2.85 ± 0.03	-2.87 ± 0.03	-2.80 ± 0.05	-2.88 ± 0.02	-2.87 ± 0.00	-2.83 ± 0.04	**-2.80 ± 0.03**
Wine	-0.98 ± 0.14	-1.04 ± 0.03	-2.66 ± 0.04	-1.93 ± 0.23	**-0.93 ± 0.01**	-0.97 ± 0.32	-0.95 ± 0.04	-0.94 ± 0.03	-1.71 ± 0.02
Yacht	-1.71 ± 0.03	-4.09 ± 0.05	-2.57 ± 0.17	-2.52 ± 0.02	**-1.55 ± 0.03**	-1.77 ± 0.01	-1.58 ± 0.04	-1.56 ± 0.03	-2.42 ± 0.01

The average RMSE and standard errors of the UCI regression are listed in Table 1. It can be seen that the performances of GSM-BNN and HS-BNN were the worst on the eight datasets, whereas the performances of the other methods were very similar. Our model provided a lower RMSE in seven of the eight datasets compared with the baselines. In addition, all the models performed well on large datasets such as 'Naval' and 'Power', whereas on small datasets such as 'Boston' and 'Yacht', the performance of each model showed inconsistent results. Our model yielded satisfactory experimental results for various datasets.

The average test log-likelihood and standard errors for the UCI regression are shown in Table 2. It can be observed that SLL-BNN outperformed the other baseline methods on 'Wine' and 'Yacht', ARD-BNN yielded satisfactory results for Energy, and AADM-BNN achieved the best predictive log-likelihood on 'Naval'. In addition, our method performed better than the baselines on the other four datasets. The regularized horseshoe prior can effectively select important features for prediction and severely shrink weights that are irrelevant to features, whereas the Gaussian scale mixture prior has the effect of regularization. These are crucial for avoiding overfitting and obtaining an accurate estimate of predictive uncertainty. These results further confirm the proposed method's effectiveness and its good

predictive and generalization performances.

4.3. Classification task

4.3.1. Toy classification dataset

Following [3], we consider a toy classification problem for the banana dataset. We trained the baseline models with two hidden layers, with 16 hidden units per layer. We compared our method with the baselines. Figure 5 shows the binary classification confidence estimates for various methods. It can be observed that the baseline methods have high uncertainty both within and away from the data support. Our method shows better confidence because it only has higher uncertainty at the decision boundaries, which decreases outside of the data support.

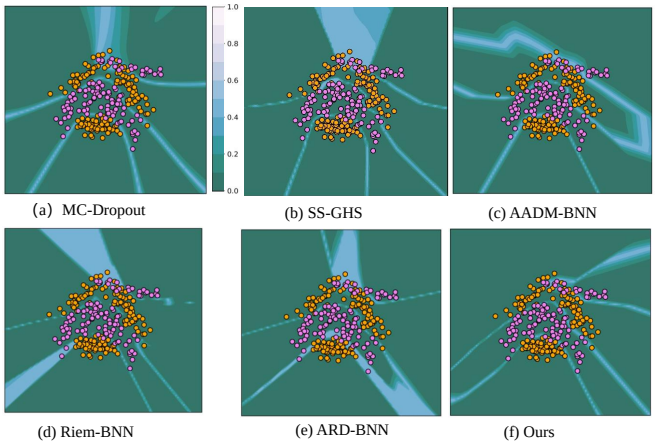


Figure 5. Classification confidence estimates on the banana dataset.

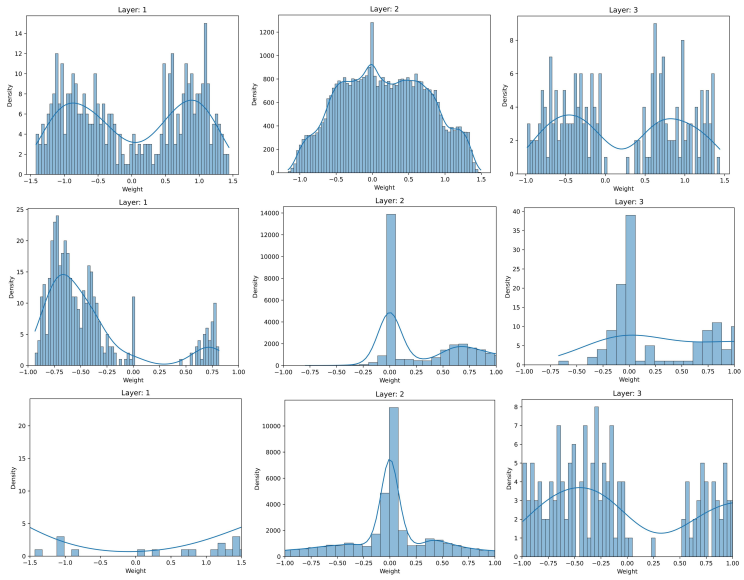


Figure 6. Posterior weight distributions of the GSM-BNN (top), HS-BNN (middle), and our method (bottom).

We further inspected the histograms of the posterior weight distributions per layer of the banana dataset. These weight distributions are shown as histograms in Figure 6. The figure shows that the weight distributions of the first and second layers of the GSM-BNN are dense, and the third layer shows a clear separation into two groups. The histogram corresponding to the HS-BNN shows sparse patterns, and there are two well-separated groups in the weight distribution of the first layer. This is more evident in the proposed method. It is evident that the horseshoe prior exhibits a more significant shrinkage of irrelevant weights. Moreover, our method generates more complex approximate posterior distributions and well-separated groups in the weight distribution of the first layer. These results indicate that the proposed model with regularized horseshoe priors is better suited for feature selection than the other models.

4.3.2. UCI classification datasets

We compared our method with the baselines on a set of six UCI classification datasets using a fully connected network with three layers, 60 hidden units per layer, and ReLU activation. The predictive distribution was estimated using MC sampling with 10 draws from the approximate posterior of each approach. In Table 3, we compare the methods in terms of their mean accuracy and standard error on the test set.

From the results in Table 3, we see that our method performs better than all the baseline methods on four datasets. The performances of the AADA-BNN on EEG and HS-BNN on Ionosphere were slightly better than those of our method. These results indicate that the proposed method has strong predictive and generalization performance.

Table 3. Average classification accuracy (%) for UCI classification datasets. Bold indicates the best results.

Method	EEG	HTRU_2	Magic	Miniboo	Ionosphere	WDBC
MC-Dropout	77.59 ± 1.3	97.84 ± 0.22	87.20 ± 0.35	93.54 ± 0.13	87.61 ± 5.70	95.31 ± 2.81
GSM-BNN [4]	76.28 ± 1.8	97.84 ± 0.29	83.50 ± 0.66	92.03 ± 0.32	80.06 ± 5.83	93.85 ± 3.17
HS-BNN [14]	78.11 ± 2.8	97.97 ± 0.21	87.17 ± 0.69	93.55 ± 0.21	**88.61 ± 6.24**	94.19 ± 2.89
SS-GHS [23]	76.16 ± 1.2	97.64 ± 0.19	86.97 ± 0.41	92.83 ± 0.12	86.42 ± 6.47	95.49 ± 1.92
SLL-BNN [10]	75.77 ± 2.2	97.83 ± 0.27	83.65 ± 0.81	92.15 ± 0.20	82.64 ± 6.34	93.85 ± 3.17
AADM-BNN [42]	**78.66 ± 2.7**	97.98 ± 0.24	85.81 ± 0.51	93.43 ± 0.17	81.76 ± 6.03	94.56 ± 2.28
Riem-BNN [3]	78.52 ± 2.4	97.88 ± 0.22	86.71 ± 0.63	93.44 ± 0.11	85.90 ± 5.65	94.84 ± 2.37
ARD-BNN [9]	78.60 ± 1.6	97.96 ± 0.21	87.03 ± 0.65	93.49 ± 0.20	86.70 ± 5.41	94.41 ± 2.03
Ours	78.48 ± 1.93	**98.00 ± 0.10**	**87.21 ± 0.42**	**93.58 ± 0.16**	85.47 ± 6.94	**95.61 ± 2.74**

The ECE and NLL are commonly used measures to assess the quality of uncertainty estimation in models. Tables 4 and 5 list the results of the average tests ECE and NLL, respectively. It can be seen that our method consistently performs better than all the baselines in terms of ECE, indicating that our approach produces better calibrated uncertainty estimates. In terms of NLL, we observed that only on the EEG dataset was our method not optimal; on all other datasets, it achieved better results than the baselines. These results indicate that the proposed method provides both good classification accuracy and excellent uncertainty estimation performance.

Table 4. Average test ECE for UCI classification datasets. Bold indicates the best results.

Method	EEG	HTRU_2	Magic	Miniboo	Ionosphere	WDBC
MC-Dropout	0.175 ± 0.022	0.053 ± 0.020	0.080 ± 0.002	0.013 ± 0.001	0.192 ± 0.025	0.163 ± 0.024
GSM-BNN [4]	0.212 ± 0.020	0.062 ± 0.020	0.088 ± 0.007	0.013 ± 0.002	0.193 ± 0.051	0.170 ± 0.039
HS-BNN [14]	0.175 ± 0.030	0.060 ± 0.017	0.078 ± 0.007	0.014 ± 0.004	0.198 ± 0.033	0.172 ± 0.027
SS-GHS [23]	0.178 ± 0.022	0.055 ± 0.027	0.084 ± 0.008	0.015 ± 0.002	0.192 ± 0.034	0.167 ± 0.019
SLL-BNN [10]	0.199 ± 0.025	0.050 ± 0.015	0.081 ± 0.005	0.014 ± 0.003	0.193 ± 0.051	0.170 ± 0.039
AADM-BNN [42]	0.175 ± 0.031	0.057 ± 0.014	0.085 ± 0.008	0.015 ± 0.004	0.209 ± 0.048	0.166 ± 0.023
Riem-BNN [3]	0.181 ± 0.029	0.057 ± 0.012	0.086 ± 0.016	0.016 ± 0.003	0.200 ± 0.042	0.167 ± 0.020
ARD-BNN [9]	0.177 ± 0.021	0.052 ± 0.015	0.082 ± 0.006	0.013 ± 0.003	0.195 ± 0.039	0.164 ± 0.025
Ours	**0.173 ± 0.024**	**0.050 ± 0.012**	**0.078 ± 0.004**	**0.012 ± 0.003**	**0.191 ± 0.037**	**0.162 ± 0.028**

Table 5. Average test NLL for UCI classification datasets. Bold indicates the best results.

Method	EEG	HTRU_2	Magic	Miniboo	Ionosphere	WDBC
MC-Dropout	3.425 ± 0.265	0.335 ± 0.036	2.274 ± 0.137	1.065 ± 0.032	2.432	0.742 ± 0.34
GSM-BNN [4]	3.781 ± 0.288	0.345 ± 0.046	2.631 ± 0.105	1.271 ± 0.051	3.178 ± 0.929	0.981 ± 0.505
HS-BNN [14]	3.490 ± 0.444	0.323 ± 0.033	2.046 ± 0.108	1.029 ± 0.034	2.816 ± 0.995	0.927 ± 0.460
SS-GHS [23]	3.775 ± 0.223	0.338 ± 0.037	2.161 ± 0.181	1.092 ± 0.020	2.864 ± 0.934	0.759 ± 0.306
SLL-BNN [10]	3.863 ± 0.349	0.347 ± 0.043	2.606 ± 0.129	1.252 ± 0.031	3.122 ± 0.901	0.981 ± 0.505
AADM-BNN [42]	**3.402 ± 0.427**	0.322 ± 0.039	2.261 ± 0.081	1.047 ± 0.028	2.908 ± 0.962	0.868 ± 0.363
Riem-BNN [3]	3.529 ± 0.433	0.324 ± 0.041	2.202 ± 0.103	1.083 ± 0.030	2.976 ± 0.940	0.824 ± 0.375
ARD-BNN [9]	3.503 ± 0.420	0.322 ± 0.034	2.199 ± 0.096	1.032 ± 0.033	2.857 ± 1.005	0.799 ± 0.442
Ours	3.430 ± 0.308	**0.319 ± 0.031**	**2.038 ± 0.068**	**1.024 ± 0.027**	**2.317 ± 1.106**	**0.699 ± 0.437**

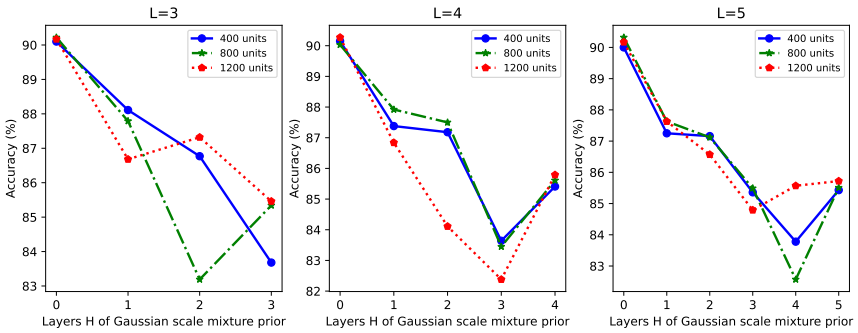
4.4. Ablation study

In this subsection, we discuss an ablation study conducted to investigate the impact of different combinations of prior distributions on model performance, and sensitivity analysis of the hyperparameters involved in the model, such as parameter H, layer-specific global shrinkage parameter b_g, etc.

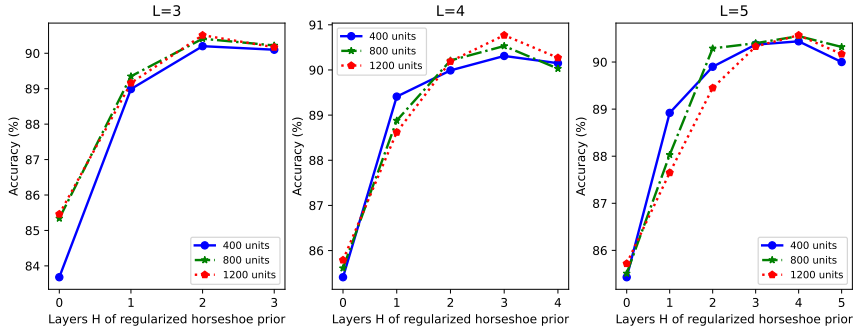
Impact of prior order and H. To analyze the effect of the combination order of the two prior distributions and the parameter H on model performance, we used three neural networks with 400, 800, and 1200 hidden units, respectively. We considered a network with $L(L = 3, 4, 5)$ layers and conducted image classification experiments on the FashionMNIST dataset. Figure 7 shows the results for different network structures and values of H.

First scenario: the weights of the first $H\,(1 \leq H < L)$ layers were given a Gaussian scale mixture prior, and the remaining layers a regularized horseshoe prior. The case $H = 0$ corresponds to all weights using a regularized horseshoe prior, while corresponds to all weights using a Gaussian scale mixture prior. As shown in Figure 7(a), the accuracy of all structures decreased as the number of Gaussian scale mixture prior layers increased. The performance of the BNNs with both Gaussian scale mixture priors and regularized horseshoe priors was worse than that of the BNNs using only

regularized horseshoe priors or only Gaussian scale mixture priors. Consider the second scenario: the weights of the first $H\,(1 \leq H < L)$ layers were assigned a regularized horseshoe prior, while the remaining layers were assigned a Gaussian scale mixture prior. $H = 0$ represents the prior of the weights given by the Gaussian scale mixture prior and $H = L\,(L = 3, 4, 5)$ represents the priors of the weights given by a regularized horseshoe prior. It can be observed that the accuracy of all structures increased with an increase in the number of layers H of the regularized horseshoe prior. Among all the network structures used, the model has the best classification performance when $H = L - 1$. Moreover, using the regularized horseshoe prior in the first H layer proved to be more effective than using the Gaussian-scale mixture prior. Increasing the number of layers using a regularized horseshoe prior can improve the performance of the model. These results demonstrate that the proposed method is effective and achieves a good performance.



(a)



(b)

Figure 7. Impacts of different combinations of the two priors and parameter H on the FashionMNIST.

Impact of hyperparameters. We conducted all ablation study experiments using a 3-hidden layer fully connected neural network with 800 hidden units per layer. The prior of the first H layers is given by a regularized horseshoe prior. Figure 8(a) shows the impact of the parameter H and hidden units on the inference times. As H increases, the model inference times decrease, whereas the more hidden units there are, the longer the inference time.

The regularized horseshoe prior provides severe shrinkage toward zero for small weights. However, owing to the presence of noise, the shrunken weights are never actually zero [15]. Figure 8(b) shows the test accuracy versus the percentage of weight removed. The weights of each network were ordered

by signal-to-noise ratio (SNR) [4] and then removed based on a threshold. The SNR is defined as $\left|\mu_{ij,l}\right|/\sigma_{ij,l} < \varsigma$, where $\left|\mu_{ij,l}\right|$ is the absolute value of the posterior mean of weight $\omega_{ij,l}$, $\sigma_{ij,l}$ is the posterior standard deviation, and ς is a positive constant. When the SNR holds, we remove the weights; that is, we set them to zero. To determine the sparsity of the network, we examined the effect of setting the variational posterior of some of the weights to zero. The GSM-BNN and HS-BNN methods are compared. Many network weights can be successfully pruned without significantly affecting performance. At least 94% of the weights were removed before there was a catastrophic decrease in the test accuracy for the GSM-BNN; the figures were 97% for the HS-BNN, and 98% for our method. Sparsity-inducing priors encourage a broad spread of weights, and using the SNR to prune proved to be highly effective. After training using the weight-pruning approach, we obtained a smaller, sparser network for the prediction.

We conducted sensitivity analysis on the number of samples T in Eq (2.7), b_0, b_g in Eq (3.3), and p_1 in Eq (3.4). Figures 8(c)–8(e) show the analysis results for these parameters. The inference time increased with the number of MC samples T, and the accuracy curve for the test set first increased and then decreased. When $T = 30$, the accuracy was highest. This indicates that we need to balance accuracy and inference time when choosing T. The model performs best when both b_0 and b_g are –9. In Eq (3.4), when $p_1 = 1$, the weight prior follows a Gaussian distribution, and the model performance is poor. When $p_1 = 0.5$, the model's performance is best.

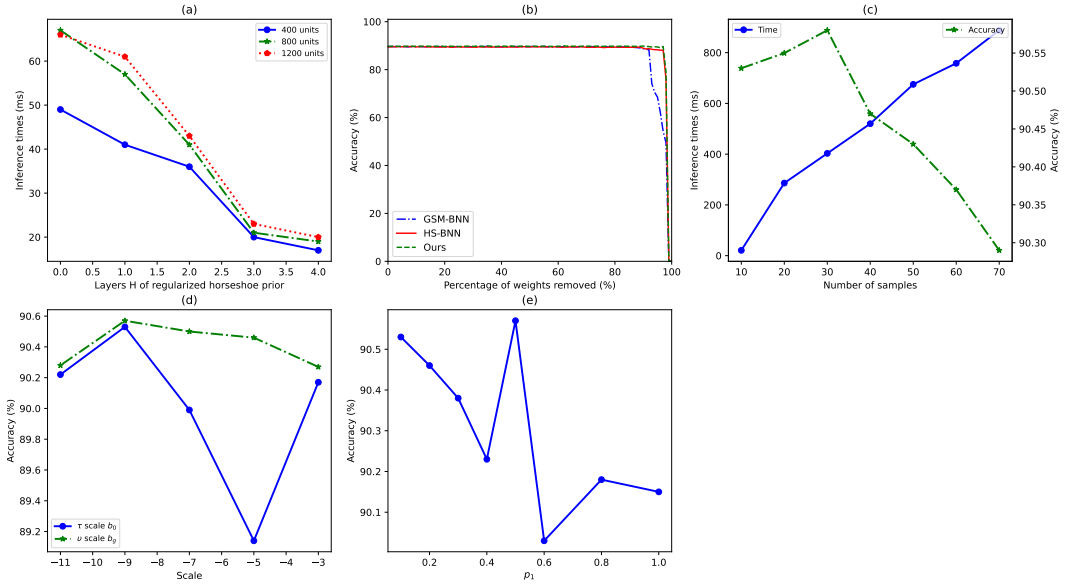


Figure 8. Sensitivity analysis of some key parameters.

4.5. Anomaly detection

In this subsection, we evaluate the performance of the proposed Bayesian model for anomaly detection tasks. The objective of anomaly detection is to identify out-of-distribution (OOD) data, using a classifier trained solely on in-distribution data [35]. When the OOD samples in the test set differ from the training set, machine learning classifiers often assign high-confidence predictions to anomalous inputs. Consequently, most anomaly detection methods rely on classifier output uncertainty as an indicator for OOD detection [35]. BNNs have proven effective for anomaly detection because

they yield more reliable uncertainty estimates than deterministic DNNs, which produce only point estimates. Hendrycks et al. [16] provided benchmarks for anomaly detection. Krueger et al. [26] demonstrated that BNNs outperform deterministic neural networks in this context. A well-calibrated BNN should express higher uncertainty for OOD inputs, enabling more accurate detection.

We used three widely known image classification datasets—CIFAR-10, CIFAR-100, and SVHN—as in-distribution training sets. A fully connected network with four hidden layers ($H = 3$) and 800 hidden units per layer was trained. During testing, inputs included both in-distribution and OOD data. For OOD examples, we used real images and synthetic datasets (random Gaussian noise) following [16]. Model predictions were converted into OOD scores using the maximum probability (Eq (2.8)) and entropy (Eq (2.10)) metrics. In-distribution data were treated as negative examples, while OOD data were treated as positive examples.

Performance was assessed using the area under the receiver operating characteristic curve (AUROC) and the area under the precision–recall curve (AUPR). The ROC curve plots the true positive rate against the false positive rate and represents the probability that an anomalous example has a higher detector score than in-distribution data. Higher AUROC values indicate better detection performance.

Table 6. Image datasets anomaly detection for entropy.

In	Out	MC-Dropout		GSM-BNN [4]		HS-BNN [14]		SS-GHS [23]		SLL-BNN [10]		AADM-BNN [42]		Riem-BNN [3]		ARD-BNN [9]		Ours	
		AUROC	AUPR	AUROC	AUPR	AUROC	AUPR	AUROC	AUPR	AUROC	AUPR	AUROC	AUPR	AUROC	AUPR	AUROC	AUPR	AUROC	AUPR
CIFAR-10	Gaussian	97.54	89.57	92.32	93.12	67.97	94.50	98.55	93.22	84.87	94.70	99.03	94.81	96.79	76.84	95.56	84.31	99.23	95.18
	CIFAR-100	62.93	33.12	58.42	21.11	63.04	21.25	60.42	21.65	57.59	21.25	57.26	20.99	61.28	22.30	61.53	21.96	55.88	22.16
	SVHN	69.72	22.54	67.58	32.24	74.27	38.67	73.02	32.58	69.00	38.67	70.10	33.93	73.78	33.09	71.00	29.22	70.94	34.01
CIFAR-100	Gaussian	96.52	97.17	92.09	41.76	82.27	99.41	99.73	97.63	80.01	32.80	99.56	97.97	98.84	89.57	99.73	98.10	99.73	99.94
	CIFAR-10	56.94	18.34	57.89	18.29	60.13	19.49	57.32	19.38	57.54	16.95	57.44	19.21	59.03	19.54	59.00	19.16	57.53	19.85
	SVHN	61.57	25.47	61.96	21.19	68.89	31.88	67.53	20.35	59.27	19.84	69.25	20.91	74.05	33.12	69.02	26.84	72.32	34.06
SVHN	Gaussian	80.27	71.53	81.45	64.05	80.93	47.90	90.39	62.17	79.02	64.31	85.41	63.61	90.20	63.31	91.09	62.31	87.78	64.14
	CIFAR-10	52.89	36.84	67.85	29.43	58.08	28.41	71.69	26.85	67.02	31.60	68.68	31.34	71.93	29.24	75.19	30.60	69.84	32.79
	CIFAR-100	54.40	39.48	69.34	31.55	60.38	28.74	72.90	29.03	67.99	35.35	70.11	34.08	72.14	29.30	76.62	34.35	70.69	35.62

Table 6 presents the anomaly detection results using entropy-based scores for CIFAR-10, CIFAR-100, and SVHN. Our method achieved consistently strong performance, outperforming baseline approaches across all three datasets. When Gaussian noise was used as OOD data, all models achieved high AUROC and AUPR values. However, the AUPR values dropped noticeably when the in- and out-of-distribution datasets were CIFAR-10 and CIFAR-100 (in either order), likely due to their similarity. Overall, the results show that our BNN approach is effective and competitive with, or superior to, baseline methods.

Table 7. Image datasets anomaly detection for maximum probability.

In	Out	MC-Dropout		GSM-BNN [4]		HS-BNN [14]		SS-GHS [23]		SLL-BNN [10]		AADM-BNN [42]		Riem-BNN [3]		ARD-BNN [9]		Ours	
		AUROC	AUPR	AUROC	AUPR	AUROC	AUPR	AUROC	AUPR	AUROC	AUPR	AUROC	AUPR	AUROC	AUPR	AUROC	AUPR	AUROC	AUPR
CIFAR-10	Gaussian	96.64	78.64	89.03	68.53	68.03	94.81	95.58	83.45	80.55	84.04	96.79	84.72	95.68	78.75	94.66	83.41	95.74	84.82
	CIFAR-100	62.41	21.30	52.57	22.79	62.22	20.99	61.55	21.24	58.02	21.56	55.79	20.90	60.47	21.04	60.51	20.37	57.11	22.52
	SVHN	68.12	29.09	57.09	25.66	72.72	34.93	70.30	28.35	66.42	30.46	66.74	30.15	70.65	30.07	70.91	29.33	66.90	30.22
CIFAR-100	Gaussian	96.03	96.44	86.85	32.08	81.66	97.95	99.32	98.46	80.01	33.56	99.24	98.11	98.59	90.73	99.28	98.01	98.46	99.11
	CIFAR-10	58.73	19.79	57.33	16.95	59.71	20.21	57.90	18.93	56.54	18.57	56.45	19.14	58.31	19.43	59.22	19.81	56.86	19.75
	SVHN	61.03	26.11	60.28	19.84	66.44	29.81	66.63	25.03	59.27	26.46	68.97	27.05	67.02	28.07	66.91	27.90	64.43	28.11
SVHN	Gaussian	73.63	66.31	85.21	71.61	78.26	45.31	90.43	61.84	89.90	63.19	83.66	60.55	86.35	59.16	88.81	55.42	83.94	47.71
	CIFAR-10	52.09	37.60	72.98	54.34	56.49	29.24	80.82	50.77	76.25	49.66	72.24	51.45	78.50	49.27	83.77	50.76	81.73	52.92
	CIFAR-100	53.18	39.35	74.19	54.08	58.81	29.30	83.28	52.94	77.33	54.53	72.82	50.17	81.52	42.85	78.36	45.65	81.85	53.43

Table 7 reports results using the maximum probability metric. Similar conclusions to those from Table 6 can be drawn, with our method performing well for most cases. Notably, the maximum

probability method yielded higher AUROC and AUPR scores than entropy when CIFAR-100 and SVHN were used as in-distribution datasets.

5. Conclusions and future work

In this study, we present a variational sparse BNN that combines a regularized horseshoe prior with a Gaussian-scale mixture prior. These priors promote weight sparsity and mitigate overfitting, improving both regularization and uncertainty quantification. Experimental results show that our model outperforms the baselines in predictive performance and uncertainty estimation, confirming its regularization effectiveness.

BNNs can estimate both aleatoric and epistemic uncertainty through carefully selected priors, thereby quantifying model confidence—a capability that demonstrates core value in high-risk fields like medical diagnosis and autonomous driving.

In medical imaging analysis (such as pathological slide recognition), uncertainty estimation serves to flag low-confidence areas, assisting physicians in reviewing potential misdiagnoses and reducing the risk of oversight. For critical applications such as cancer screening, the system can differentiate between data uncertainty (such as image noise) and model epistemic uncertainty. This enables dynamic allocation of secondary examination resources: cases exhibiting high epistemic uncertainty should be prioritized for manual review, while those with high data uncertainty may warrant repeat testing.

When implemented on multi-source sensor data (including LiDAR and camera inputs), Bayesian uncertainty estimation dynamically assesses environmental perception reliability. For instance, in scenarios with elevated uncertainty during adverse weather conditions, the system can activate conservative driving protocols—such as speed reduction or collision warnings—to prevent hazardous misjudgments of pedestrian positions.

The main limitations of this work are: (i) the absence of theoretical guarantees for selecting H; (ii) sensitivity to hyperparameter settings, with only a limited analysis conducted; and (iii) experiments restricted to relatively simple neural networks. Future work should explore the method's applicability to more complex architectures and larger datasets. Given the strong influence of prior choice on BNN performance, further investigation into the relationship between different priors, weight redundancy, and regularization is warranted.

Author contributions

Xu Chen: Writing—review and editing, supervision, conceptualization, funding acquisition; Lilong Sima: Writing—review and editing, methodology, data curation, software; Zhen Wei: Writing—review and editing, data curation, software; Xingde Duan: Methodology, conceptualization, investigation; Ping Feng: Conceptualization, investigation, visualization; Fuhong Song: Conceptualization, investigation, visualization. All authors have read and agreed to the published version of the manuscript.

Use of Generative-AI tools declaration

The authors declare they have not used Artificial Intelligence (AI) tools in the creation of this article.

Acknowledgments

This study was supported in part by the Guizhou Provincial Major Scientific and Technological Program (Grant No. QKHCG[2024]ZD018), Guizhou Provincial Science and Technology Project (Grant Nos. QKHBasic-ZK[2024]694, QKHBasic-ZK[2021]308), and the Guizhou University of Finance and Economics Introduced Talent Research Start-Up Project (Grant No. 2023YJ21).

Conflict of interest

All authors declare no conflicts of interest.

References

1. I. Abdullayev, E. Akhmetshin, I. Kosorukova, E. Klochko, W. Cho, G. P. Joshi, Modeling of extended osprey optimization algorithm with Bayesian neural network: An application on Fintech to predict financial crisis, *AIMS Mathematics*, **9** (2024), 17555–17577. https://doi.org/10.3934/math.2024853
2. J. C. Bai, Q. F. Song, G. Cheng, Efficient variational inference for sparse deep learning with theoretical guarantee, In: *Proceedings of the 34th International Conference on Neural Information Processing Systems*, Red Hook: Curran Associates Inc., 2020, 466–476.
3. F. Bergamin, Pablo. Moreno-Muñoz, S. Hauberg, G. Arvanitidis, Riemannian Laplace approximations for Bayesian neural networks, *37th Annual Conference on Neural Information Processing Systems*, New Orleans, US, 2023.
4. C. Blundell, J. Cornebise, K. Kavukcuoglu, D. Wierstra, Weight uncertainty in neural networks, In: *Proceedings of the 32nd International Conference on Machine Learning*, New York: PMLR, 2015, 1613–1622.
5. C. M. Carvalho, N. G. Polson, J. G. Scott, Handling sparsity via the horseshoe, In: *Proceedings of the Twelfth International Conference on Artificial Intelligence and Statistics*, New York: PMLR, 2009, 73–80.
6. Y. Cao, Y. G. Kao, Z. Wang, X. S. Yang, J. H. Park, W. Xie, Sliding mode control for uncertain fractional-order reaction-diffusion memristor neural networks with time delays, *Neural Networks*, **178** (2024), 106402. https://doi.org/10.1016/j.neunet.2024.106402
7. X. Chen, C. C. Liu, Y. Zhao, Z. Y. Jia, G. Jin, Improving adversarial robustness of Bayesian neural networks via multi-task adversarial training, *Inform. Sciences*, **592** (2022), 156–173. https://doi.org/10.1016/j.ins.2022.01.051
8. X. Chen, Y. Zhao, C. C. Liu, Medical image segmentation using scalable functional variational Bayesian neural networks with Gaussian processes, *Neurocomputing*, **500** (2022), 58–72. https://doi.org/10.1016/j.neucom.2022.05.055
9. N. Dabiran, B. Robinson, R. Sandhu, Mo. Khalil, D. Poirel, A. Sarkar, Sparse Bayesian neural networks for regression: Tackling overfitting and computational challenges in uncertainty quantification, arXiv:2310.15614, (2023).

10. E. Daxberger, E. Nalisnick, J. U. Allingham, J. Antorán, J. Hernández-Lobato, Bayesian deep learning via subnetwork inference, In: *Proceedings of the 38th International Conference on Machine Learning*, New York: PMLR, 2021, 2510–2521.
11. G. Franchi, A. Bursuc, E. Aldea, S. Dubuisson, I. Bloch, Encoding the latent posterior of Bayesian neural networks for uncertainty quantification, *IEEE T. Pattern Anal.*, **46** (2024), 2027–2040. https://doi.org/10.1109/TPAMI.2023.3328829
12. B. Ghoshal, A. Tucker, B. Sanghera, W. L. Wong, Estimating uncertainty in deep learning for reporting confidence to clinicians in medical image segmentation and disease detection, *Comput. Intell.*, **37** (2021), 701–734. https://doi.org/10.1111/coin.12411
13. Y. Gal, R. Islam, Z. Ghahramani, Deep Bayesian active learning with image data, In: *Proceedings of the 34th International Conference on Machine Learning*, New York: PMLR, 2017, 1183–1192.
14. S. Ghosh, Y. Yao, F. Doshi-Velez, Model selection in Bayesian neural networks via horseshoe priors, *J. Mach. Learn. Res.*, **20** (2019), 1–46.
15. S. Ghosh, Y. Yao, F. Doshi-Velez, Structured variational learning of Bayesian neural networks with horseshoe priors, In: *Proceedings of the 35th International Conference on Machine Learning*, New York: PMLR, 2018, 1739–1748.
16. D. Hendrycks, K. Gimpel, A baseline for detecting misclassified and out-of-distribution examples in neural networks, In: *International Conference on Learning Representations*, 2017, 1–12.
17. J. M. Hernández-Lobato, D. Hernández-Lobato, A. Suárez, Expectation propagation in linear regression models with spike-and-slab priors, *Machine Learning*, **99** (2015), 437–487. https://doi.org/10.1007/s10994-014-5475-7
18. J. M. Hernández-Lobato, Y. Z. Li, M. Rowland, T. Bui, D. Hernández-Lobato, R. Turner, Black-box alpha-divergence minimization, In: *Proceedings of the 33rd International Conference on Machine Learning*, New York: PMLR, 2016, 1511–1520.
19. A. Hubin, G. Storvik, Variational inference for Bayesian neural networks under model and parameter uncertainty, arXiv:2305.00934, (2023).
20. J. Ingraham, D. S. Marks, Variational inference for sparse and undirected models, In: *Proceedings of the 34th International Conference on Machine Learning*, New York: PMLR, 2017, 1607–1616.
21. D. R. Insua, R. Naveiro, V. Gallego, J. Poulos, Adversarial machine learning: Bayesian perspectives, *J. Am. Stat. Assoc.*, **118** (2023), 2195–2206. https://doi.org/10.1080/01621459.2023.2183129
22. S. Jantre, S. Bhattacharya, T. Maiti, Layer adaptive node selection in Bayesian neural networks: Statistical guarantees and implementation details, *Neural Networks*, **167** (2023), 309–330. https://doi.org/10.1016/j.neunet.2023.08.029
23. S. Jantre, S. Bhattacharya, T. Maiti, Spike-and-slab shrinkage priors for structurally sparse Bayesian neural networks, *IEEE T. Neur. Net. Lear.*, **36** (2025), 11176–11188. https://doi.org/10.1109/TNNLS.2024.3485529
24. A. Kendall, Y. Gal, What uncertainties do we need in Bayesian deep learning for computer vision?, In: *Proceedings of the 31st International Conference on Neural Information Processing Systems*, Red Hook: Curran Associates Inc., 2017, 5580–5590.

25. D. P. Kingma, M. Welling, Auto-encoding variational Bayes, *International Conference on Learning Representations*, Banff, Canada, 2014.

26. D. Krueger, C.-W. Huang, R. Islam, R. Turner, A. Lacoste, A. Courville, Bayesian hypernetworks, *Second workshop on Bayesian Deep Learning (NIPS 2017)*, Long Beach, USA, 2017.

27. Y. Z. Li, Y. Gal, Dropout inference in Bayesian neural networks with alpha-divergences, In: *Proceedings of the 34th International Conference on Machine Learning*, New York: PMLR, 2017, 2052–2061.

28. Z. Li, L. Bin, On weighted residual varextropy: characterization, estimation and application, *AIMS Mathematics*, **10** (2025), 11234–11259. https://doi.org/10.3934/math.2025509

29. C. Louizos, K. Ullrich, M. Welling, Bayesian compression for deep learning, In: *Proceedings of the 31st International Conference on Neural Information Processing Systems*, Red Hook: Curran Associates Inc., 2017, 3290–3300.

30. H. Li, Y. G. Kao, H. B. Bao, Y. Q. Chen, Uniform stability of complex-valued neural networks of fractional order with linear impulses and fixed time delays, *IEEE T. Neur. Net. Lear.*, **33** (2022), 5321–5331. https://doi.org/10.1109/TNNLS.2021.3070136

31. P. F. Li, Q. H. Hu, X. F. Wang, Federated learning meets Bayesian neural network: Robust and uncertainty-aware distributed variational inference, *Neural Networks*, **185** (2025), 107135. https://doi.org/10.1016/j.neunet.2025.107135

32. Y. D. Lin, Q. T. Zhang, B. Gao, J. S. Tang, P. Yao, C. X. Li, et al., Uncertainty quantification via a memristor Bayesian deep neural network for risk-sensitive reinforcement learning, *Nat. Mach. Intell.*, **5** (2023), 714–723. https://doi.org/10.1038/s42256-023-00680-y

33. M. Magris, A. Iosifidis, Bayesian learning for neural networks: An algorithmic survey, *Artif. Intell. Rev.*, **56** (2023), 11773–11823. https://doi.org/10.1007/s10462-023-10443-1

34. A. Malinin, M. Gales, Predictive uncertainty estimation via prior networks, In: *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, Red Hook: Curran Associates Inc., 2018, 7047–7058.

35. J. Mitros, A. Pakrashi, B. M. Namee, Ramifications of approximate posterior inference for Bayesian deep learning in adversarial and out-of-distribution settings, In: *Computer Vision–ECCV 2020 Workshops*, Cham: Springer, 2020, 71–87. https://doi.org/10.1007/978-3-030-66415-2_5

36. E. T. Nalisnick, On prior for Bayesian neural networks, PhD Thesis, UC Irvine, 2018.

37. E. T. Nalisnick, J. M. Hernandez-Lobato, P. Smyth, Dropout as a structured shrinkage prior, In: *Proceedings of the 36th International Conference on Machine Learning*, New York: PMLR, 2019, 4712–4722.

38. S. W. Ober, L. Aitchison, Global inducing point variational posteriors for Bayesian neural networks and deep Gaussian processes, In: *Proceedings of the 38th International Conference on Machine Learning*, New York: PMLR, 2021, 8248–8259.

39. H. Overweg, A.-L. Popkes, A. Ercole, Y. Z. Li, J. M. Hernández-Lobato, Y. Zaykov, et al., Interpretable outcome prediction with sparse Bayesian neural networks in intensive care, arXiv:1905.02599, (2019).

40. J. Piironen, A. Vehtari, On the hyperprior choice for the global shrinkage parameter in the horseshoe prior, In: *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, New York: PMLR, 2017, 905–913.

41. I. Qureshi, J. H. Yan, Q. Abbas, K. Shaheed, A. B. Riaz, A. Wahid, et al., Medical image segmentation using deep semantic-based methods: A review of techniques, applications and emerging trends, *Inform. Fusion*, **90** (2023), 316–352. https://doi.org/10.1016/j.inffus.2022.09.031

42. S. Rodríguez-Santana, D. Hernández-Lobato, Adversarial α-divergence minimization for Bayesian approximate inference, *Neurocomputing*, **471** (2022), 260–274. https://doi.org/10.1016/j.neucom.2020.09.076

43. S. Y. Sun, G. D. Zhang, J. X. Shi, Functional variational Bayesian neural networks, *Bayesian Deep Learning (NeurIPS 2018)*, Montréal, Canada, 2019.

44. M. Sarıyıldız, K. Alahari, D. Larlus, Y. Kalantidis, Fake it till you make it: Learning transferable representations from synthetic imagenet clones, *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Vancouver, Canada, 2023, 8011–8021. https://doi.org/10.1109/CVPR52729.2023.00774

45. L. Skaaret-Lund, G. Storvik, A. Hubin, Sparsifying Bayesian neural networks with latent binary variables and normalizing flows, arXiv:2305.03395, (2023).

46. F. B. Smith, A. Kirsch, S. Farquhar, Y. Gal, A. Foster, T. Rainforth, Prediction-oriented Bayesian active learning, In: *Proceedings of The 26th International Conference on Artificial Intelligence and Statistics*, New York: PMLR, 2023, 7331–7348.

47. Y. Sun, Q. F. Song, F. M. Liang, Consistent sparse deep learning: theory and computation, *J. Am. Stat. Assoc.*, **117** (2022), 1981–1995. https://doi.org/10.1080/01621459.2021.1895175

48. Y. Yang, D. D. Guo, B. Chen, D. X. Hu, Continual learning with Bayesian compression for shared and private latent representations, *Neural Networks*, **185** (2025), 107167. https://doi.org/10.1016/j.neunet.2025.107167

49. C. Zhang, J. Butepage, H. Kjellstrom, S. Mandt, Advances in variational inference, *IEEE T. Pattern Anal.*, **41** (2019), 2008–2026. https://doi.org/10.1109/TPAMI.2018.2889774

50. Z. D. Zhu, K. X. Lin, A. K. Jain, J. Y. Zhou, Transfer learning in deep reinforcement learning: A survey, *IEEE T. Pattern Anal.*, **45** (2023), 13344–13362. https://doi.org/10.1109/TPAMI.2023.3292075