Heidelberg University

Hardware and Artificial Intelligence Lab

Master Thesis

Multi-Level Quantization of Stochastic Variational Inference based Bayesian Neural Networks

Name:	Yong Wu
Matriculation number:	3770613
Supervisor:	Holger Fröning
Date of submission:	May 14, 2025

DECLARATION

I hereby certify that I have written the work myself and that I have not used any sources or aids other than those specified and that I have marked what has been taken over from other people's works, either verbatim or in terms of content, as foreign. I also certify that the electronic version of my thesis transmitted completely corresponds in content and wording to the printed version. I agree that this electronic version is being checked for plagiarism at the university using plagiarism software.

Heidelberg, May 14, 2025

Yong Wu

ABSTRACT

Bayesian Neural Networks (BNNs) integrate the representational power of standard neural networks with the uncertainty estimation capabilities of Bayesian Inference, offering a robust framework to address challenges such as overconfidence and overfitting. However, the inherent complexity of BNNs —due to the use of weight distributions— renders the process computationally intensive, thereby hindering their practical deployment, especially on edge devices. To overcome these challenges from the perspectives of edge deployment and inference speed, this thesis investigates a series of quantization strategies specifically tailored for BNNs.

As the main benchmark we utilize a synthetic dataset for a regression task. This dataset specifically allows us to characterize both regression performance and uncertainty prediction of aleatoric and epistemic uncertainty.

We first analyze the crucial role of input representation in the quantization process and introduce partition quantization based on thermometer coding to mitigate the impact of input quantization errors. Building on this foundation, we propose three quantization methods: one based on the quantization of values sampled from the distributions of the BNN, another leveraging the mean and variance of the utilized Gaussian distributions, and a combined strategy that integrates both approaches. Each method is evaluated based on its impact on model performance under a fixed bit-width setting. Subsequently, we conduct a deeper analysis of how varying quantization bit-widths influence accuracy and uncertainty estimation.

Extensive experiments demonstrate that our proposed quantization techniques substantially reduce computational complexity while maintaining prediction reliability, underscoring their potential for achieving efficient and robust BNN deployment in real-world, resource-constrained environments.

ZUSAMMENFASSUNG

Bayesianische Neuronale Netze (BNNs) vereinen die Ausdrucksstärke klassischer neuronaler Netze mit der Fähigkeit zur Unsicherheitsquantifizierung aus der Bayesschen Inferenz und bieten damit ein robustes Rahmenwerk zur Bewältigung von Herausforderungen wie Überkonfidenz und Overfitting. Die Verwendung von Gewichtungsverteilungen führt jedoch zu erheblichem Rechenaufwand, was die praktische Einsatzfähigkeit von BNNs – insbesondere auf ressourcenbeschränkten Edge-Geräten – stark einschränkt. Zur Bewältigung dieser Einschränkungen in Bezug auf Effizienz und Inferenzgeschwindigkeit untersucht diese Arbeit eine Reihe von Quantisierungsstrategien, die speziell auf BNNs zugeschnitten sind.

Als Haupt-Benchmark verwenden wir einen synthetischen Datensatz für eine Regressionsaufgabe, der eine differenzierte Bewertung sowohl der Vorhersageleistung als auch der Unsicherheitsabschätzung – einschließlich aleatorischer und epistemischer Unsicherheiten – ermöglicht.

Zunächst analysieren wir die entscheidende Rolle der Eingabedarstellung im Quantisierungsprozess und stellen eine Partitionierungsquantisierung basierend auf Thermometerkodierung vor, um den Einfluss von Eingabequantisierungsfehlern zu minimieren. Darauf aufbauend schlagen wir drei Quantisierungsmethoden vor: eine auf der Quantisierung von Stichproben aus den Verteilungen des BNN basierende Methode, eine zweite, die Mittelwert und Varianz der verwendeten Gaußschen Verteilungen nutzt, sowie eine kombinierte Strategie, die beide Ansätze integriert. Jede Methode wird hinsichtlich ihrer Auswirkungen auf die Modellleistung bei festgelegter Bitbreite bewertet. Anschließend erfolgt eine tiefere Analyse, wie unterschiedliche Quantisierungsbitbreiten die Genauigkeit und die Unsicherheitsabschätzung beeinflussen.

Umfangreiche Experimente zeigen, dass die vorgeschlagenen Quantisierungstechniken die rechnerische Komplexität deutlich reduzieren und gleichzeitig die Zuverlässigkeit der Vorhersagen bewahren. Dies unterstreicht ihr Potenzial für einen effizienten und robusten Einsatz von BNNs in realen, ressourcenbeschränkten Umgebungen.

CONTENTS

1 INTRODUCTION

1.1 MOTIVATION

Deep neural networks (DNNs) have revolutionized numerous fields with their remarkable performance on complex tasks (Gawlikowski et al., 2022). However, these powerful models face significant limitations that restrict their applicability in critical domains. Specifically, DNNs frequently exhibit overconfidence in their predictions (Guo et al., 2017) and demonstrate vulnerability to distribution shifts between training and deployment environments (Ovadia et al., 2019). These shortcomings raise serious concerns for applications where reliability is paramount, such as healthcare diagnostics and autonomous vehicle navigation (Varshney and Alemzadeh, 2017).

Bayesian Neural Networks (BNNs) emerged as a promising framework to address these challenges by incorporating uncertainty quantification into deep learning architectures. Unlike conventional DNNs, BNNs represent model parameters as probability distributions rather than point estimates, enabling principled uncertainty estimation and enhanced robustness against overfitting (Mitros and Namee, 2019). This probabilistic approach allows practitioners to distinguish between aleatoric uncertainty (inherent data noise) and epistemic uncertainty (model knowledge gaps), providing valuable insights for decision-making processes (Gawlikowski et al., 2022; Kendall and Gal, 2017).

Among various Bayesian inference techniques, Stochastic Variational Inference (SVI) offers a compelling compromise between computational tractability and uncertainty representation quality (Hoffman et al., 2013). By approximating complex posterior distributions with simpler, parameterized alternatives, SVI enables scalable Bayesian reasoning in neural networks while preserving essential uncertainty information (Blundell et al., 2015). This approach has demonstrated promising results across diverse applications (Kendall and Gal, 2017), with early work by Graves (2011) establishing practical foundations for variational inference in neural networks. Together, these advances highlight the significant potential of Bayesian methodologies for improving uncertainty quantification in deep learning systems.

Despite these theoretical advantages, implementing SVI in practice presents significant computational challenges. When replacing weights or activations of DNNs with probability distributions, the resulting computations become substantially more complex and in some cases even intractable, thus heavily increasing the computational cost compared to traditional DNNs (Sharma and Jennings, 2020). Although

SVI can extend the BNN method to large data sets with good results, their practical deployment is hindered by high computational resource demands and slow inference speed (Mohamad, Bouchachia, and Sayed-Mouchaweh, 2018). As Mohamad, Bouchachia, and Sayed-Mouchaweh (2018) note, "stochastic optimization improves the performance of VI, its serial employment prevents scaling up the inference and harnessing distributed resources," highlighting fundamental scalability challenges in Bayesian methods.

To address this dilemma, neural network quantization techniques (Gholami et al., 2021; Jacob et al., 2017; Jain et al., 2020) have garnered significant attention in recent years, offering substantial reductions in computational and memory requirements by reducing the precision of model parameters (from floating-point to fixed-point representations). For resource-constrained environments such as edge devices used in medical diagnostics or autonomous systems, quantized BNNs could potentially enable robust uncertainty-aware decision making without prohibitive computational costs. However, existing quantization methods have primarily focused on deterministic neural networks, with little consideration for effectively preserving uncertainty representations in Bayesian inference. This is particularly concerning since uncertainty estimates in BNNs depend critically on the fidelity of the posterior distribution, which may be significantly distorted through naive quantization approaches.

This research gap motivates our exploration of novel approaches to Bayesian Neural Network quantization, aimed at achieving both computational efficiency and accurate uncertainty estimation. Our work is driven by key questions: How can the expressiveness of posterior distributions be maintained during quantization? How do low-precision representations affect the quality of uncertainty estimates? Are there specific quantization strategies better suited to the unique demands of Bayesian inference?

1.2 GOALS

In this work, we aim to explore efficient low-precision quantization approaches for SVI while preserving its uncertainty representation capabilities. Through a systematic approach, we analyze the impact of quantization on Bayesian uncertainty representation and propose multi-level quantization strategies specifically designed for SVI.

We investigate quantization at four critical junctures: input quantization, activation quantization, quantization of probabilistic distribution parameters (means and variances), and quantization of sampled values from these distributions. Based on this framework, we analyze the feasibility of uncertainty-aware Bayesian inference on resource-

constrained hardware platforms such as BrainScaleS-2 (Pehle et al., 2022).

The primary objective of this research is to establish a comprehensive understanding of how different quantization levels and strategies affect SVI performance and uncertainty estimation quality, while providing practical solutions to reduce computational and memory costs.

Our contributions can be summarized as follows:

- **Input Representation Optimization:** We analyze the impact of non-negativity constraints in input quantization and propose a novel Sectional Thermometer Encoding approach that extends traditional thermometer encoding (Buckman et al., 2018). Our method divides the input space into optimally-sized sections based on the data distribution, with separate handling for positive and negative values. Unlike conventional quantization that simply adds bias for non-negativity, our approach preserves relative distances within each section while efficiently utilizing the limited 5-bit representation. This partitioned encoding strategy adaptively scales to different input feature distributions and bit-widths, significantly reducing information loss during quantization.

- **Multi-level Quantization Strategies:** We propose and evaluate three complementary quantization methods: one based on quantizing samples from probability distributions, another leveraging quantization of Gaussian distribution means and variances, and a combined approach integrating the strengths of both. Each method is specifically optimized for uncertainty preservation.

- **Bit-width Impact Analysis:** We comprehensively evaluate how quantization precision, ranging from very low to high bit-widths, affects model performance, uncertainty estimation quality, and resource consumption. We provide practical guidelines for optimizing activation functions, gradient computation, and weight range, achieving an optimal balance between computational efficiency and model quality.

Through these contributions, our work provides important theoretical guidance and practical techniques for building computationally efficient Bayesian Neural Networks with reliable uncertainty estimation capabilities, particularly in resource-constrained environments.

1.3 STRUCTURE

This thesis is organized as follows:

First, in Chapter 2, we provide an overview of Bayesian Neural Networks, detailing how different inference methods can be used to

obtain weight distributions and how to effectively quantify uncertainty. Additionally, this chapter introduces fundamental concepts of neural network quantization, establishing the necessary theoretical foundation for understanding our research.

Chapter 3 presents a comprehensive review of related work in the fields of Bayesian Neural Networks, uncertainty quantification, and neural network quantization, highlighting recent advances and identifying research gaps that motivate our current investigation.

In Chapter 4, we introduce the datasets and evaluation metrics employed in our experiments, including regression tasks such as the Noisy Sine Wave dataset and classification tasks like Two Moons and DirtyMNIST, along with specific quality metrics designed to assess both predictive accuracy and uncertainty quantification.

Chapter 5 explores various input representation methods for low-dimensional continuous value data, presenting approaches based on dense layers, binary sequence encoding, and sectional encoding techniques, with comprehensive experimental results demonstrating their relative effectiveness.

Chapter 6 details our proposed quantization methods at different levels of granularity, including Sample-Based Quantization, Parameter-Based Quantization, and Fully Integrated Quantization, with experimental validations across multiple datasets.

In Chapter 7, we investigate optimization strategies specifically designed for quantized SVI, including the effects of specialized activation functions like Alpha-Sigmoid and Alpha-Sine, as well as weight clipping techniques that enhance quantization efficiency while preserving model performance.

Finally, Chapter 8 presents a systematic bit-width sensitivity analysis, examining how different quantization methods perform across various precision levels, providing practical insights into the minimum bit-width requirements for maintaining both predictive accuracy and high-quality uncertainty estimation.

2 BACKGROUND

In this chapter, we cover necessary background for this work. First, we introduce Variational Inference (Section 2.1). Then, we describe how uncertainty estimates can be obtained, especially when dividing the total uncertainty into the aleatoric component (inherent in the input) and the epistemic component (from the model weights) in Section 2.2. The probabilistic programming framework Pyro used in this work is introduced in Section 2.3. Then, we cover the fundamentals of quantization in Section 2.4. Lastly, we introduce Brevitas, the quantization framework used in this research, in Section 2.5.

2.1 VARIATIONAL INFERENCE

2.1.1 Bayesian Neural Networks

In contrast to conventional neural networks, where each parameter is represented by a single point estimate, Bayesian Neural Networks (BNNs) characterize weight parameters through probability distributions (Jospin et al., 2022). The primary objective is to derive the posterior weight distribution $p(w|D)$ conditioned on the observed data (Blei, Kucukelbir, and McAuliffe, 2017). As their designation suggests, Bayesian Neural Networks are fundamentally grounded in Bayes' theorem, which establishes that posterior beliefs are modulated by prior beliefs (Jospin et al., 2022):

$$p(w|D) = \frac{p(D|w)p(w)}{p(D)} \tag{2.1}$$

Equation (2.1) represents Bayes' theorem applied to neural networks, where $p(w|D)$ denotes the posterior probability distribution of weights w given the observed data D. The term $p(D|w)$ represents the likelihood function, measuring how well the weights explain the observed data. The prior distribution $p(w)$ encapsulates our initial beliefs about the weights before observing any data, while $p(D)$ is the evidence or marginal likelihood, acting as a normalizing constant.

Using Equation (2.1), prior beliefs can be updated with observed data, resulting in a posterior weight distribution that combines prior beliefs and observed information. This posterior distribution enables Bayesian Neural Networks (BNNs) to quantify prediction uncertainty, a critical advantage over deterministic neural networks.

However, computing the exact posterior distribution is typically intractable in practice. The denominator $p(D) = \int p(D|w)p(w)\mathrm{d}w$ involves a high-dimensional, non-convex integral that becomes computationally infeasible for complex models (Jospin et al., 2022). This fundamental challenge necessitates approximate inference methods, with variational inference being one of the most effective approaches for BNNs (Mullachery, Khera, and Husain, 2018).

OBTAINING BNN PREDICTIONS Once a posterior distribution over weights $p(w|D)$ (or an approximation thereof) is obtained, predictions in Bayesian Neural Networks are made through marginalization over the weight distribution. For a new input x_*, the predictive distribution is given by (Wilson and Izmailov, 2022):

$$p(y_*|x_*, D) = \int p(y_*|x_*, w)p(w|D)\mathrm{d}w \tag{2.2}$$

Equation (2.2) illustrates the Bayesian model averaging process, where $p(y_*|x_*, D)$ represents the predictive distribution for a new output y_* given a new input x_* and the training data D. This integration effectively averages predictions over all possible weight configurations, weighted by their posterior probabilities $p(w|D)$. The term $p(y_*|x_*, w)$ denotes the likelihood of the output given the input and a specific weight configuration.

For regression tasks, assuming a Gaussian likelihood $p(y_*|x_*, w) = \mathcal{N}(y_*|f^w(x_*), \sigma^2)$, where f^w represents the neural network with weights w and σ^2 is the noise variance, the predictive distribution is obtained by marginalizing over the posterior distribution of network weights. This Bayesian approach provides a comprehensive uncertainty quantification framework that deterministic neural networks inherently lack.

The predictive mean is computed as:

$$\mathbb{E}[y_* \mid x_*, \mathcal{D}] = \int f^w(x_*)\, p(w \mid \mathcal{D})\, \mathrm{d}w \tag{2.3}$$

This integral represents a weighted average of neural network outputs $f^w(x_*)$ across all possible weight configurations, with weights determined by the posterior probability $p(w|\mathcal{D})$. Unlike point estimation methods in traditional neural networks, this approach accounts for the uncertainty in weight values.

The predictive variance is given by:

$$\mathrm{Var}[y_* \mid x_*, \mathcal{D}] = \int (f^w(x_*) - \mathbb{E}[y_* \mid x_*, \mathcal{D}])^2 p(w \mid \mathcal{D})\, \mathrm{d}w + \sigma^2 \tag{2.4}$$

The variance in Equation (2.4) has two distinct components: the first term quantifies epistemic uncertainty, which stems from model parameter uncertainty and can theoretically be reduced with additional data; the second term σ^2 represents aleatoric uncertainty, capturing

the inherent noise and stochasticity in the data that cannot be reduced through additional observations. This decomposition offers valuable insights for uncertainty-aware decision-making in various application domains, which we will analyze further in Section 2.2.

By explicitly modeling both sources of uncertainty, Bayesian Neural Networks provide a more nuanced and reliable uncertainty quantification framework compared to their deterministic counterparts, enabling more informed decision-making in high-stakes domains where uncertainty awareness is critical.

In practice, since the exact posterior $p(w|D)$ is typically intractable, predictions are made using approximate methods. When variational inference is employed, the true posterior is approximated by a variational distribution $q_\phi(w)$, and the predictive distribution becomes (Blundell et al., 2015):

$$p(y_*|x_*, D) \approx \int p(y_*|x_*, w) q_\phi(w) \mathrm{d}w \tag{2.5}$$

This integral is usually approximated using Monte Carlo sampling (Gal and Ghahramani, 2016):

$$p(y_*|x_*, D) \approx \frac{1}{S} \sum_{s=1}^{S} p(y_*|x_*, w^{(s)}), \quad w^{(s)} \sim q_\phi(w) \tag{2.6}$$

where S is the number of weight samples drawn from the variational distribution. This approximation forms the basis for practical implementation of BNN predictions and uncertainty estimation in variational inference frameworks.

2.1.2 Variational Inference

Variational Inference (VI) offers an elegant solution to the intractability problem in Bayesian Neural Networks by recasting inference as an optimization task (Blei, Kucukelbir, and McAuliffe, 2017). Instead of directly computing the true posterior distribution $p(w|D)$, VI approximates it with a simpler, parameterized distribution $q_\phi(w)$ chosen from a tractable family. The objective is to find the parameters ϕ that minimize the Kullback-Leibler (KL) divergence (Shlens, 2014) between the approximate and true posterior:

$$\phi^* = \arg\min_\phi \mathrm{KL}(q_\phi(w)||p(w|D)) \tag{2.7}$$

The KL divergence, which measures the dissimilarity between two probability distributions, is defined as:

$$\mathrm{KL}(q_\phi(w)||p(w|D)) = \mathbb{E}_{q_\phi(w)}\left[\log \frac{q_\phi(w)}{p(w|D)}\right] \tag{2.8}$$

However, directly minimizing this expression is not feasible since it requires knowledge of the very posterior $p(w|D)$ we are trying to approximate. Through mathematical manipulation, this optimization problem can be reformulated as the maximization of the Evidence Lower Bound (ELBO):

$$\text{ELBO}(\phi) = \mathbb{E}_{q_\phi(w)}[\log p(D|w)] - \text{KL}(q_\phi(w)||p(w)) \tag{2.9}$$

The ELBO consists of two terms: the first term represents the expected log-likelihood, encouraging the approximate posterior to explain the observed data well, while the second term acts as a regularizer, keeping the approximate posterior close to the prior distribution.

To understand why maximizing the ELBO is equivalent to minimizing the KL divergence between $q_\phi(w)$ and $p(w|D)$, we can start by expanding the KL divergence expression (Jordan et al., 1998; Bishop, 2006):

$$\begin{aligned}\text{KL}(q_\phi(w)||p(w|D)) &= \mathbb{E}_{q_\phi(w)}\left[\log \frac{q_\phi(w)}{p(w|D)}\right] \\ &= \mathbb{E}_{q_\phi(w)}[\log q_\phi(w)] - \mathbb{E}_{q_\phi(w)}[\log p(w|D)]\end{aligned} \tag{2.10}$$

Using Bayes' theorem, we can expand the posterior term:

$$\begin{aligned}\mathbb{E}_{q_\phi(w)}[\log p(w|D)] &= \mathbb{E}_{q_\phi(w)}\left[\log \frac{p(D|w)p(w)}{p(D)}\right] \\ &= \mathbb{E}_{q_\phi(w)}[\log p(D|w)] + \mathbb{E}_{q_\phi(w)}[\log p(w)] \\ &\quad - \mathbb{E}_{q_\phi(w)}[\log p(D)]\end{aligned} \tag{2.11}$$

Since $p(D)$ is a constant with respect to w, we have $\mathbb{E}_{q_\phi(w)}[\log p(D)] = \log p(D)$. Substituting this back into Equation (2.10) yields:

$$\begin{aligned}\text{KL}(q_\phi(w)||p(w|D)) &= \mathbb{E}_{q_\phi(w)}[\log q_\phi(w)] - \mathbb{E}_{q_\phi(w)}[\log p(D|w)] \\ &\quad - \mathbb{E}_{q_\phi(w)}[\log p(w)] + \log p(D) \\ &= -\mathbb{E}_{q_\phi(w)}[\log p(D|w)] + \text{KL}(q_\phi(w)||p(w)) \\ &\quad + \log p(D)\end{aligned} \tag{2.12}$$

Through further rearrangement of Equation (2.12), we arrive at:

$$\text{KL}(q_\phi(w)||p(w|D)) + \mathbb{E}_{q_\phi(w)}[\log p(D|w)] - \text{KL}(q_\phi(w)||p(w)) = \log p(D) \tag{2.13}$$

Since $\log p(D)$ is the logarithm of the marginal likelihood (also known as the evidence), we can recognize that:

$$\begin{aligned}\text{ELBO}(\phi) &= \mathbb{E}_{q_\phi(w)}[\log p(D|w)] - \text{KL}(q_\phi(w)||p(w)) \\ &= \log p(D) - \text{KL}(q_\phi(w)||p(w|D))\end{aligned} \tag{2.14}$$

This derivation reveals a crucial insight: since $\log p(D)$ is constant with respect to ϕ, maximizing the ELBO is mathematically equivalent to minimizing the KL divergence between the approximate posterior $q_\phi(w)$ and the true posterior $p(w|D)$ (Bishop, 2006). As $\mathrm{KL}(q_\phi(w)||p(w|D)) \geq 0$, with equality if and only if $q_\phi(w) = p(w|D)$ almost everywhere, the ELBO is indeed a lower bound on the log evidence, i.e., $\mathrm{ELBO}(\phi) \leq \log p(D)$.

This equivalence provides the theoretical foundation for variational inference in Bayesian Neural Networks (Graves, 2011; Blundell et al., 2015). By optimizing the ELBO, we obtain an approximate posterior that balances data fit (through the expected log-likelihood term) and simplicity (through the KL divergence to the prior).

VARIATIONAL DISTRIBUTION The choice of variational distribution is a critical element in the variational inference framework, directly impacting both the quality of posterior approximation and computational efficiency. The ideal variational distribution strikes a balance between expressiveness and tractability—it must be sufficiently flexible to capture the essential characteristics of the true posterior while remaining computationally manageable (Blei, Kucukelbir, and McAuliffe, 2017).

Several families of variational distributions have emerged in the literature. The mean-field approximation assumes that latent variables are independent of each other:

$$q_\phi(w) = \prod_{i=1}^{M} q_{\phi_i}(w_i) \tag{2.15}$$

In Bayesian Neural Networks, this often manifests as a product of independent Gaussian distributions:

$$q_\phi(w) = \prod_{i=1}^{M} \mathcal{N}(w_i|\mu_i, \sigma_i^2) \tag{2.16}$$

While computationally efficient, this independence assumption fails to capture important correlations among parameters (Attias, 1999).

Structured mean-field approaches provide a middle ground by maintaining dependencies within defined variable groups (Wiegerinck, 2013):

$$q_\phi(w) = \prod_{k=1}^{K} q_{\phi_k}(w_{G_k}) \tag{2.17}$$

where $\{G_1, G_2, \ldots, G_K\}$ represents a partitioning of weights into groups.

For more expressive representations, normalizing flows (Rezende and Mohamed, 2016) transform simple distributions into complex ones through a series of invertible transformations:

$$w_K = f_K \circ f_{K-1} \circ \cdots \circ f_1(w_0), \quad w_0 \sim q_0(w_0) \tag{2.18}$$

In practical applications for Bayesian Neural Networks, researchers have explored various approaches to balance expressiveness and efficiency. Louizos and Welling (2017) proposed multiplicative normalizing flows that capture complex dependencies while maintaining computational feasibility:

$$w = \mu + \sigma \odot z_K \tag{2.19}$$

where z_K is produced by a normalizing flow and $\odot$ represents element-wise multiplication.

The expressiveness-efficiency tradeoff manifests clearly in covariance structures. Diagonal Gaussian approximations require only $2M$ parameters for a network with M weights, but severely constrain the representational capacity. Full-covariance Gaussians, while more expressive, require $O(M^2)$ parameters, quickly becoming intractable for modern neural networks.

Recent innovations include structured approximations with constrained covariance matrices, such as low-rank plus diagonal decompositions:

$$\Sigma = \text{diag}(\sigma^2) + UU^T \tag{2.20}$$

where $U \in \mathbb{R}^{M \times r}$ with $r \ll M$, providing a practical compromise between expressiveness and computational efficiency (Diederik P. Kingma, Salimans, and Welling, 2015).

The appropriate selection of variational distribution ultimately depends on the specific inference task, model architecture, dataset characteristics, and available computational resources.

KL ANNEALING IN VARIATIONAL INFERENCE In variational inference, particularly when training complex Bayesian Neural Networks, optimizing the ELBO may face a common challenge: the KL divergence term can excessively penalize the variational posterior distribution for deviating from the prior distribution, especially during the early stages of training. In such cases, the model might prematurely converge to suboptimal solutions, resulting in a variational posterior distribution that fails to adequately capture patterns in the data. To address this issue, researchers have proposed the KL annealing technique (Bowman et al., 2016; Sønderby et al., 2016).

The core idea behind KL annealing is to reduce the influence of the KL divergence term during the initial training phase and then gradually increase its weight as training progresses, until it reaches

the full weight in the original ELBO formula. The modified ELBO objective function can be expressed as:

$$\text{ELBO}_\beta(\phi) = \mathbb{E}_{q_\phi(w)}[\log p(D|w)] - \beta \cdot \text{KL}(q_\phi(w)||p(w)) \tag{2.21}$$

where β is a time-varying weight coefficient, typically starting from a small value close to 0 and gradually increasing to 1 as training proceeds. This approach allows the model to primarily focus on data fitting (through the expected log-likelihood term) during the early stages of training, while progressively introducing regularization constraints (through the KL divergence term) in later stages.

The choice of annealing schedule significantly impacts training dynamics. Common annealing strategies include linear annealing:

$$\beta_t = \min(1, \frac{t}{T}) \tag{2.22}$$

and sigmoid annealing:

$$\beta_t = \text{sigmoid}(\alpha(t - T/2)) \tag{2.23}$$

where t is the current training step, T is the predetermined annealing period, and α controls the steepness of the sigmoid function.

Higgins et al. (2017) proposed β-VAE, which treats the KL weight as a fixed hyperparameter to control the degree of disentanglement in learned representations, which can be viewed as a special case of KL annealing. In contrast, Fu et al. (2019) proposed a cyclical annealing strategy, periodically adjusting β throughout the training process to help the model escape local optima.

In Bayesian Neural Networks, KL annealing has been proven effective in mitigating the "posterior collapse" problem, where the variational posterior distribution prematurely degenerates to the prior distribution (Chen et al., 2017). Zhang et al. (2018) demonstrated how progressive KL annealing helps models gradually build complex posterior approximations, particularly in hierarchical Bayesian models.

2.1.3 Stochastic Variational Inference

For large-scale datasets, traditional variational methods become computationally prohibitive as they require processing the entire dataset for each parameter update. Hoffman et al. (2013) proposed Stochastic Variational Inference (SVI) to address this limitation by incorporating stochastic optimization principles.

The gradient of the ELBO can be expressed as an expectation with respect to the variational distribution:

$$\nabla_\phi \text{ELBO}(\phi) = \mathbb{E}_{q_\phi(w)}[\nabla_\phi \log q_\phi(w)(\log p(w, D) - \log q_\phi(w))] \quad (2.24)$$

This Equation (2.24) allows for a Monte Carlo approximation using samples drawn from the variational distribution:

$$\nabla_\phi \text{ELBO}(\phi) \approx \frac{1}{K} \sum_{k=1}^{K} \nabla_\phi \log q_\phi(w^{(k)})(\log p(w^{(k)}, D) - \log q_\phi(w^{(k)})) \quad (2.25)$$

where $w^{(k)} \sim q_\phi(w)$ for $k = 1, 2, \ldots, K$.

While these stochastic gradients provide computational efficiency, they typically exhibit high variance, potentially impeding convergence (Diederik P Kingma and Welling, 2022). To mitigate this issue, researchers have developed the reparameterization technique, which is particularly effective for continuous latent variables.

For variational distributions from location-scale families, such as Gaussian distributions $\mathcal{N}(w|\mu_\phi, \sigma_\phi^2)$, we can reparameterize the random variable w as:

$$w = \mu_\phi + \sigma_\phi \cdot \epsilon, \quad \epsilon \sim \mathcal{N}(0, 1) \quad (2.26)$$

This transformation separates the source of randomness (ϵ) from the parameters being optimized (ϕ), enabling more stable gradient computation:

$$\nabla_\phi \mathbb{E}_{q_\phi(w)}[f(w)] = \mathbb{E}_{\mathcal{N}(\epsilon|0,1)}[\nabla_\phi f(\mu_\phi + \sigma_\phi \cdot \epsilon)] \quad (2.27)$$

The reparameterization approach significantly reduces gradient variance and facilitates more efficient optimization (Titsias and Lázaro-Gredilla, 2014), yielding substantial improvements in convergence rates compared to standard score function estimators.

Recent advancements have extended these techniques to more complex variational families (Figurnov, Mohamed, and Mnih, 2019) and developed adaptive learning rate schedules (Ranganath, Gerrish, and Blei, 2013), further enhancing the practical applicability of SVI across diverse domains including deep generative modeling and Bayesian Neural Networks.

2.2 UNCERTAINTY

Building upon the BNN foundations discussed earlier, this section delves into uncertainty quantification—a key advantage of Bayesian approaches over deterministic neural networks.

2.2.1 Understanding Predictive Uncertainty

BNNs generate complete predictive distributions rather than mere point estimates, enabling robust uncertainty quantification that is crucial for applications requiring reliable decision-making under information constraints (Gal and Ghahramani, 2016; Kendall and Gal, 2017).

The predictive uncertainty can be systematically decomposed into two distinct components:

- Aleatoric uncertainty captures inherent data randomness and noise. This irreducible uncertainty persists regardless of model improvements and typically stems from measurement errors, stochastic data-generating processes, and input ambiguities (Hüllermeier and Waegeman, 2021).
- Epistemic uncertainty reflects model knowledge limitations regarding the underlying data-generating mechanism. Unlike aleatoric uncertainty, this component can be reduced through additional training data, more expressive model architectures, and improved parameter estimation techniques (Blundell et al., 2015).

This decomposition offers valuable insights for practical applications. Elevated epistemic uncertainty typically signals out-of-distribution inputs or unfamiliar data patterns, while high aleatoric uncertainty indicates inputs with intrinsic noise or ambiguity (Hüllermeier and Waegeman, 2021).

The following sections present methodologies for disentangling these uncertainty components in both regression and classification contexts. While classification benefits from established information-theoretic metrics that can directly decompose uncertainty from softmax outputs, regression requires specific model adjustments to effectively separate aleatoric and epistemic contributions to the overall predictive uncertainty.

2.2.2 Uncertainty in Regression

In regression tasks, BNN models typically output single values for each prediction. Although we can estimate the spread of the predictive distribution by sampling multiple outputs, distinguishing between epistemic and aleatoric components presents a significant challenge without additional model modifications.

To effectively decompose uncertainty in regression, researchers have developed an elegant approach that explicitly models aleatoric uncertainty through additional parameters. Specifically, the predictive distribution $p(y|x, \mathcal{D})$ is parameterized as a heteroscedastic Gaussian $\mathcal{N}(\mu(x), \sigma^2(x))$, where both the mean $\mu(x)$ and variance $\sigma^2(x)$ depend

on the input. This is implemented by modifying the BNN architecture to include two parallel output layers: one predicting the mean $\mu(x)$ and another predicting the variance $\sigma^2(x)$ (Kendall and Gal, 2017; Ovadia et al., 2019; Gustafsson, Danelljan, and Schön, 2020; Seitzer et al., 2022). The predicted variance $\sigma^2(x)$ directly captures the input-dependent aleatoric uncertainty, while the epistemic component can be derived by subtracting this aleatoric uncertainty from the total predictive uncertainty.

The computational procedure follows a systematic workflow: For an input x, the BNN performs a standard forward pass with dual outputs—predicted mean $\mu(x)$ and predicted variance $\sigma^2(x)$. These parameters define a Gaussian distribution from which the final output y is sampled. To capture the full predictive distribution, this process is repeated N times using different weight sets sampled from the posterior distribution. The total uncertainty is then computed as the variance across all N final outputs y_n, while the aleatoric uncertainty corresponds to the average of the N input-dependent variances $\sigma_n^2(x)$.

From a mathematical perspective, this uncertainty decomposition can be formalized as:

$$\mathrm{Var}[y|x,\mathcal{D}] = \underbrace{\mathrm{Var}_w[\mu_w(x)]}_{\text{epistemic}} + \underbrace{\mathbb{E}_w[\sigma_w^2(x)]}_{\text{aleatoric}} \tag{2.28}$$

In Equation (2.28), $\mu_w(x)$ represents the mean prediction with weights w, and $\sigma_w^2(x)$ represents the corresponding predicted variance. This equation elegantly separates uncertainty into two components:

- The epistemic term $\mathrm{Var}_w[\mu_w(x)]$ quantifies the variance of predicted means across different weight configurations, reflecting model uncertainty.
- The aleatoric term $\mathbb{E}_w[\sigma_w^2(x)]$ represents the expected variance across all weight samples, capturing inherent data noise.

For practical implementation, the variance is often parameterized as $\log \sigma^2(x)$ rather than directly as $\sigma^2(x)$. This logarithmic transformation ensures numerical stability and guarantees positive variance values. The uncertainty components are then estimated through multiple forward passes using different weight samples from the approximate posterior.

This principled uncertainty decomposition framework is particularly valuable in safety-critical applications, where understanding the distinct sources of predictive uncertainty enables more informed risk assessment and mitigation strategies.

2.2.3 Uncertainty in Classification

In classification tasks, Bayesian Neural Networks provide predictive distributions over class probabilities rather than deterministic class assignments. This enables a more nuanced understanding of model confidence and facilitates the decomposition of predictive uncertainty into its constituent components (Kendall and Gal, 2017; Gal and Ghahramani, 2016).

ENTROPY-BASED UNCERTAINTY QUANTIFICATION For classification problems, information-theoretic metrics offer a principled approach to quantifying uncertainty. The primary metric for measuring the total predictive uncertainty is the Shannon entropy (Shannon, 1948) of the approximate predictive distribution $p(y|x, \mathcal{D})$:

$$H(y|x, \mathcal{D}) = -\sum_{c=1}^{C} p(y = c|x, \mathcal{D}) \log p(y = c|x, \mathcal{D}) \tag{2.29}$$

where C represents the number of classes. The Shannon entropy possesses several desirable properties for uncertainty quantification (Wimmer et al., 2023): (1) it is non-negative, (2) reaches its maximum value for a uniform distribution, and (3) remains invariant to class permutations. Intuitively, entropy is minimal when predictions are concentrated on a single class and maximal when distributed uniformly across all classes.

UNCERTAINTY DECOMPOSITION A key advantage of entropy-based uncertainty metrics is their ability to be mathematically decomposed into aleatoric and epistemic components. Following the derivation by Depeweg et al. (2018), the Shannon entropy can be expressed as:

$$H(y|x, \mathcal{D}) = I[y, w|x, \mathcal{D}] + \mathbb{E}_{p(w|\mathcal{D})}[H[y|x, w]] \tag{2.30}$$

This decomposition separates total uncertainty into two distinct terms:

- The expected softmax entropy $\mathbb{E}_{p(w|\mathcal{D})}[H[y|x, w]]$ represents the aleatoric uncertainty component. It captures the average uncertainty in predictions across different weight configurations, reflecting the inherent noise or ambiguity in the input data.
- The mutual information $I[y, w|x, \mathcal{D}]$ quantifies the epistemic uncertainty. It measures the reduction in prediction uncertainty that would be achieved if we knew the true model parameters, thus representing uncertainty attributable to model knowledge limitations.

PRACTICAL IMPLEMENTATION In practice, these uncertainty components can be estimated using Monte Carlo sampling from the posterior

distribution. With N weight samples $w_n{}_{n=1}^{N}$ from the approximate posterior $q(w|\mathcal{D})$, the aleatoric uncertainty component is computed as:

$$\mathbb{E}_{p(w|\mathcal{D})}[H[y|x,w]] \approx -\frac{1}{N}\sum n = 1^N \sum_{c=1}^{C} p(y=c|x,w_n)\log p(y=c|x,w_n) \quad (2.31)$$

This expression calculates the average entropy of the per-sample softmax outputs. The total uncertainty is estimated by first averaging the softmax probabilities across samples to obtain the predictive distribution, then computing its entropy:

$$H(y|x,\mathcal{D}) \approx -\sum_{c=1}^{C}\left(\frac{1}{N}\sum_{n=1}^{N} p(y=c|x,w_n)\right)\log\left(\frac{1}{N}\sum_{n=1}^{N} p(y=c|x,w_n)\right) \quad (2.32)$$

Finally, the epistemic uncertainty component is derived by subtracting the aleatoric component from the total uncertainty:

$$I[y,w|x,\mathcal{D}] = H(y|x,\mathcal{D}) - \mathbb{E}_{p(w|\mathcal{D})}[H[y|x,w]] \quad (2.33)$$

2.3 PYRO: PROBABILISTIC PROGRAMMING FRAMEWORK

PYRO: DEEP UNIVERSAL PROBABILISTIC PROGRAMMING Probabilistic programming languages (PPLs) have emerged as powerful tools for constructing complex probabilistic models in artificial intelligence research. Pyro (Bingham et al., 2018) is a deep universal probabilistic programming framework built on Python and PyTorch (Paszke et al., 2019) that balances expressivity, scalability, flexibility, and minimality. As noted by Bingham et al. (2018), Pyro allows researchers to concisely describe models with data-dependent control flow and complex latent variable structures while leveraging GPU-accelerated tensor operations for efficient computation.

DESIGN PRINCIPLES AND ARCHITECTURE The core design of Pyro introduces two primary language primitives: `pyro.sample` for annotating calls to functions with internal randomness, and `pyro.param` for registering learnable parameters with inference algorithms. Pyro models can contain arbitrary Python code and interact with it in arbitrary ways, making it highly expressive for complex model specification. Importantly, Pyro is built on Poutine, a library based on algebraic effects and handlers (Pretnar, 2015) that implement individual control and book-keeping operations, creating a clean separation between model specifications and inference algorithms.

Figure 2.1 presents an implementation of a Bayesian Neural Network using Pyro's `PyroModule` interface, illustrating the object-oriented approach to model specification. This example demonstrates Pyro's flexibility in constructing hierarchical Bayesian models, where neural network parameters are treated as random variables with prior distributions, enabling uncertainty quantification in predictions.

```python
class BayesianNN(PyroModule):
    def __init__(self, in_features: int, out_features: int = 1,
                 hidden_layer_list: list = [100], act_function: str =
                     "relu",
                 device = "cuda") -> None:
        super().__init__()
        # Define priors for weight parameters
        self.std = torch.tensor(1.).float().to(device)
        self.mean = torch.tensor(0.).float().to(device)
        # Construct network with Bayesian layers
        self.hidden_layers = nn.ModuleList()
        for out in hidden_layer_list:
            self.hidden_layers.append(
                self._make_linear(in_features, out)
            )
            in_features=out

        # Output layers for mean and variance predictions
        self.output_mean = self._make_linear(hidden_layer_list[-1],
             out_features)
        self.output_var = self._make_linear(hidden_layer_list[-1],
            out_features)
        self.act=act_function

    def _make_linear(self, in_feature, out_feature):
        return PyroLinear(in_feature, out_feature, self.mean, self.
            std, bias=True)

    def forward(self, x, y=None):
        for layer in self.hidden_layers:
            x = self.act(layer(x))

        # Compute mean and variance of output distribution
        mean = self.output_mean(x)
        var = torch.exp(self.output_var(x)) # Ensuring positive
            variance

        with pyro.plate("data", subsample=x, size=self.data_size,
            dim=-2):
            out = pyro.sample("out", dist.Normal(mean, variance),
                obs=y)

        return out
# Using AutoGuide for automatic variational inference
model = BayesianNN(in_features=10, hidden_layer_list=[64, 32])
guide = AutoDiagonalNormal(model)
# Setup inference
optimizer = Adam({"lr": 0.01})
svi = SVI(model, guide, optimizer, loss=Trace_ELBO())
```

Figure 2.1: A Pyro implementation of a Bayesian Neural Network using the object-oriented `PyroModule` interface. The `BayesianNN` class defines the model architecture with probabilistic layers. Inference is performed using Pyro's `AutoDiagonalNormal` guide, which automatically constructs a variational distribution over all sample sites in the model.

INFERENCE MECHANISMS Pyro implements several generic probabilistic inference algorithms, including the No U-turn Sampler (NUTS) (Homan and Gelman, 2014), a variant of Hamiltonian Monte Carlo. However, its primary inference algorithm is gradient-based stochastic variational inference (SVI) (Ranganath, Gerrish, and Blei, 2013), which uses stochastic gradient descent to optimize Monte Carlo estimates of divergence measures between approximate and true posterior distributions. This approach enables Pyro to scale to complex, high-dimensional models through GPU-accelerated tensor math and to large datasets through stochastic gradient estimates computed over mini-batches of data.

COMPARISON WITH OTHER FRAMEWORKS Pyro distinguishes itself from other PPLs such as Stan (Carpenter et al., 2017), Church (N. Goodman et al., 2014), and Edward (Tran et al., 2017) through its unique combination of design principles. Unlike Stan, which focuses on a restricted class of models with automated inference, Pyro provides more expressive power with dynamic control flow. Compared to Church and webPPL (Ritchie, Horsfall, and N. D. Goodman, 2016), Pyro offers superior scalability through GPU acceleration and subsampling. Experimental evaluations have demonstrated Pyro's effectiveness in implementing state-of-the-art models, including variational autoencoders and deep Markov models, with moderate performance overhead compared to native PyTorch implementations that diminishes as tensor operation complexity increases.

2.4 FUNDAMENTALS OF QUANTIZATION

Quantization is a technique for reducing the computational complexity of deep learning models by converting their floating-point representations to lower-precision integer representations. This section introduces the fundamental concepts, methodologies, and applications of quantization in neural network deployment.

2.4.1 Principles of Quantization

Quantization is the process of mapping continuous values to a discrete set of values. In neural networks, quantization primarily targets weights and activations, converting high-precision floating-point numbers (typically 32-bit) to lower-precision representations (such as 8-bit, 4-bit, or even 2-bit integers) (Gholami et al., 2021). The basic quantization operation can be represented by the following formula:

$$Q(r) = \text{round}\left(\frac{r}{S}\right) + Z \tag{2.34}$$

where r represents the original floating-point value, S is the quantization scale factor, Z is the zero-point offset, and $Q(r)$ is the quantized integer value. The dequantization process can be represented as:

$$r \approx D(q) = S \cdot (q - Z) \tag{2.35}$$

where q is the quantized value and $D(q)$ is the dequantized floating-point approximation (Jacob et al., 2017).

The core objective of quantization is to reduce model size and computational complexity while preserving model performance as much as possible. Based on the precision range, quantization can be categorized into binary (1-bit), ternary (2-bit), and integer quantization (e.g., INT8, INT4) (Hubara et al., 2016).

2.4.2 Quantization Methods

Quantization methods can be broadly classified into Post-Training Quantization (PTQ) and Quantization-Aware Training (QAT) (Krishnamoorthi, 2018; Nagel et al., 2020).

POST-TRAINING QUANTIZATION (PTQ) Post-Training Quantization is applied directly to pre-trained models without requiring retraining or fine-tuning. The main steps of PTQ include: calibration, determination of quantization parameters (such as scale and zero-point), and application of quantization (Banner et al., 2019). The advantage of PTQ lies in its simple implementation and low computational cost, but performance degradation may be significant for low-bitwidth quantization (below 4-bit) or complex tasks.

For the quantization of weights w in each layer, a common approach is to determine their range $[w_{min}, w_{max}]$ and then calculate the quantization scale $S_w = \frac{w_{max} - w_{min}}{2^b - 1}$, where b is the quantization bitwidth (Nagel et al., 2020).

QUANTIZATION-AWARE TRAINING (QAT) Quantization-Aware Training considers quantization effects during the training phase by simulating quantization operations in the forward pass while using Straight-Through Estimator (STE) in the backward pass to handle the non-differentiable nature of quantization operations (Bengio, Léonard, and Courville, 2013). The general form of QAT is:

$$\begin{aligned} &\text{Forward: } \hat{w} = Q(w) \\ &\text{Backward: } \frac{\partial \mathcal{L}}{\partial w} = \frac{\partial \mathcal{L}}{\partial \hat{w}} \cdot \frac{\partial \hat{w}}{\partial w} \approx \frac{\partial \mathcal{L}}{\partial \hat{w}} \end{aligned} \tag{2.36}$$

where $\mathcal{L}$ is the loss function, w is the original weight, and $\hat{w}$ is the quantized weight. QAT can significantly mitigate accuracy loss caused

by quantization, especially for low-bitwidth quantization (Zhou et al., 2018). However, QAT requires retraining the model, which incurs higher computational costs.

2.5 BREVITAS: QUANTIZATION FRAMEWORK

Brevitas (Pappalardo, 2023) is an open-source quantization framework built on PyTorch that enables the training of quantization-aware neural networks. This section explores its architecture, core components, and integration with the PyTorch ecosystem. Brevitas follows a modular approach to quantization, implementing quantizers as PyTorch modules that can be seamlessly integrated into existing neural network architectures. The framework's design philosophy centers on providing flexibility while maintaining compatibility with hardware deployment targets (Pappalardo, 2023). Unlike post-training quantization tools, Brevitas focuses on quantization-aware training (QAT), allowing models to learn to compensate for quantization effects during the training process. The core abstraction in Brevitas is the quantizer, which transforms floating-point tensors into quantized representations while providing differentiable operations for backpropagation. These quantizers can be applied to weights, activations, and biases, with configurable bit-width, scaling, and clamping mechanisms (Micikevicius et al., 2022).

The Brevitas quantization call stack for a linear layer demonstrates how the framework intercepts the normal forward propagation process in PyTorch and replaces it with quantized operations. When a `QuantLinear` layer executes a forward pass, the following sequence occurs: (1) The model's forward call initiates a series of wrapped calls that eventually reach the `QuantLinear.forward` method. (2) The method sets a global flag `IS_INSIDE_QUANT_LAYER=True` to indicate the execution context. (3) Input tensors are unpacked and potentially quantized depending on configuration settings. (4) The weight injector intercepts the original weight tensor and applies quantization, replacing floating-point weights with their quantized counterparts. (5) The standard PyTorch `torch.linear()` function executes using the quantized weights and potentially quantized inputs. (6) The output is quantized if specified in the layer configuration. (7) Before returning, the global quantization flag is reset to `False`. This architecture allows Brevitas to maintain backward compatibility with PyTorch while introducing quantization behaviors, revealing how the framework integrates with PyTorch's module system to seamlessly inject quantization operations into standard neural network architectures.

2.5.1 Key Features and Components

Brevitas provides several key components that facilitate quantization-aware training:

QUANTIZERS Brevitas implements various quantizers, including binary, ternary, and fixed-point representations with configurable bit-width. Each quantizer implements a forward method that applies the quantization function to its input tensor. The general quantization process follows three steps (Pappalardo, 2023):

$$Q(x) = \text{round}\left(\text{clamp}\left(\frac{x}{s}, min\_val, max\_val\right)\right) \cdot s \tag{2.37}$$

where s represents the scale factor, and min_val and max_val define the quantization range.

SCALE FACTORS Scale factor determination is crucial for quantization performance. Brevitas offers various scaling strategies:

- **Statistical scaling**: Based on tensor statistics like min/max or percentiles
- **Parameter scaling**: Learnable scale factors that adjust during training
- **Power-of-two scaling**: Constraining scale factors to powers of two for hardware efficiency

QUANTIZED LAYERS Brevitas provides quantized versions of common PyTorch layers, including `QuantLinear`, `QuantConv1d/2d`, and `QuantReLU`. These layers integrate the quantization process into their forward pass, as described in the implementation sequence above.

2.5.2 Hardware-Aware Quantization

A distinctive feature of Brevitas is its focus on hardware-aware quantization. The framework provides export capabilities to hardware description formats like FINN (Blott et al., 2018) and ONNX, enabling direct deployment to FPGAs and other specialized hardware accelerators. These export functions translate the quantization parameters learned during training into hardware-compatible configurations.

The quantizers in Brevitas can be constrained to specific formats required by hardware platforms. For instance, certain accelerators may require power-of-two scale factors or specific bit alignments, which Brevitas can accommodate through appropriate configuration (Umuroglu et al., 2017).

3 RELATED WORK

This chapter will review the latest research progress in the fields of Bayesian Neural Networks and neural network quantization. It is worth noting that although both fields have rich research results, there is currently a lack of standardized research work on Bayesian Neural Network quantization. Therefore, we will review the progress of these two fields separately.

3.1 BAYESIAN NEURAL NETWORKS AND UNCERTAINTY QUANTIFICATION

In recent years, Bayesian Neural Networks (BNNs) have gained increasing attention due to their unique advantage in naturally capturing predictive uncertainty. This capability is particularly valuable as machine learning systems are increasingly deployed in everyday life and high-risk applications (Gawlikowski et al., 2022). Since the introduction of MC-Dropout (Gal and Ghahramani, 2016) and Deep Ensembles (Lakshminarayanan, Pritzel, and Blundell, 2017), which promise to provide uncertainty estimates with relatively low additional overhead, the application of BNNs in large-scale environments has become feasible.

BNNs have been applied across a wide range of domains. In the medical field, BNNs have been utilized for body sway detection (Silva et al., 2021), ischemic stroke lesion segmentation (M, Santhi, and Ramasamy, 2024), and diabetic retinopathy detection (Akram et al., 2025). Additionally, they have demonstrated practical value in hyperspectral image object detection (Klein et al., 2022). In computer vision tasks such as image classification (Joshaghani et al., 2023), depth regression, and semantic segmentation (Kendall and Gal, 2017; Gustafsson, Danelljan, and Schön, 2020), BNNs have also shown excellent performance. Research indicates that incorporating aleatoric uncertainty into BNNs can improve model accuracy beyond standard neural networks. Overall, many authors note that their proposed BNN solutions are more robust than standard neural network baselines and provide additional value through uncertainty or confidence estimates.

Given the variety of inference methods for BNNs, each with its unique assumptions and precision levels, it is necessary to compare and quantify the prediction and uncertainty quality of each method to select the appropriate approach for specific tasks. Many researchers focus on approximate Bayesian methods such as MC-Dropout and

ensembles, as these methods can be easily scaled to larger models and datasets (Gustafsson, Danelljan, and Schön, 2020). Whether MC-Dropout and ensembles are effective approximations of more precise BNN methods (such as MCMC) remains controversial in the literature. Wilson and Izmailov (2022) and Staber and Veiga (2023) argue that deep ensembles provide well-calibrated predictive distributions that appear superior to standard methods (such as VI). However, Yao et al. (2019) and Izmailov et al. (2021) found that approximate Bayesian methods sometimes struggle to capture the true posterior and do not necessarily cover all uncertainties when diversity is insufficient. Furthermore, Izmailov et al. (2021), in their extensive study of BNN posteriors, discovered that deep ensembles can provide good generalization and uncertainty estimates, but their predictive distributions differ from those generated by HMC.

When comparing the predictive quality of BNN inference methods such as HMC, stochastic MCMC, and deep ensembles in practical tasks, Li, Rudner, and Wilson (2024) found that HMC BNNs generally outperform other methods. However, rankings varied across different datasets, indicating that there is not always a single best choice. Additionally, model hyperparameters (such as network architecture or activation function selection) were shown to significantly impact uncertainty estimates in HMC BNNs. Foong et al. (2020) used HMC-inferred BNNs as the gold standard for predictive uncertainty, comparing them with mean-field SVI and MC-Dropout. A key finding of this study was that, even in simple examples where data clusters are surrounded by regions without data, mean-field SVI and MC-Dropout were overconfident between data clusters and failed to provide reasonable uncertainty estimates in these regions. The work of Wilson and Izmailov (2022) and Yao et al. (2019) further confirmed this behavior of SVI. Unfortunately, this issue cannot be resolved with deeper models. Since HMC models did not exhibit this behavior, Foong et al. (2020) hypothesized that this was due to the approximate nature of VI and MC-Dropout, which cannot be mitigated by adjusting priors, regularization, or optimizers.

In the more specific task of hyperspectral image object detection, Ries, Adams, and Zollweg (2023) compared the predictive performance of HMC and SVI. Consistent with previous research, SVI performed slightly worse than the gold standard HMC. Although SVI models required more tuning work to provide good results, Ries, Adams, and Zollweg (2023) considered it an efficient alternative to HMC. As many studies focus on classification tasks, Staber and Veiga (2023) aimed to compare the quality of BNN inference methods on four synthetic regression tasks. In their work, stochastic gradient MCMC, MC-Dropout, and deep ensembles were compared against each other, with HMC considered the gold standard. They demonstrated that stochastic gradient MCMC and deep ensembles could provide the best

predictive performance and uncertainty estimates, while MC-Dropout results were comparatively poor.

Ovadia et al. (2019) evaluated various DNNs and BNNs for uncertainty estimation under dataset shift, where ideally, models should be able to detect dataset shift through high uncertainty estimates, indicating when difficulties are encountered. While calibrated DNNs and some BNNs failed to provide reasonable uncertainty estimates, SVI, deep ensembles, and MC-Dropout performed better in this regard. Mitros and Namee (2019) also found a lack of calibration in standard NNs for popular image classification tasks, while selected BNN alternatives were better calibrated and more robust. Additionally, Ovadia et al. (2019) found that SVI could produce promising uncertainty estimates corresponding to predictive quality but was difficult to use due to complexity. Consequently, deep ensembles consistently provided the best predictions and uncertainty estimates on larger datasets and tasks such as ImageNet and news category prediction.

Unlike the aforementioned work that only assessed total uncertainty estimates, Valdenegro-Toro and Saromo (2022) compared the ability of approximate BNN inference methods to decompose uncertainty into aleatoric and epistemic components. Their expectation was that, since aleatoric uncertainty depends only on the input, it should be similar across all methods and independent of epistemic uncertainty. However, MC-Dropout, ensembles, and particularly Flipout (a VI method that adds weight perturbations to mini-batches) entangled uncertainties to some degree, suggesting that the distinction of uncertainties is not as clear in most models. Similarly, deep ensembles provided the best results among approximate Bayesian methods.

3.2 NEURAL NETWORK QUANTIZATION

Neural network quantization has emerged as a critical technique for reducing model size, memory footprint, and inference latency—key concerns in deploying deep learning models on edge devices and embedded systems. Unlike other model compression techniques such as pruning and knowledge distillation, quantization reduces the precision of weights and activations, typically from 32-bit floating-point to 8-bit integer or even lower precision representations, without significantly degrading model performance.

A significant body of work has focused on developing quantization techniques that minimize the accuracy loss associated with low-precision computation. Early methods such as Post-Training Quantization (PTQ) (Jacob et al., 2017) perform quantization without retraining and are attractive due to their simplicity and low computational overhead. However, these methods may not generalize well to all architectures or datasets. To address this, Quantization-Aware Training (QAT)

methods (Krishnamoorthi, 2018) incorporate quantization effects into the forward pass during training while keeping the backward pass in full precision. This approach helps the model adapt to quantization noise and generally yields higher accuracy compared to PTQ.

In addition to precision reduction, researchers have explored mixed-precision quantization (Wang et al., 2019), where different layers or operations within a network are assigned different bit-widths according to their sensitivity. Hardware-aware neural architecture search (NAS) methods are often employed in these works to find the optimal trade-off between efficiency and accuracy, tailored to specific deployment targets such as CPUs, GPUs, or custom accelerators.

More recent work has also explored quantization for large-scale models, including transformers and vision models. For example, GPTQ (Frantar et al., 2023) introduces a post-training quantization approach tailored for transformer models, achieving low-bit quantization (down to 4-bit) while preserving performance. Similarly, LLM.int8() (Dettmers et al., 2022) demonstrates that with careful calibration and quantization strategies, even large language models (LLMs) can be quantized effectively without retraining.

A key challenge in quantization remains the trade-off between computational savings and predictive performance. Techniques such as per-channel quantization, outlier-aware quantization (Zadeh et al., 2020), and learning-based quantizer design aim to address this issue by reducing quantization error in critical components of the network. Moreover, some works emphasize the importance of robust evaluation protocols for quantized models, especially under distribution shift and adversarial conditions (Lin, Gan, and Han, 2019), as quantization may increase model sensitivity to input perturbations.

Despite substantial progress in the quantization of standard deterministic neural networks, there is currently a lack of systematic research addressing the quantization of Bayesian neural networks. The stochastic nature of BNNs and their reliance on weight distributions rather than point estimates introduce unique challenges not present in conventional models. For instance, the interaction between quantization noise and uncertainty estimates remains an open problem. Additionally, the need to preserve uncertainty calibration while reducing numerical precision is a largely unexplored area. Consequently, this work aims to bridge this gap by investigating the intersection of quantization techniques and Bayesian inference methods.

4 DATASETS AND EVALUATION METRICS

4.1 REGRESSION TASKS

4.1.1 Noisy Sine Wave

The noisy sine wave regression task serves as an informative benchmark for evaluating model predictions and uncertainty estimation capabilities. This one-dimensional problem facilitates straightforward visualization and interpretation of both predictive accuracy and uncertainty calibration. Such noisy sine wave variants have been widely adopted in the literature as illustrative examples for uncertainty quantification, appearing in seminal works by Gustafsson, Danelljan, and Schön (2020), Depeweg et al. (2018), and Seitzer et al. (2022).

The synthetic nature of this dataset allows for precise control over uncertainty characteristics. In our implementation, we define a sine function over the domain [-22.4, 22.4], partitioning this range into nine distinct regimes with varying uncertainty properties (see Figure 4.1). Each regime is deliberately designed to exhibit specific epistemic and aleatoric uncertainty patterns, enabling systematic evaluation of model behavior under different uncertainty conditions.

Epistemic uncertainty, which reflects model knowledge limitations, is manipulated through intentional data distribution. Regions devoid of training samples naturally induce higher epistemic uncertainty, as models must extrapolate without evidence. While establishing ground truth for epistemic uncertainty remains an open challenge in Bayesian Neural Networks research (Hüllermeier and Waegeman, 2021), we can reasonably expect uncertainty predictions to correlate inversely with data density.

Aleatoric uncertainty, which captures inherent data randomness, is introduced by incorporating Gaussian noise with regime-specific standard deviations into the target values. This controlled noise injection provides an explicit ground truth for aleatoric uncertainty—namely, the standard deviation of the added noise. Consequently, regimes with higher noise levels should elicit proportionally higher aleatoric uncertainty predictions from well-calibrated models.

4.1.2 Quality Metrics

Our evaluation framework addresses two critical aspects of model performance: prediction accuracy and uncertainty calibration. For

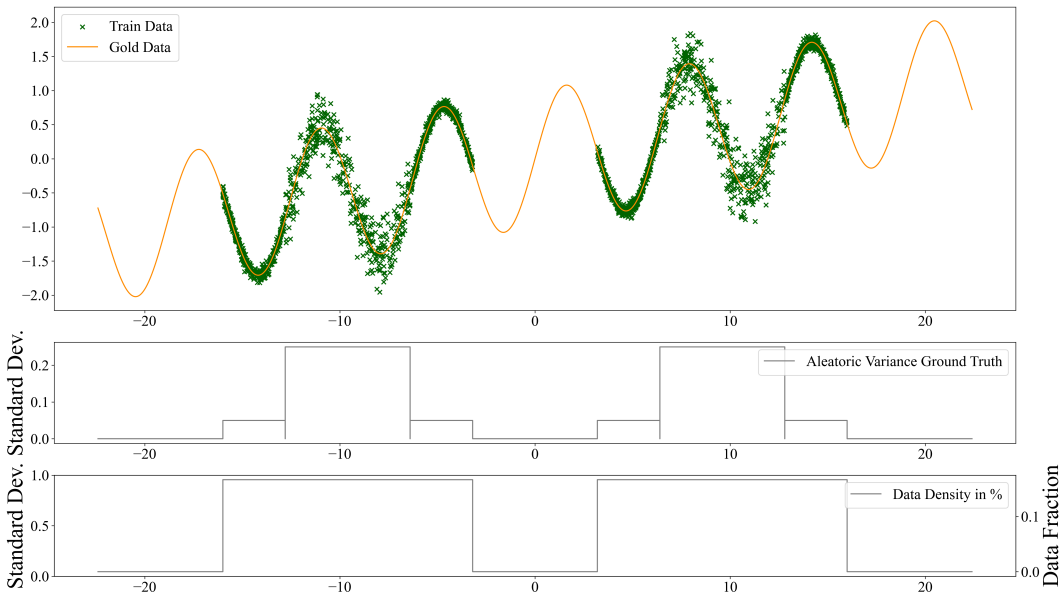


Figure 4.1: Noisy sine example with 2000 training data points (green crosses) and the underlying sine function $f(x) = \sin(x) + 0.05 \cdot x$ (gold line). The domain $[-22.4, 22.4]$ is partitioned into nine distinct regimes. The middle plot shows the ground truth aleatoric uncertainty (standard deviation) across different regimes. The bottom plot displays data density, indicating regions with and without training samples to induce varying levels of epistemic uncertainty.

assessing prediction accuracy in this regression context, we employ the mean squared error (MSE) metric, which quantifies the deviation between predicted and actual values.

For uncertainty evaluation, we analyze both aleatoric and epistemic components. The accuracy of aleatoric uncertainty estimates is assessed by comparing predicted variance against the known ground truth variance of the injected noise. Additionally, we employ the negative log-likelihood (NLL) for heteroscedastic Gaussian distributions:

$$\text{NLL}_{\text{Gaussian}} = \frac{1}{2}\left(\frac{(y-\mu)^2}{\sigma^2} + \log \sigma^2\right) \tag{4.1}$$

where y represents the target value, while μ and σ^2 denote the model's predicted mean and variance, respectively. This comprehensive metric evaluates the likelihood of observed data under the model's predictive distribution, thereby integrating both predictive accuracy and uncertainty calibration into a single measure.

While ground truth for epistemic uncertainty remains elusive, we qualitatively assess these predictions by examining their correlation with data density. Through visual inspection of uncertainty decomposition plots alongside data distribution visualizations, we can evaluate

whether models appropriately increase epistemic uncertainty estimates in data-sparse regions.

4.2 CLASSIFICATION TASKS

4.2.1 Two Moons

The Two Moons dataset is a canonical binary classification problem with interleaving half-moon shapes, providing both visual intuitiveness and sufficient complexity to challenge linear classifiers.

In our implementation, we partition the input space into three regions for uncertainty quantification evaluation: in-domain regions (with dense training samples), near-OOD regions (boundary areas with diminishing data density), and OOD regions (spaces far removed from training data).

To delineate these regions, we employ the Alpha Shape algorithm (Edelsbrunner, Kirkpatrick, and Seidel, 1983), which constructs generalized boundaries around point sets. This method outperforms traditional convex hull approaches by accommodating the non-convex, crescent-shaped clusters in the Two Moons dataset. We further refine this classification with a k-nearest neighbor distance metric to address potential sparsity within the in-domain regions.

Our dataset preparation begins with generating clean moon-shaped clusters, adding controlled Gaussian noise to simulate aleatoric uncertainty, and extending the test region beyond the training distribution to capture OOD behaviors. As shown in Figure (4.2), test points are categorized based on their relationship to Alpha Shape boundaries and proximity to training points. Importantly, while we identify near-OOD regions, they are deliberately excluded from final quantitative evaluation. This decision reflects that: (1) the boundary between in-distribution and OOD data exists as a continuum rather than a discrete transition; (2) near-OOD delineation depends on parameter choices; and (3) including this transitional region could obscure the distinction between handling in-distribution aleatoric uncertainty versus recognizing novel data patterns. By evaluating only in-distribution and clear OOD regions, we obtain more definitive performance measures.

4.2.2 DirtyMNIST

Building upon the classic MNIST dataset of handwritten digits (Lecun et al., 1998), DirtyMNIST introduces deliberate ambiguity and noise to simulate real-world data challenges. Developed by Mukhoti et al. (2021), this dataset specifically targets the evaluation of uncertainty decomposition capabilities in classification models.

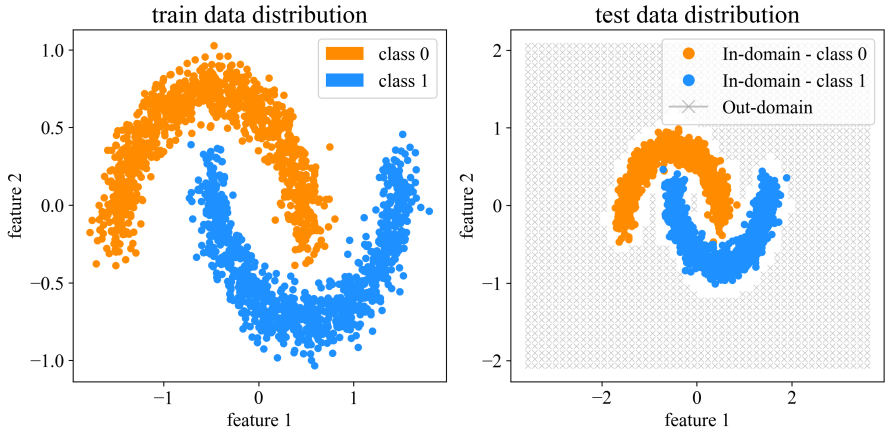


Figure 4.2: Two Moon dataset consists of 2000 training samples (shown as orange and blue dots), forming the classic double-moon shape distribution. The left plot displays the training data distribution, where two classes (class 0 and class 1) form interleaved crescent shapes. The feature space domain [-3, 3] × [-2, 2] is partitioned into in-domain and out-domain regions. The right plot shows the test data distribution, clearly identifying in-domain samples and out-domain regions (represented by gray grid). This design makes the Two Moons dataset particularly suitable for studying the performance of classification algorithms when handling non-linear decision boundaries and out-of-domain generalization capabilities.

The fundamental innovation of DirtyMNIST lies in its integration of ambiguous samples—digits that exhibit characteristics of multiple classes simultaneously. These samples are systematically generated using a Variational Autoencoder (VAE) framework (Diederik P Kingma and Welling, 2022), where the latent representations of distinct digit classes are linearly interpolated before decoding. This process produces images that genuinely embody features of multiple digit classes, creating inherent classification ambiguity.

To establish ground truth labels for these ambiguous samples, Mukhoti et al. (2021) employ an ensemble of pre-trained MNIST classifiers. The labels with the highest predicted probabilities across the ensemble are assigned as targets, reflecting the genuine classification difficulty these samples present. Importantly, samples producing excessive uncertainty in the ensemble (termed "junk images") are filtered out, ensuring that the ambiguity remains structured rather than arbitrary.

The final DirtyMNIST dataset achieves a balanced composition, containing equal proportions of standard MNIST samples and the generated ambiguous samples. This design enables systematic evaluation of aleatoric uncertainty handling, as models must appropriately express uncertainty when confronted with genuinely ambiguous class assignments. In total, the dataset comprises 120,000 training samples

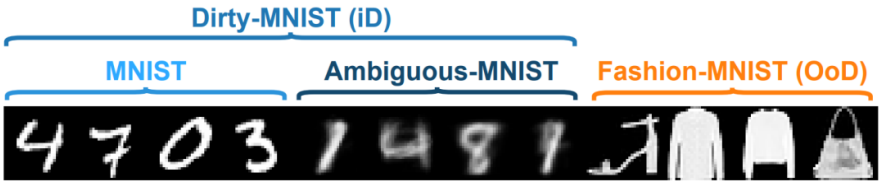


Figure 4.3: DirtyMNIST: dataset and uncertainty overview from Mukhoti, J., Kirsch, A., van Amersfoort, J., Torr, P. H. S., and Gal, Y. (2021): Deterministic neural networks with appropriate inductive biases capture epistemic and aleatoric uncertainty.

and 20,000 test samples, providing sufficient scale for robust model training and evaluation.

To complement the aleatoric uncertainty assessment, Mukhoti et al. (2021) propose using Fashion-MNIST (Xiao, Rasul, and Vollgraf, 2017) as an out-of-distribution test set. While Fashion-MNIST maintains the same image dimensions and format as MNIST, it depicts entirely different object categories (clothing items rather than digits). This categorical shift presents an ideal test case for epistemic uncertainty, as models trained on digit recognition should express high knowledge uncertainty when confronted with previously unseen object categories.

The DirtyMNIST framework thus provides a comprehensive testbed for uncertainty quantification in classification, with controlled sources of both aleatoric uncertainty (through ambiguous samples) and epistemic uncertainty (through OOD evaluation). This enables systematic assessment of a model's ability to distinguish between intrinsic data ambiguity and knowledge limitations—a critical capability for reliable deployment in real-world scenarios.

4.2.3 Quality Metrics

For classification tasks, we employ a comprehensive evaluation framework that addresses both predictive accuracy and uncertainty calibration. The primary predictive performance metric is classification accuracy, computed as the proportion of correctly classified samples in the test set:

$$\text{Accuracy} = \frac{1}{N}\sum_{i=1}^{N} 1[\hat{y}_i = y_i] \tag{4.2}$$

where $\hat{y}_i$ represents the predicted class for sample i, y_i denotes the true class, and $1[\cdot]$ is the indicator function that equals 1 when the condition is satisfied and 0 otherwise.

To assess model confidence and its alignment with predictive accuracy, we implement several specialized uncertainty metrics. For Bayesian models that approximate posterior weight distributions, we

measure predictive unanimity—the proportion of sampled weight configurations that agree on the final predicted class:

$$\text{Unanimity}(x) = \frac{1}{N} \sum_{n=1}^{N} 1[\hat{y}_n = c] \tag{4.3}$$

where $\hat{y}_n$ represents the predicted class using weight sample n, c denotes the final averaged class prediction, and N is the total number of weight samples. This metric provides insight into the consistency of predictions across the approximated posterior, with higher unanimity indicating greater confidence.

For uncertainty decomposition, we use the Shannon entropy, softmax entropy and mutual information as described in Chapter 2.2.

5 INPUT REPRESENTATION FOR LOW DIMENTIONAL CONTINUOUS VALUE DATA

In exploring the capabilities of the BrainScaleS-2 system, we identified a key challenge: effectively implementing quantized neural network models with limited bit precision. A critical aspect of this challenge is developing suitable input representation methods that can convert continuous-valued data into formats compatible with hardware constraints while preserving essential information. For the BrainScaleS-2 system specifically, this requires representing dataset inputs as 5-bit unsigned values, presenting unique encoding challenges (Pehle et al., 2022). To illustrate the challenges of quantization constraints, we conducted experiments applying BrainScaleS-2 quantization requirements directly to a standard MLP model. Figure (5.1) demonstrates the severe performance degradation that occurs when implementing 5-bit unsigned quantization on continuous-valued data. As evident in the visualization, the quantized model produces step-like predictions that fail to capture the smooth nature of the sine wave function. This performance limitation is particularly pronounced around zero-crossing regions, where the unsigned quantization constraint forces the model to make abrupt transitions. The significant gap between the ground truth data (yellow line) and model predictions (blue line) highlights the fundamental inadequacy of naive quantization approaches, despite the model having sufficient training points (green markers) to theoretically learn the underlying pattern. This visualization clearly demonstrates why more sophisticated input representation methods are essential for effective neuromorphic computing under strict bit-width constraints.

A straightforward approach might be to simply add a bias to ensure non-negativity of inputs. However, the low-dimensional nature of many datasets introduces significant limitations with this method. When using 5-bit representation (allowing only 32 distinct values), a single-feature input space like our Noisy Sine Wave dataset (4.1.1) becomes severely discretized, potentially losing critical information in the quantization process. Even for two-dimensional inputs such as the Two Moons dataset (4.2.1), the effective input space is limited to just 1,024 possible value combinations. This discretization disrupts data continuity and substantially reduces information content, potentially degrading model performance.

These constraints motivate our exploration of more sophisticated input representation methods that can maximize information preser-

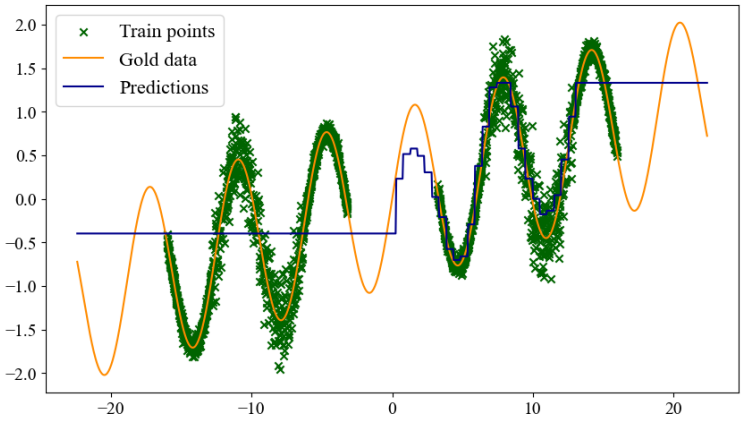


Figure 5.1: Performance of a MLP Model with BrainScale-2 Quantization on Noisy Sine Wave Regression: The direct application of 5-bit unsigned quantization severely constrains the model's ability to capture the continuous nature of the sine wave function, resulting in step-like predictions (blue line) that fail to accurately follow the ground truth data (yellow line), especially noticeable around the zero crossing regions.

vation within the restricted bit-width requirements of neuromorphic hardware. An optimal input representation method for the BrainScaleS-2 system should satisfy several key criteria:

- **Simplicity and efficiency:** The representation should not require extensive computational overhead, as this would counteract the energy efficiency advantages of neuromorphic computing. The transformation process should be straightforward to implement and execute rapidly.
- **Non-negative unique mapping:** The representation must map the original input domain to a non-negative domain suitable for the hardware constraints, ideally maintaining a one-to-one correspondence between original and transformed values to prevent information loss through mapping collisions.
- **Precision preservation:** The transformation should minimize quantization error while maintaining the essential statistical properties and relationships within the original data. This includes preserving relative distances between data points and the overall distribution characteristics.

To address these requirements, this chapter explores various input representation strategies for low-dimensional continuous data in the context of neuromorphic computing. We investigate several innovative approaches including:

- **Input representation via Dense layer:** Utilizing neural network layers to learn optimal transformations that preserve essential information while conforming to bit-width constraints.
- **Input representation via Binary Sequence:** Encoding continuous values through strategic binary representations that maximize information retention in limited bit spaces.
- **Input representation via Sectional Thermometer Encoding:** Adapting thermometer coding principles with sectional strategies to enhance representation capabilities for various data distributions.

By using the Noisy Sine Wave dataset (see Figure 4.1.1) as our primary experimental benchmark, we can visually assess and quantitatively evaluate different representation methods. Additionally, to isolate the effects of input representation from uncertainty estimation complexities, we conduct our experiments using a standard Multilayer Perceptron (MLP) architecture rather than Bayesian models. Due to the sensitivity of Bayesian Neural Networks to activation functions (Tempczyk et al., 2022; Fakhfakh and Chaari, 2023), we consistently employ Sigmoid activation functions across all experiments with the Noisy Sine Wave dataset to ensure optimal performance in subsequent Bayesian implementations. This standardization guarantees more stable and comparable results across different input representation methods while maintaining consistency in the neural network's behavior.

Through comparative analysis, we aim to identify representation methods that strike an optimal balance between hardware compatibility and information preservation, thereby enabling more effective implementation of machine learning models on systems with strict bit-width constraints.

5.1 INPUT REPRESENTATION VIA DENSE LAYER

Neural networks with dense layers have demonstrated remarkable capabilities in learning complex transformations across various domains. In the context of input representation for neuromorphic hardware with limited bit precision, dense layers offer a promising approach. The fundamental principle behind using dense layers for input representation is to leverage their ability to map an input space to a different feature space through learnable parameters.

For continuous-valued data with limited dimensionality, dense layers can be trained to transform the original input values into a representation that maximizes information retention while satisfying hardware constraints. Specifically, for the BrainScaleS-2 system's 5-bit precision limitation, dense layers can learn to map continuous inputs

to a discrete 32-value space (5-bit unsigned integers ranging from 0 to 31) in a way that preserves the most discriminative information for downstream tasks.

The theoretical advantage of this approach lies in its adaptability and task-specificity. Unlike fixed encoding schemes, dense layers can learn transformations that are optimal for a particular dataset and task, potentially capturing the most relevant patterns and features while discarding noise. Furthermore, dense layers can automatically discover non-linear transformations that might be difficult to design manually, making them particularly suitable for complex data distributions.

5.1.1 Implementation Approaches

We explore two distinct approaches for implementing input representation via dense layers, each with its own advantages and limitations: the End-to-End Training Method and the Autoencoder Training Method.

END-TO-END TRAINING METHOD The End-to-End Training Method integrates input representation directly into the task-specific neural network architecture. This approach involves prepending specialized dense layers to the main MLP network, creating a unified model where both representation and task-specific components are trained simultaneously through gradient-based optimization. As illustrated in Figure (5.2), the dense layer processes the original continuous inputs at full precision before passing its outputs to the downstream MLP, which may operate at either full precision or with quantization constraints to simulate neuromorphic hardware limitations. The joint

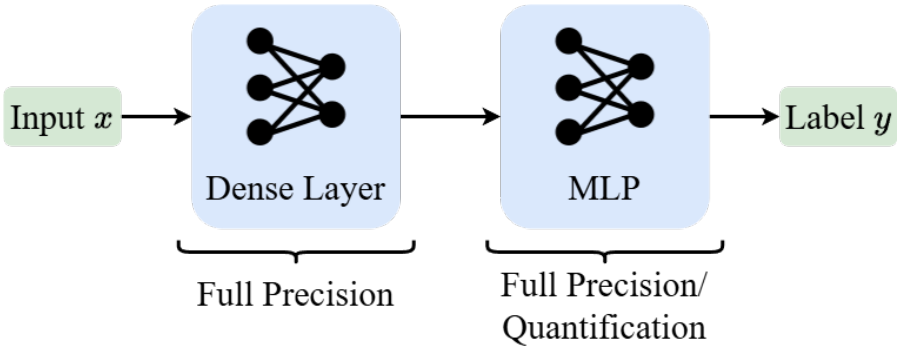


Figure 5.2: End-to-end training architecture with input representation via dense layer. The dense layer operates at full precision while the downstream MLP can operate at either full precision or with quantization constraints.

optimization process offers distinct advantages for neuromorphic computing applications. By directly minimizing the task loss function, the representation layers learn to capture precisely those input features most critical for the specific prediction task, potentially improving

overall performance within the hardware constraints. The end-to-end approach streamlines the training pipeline, requiring only a single optimization process rather than separate training phases. Additionally, joint optimization enables the discovery of complex, synergistic interactions between the representation and task-specific components that might remain undiscovered in isolated training regimes.

Despite these benefits, the end-to-end method presents notable challenges. The resulting representations tend to be highly specialized to the particular task and training distribution, potentially limiting their transferability to other related problems. The integrated optimization also complicates the analysis of representation quality as an independent factor, making it difficult to isolate the effects of the representation from those of the task-specific network components when evaluating overall system performance.

AUTOENCODER TRAINING METHOD The Autoencoder Training Method employs a distinct two-phase approach to input representation. As illustrated in Figure (5.3), this method decouples the representation learning process from the task-specific training, allowing for more generalized feature extraction.

In the initial pre-training phase, we construct a complete autoencoder architecture consisting of two components: an encoder network implemented as a dense MLP that transforms the original continuous input into an intermediate representation x_r, and a decoder network that attempts to reconstruct the original input from this intermediate representation. This autoencoder system is trained end-to-end to minimize reconstruction error, effectively learning to compress and decompress the input data while preserving essential information. The training objective encourages the intermediate representation to capture the most salient features of the input distribution rather than features specific to any downstream task.

Following successful convergence of the autoencoder training, we extract only the encoder component and discard the decoder. This pre-trained encoder then functions as a fixed feature transformation module in the second phase, where it processes raw inputs before they enter a separately trained task-specific MLP. The task network is optimized using these transformed representations to perform the target prediction task without modifying the pre-trained encoder weights.

This methodology offers several compelling advantages for neuromorphic computing applications. The representation learning focuses on preserving general input characteristics rather than task-specific features, potentially resulting in more robust and transferable encodings. The clear separation between representation learning and task learning enables more transparent analysis of how different input encoding strategies affect downstream performance. Furthermore, the

pre-trained encoder can serve as a universal front-end for multiple task-specific networks, promoting modular system design and potentially reducing overall training requirements for multiple related tasks.

Despite these benefits, the autoencoder approach introduces certain trade-offs. The two separate training phases increase computational overhead compared to the end-to-end method. Additionally, since the representation is optimized for reconstruction fidelity rather than task performance, there may be scenarios where task-critical features receive insufficient emphasis in the learned representation, potentially limiting performance on specific tasks compared to representations directly optimized for those tasks.

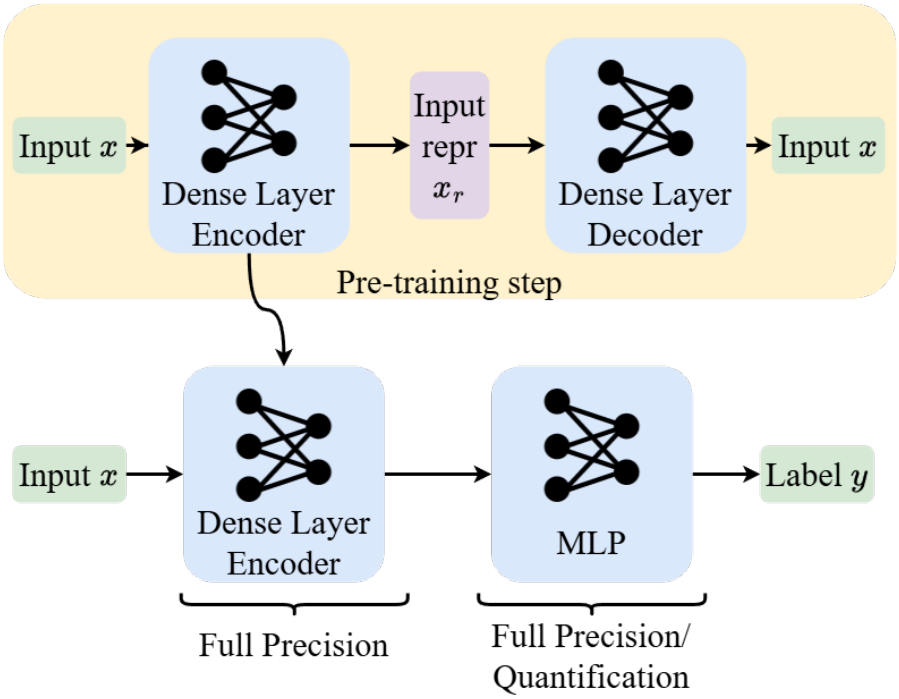


Figure 5.3: Autoencoder training method for input representation. Top: Pre-training phase where an encoder-decoder architecture learns to reconstruct the input through an intermediate representation x_r. Bottom: Application phase where only the pre-trained encoder is used to transform inputs for the task-specific MLP.

5.1.2 Experiments and Results

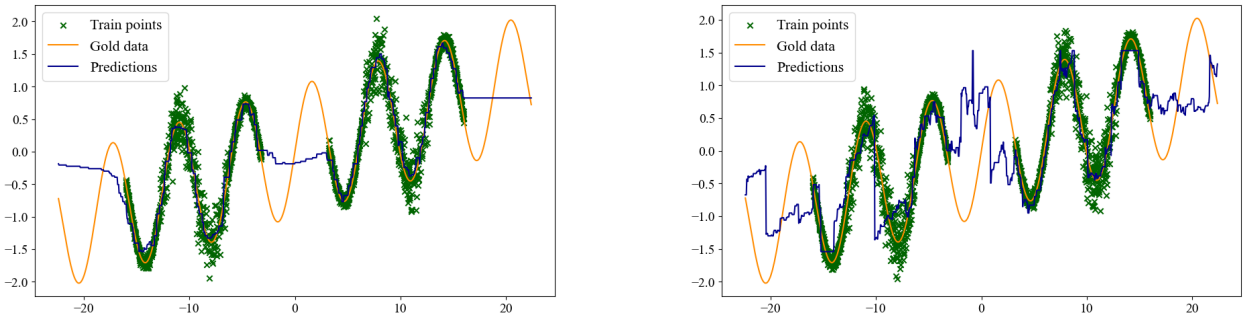
To evaluate the performance of this method on a unified scale, we conducted experiments with quantized versions on the Noisy Sine Wave dataset.

EXPERIMENT SETTING We conducted a comprehensive evaluation of both input representation methods using consistent experimental configurations:

- **Dataset:** 2,000 training samples, 1,000 testing samples, batch size 64.
- **Model Architecture:** Quantizable MLP with 50 neurons in the hidden layer, using sigmoid activation functions throughout to ensure compatibility with subsequent Bayesian implementations.
- **Quantization Parameters:** 5-bit unsigned representation for input quantization, 8-bit signed representation for output quantization, and 7-bit signed representation for weight quantization.

For the specific training methodologies, we implemented distinct protocols:

- **End-to-End Method:** Single training phase of 300 epochs with learning rate 0.002, jointly optimizing both the representation layers and task-specific layers.
- **Autoencoder Method:** Two-phase training approach consisting of:
  - Autoencoder pre-training: 300 epochs with learning rate 0.002, optimizing for input reconstruction
  - Task-specific training: 300 epochs with the same learning rate, training only the MLP while keeping the pre-trained encoder fixed



(a) End-to-End **(b)** Autoencoder

Figure 5.4: Comparative Performance of End-to-End and Autoencoder Approaches in Quantized Neural Networks for Noisy Sine Wave Regression: The End-to-End model (a) shows significantly better performance compared to the Autoencoder approach (b), although still exhibiting notable non-smoothness across the prediction domain.

EXPERIMENT RESULTS AND ANALYSIS Figure (5.4) presents a visual comparison between the End-to-End and Autoencoder approaches for input representation in quantized neural networks applied to the noisy

sine wave regression task. The results reveal substantial differences in performance characteristics between these two methodologies.

The End-to-End approach (Figure 5.4a) demonstrates significantly superior predictive accuracy compared to the Autoencoder method. The prediction curve closely follows the ground truth data across most of the input domain, capturing the underlying sinusoidal pattern with reasonable fidelity. However, despite this overall strong performance, the End-to-End model exhibits notable non-smoothness in its predictions, particularly visible at the boundaries of the input domain (near $x = -20$ and $x = 20$). This non-smoothness suggests that while the joint optimization effectively maps inputs to correct outputs for most data points, it may struggle with generalization at domain extremities where training data might be sparser.

In contrast, the Autoencoder approach (Figure 5.4b) shows markedly inferior performance. The prediction curve exhibits significant oscillatory behavior that deviates substantially from the ground truth, particularly in regions where the End-to-End model performs well. These oscillations suggest that the representation learned by the autoencoder fails to preserve critical features necessary for accurate regression. The pronounced instability in predictions, especially in the left portion of the input domain ($x < -10$), indicates that the two-phase training methodology may be fundamentally limited in its ability to capture task-relevant features when operating under severe quantization constraints.

The comparative performance disparity between these approaches highlights a critical insight for neuromorphic computing applications: When operating under strict bit-precision limitations (5-bit input quantization in this case), the End-to-End method's ability to jointly optimize representation and task-specific components provides a substantial advantage over the decoupled learning strategy of the Autoencoder method. This suggests that for hardware-constrained neuromorphic systems, the synergistic interaction between representation learning and task optimization may be essential for preserving performance within severe quantization constraints.

These findings align with theoretical expectations, as the End-to-End method directly optimizes for task performance rather than reconstruction fidelity, allowing it to prioritize task-critical features in its learned representation. The results demonstrate that while the Autoencoder approach offers theoretical advantages in transferability and modularity, these benefits may not translate to practical performance improvements in the highly constrained setting of neuromorphic hardware implementation for regression tasks.

5.2 INPUT REPRESENTATION VIA BINARY SEQUENCE

Binary sequence representation offers an alternative approach to encoding continuous-valued inputs for neuromorphic hardware with strict bit-width limitations. This section explores two primary binary encoding strategies: IEEE754 floating-point representation and Fixed-Point Number representation. Both methods aim to effectively convert continuous data into 5-bit unsigned integers compatible with the BrainScaleS-2 system while maximizing information preservation.

5.2.1 IEEE754 Floating-Point Representation

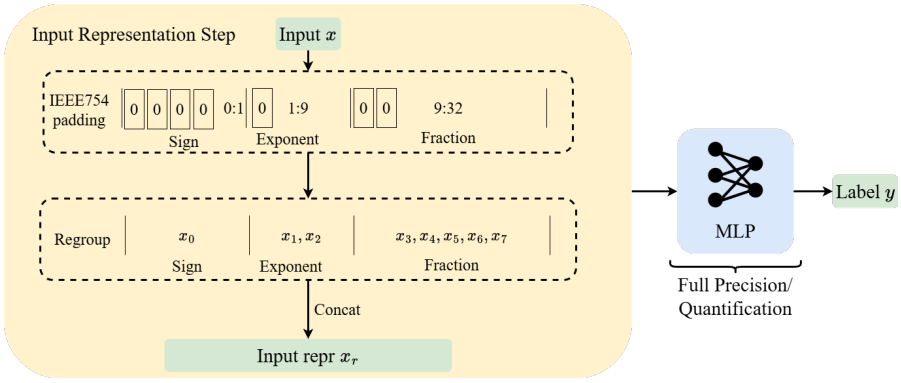


Figure 5.5: IEEE754 Floating-Point representation for neural network inputs. The diagram illustrates the process of converting raw input values into a structured representation through IEEE754 padding, regrouping of sign, exponent, and fraction components, and concatenation to produce the final input representation for MLP processing.

The IEEE754 standard provides a precise method for encoding real numbers in binary format. For a single-precision 32-bit floating-point representation, a real number x is encoded as:

$$x = (-1)^s \times 2^{(e-127)} \times (1 + f) \tag{5.1}$$

where s is the sign bit (0 for positive, 1 for negative), e is the 8-bit exponent (ranging from 0 to 255), and f is the 23-bit fraction or mantissa. The bias of 127 allows for representing both positive and negative exponents.

For neuromorphic hardware implementation, we transform this representation into a structured feature vector. Let x_r represent our input representation:

$$x_r = [x_0, x_1, x_2, x_3, x_4, x_5, x_6, x_7] \tag{5.2}$$

where each x_i is a 5-bit unsigned integer derived from the binary representation. Here, x_0 corresponds to the sign bit, x_1 and x_2 encode the

exponent, and x_3 through x_7 represent the fraction components. This decomposition preserves the mathematical properties of the floating-point representation while making it compatible with our hardware constraints. The transformation process is illustrated in Figure (5.5), showing how the continuous input x is converted through IEEE754 padding and regrouping steps before final concatenation.

5.2.2 Fixed-Point Number Representation

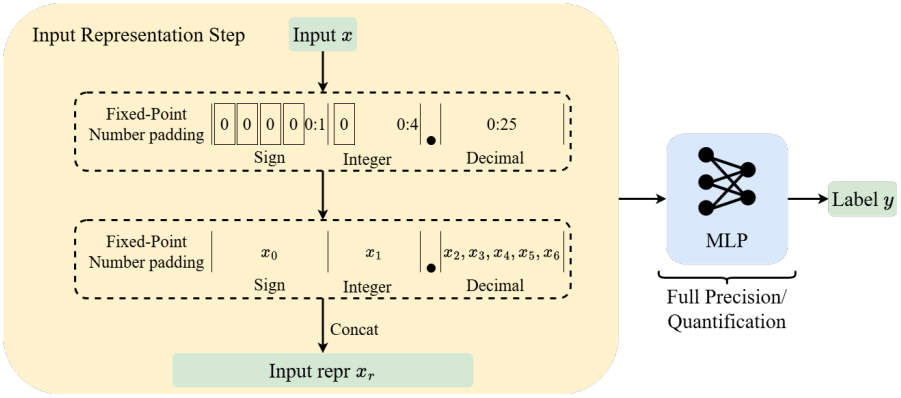


Figure 5.6: Fixed-point number representation for neural network inputs. The diagram illustrates the process of transforming input values using fixed-point number format with sign, integer, and decimal components. The representation includes padding, regrouping into structured components, and concatenation to form the final input representation for MLP processing.

Fixed-point representation allocates a fixed number of bits for the integer and fractional parts of a real number. For a number x with m integer bits and n fractional bits, the fixed-point representation is defined as:

$$x = \text{sign} \times \left(\sum_{i=0}^{m-1} b_i \times 2^i + \sum_{j=1}^{n} b_{-j} \times 2^{-j} \right) \tag{5.3}$$

where b_i represents the binary digit at position i, and the sign determines whether the value is positive or negative.

In our implementation, we partition the representation into three distinct components:

$$x_r = [x_0, x_1, x_2, x_3, x_4, x_5, x_6] \tag{5.4}$$

where each x_i is a 5-bit unsigned integer derived from the binary representation. Here, x_0 encodes the sign bit, x_1 represents the integer part, and x_2 through x_6 encode the decimal component with a precision of 2^{-25}. This approach offers deterministic precision control, as the positions of the binary point are fixed regardless of the magnitude of the input value. Unlike floating-point representation, this

method maintains uniform quantization steps across the entire range of representable values, which can be advantageous for certain neural network applications. Figure (5.6) demonstrates the transformation process from the continuous input to the structured fixed-point representation. The choice of 2^{-25} precision for the decimal component can be referenced in Appendix (a.1) for empirical error calculations. Although both 23-bit and 25-bit precision yield zero error rates (with 23-bit showing $error_{average} = 0.0$ and 25-bit showing $error_{average} = 0.0$) , we opted for 25-bit precision due to better hardware alignment and grouping considerations. This choice optimizes the data structure organization when implementing our representation scheme on the BrainScaleS-2 neuromorphic hardware.

5.2.3 Experiments and Results

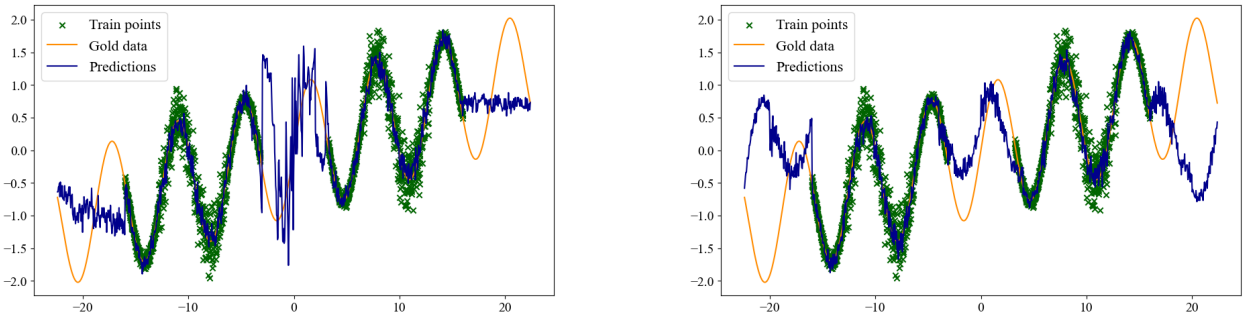
To evaluate the performance of binary sequence representation methods on a unified scale, we conducted experiments with quantized versions on the Noisy Sine Wave dataset.

EXPERIMENT SETTING The same experimental settings as in the previous section are used, except that we specifically focus on comparing IEEE754 floating-point and fixed-point number representation methods. Both representations were evaluated under identical conditions to isolate their performance differences.

The key distinction in our experiments was solely the input representation method, with all other factors (dataset configuration, model architecture, and quantization parameters) held constant. This controlled approach allowed us to accurately assess the efficacy of each binary sequence representation technique on neuromorphic hardware with strict bit-width limitations.

EXPERIMENT RESULTS AND ANALYSIS Figure (5.7) presents a visual comparison between IEEE754 floating-point and Fixed-Point number representations in quantized neural networks applied to the Noisy Sine Wave regression task. The results reveal important insights into how different numerical formats influence prediction quality under the 5-bit unsigned integer constraints imposed by neuromorphic hardware like BrainScaleS-2.

The IEEE754 representation (Figure 5.7a) demonstrates significantly compromised prediction smoothness. The prediction curve exhibits pronounced jagged artifacts, particularly visible around $x = 0$ and between $x = 5$ and $x = 10$. These oscillations can be attributed to the fundamental structure of IEEE754 encoding, where the sign bit, exponent, and mantissa components are regrouped into 5-bit unsigned integers. This decomposition, while preserving the mathematical range, creates inherent discontinuities in the numerical representation when



(a) IEEE754 **(b)** Fixed-Point Number

Figure 5.7: Comparison of IEEE754 and Fixed-Point Number Representations in Quantized Neural Networks for Noisy Sine Wave Regression: While both methods achieve good data fitting within the training domain, the predictions exhibit significant jagged artifacts, with IEEE754 representation (a) showing more severe non-smoothness compared to the Fixed-Point approach (b).

quantized, as adjacent values in the original space may have substantially different binary encodings after the IEEE754 transformation and regrouping process.

In contrast, the Fixed-Point representation (Figure 5.7b) shows markedly improved smoothness in its predictions. While still exhibiting some non-smooth behavior, the prediction curve follows the ground truth data with significantly fewer extreme oscillations. This improved stability stems from the fixed-point encoding's deterministic precision control, where the binary point positions remain fixed regardless of input magnitude. The partitioning of the representation into sign, integer, and decimal components with uniform quantization steps provides more consistent numerical transitions under quantization constraints.

The comparative performance difference between these numerical representations highlights a critical insight for neuromorphic computing: the binary encoding methodology significantly affects how neural networks handle the transitions between adjacent numerical values. The IEEE754 format's variable precision (higher precision near zero, lower precision for larger magnitudes) creates vulnerabilities to quantization artifacts, as small changes in the exponent can result in disproportionately large value jumps. The Fixed-Point approach, with its 2^{-25} precision for the decimal component, maintains uniform quantization steps across the representable range, resulting in relatively smoother predictions despite the severe bit-width limitations.

These findings suggest that for hardware-constrained neuromorphic systems, the choice of numerical representation profoundly impacts prediction stability. While both approaches achieve reasonable data fit-

ting within the training domain, the Fixed-Point representation offers superior robustness against oscillatory artifacts, making it more suitable for practical deployment in quantization-limited neuromorphic computing applications where prediction smoothness is valued.

5.3 INPUT REPRESENTATION VIA SECTIONAL ENCODING

Our experiments with dense layer approaches and binary sequence representations have revealed the importance of maintaining smooth transitions between adjacent numerical values when working under strict bit-width constraints. While these methods demonstrate various degrees of effectiveness, they still exhibit undesirable oscillatory behavior in regression tasks, particularly around critical transitions such as zero crossings in our sine wave experiments. This suggests that the representation method's ability to preserve ordinal relationships between values plays a crucial role in achieving smooth predictions.

Building upon these insights, we propose a sectional encoding strategy that explicitly preserves the ordering of numerical values while conforming to the 5-bit unsigned integer constraints of the BrainScaleS-2 system. Our approach partitions the input domain into discrete sections and applies specialized encoding within each section, with provisions for handling both positive and negative values effectively. We explore two variants of this approach: Sectional Zero-Filling Encoding and Sectional Thermometer Encoding.

5.3.1 Sectional Zero-Filling Encoding

Sectional Zero-Filling Encoding represents a straightforward implementation of the sectional approach. In this method, we first divide the input domain into m discrete sections or "percentiles," with a dividing point (typically zero) separating positive and negative domains. For datasets spanning both positive and negative values, we implement separate partitioning schemes on either side of this dividing point.

Given an input value x, we determine which partition $[p_i, p_{i+1})$ it falls into and generate an encoding vector where only the corresponding section contains a non-zero value:

$$Z(x)_j = \begin{cases} \frac{x-p_i}{p_{i+1}-p_i} \cdot (v_{max} - v_{min}) + v_{min} & \text{if } j = i \\ v_{min} & \text{if } j \neq i \end{cases} \tag{5.5}$$

where v_{min} and v_{max} define the range of representable values (0 and 31 respectively for 5-bit unsigned integers). The normalized value within the active section represents the position of x relative to the section boundaries, providing a fine-grained encoding within each partition.

It is worth noting that setting $v_{min} = 0$ introduces a potential issue: multiple distinct input values could map to identical all-zero representation vectors when they fall outside the partitioning range. To address this ambiguity problem, we reserve zero as a special indicator value, effectively limiting our quantization to 30 distinct values instead of the full 32 values available with 5 bits. While this slightly increases quantization error, it ensures that an all-zero representation uniquely corresponds to a specific input value, preserving the one-to-one mapping between inputs and representations that is crucial for effective learning.

For negative values, we apply this encoding to the absolute value and then place the result in the designated negative sections of our representation vector, effectively creating separate encoding spaces for positive and negative values. This approach produces a sparse representation where most elements are at the minimum value, with only one active section per input value. While Sectional Zero-Filling Encoding preserves the basic ordering of values through section allocation, it creates sharp discontinuities at section boundaries. Since adjacent values that fall into different sections activate completely different portions of the representation vector, the neural network must learn to bridge these discontinuities during training, potentially limiting the smoothness of predictions.

5.3.2 Sectional Thermometer Encoding

To address the discontinuity issues of Zero-Filling encoding, we developed Sectional Thermometer Encoding, a more sophisticated approach inspired by traditional thermometer coding techniques used in analog-to-digital conversion (Buckman et al., 2018).

Standard thermometer encoding represents a normalized value $v \in [0, 1]$ using n bits by setting the first $\lfloor v \cdot n \rfloor$ bits to 1 and the remaining bits to 0. This creates a representation where adjacent values differ by exactly one bit, reducing quantization errors and maintaining clear ordinal relationships. We extend this principle to our sectional framework, creating a hybrid approach that combines the benefits of thermometer encoding with adaptive sectional partitioning.

For an input value x falling into partition $[p_i, p_{i+1})$, Sectional Thermometer Encoding generates a representation where:

$$T(x)_j = \begin{cases} v_{max} & \text{if } j < i \\ \frac{x - p_i}{p_{i+1} - p_i} \cdot (v_{max} - v_{min}) + v_{min} & \text{if } j = i \\ v_{min} & \text{if } j > i \end{cases} \tag{5.6}$$

This creates a representation where all sections before the active section are fully activated (set to maximum value), the active section

contains a normalized value representing the position within that section, and all subsequent sections remain inactive (at minimum value).

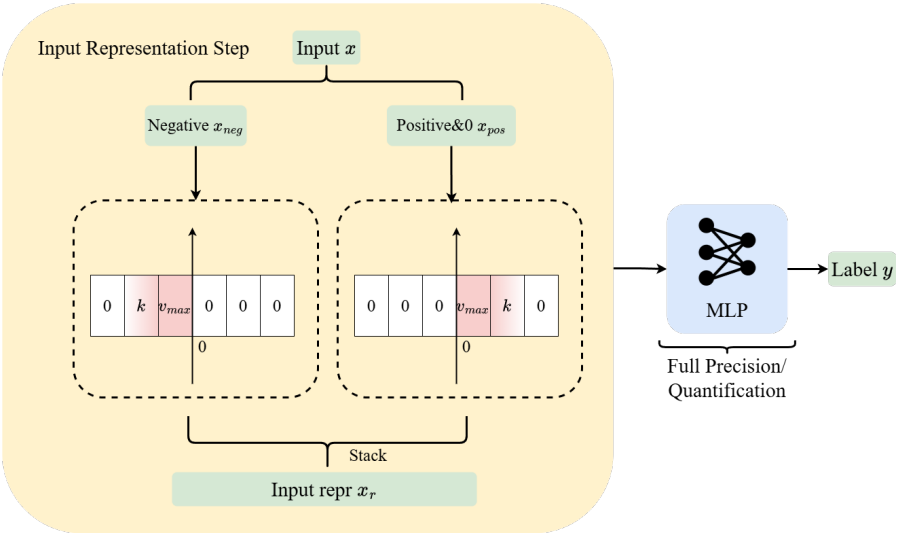


Figure 5.8: Sectional Thermometer Encoding for neural network inputs. The diagram illustrates the process of partitioning the input domain with specialized handling for negative and positive regions. Input values are first classified as positive or negative, then encoded using thermometer representation where the normalized values within each section are calculated according to Equation (5.6). As shown in the figure, v_{max} represents the maximum value (fully activated section), while k represents the normalized value in the active section calculated using the formula in Equation (5.6). The encoding finally results in a concatenated feature vector compatible with hardware constraints.

For handling both positive and negative values, we employ a sign-aware approach as illustrated in Figure (5.8). For negative values, we compute the thermometer encoding on the absolute value and then flip the order of representation (reversing the sequence of sections), ensuring that values close to zero from either direction have similar encodings. This special handling of the zero-crossing region helps maintain continuity where many other methods struggle.

The thermometer approach offers several advantages over zero-filling encoding. First, it provides enhanced information density, as each encoded value implicitly carries information about all section boundaries it has exceeded. Second, it creates smoother transitions between sections, as adjacent values that fall into different sections still share many common activated sections in their representations. These properties facilitate more consistent gradient propagation during training and potentially improve the neural network's ability to generalize across the input domain.

For our implementation with the BrainScaleS-2 hardware constraints, we use a representation range of $[0, 31]$ corresponding to 5-bit un-

signed integers, with 12 total partitions dynamically allocated between negative and positive domains based on data distribution. Our allocation algorithm assigns partitions proportionally to the range span of each domain, ensuring appropriate resolution throughout the input space.

5.3.3 Experiments and Results

To evaluate the performance of our sectional encoding methods on a unified scale, we conducted experiments with quantized versions on the Noisy Sine Wave dataset.

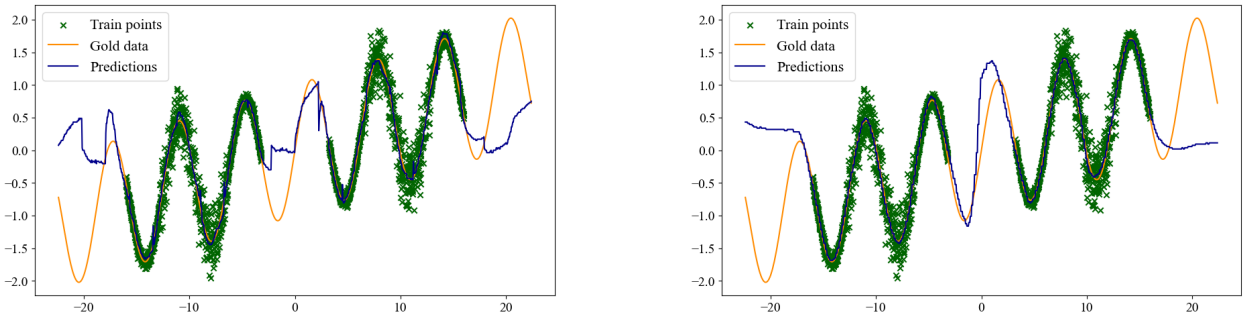
EXPERIMENT SETTING We conducted a comprehensive evaluation of both Zero-Filling and Thermometer sectional encoding methods using consistent experimental configurations:

- **Dataset:** 2,000 training samples, 1,000 testing samples, batch size 64.
- **Model Architecture:** Quantizable MLP with 50 neurons in the hidden layer, using sigmoid activation functions throughout to ensure compatibility with subsequent Bayesian implementations.
- **Quantization Parameters:** 5-bit unsigned representation for input quantization, 8-bit signed representation for output quantization, and 7-bit signed representation for weight quantization.
- **Sectional Encoding Configuration:** 12 total partitions dynamically allocated between negative and positive domains based on data distribution, with encoding range [0,31] to fully utilize the 5-bit representation.

The key distinction in our experiments was the input representation method, with all other factors held constant. This approach allowed us to isolate and accurately assess the efficacy of each sectional encoding technique on neuromorphic hardware with strict bit-width limitations.

EXPERIMENT RESULTS AND ANALYSIS Figure (5.9) illustrates the performance comparison between Sectional Zero-Filling and Sectional Thermometer Encoding approaches in addressing the Noisy Sine Wave regression challenge. These results provide valuable insights into how specialized input encoding strategies can further improve prediction smoothness within the 5-bit unsigned integer constraints of neuromorphic hardware systems like BrainScaleS-2.

The Sectional Zero-Filling encoding (Figure 5.9a) demonstrates improved prediction quality compared to previous numerical representations. While achieving reasonable fitting within the training domain, this approach still exhibits noticeable discontinuities at boundary regions between encoding sections. These artifacts manifest as abrupt



(a) Zero-Filling **(b)** Thermometer

Figure 5.9: Comparison of Sectional Zero-Filling and Sectional Thermometer Encoding in Quantized Neural Networks for Noisy Sine Wave Regression: Both methods achieve adequate fitting within the training domain, but Sectional Thermometer Encoding (b) demonstrates superior smoothness properties with virtually no significant discontinuities, capturing the continuous nature of the data more accurately. In contrast, the Zero-Filling approach (a) exhibits more pronounced jagged artifacts and non-smooth transitions at boundary regions.

value transitions, particularly visible around $x = 0$ and near the extremities of the input domain. This behavior can be attributed to the encoding strategy's sectional transitions, where adjacent values across section boundaries may have significantly different binary representations despite their numerical proximity in the original space.

In contrast, the Sectional Thermometer Encoding (Figure 5.9b) yields remarkably superior prediction smoothness. The prediction curve follows the ground truth with minimal oscillatory artifacts, maintaining consistent smoothness across the entire domain. This enhanced performance stems from the thermometer code's intrinsic ordinal property, where adjacent numerical values maintain similar binary patterns, differing by only one bit flip. By preserving the distance relationships between encoded values, the thermometer approach ensures that small changes in the input produce correspondingly small changes in the network's internal representations, resulting in the observed continuity of predictions.

The comparative analysis reveals that Sectional Thermometer Encoding offers substantial advantages for neuromorphic computing applications requiring smooth regression capabilities. Unlike the Zero-Filling method, which primarily addresses the representational range without fully resolving transition discontinuities, the thermometer approach directly addresses the core challenge of maintaining ordinal relationships through its monotonic encoding structure. This property proves especially valuable at boundary transitions between encoding

sections, where other methods typically struggle with representational inconsistencies.

These findings highlight the critical importance of explicitly preserving numerical ordinality in binary encodings for quantization-constrained neural networks. While traditional numerical representations focus primarily on range and precision considerations, our experiments demonstrate that maintaining smooth transition properties between adjacent values plays an equally fundamental role in achieving robust regression performance under severe bit-width limitations. The Sectional Thermometer Encoding thus represents a significant advancement for neuromorphic systems, enabling more reliable analog function approximation within the strict hardware constraints of platforms like BrainScaleS-2.

5.4 COMPARATIVE ANALYSIS OF INPUT REPRESENTATION METHODS

Method		MSE	average convert error	
Main Method	Sub Method		train	test
Baseline (Full precision)		0.0294	0.0	0.0
Dense Layer	End-to-End	0.0335	–	–
	Autoencoder	0.0937	0.0242	0.0281
Binary sequence	IEEE754 Floating-Point	0.0161	0.0	0.0
	Fixed-Point Number	0.0155	0.0	0.0
Sectional Encoding	Zero-Filling	0.0251	2.41×10^{-9}	4.13×10^{-9}
	Thermometer	0.0232	2.39×10^{-9}	4.08×10^{-9}

Table 5.1: Quantitative Comparison of Different Input Representation Methods for Noisy Sine Wave Regression: Performance metrics include Mean Squared Error (MSE) for training data evaluation and average conversion error for both training and test datasets. The Binary sequence methods achieve the lowest training MSE, while Sectional Thermometer Encoding approaches demonstrate negligible conversion errors and competitive training MSE values, balancing numerical accuracy with prediction smoothness under 5-bit unsigned integer constraints.

Our investigation of various input representation methods for quantized neural networks reveals important trade-offs between information preservation, computational efficiency, and prediction quality. Table (5.1) presents quantitative metrics for each approach, providing a foundation for our comparative analysis. The average conversion error metric quantifies the mean absolute difference between original input values and their reconstructed versions after passing through

the quantization and representation process, effectively measuring information loss during the conversion process. To complement these numerical findings, Table (5.2) offers a qualitative evaluation across multiple dimensions that are critical for neuromorphic implementations.

The Dense Layer approaches demonstrate substantially different characteristics between their implementation variants. The End-to-End method achieves reasonable prediction accuracy (MSE = 0.0335) and maintains the advantage of learning task-specific representations. However, this comes at the cost of increased training complexity due to joint optimization of representation and task-specific parameters. The Autoencoder method exhibits the highest MSE (0.0937) among all tested approaches, suggesting fundamental limitations in decoupled learning strategies under severe quantization constraints. Additionally, its two-phase training procedure introduces significant computational overhead, making it less practical for resource-constrained environments.

Binary Sequence representations achieve the lowest MSE values overall, with Fixed-Point Number representation (MSE = 0.0155) slightly outperforming IEEE754 Floating-Point (MSE = 0.0161). This superior numerical accuracy can be attributed to their comprehensive encoding schemes that efficiently utilize available bits to represent numerical values. However, as clearly demonstrated in Figure (5.7), both methods—especially IEEE754—exhibit significant oscillatory artifacts in their predictions. This suggests that while these approaches excel at minimizing average error, they struggle with maintaining smooth transitions between adjacent values, potentially limiting their applicability in scenarios where prediction continuity is crucial.

Sectional Thermometer Encoding approaches demonstrate a compelling balance between accuracy and smoothness. While not achieving the lowest absolute MSE values, they deliver competitive performance (MSE = 0.0232 for Thermometer encoding) while significantly outperforming other methods in prediction smoothness. Their negligible conversion errors ($\sim 10^{-9}$) indicate high fidelity in the encoding-decoding process, suggesting minimal information loss during representation transformation. Most importantly, as evident in Figure (5.9), the Thermometer variant exhibits exceptional smoothness properties, virtually eliminating the discontinuities that plague other methods, particularly at boundary transitions.

The comparative assessment reveals that each representation method presents distinct advantages depending on application priorities. For applications where absolute numerical precision is paramount, Binary Sequence methods offer the best performance. Where smooth predictions are critical, Sectional Thermometer Encoding provides superior results. The End-to-End Dense Layer approach offers a middle ground

that may be suitable for scenarios requiring adaptation to specific datasets or tasks.

For neuromorphic computing applications on hardware with severe bit-width constraints like the BrainScaleS-2 system, our findings suggest that Sectional Thermometer Encoding represents the most balanced approach. Its explicit preservation of ordinality between adjacent values, combined with competitive MSE performance and minimal conversion error, makes it particularly well-suited for regression tasks requiring continuous predictions. The implementation simplicity and low computational overhead further enhance its practical viability for deployment in energy-efficient neuromorphic systems.

Method		Training	Computational	Prediction	Mapping	Preservation of
Main Method	Sub Method	Complexity	Overhead	Smoothness	Uniqueness	Ordinality
Dense Layer	End-to-End	High	Medium	Medium	High	Medium
	Autoencoder	Very High	High	Low	Medium	Low
Binary Sequence	IEEE754	Low	Low	Low	High	Low
	Fixed-Point	Low	Low	Medium	High	Medium
Sectional Encoding	Zero-Filling	Medium	Low	Medium	High	High
	Thermometer	Medium	Low	High	High	Very High

Table 5.2: Qualitative Comparison of Input Representation Methods for Neuromorphic Computing: The analysis evaluates each approach across dimensions critical for implementation on hardware with strict bit-width constraints. Training Complexity refers to the difficulty of optimizing model parameters; Computational Overhead measures the processing resources required for encoding; Prediction Smoothness assesses continuity in output predictions; Mapping Uniqueness evaluates the preservation of one-to-one correspondence between original and transformed values; and Preservation of Ordinality measures how well the relative ordering of values is maintained in the representation.

These insights inform future development of quantized neural networks for neuromorphic hardware, highlighting the critical importance of representation methods that preserve ordinality and smooth transitions between adjacent values, rather than focusing exclusively on minimizing average numerical error. Such considerations become increasingly important as neuromorphic computing advances toward more complex applications requiring analog function approximation under strict hardware constraints.

5.5 INPUT REPRESENTATION FOR HIGH-DIMENSIONAL IMAGE DATA: THE CASE OF DIRTYMNIST

While the previous sections focused on low-dimensional continuous-valued data, high-dimensional discrete data such as images present different challenges and opportunities for input representation in neuromorphic computing. The DirtyMNIST dataset, with its 28×28 pixel input structure (resulting in 784 dimensions), offers substantially richer feature space compared to the low-dimensional datasets previously examined. This dimensional abundance fundamentally alters the approach required for effective input representation under hardware constraints.

For high-dimensional image data, the quantization challenge is mitigated by the inherent redundancy and distributed information content across the numerous input features. This characteristic allows for a simpler input representation approach while maintaining model performance. Specifically, for DirtyMNIST implementation on BrainScaleS-2 hardware, we employ a straightforward non-negative offset transformation that merely shifts the input values to comply with the unsigned integer requirement:

$$x_r = x - \min(X) \tag{5.7}$$

where x_r represents the transformed input, x is the original input value and $\min(X)$ is the global minimum value across the entire dataset. This approach maintains the relative distances between data points while fulfilling the non-negative constraint with minimal computational overhead.

5.5.1 Experimental Design and Results

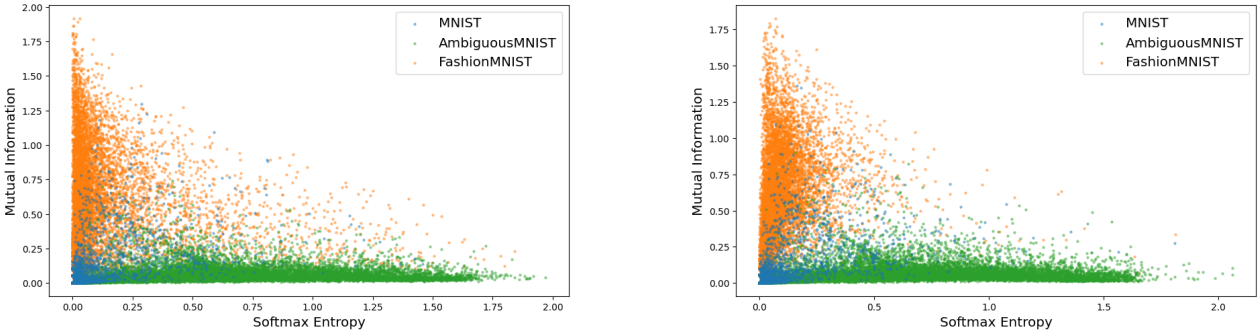
To systematically evaluate the impact of input representation on high-dimensional image data, we conducted experiments using Bayesian Neural Networks (BNNs) rather than deterministic networks. This methodological choice was deliberate, as BNNs provide richer evaluation metrics including uncertainty quantification, which allows for more comprehensive assessment of representation quality beyond mere accuracy metrics.

EXPERIMENTAL SETTINGS All experiments were performed with the following settings:

- **Dataset:** DirtyMNIST was used for training, MNIST, AmbiguousMNIST, and FashionMNIST were used for testing, and the batch size was set to 100.
- **Model:** Two hidden layers of MLP, each containing 100 neurons, using Relu activation function.

- **Training:** 1500 epochs of pre-training (lr=0.001), 1000 epochs of main training (lr=0.0001), 1 particle for training, and random_seed was set to 42.
- **KL Annealing:** Maximum weight 0.25, using an annealing scheme focused on the end of training, so that the KL loss is almost inactive in the early stages of training and only increases near the end of training.

The only difference between the two experimental conditions was the input representation: one experiment used raw pixel values directly, while the other applied a non-negative offset transformation by subtracting the global minimum value across the dataset from each input.



(a) Without Offset **(b)** With Offset

Figure 5.10: Uncertainty Decomposition for BNN Trained on DirtyMNIST: Scatter plots illustrate the relationship between predictive uncertainty (Softmax Entropy) and epistemic uncertainty (Mutual Information) across three datasets. The comparison between (a) without offset transformation and (b) with non-negative offset transformation shows minimal differences in uncertainty profiles, demonstrating that simple offset representation preserves the statistical properties of high-dimensional image data under hardware constraints.

EXPERIMENT RESULTS AND ANALYSIS Figures (5.10a) and (5.10b) present the visual decomposition of uncertainty for Bayesian Neural Networks trained without and with offset transformation, respectively, under full precision conditions without quantization constraints. Both visualizations reveal three distinct clusters corresponding to the three datasets, characterized by their unique uncertainty signatures. The uncertainty distribution for FashionMNIST in Figure (5.10b) demonstrates improved cluster cohesion and more pronounced separation from the other datasets compared to Figure (5.10a), suggesting that the offset transformation enhances the model's ability to discriminate

out-of-distribution samples with more consistent uncertainty quantification. This improved clustering behavior indicates that even in full precision implementations, the simple non-negative offset transformation contributes to the statistical properties of uncertainty representation for high-dimensional image data, establishing a strong foundation for subsequent hardware-constrained deployments.

Method	MNIST					AmbiguousMNIST					FashionMNIST				
	A ↑	U ↑	P.U.	S.E.	M.I.	A ↑	U ↑	P.U.	S.E.	M.I.	A ↑	U ↑	P.U.	S.E.	M.I.
Without Offset	0.97	0.98	0.05	0.02	0.03	0.51	0.74	0.71	0.65	0.06	0.07	0.78	0.57	0.11	0.46
With Offset	0.98	0.98	0.05	0.03	0.03	0.51	0.73	0.71	0.65	0.07	0.09	0.77	0.56	0.08	0.48

Table 5.3: Performance comparison of models with and without input offset transformation across different MNIST datasets. ↑ indicates that higher values are better. A: Accuracy, U: Unanimity, P.U.: Predictive Uncertainty, S.E.: Softmax Entropy, M.I.: Mutual Information. Results demonstrate minimal differences between the two input representation approaches, with slight improvements in accuracy for MNIST and FashionMNIST when using the offset transformation. Both approaches maintain similar uncertainty profiles, with the most notable differences observed in the FashionMNIST dataset, confirming that simple non-negative offset is sufficient for representing high-dimensional image data.

These findings highlight a key insight for neuromorphic computing implementations: high-dimensional image data possesses inherent resilience to simplistic input transformations due to its distributed information content. Unlike low-dimensional continuous data, where precise representation of each feature is critical, image data distributes information across hundreds of dimensions, providing natural redundancy. This characteristic makes high-dimensional data particularly well-suited for implementation on hardware-constrained neuromorphic systems with strict bit-width limitations. While sophisticated encoding strategies are essential for low-dimensional continuous data, our results suggest that high-dimensional image data can be effectively processed with minimal preprocessing overhead, allowing for more efficient deployment of computer vision applications on neuromorphic hardware platforms.

6 DIFFERENT LEVELS OF QUANTIZATION METHODS

The methods introduced in Chapter 2 are applied to three machine learning tasks to evaluate and compare the quality of predictive outputs and uncertainties. The first task involves a regression problem on a noisy sine wave, the second task focuses on classification using the Two Moons dataset, while the third task classifies DirtyMNIST images (Mukhoti et al., 2021). As all three tasks inherently contain aleatoric and epistemic uncertainty, these relatively simple problems are well-suited for quantization and, particularly, for visualizing each model's performance. Specifically, for the noisy sine wave and Two Moons datasets, we employ the input representation methods introduced in Chapter 5 for preprocessing. For DirtyMNIST, we preprocess the data by computing the global minimum of the dataset and subtracting that value from each item in the dataset to maintain positive values. This chapter aims to describe the impact of different quantization levels on BNN inference performance from a qualitative perspective.

Before detailing each quantization approach, it is important to understand why we explore these three distinct levels. Traditional quantization methods for DNNs cannot be directly applied to BNNs due to the probabilistic nature of their weights. Our three-level approach allows us to systematically evaluate different trade-offs between computational efficiency and the preservation of uncertainty information.

We utilize the Pyro framework introduced in Section 2.3 as the underlying framework for different levels of quantization. As illustrated in Figure (6.1), the modules that can be quantized in the forward pass are similar to those in traditional DNNs. However, the key distinction lies in the weight distributions, which can be quantized from three different perspectives:

- **Sample-Based Quantization** The sampled value of weights distribution.
- **Parameter-Based Quantization:** The parameter (mean and standard deviation) of weights distribution.
- **Fully Integrated Quantization** A comprehensive approach considering the mean, standard deviation, and sampled values of weights distribution.

In the following sections, we will elucidate how this framework integrates with Brevitas, as described in Section 2.5, and present quantitative experimental analyses. Further exploration of optimization

techniques to enhance the performance of these three quantization levels will be elaborated in Chapter 7.

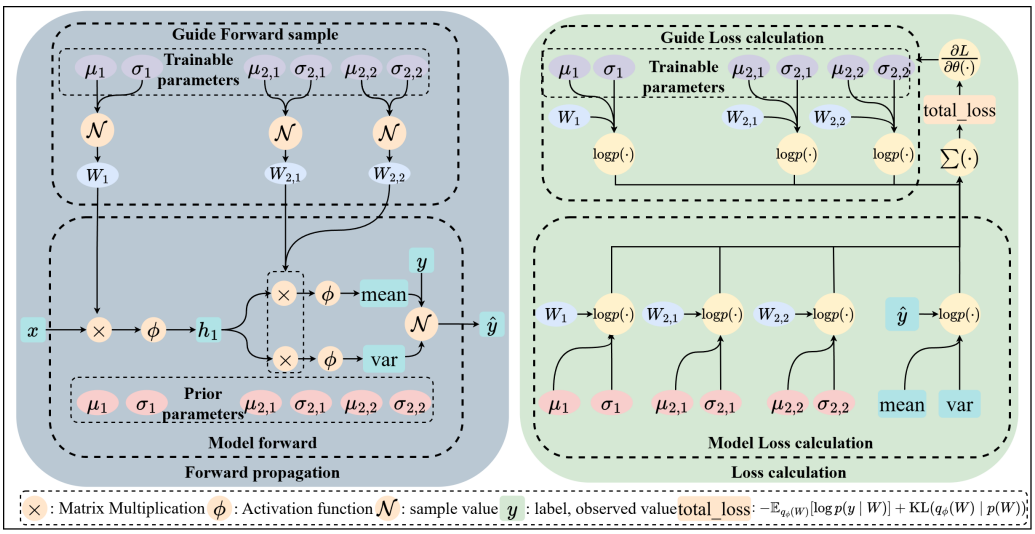


Figure 6.1: SVI forward sampling and ELBO loss calculation process based on Pyro. The figure demonstrates the key components and their interactions during the sampling process, including prior distributions, approximate posterior distributions, sampled weights, and loss calculation.

6.1 SAMPLE-BASED QUANTIZATION METHOD

Among the three quantization levels previously discussed, directly quantizing the sampled values from distributions is relatively straightforward to implement, as this approach closely resembles quantization in DNNs. To ensure compatibility with BSS-2 hardware specifications (Pehle et al., 2022), we implemented 5-bit unsigned input quantization, 8-bit signed output quantization, and 7-bit signed weight quantization. These specific bit-width choices are based on hardware constraints and preliminary experimental results, optimizing resource usage while meeting accuracy requirements. Figure (6.2) illustrates our quantization implementation, where each linear layer requires input quantization for incoming values, weight quantization for the sampled distribution values, and output quantization for the layer outputs.

Mathematically, our sampling-based quantization approach can be formulated as:

$$\begin{aligned} w_s &\sim \mathcal{N}(\mu, \sigma^2) \\ w_q &= Q(w_s; b_w, s_w, z_w) \end{aligned} \tag{6.1}$$

where w_s represents the sampled weight from the normal distribution parameterized by mean μ and standard deviation σ, and w_q denotes

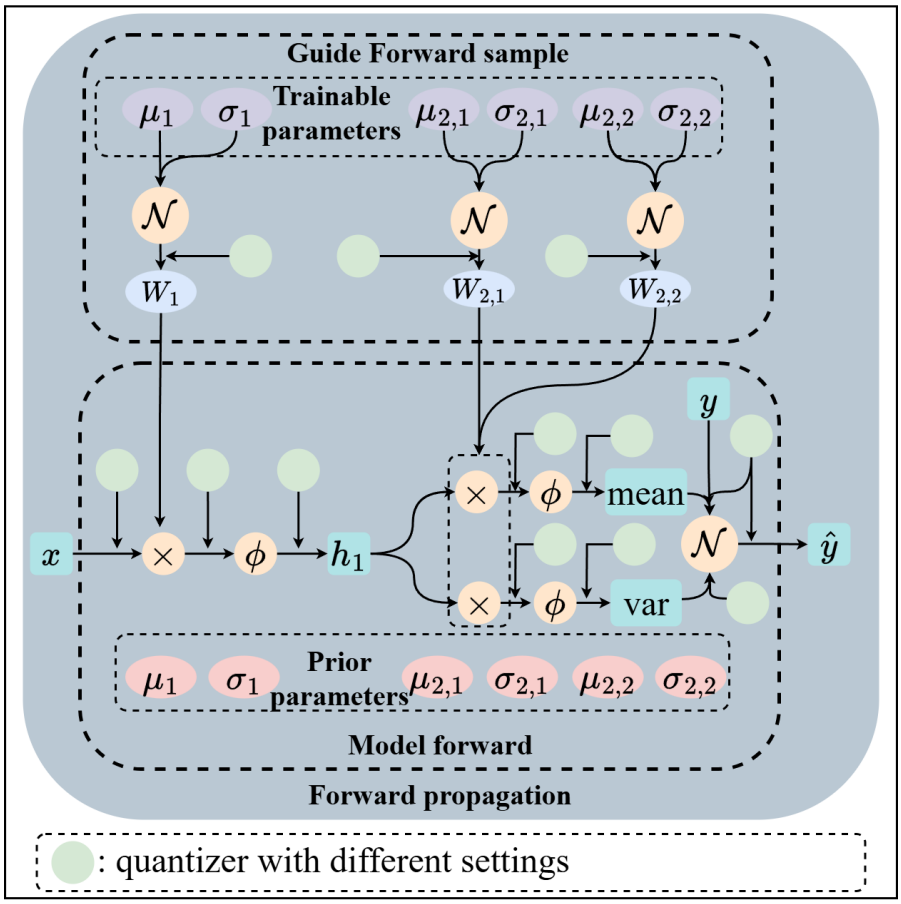


Figure 6.2: Weighted distribution sampling value quantization based on fusion of Brevitas and Pyro. The figure shows the process of applying quantization after sampling from weight distributions, including the complete flow of input quantization, weight quantization, and output quantization.

the quantized weight. The quantization function $Q(\cdot)$ is defined by bit-width b_w, scaling factor s_w, and zero-point z_w parameters.

The quantization process can be further expressed as:

$$w_q = s_w \cdot \text{clamp}\left(\left\lfloor \frac{w_s}{s_w} \right\rceil + z_w, n_{min}, n_{max}\right) \tag{6.2}$$

where $\lfloor \cdot \rceil$ represents rounding to the nearest integer, n_{min} and n_{max} denote the minimum and maximum representable values given the bit-width b_w. For our 7-bit signed weight quantization, these bounds are $n_{min} = -2^6$ and $n_{max} = 2^6 - 1$.

Our implementation leverages the Brevitas framework, which provides efficient quantization mechanisms with minimal user intervention. The default quantizers in our system utilize `Uint8ActPerTensorFloat` through `Int8ActPerTensorFloat` to initialize activation scaling param-

eters. The scaling factor s is determined through statistical analysis:

$$s = \frac{\text{percentile}(|x|, 99.999)}{2^{b-1} - 1} \tag{6.3}$$

where $\text{percentile}(|x|, 99.999)$ computes the 99.999 percentile of absolute values of activation x over approximately 300 training steps.

This statistical collection serves as an initial calibration phase, though it differs from conventional calibration approaches in one key aspect: quantization remains enabled during statistics collection, whereas traditional methods first collect floating-point statistics before enabling quantization. The collected statistics form an exponential moving average:

$$\text{EMA}_t = \alpha \cdot \text{stat}_t + (1 - \alpha) \cdot \text{EMA}_{t-1} \tag{6.4}$$

where stat_t represents the statistics from the current batch, and α is the smoothing factor. Upon completion of the collection phase, this EMA initializes the learned `nn.Parameter` values internally.

The quantizer behavior during this process varies between training and evaluation modes, similar to batch normalization. When in `train()` mode, the quantizer returns statistics specific to the current batch:

$$s_{train} = \frac{\text{percentile}(|x_{batch}|, 99.999)}{2^{b-1} - 1} \tag{6.5}$$

Conversely, in `eval()` mode, it returns the exponential moving average of collected statistics:

$$s_{eval} = \frac{\text{EMA}}{2^{b-1} - 1} \tag{6.6}$$

Once the collection phase concludes, both execution modes return the learned parameters.

This sampling-based quantization approach offers a practical balance between implementation complexity and model performance preservation. By focusing on quantizing the sampled values rather than the distribution parameters themselves, we maintain the fundamental Bayesian characteristics of the network while enabling significant computational efficiency gains. The integration with variational inference can be expressed in the quantized evidence lower bound (ELBO):

$$\text{ELBO}Q = \mathbb{E}q_\phi(w_q|x)[\log p(y|x, w_q)] - \text{KL}(q_\phi(w|x) \parallel p(w)) \tag{6.7}$$

where $q_\phi(w_q|x)$ represents the quantized approximate posterior. Our experimental results demonstrate that this approach effectively preserves both predictive accuracy and uncertainty estimation capabilities across our benchmark tasks.

6.2 PARAMETER-BASED QUANTIZATION METHOD

While sampling-based quantization provides a straightforward approach, directly quantizing the distribution parameters offers potential advantages in terms of computational efficiency and consistency. This section details our implementation of distribution parameter quantization compatible with the BSS-2 neuromorphic hardware constraints (Pehle et al., 2022).

In Bayesian Neural Networks, weight distributions are typically parameterized by mean μ and variance σ^2 or standard deviation σ. Our distribution parameter quantization approach targets these parameters directly, as illustrated in Figure (6.3).

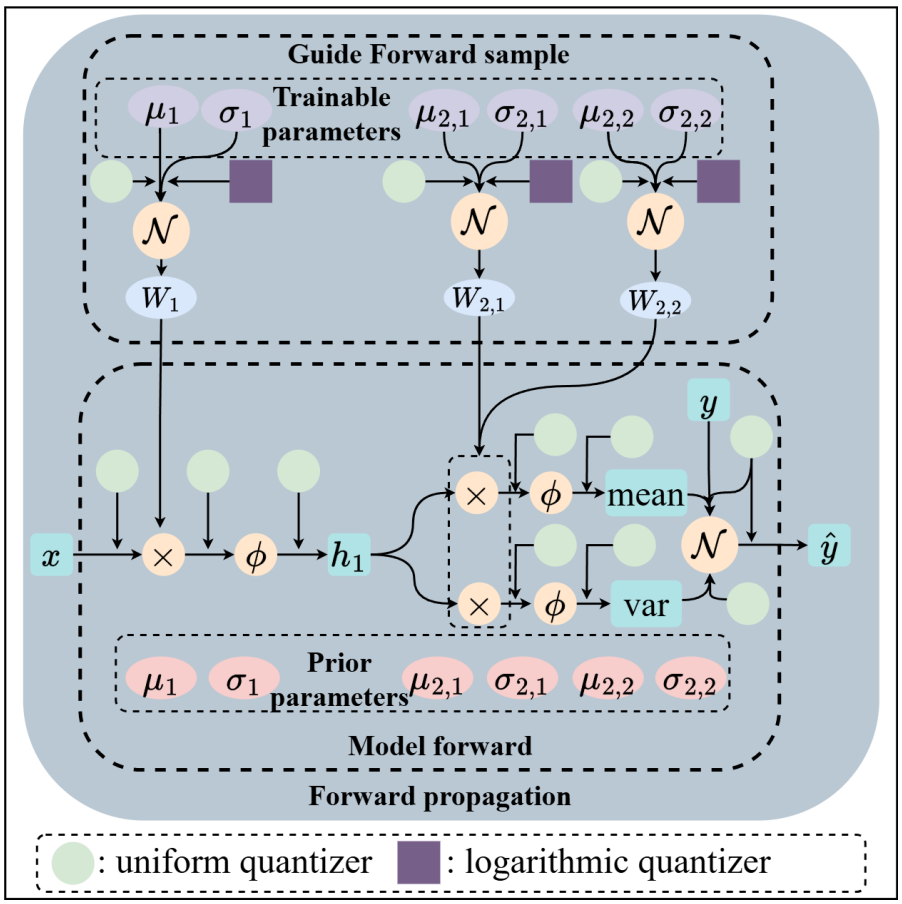


Figure 6.3: Distribution parameter quantization framework showing separate quantization paths for mean and standard deviation parameters, with logarithmic quantization applied specifically to standard deviation parameters. This approach allows for more precise representation of a wide range of standard deviation values, which is particularly important for uncertainty estimation in high-confidence predictions.

Mathematically, the distribution parameter quantization can be formulated as:

$$\begin{aligned} \mu_q &= Q_{linear}(\mu; b_\mu, s_\mu, z_\mu) \\ \sigma_q &= Q_{log}(\sigma; b_\sigma, s_\sigma, z_\sigma) \end{aligned} \tag{6.8}$$

where μ_q and σ_q represent the quantized mean and standard deviation parameters, respectively. The quantization functions are parameterized by their respective bit-widths (b_μ, b_σ), scaling factors (s_μ, s_σ), and zero-points (z_μ, z_σ).

For compatibility with BSS-2 hardware, we implemented 7-bit signed quantization for both mean parameters and standard deviation parameters. The quantization process for mean parameters follows conventional linear quantization:

$$\mu_q = s_\mu \cdot \text{clamp}\left(\left\lfloor \frac{\mu}{s_\mu} \right\rceil + z_\mu, -2^6, 2^6 - 1\right) \tag{6.9}$$

However, standard deviation parameters present unique challenges that make linear quantization suboptimal. Standard deviation values are strictly positive and typically span multiple orders of magnitude, ranging from 10^{-6} to 10^2 in our experiments. This wide dynamic range is critical for accurate uncertainty estimation, as standard deviation directly represents the model's confidence in its predictions. Additionally, the non-negative nature of standard deviation means that conventional linear quantization with signed representation would inefficiently utilize nearly half of the available quantization levels. Figure (6.4) illustrates the quantization error comparison between linear and logarithmic quantization schemes for standard deviation parameters.

As shown in the Figure (6.4), linear quantization introduces substantial relative errors for small standard deviation values, which are prevalent in well-trained models for confident predictions. To address these limitations, we implement logarithmic quantization for standard deviation parameters:

$$\sigma_q = \exp\left(s_\sigma \cdot \text{clamp}\left(\left\lfloor \frac{\log(\sigma)}{s_\sigma} \right\rceil + z_\sigma, -2^6, 2^6 - 1\right)\right) \tag{6.10}$$

This logarithmic approach in Equation (6.10) offers two significant advantages: First, it enables more efficient representation of the wide dynamic range of standard deviation values while preserving the relative precision across different scales. Second, by transforming the strictly positive standard deviation values into the real domain through the logarithm function, we can fully utilize the entire signed 7-bit quantization range (-2^6 to $2^6 - 1$), effectively doubling the number of quantization levels available compared to using unsigned quantization for the original standard deviation values. The scaling factor for logarithmic quantization is calculated as:

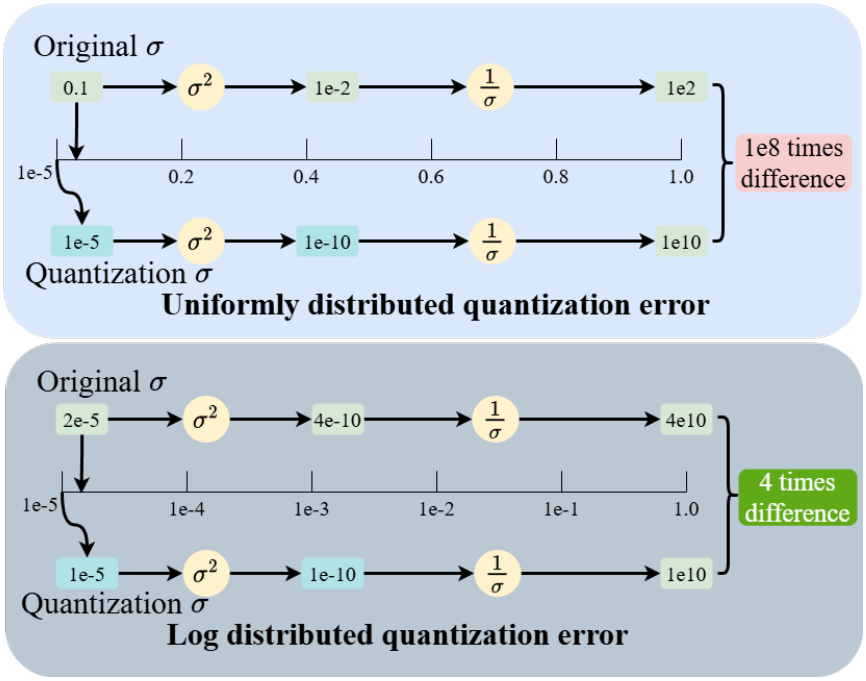


Figure 6.4: Comparison of quantization error between linear (uniform) quantization and logarithmic quantization of the standard deviation parameter at different scales when computing log_prob. Due to the involvement of $1/\sigma^2$ term in log-probability calculations, quantization errors are significantly amplified, especially for smaller standard deviation values. Linear quantization shows substantially higher relative errors in this scenario, which is critical for uncertainty estimation in high-confidence predictions.

$$s_\sigma = \frac{\log(\sigma_{max}) - \log(\sigma_{min})}{2^7 - 2} \tag{6.11}$$

where σ_{max} and σ_{min} represent the maximum and minimum standard deviation values observed during calibration. This approach ensures uniform relative precision across the entire range of standard deviation values, with particular benefits for small standard deviation values that are critical for uncertainty estimation in confident predictions.

For mean parameters, the scaling factor is determined through conventional statistical analysis:

$$s_\mu = \frac{\text{percentile}(|\mu|, 99.999)}{2^{b_\mu - 1} - 1} \tag{6.12}$$

This approach implements a statistically-driven uniform quantization for mean parameters, which is essential because the distribution of mean values across neural network weights often exhibits unpredictable patterns that vary significantly between layers and during

training. Unlike standard deviation parameters, which follow strictly positive distributions with known characteristics, mean parameters can span both positive and negative domains with highly task-specific distributions. The statistical method leverages the actual empirical distribution of the mean values by capturing the 99.999 percentile, effectively establishing a quantization range that accommodates the vast majority of values while minimizing outlier effects.

During inference, sampling occurs from the quantized distributions:

$$w \sim \mathcal{N}(\mu_q, \sigma_q^2) \tag{6.13}$$

The logarithmic quantization of standard deviation parameters results in significant improvements in uncertainty estimation quality, particularly for high-confidence predictions where precise representation of small standard deviation values is crucial.

The modified evidential lower bound for this approach becomes:

$$\text{ELBO}_Q = \mathbb{E}_{q_\phi(w|\mu_q,\sigma_q^2,x)}[\log p(y|x,w)] - \text{KL}(q_\phi(w|\mu_q,\sigma_q^2,x) \parallel p(w)) \tag{6.14}$$

This parameter-based quantization approach with specialized treatment for standard deviation parameters offers competitive performance while reducing the computational overhead associated with repeated sampling during training.

6.3 FULLY INTEGRATED QUANTIZATION METHOD

After exploring both distribution parameter quantization and sampling-based quantization methods independently, we further combine the two approaches to maximize hardware efficiency and increase inference speed. As shown in Figure (6.5), this integrated framework simply applies both quantization methods in sequence.

Our integrated approach can be mathematically formulated as:

$$\begin{aligned} \mu_q &= Q(\mu; b_\mu, s_\mu, z_\mu) \\ \sigma_q &= Q_{log}(\sigma; b_\sigma, s_\sigma, z_\sigma) \\ w_s &\sim \mathcal{N}(\mu_q, \sigma_q^2) \\ w_q &= Q(w_s; b_w, s_w, z_w) \end{aligned} \tag{6.15}$$

where μ_q and σ_q represent the quantized mean and standard deviation parameters, w_s is the weight sampled from the quantized distribution, and w_q is the final quantized weight after sampling. The entire process can be divided into three simple steps:

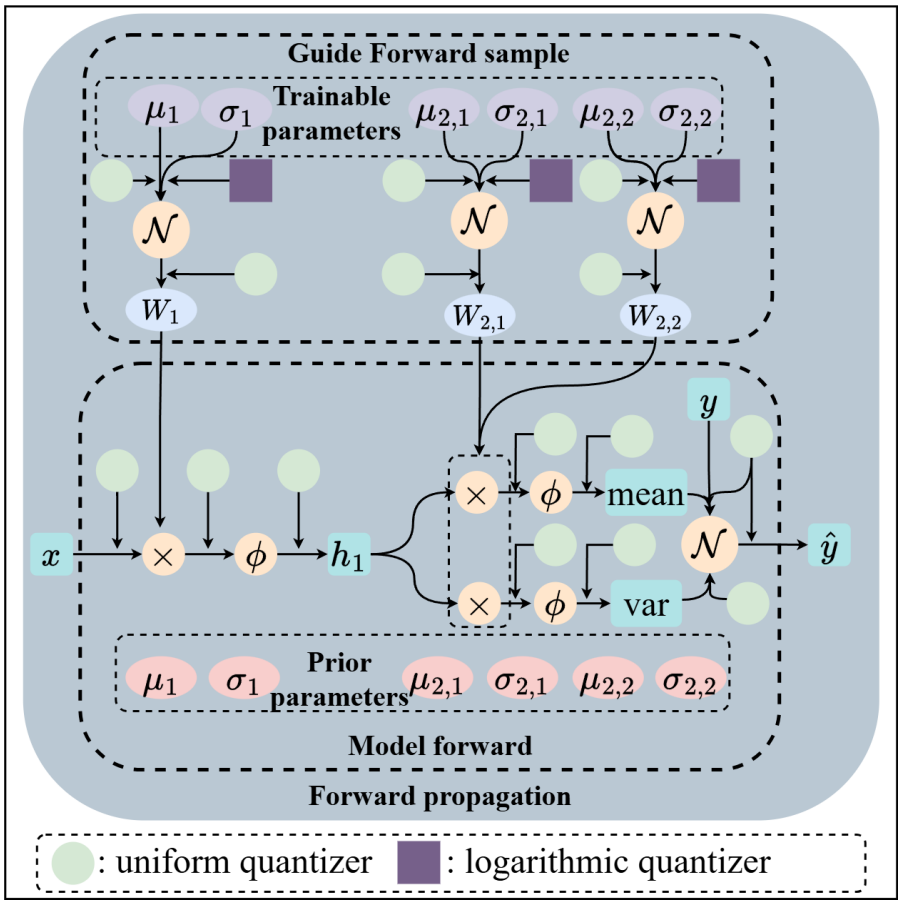


Figure 6.5: Integrated quantization framework showing the sequential application of distribution parameter quantization followed by sampling value quantization

1. First, the distribution parameters (μ and σ) are quantized
2. Weight samples are drawn from the quantized distributions
3. The sampled weights are quantized before being used in the forward pass

In this integrated framework, we apply 7-bit signed quantization for both mean parameters and sampled weights, while using 7-bit signed logarithmic quantization for standard deviation parameters. This quantization approach introduces quantization effects at multiple stages of the Bayesian inference process, which may have compound effects on predictive accuracy and uncertainty estimation.

The evidence lower bound for this integrated method can be represented as:

$$\begin{aligned} \text{ELBO}_Q = \mathbb{E}_{q_\phi(w_q|\mu_q,\sigma_q^2,x)}[\log p(y|x,w_q)] \\ - \text{KL}(q_\phi(w|\mu_q,\sigma_q^2,x) \parallel p(w)) \end{aligned} \tag{6.16}$$

6.4 EXPERIMENTS AND RESULTS

To comprehensively evaluate the performance of the above precision measurement methods, we conduct detailed experiments on the three tasks mentioned above.

6.4.1 Noisy Sine Wave

EXPERIMENT SETTING All experiments shared these settings:

- **Dataset:** 2,000 training samples, 2,000 testing samples, batch size 64
- **Model:** MLP with input representation input, 50 neurons in hidden layer, sigmoid activation (alpha-sigmoid activation for quantization experiments, a modified sigmoid function enabling smoother quantization, detailed in Section 7.1)
- **Training:** 300 epochs pretraining (lr=0.02), 1,000 epochs main training (lr=0.005), 1 particle for training, random_seed is set to 42.
- **KL Annealing:** Maximum weight 0.025, with an end-focused schedule that keeps the KL loss nearly inactive in early epochs and ramps it up only near the end of training.

The model quantization parameters conform to the BSS-2 hardware specifications, with 5-bit unsigned representation for input quantization, 8-bit signed representation for output quantization, and 7-bit signed representation for weight quantization.

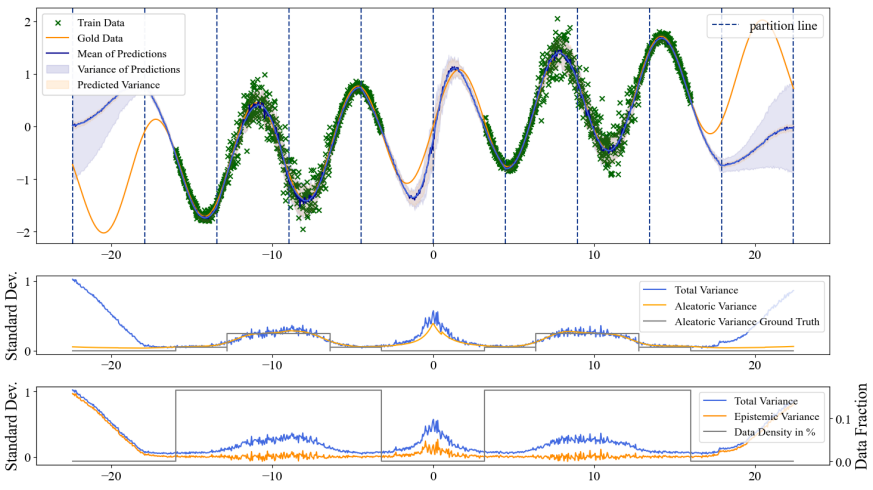


Figure 6.6: Noisy sine wave baseline based on input representation

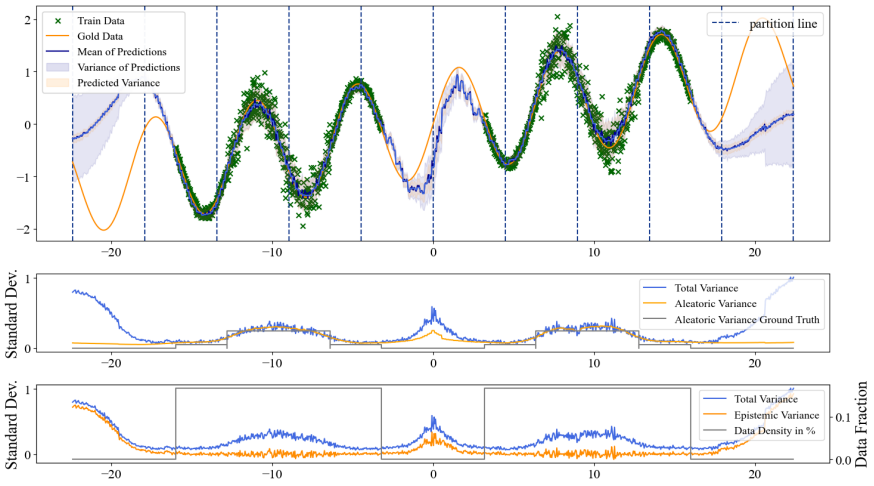


Figure 6.7: Noisy Sine Wave quantized by dirstibution sampling values

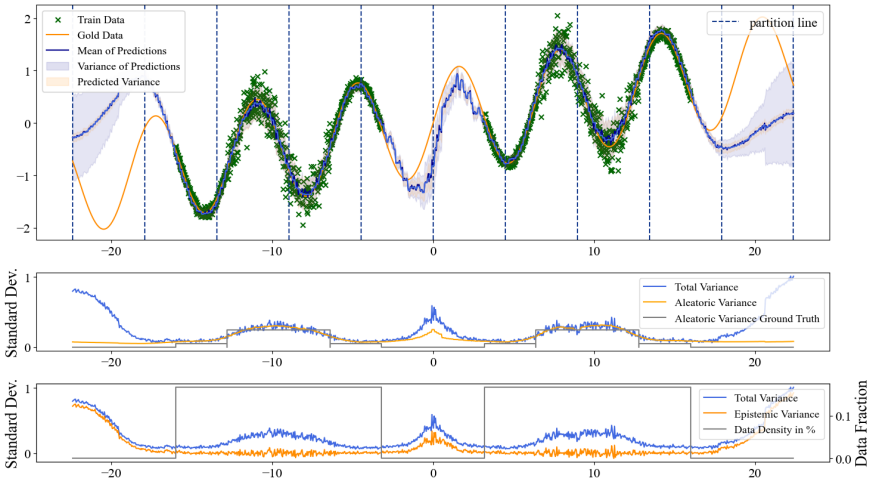


Figure 6.8: Noisy Sine Wave quantized by dirstibution parameters

EXPERIMENT RESULTS AND ANALYSIS The quantitative comparison presented in Table 6.1, along with the visualizations in Figures (6.6-6.9), provides significant insights into the effects of different quantization methods on model performance and uncertainty estimation.

The baseline model achieves superior overall accuracy with the lowest MSE (3.38×10^{-4}) and negative log-likelihood (-0.0151), confirming the expected performance degradation when quantization is introduced. As evident in Figure (6.6), the baseline model's prediction mean closely matches the target values, denoted as gold data in the figure legend, with well-calibrated uncertainty bands.

Interestingly, the distribution sampling value and distribution parameter quantization methods are slightly better than the baseline method in terms of aleatoric mean square error (MSE) (2.38×10^{-4} vs 2.41×10^{-4}). This shows that the introduction of quantization error and quantization-aware training better allows the model to express

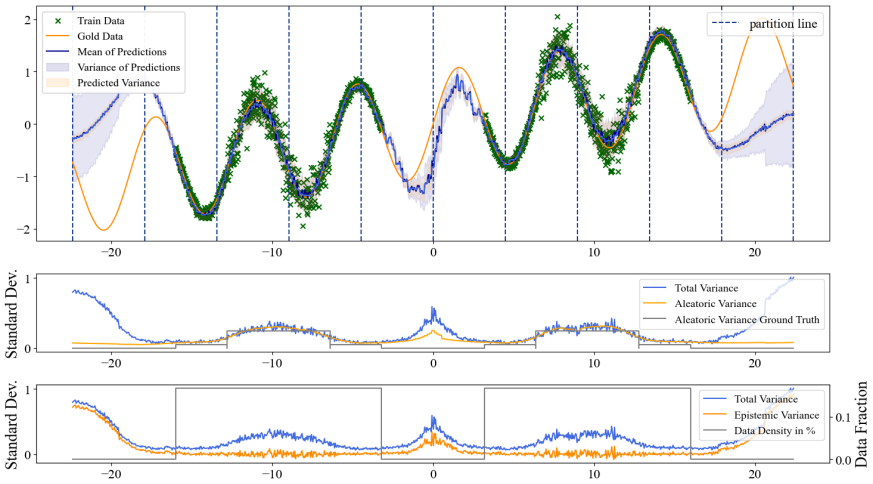


Figure 6.9: Noisy Sine Wave quantized by fully integrated quantization method

Method	MSE ($\times 10^{-4}$) ↓	NLL ↓	ELBO ↓	Aleatoric MSE ($\times 10^{-4}$) ↓
Baseline	**3.38**	**-0.0151**	**-15.23**	2.41
Distribution sampling values	3.89	-0.0117	-9.45	**2.38**
Distribution parameters	3.86	-0.0119	-9.43	**2.38**
Integrated	3.91	-0.0119	-8.94	2.42

Table 6.1: Comparative analysis of quantization techniques on model performance with Noisy Sine Wave data(Training metrics)

aleatoric uncertainty. However, Integrated quantization is slightly worse than the baseline method in terms of aleatoric mean square error (MSE) (2.42×10^{-4}), which shows that the combination of quantization methods is not a simple linear accumulation, but has a more complex relationship.

The ELBO metrics in the table reveal a trade-off introduced by quantization. The baseline model achieves the best ELBO performance (-15.23), while all quantization methods show degraded ELBO values. The integrated quantization method has the most pronounced degradation with an ELBO of -8.94, which is approximately 41% higher than the baseline model. This suggests that while quantization may preserve or even enhance certain aspects of BNN performance like aleatoric uncertainty estimation, it can negatively impact the variational inference framework's overall optimization objective. As ELBO (Evidence Lower Bound) represents a lower bound on the model evidence, higher values (closer to zero) indicate worse approximation of the true posterior distribution. This degradation likely stems from the reduced expressivity of the quantized model, which constrains its ability to fully capture the complexity of the posterior distribution.

A particularly notable phenomenon observed across all quantized models is the introduction of significant non-smoothness in specific

regions of the input domain. Specifically, the predictions become markedly irregular in three distinct regions: $x \in (-22.4, -16) \cup (-3.2, 3.2) \cup (16, 22.4)$. This non-smoothness manifests as high-frequency oscillations in the prediction mean and amplified uncertainty bands, contrasting sharply with the baseline model's smooth predictions.

This region-specific degradation can be attributed to the interaction between quantization errors and data distribution characteristics. In the central region (-3.2,3.2), where the sine wave exhibits rapid changes in slope and curvature, quantization introduces substantial stair-stepping artifacts. The fixed-point representation cannot adequately capture the continuous gradient information, resulting in oscillatory behavior as the model attempts to approximate the underlying function with limited numerical precision.

The difference in fluctuation patterns between the two peripheral regions and the central area is primarily attributable to the input representation encoding method. The key factor is that only the central region's input features are non-zero due to the encoding scheme, causing even small value changes to have proportionally larger effects since other input features are zero. In contrast, as encoding extends from the center to both sides, other input features become non-zero, causing changes in input feature values to represent a smaller proportion of the overall signal. This difference in relative signal contribution significantly impacts how quantization errors manifest across different regions.

The three quantization approaches demonstrate distinctive characteristics in the noisy sine wave experiment:

- **Sample-Based Quantization Method:** This approach maintains reasonably good overall MSE (3.89×10^{-4}) and achieves excellent aleatoric uncertainty estimation (2.38×10^{-4}). The implementation simplicity makes it attractive for applications where computational efficiency is prioritized over absolute prediction smoothness.

- **Parameter-Based Quantization Method:** Applying quantization to distribution parameters achieves slightly better overall MSE (3.86×10^{-4}) compared to sampling value quantization while maintaining the same excellent aleatoric uncertainty estimation (2.38×10^{-4}). The logarithmic quantization of standard deviation parameters proves particularly effective in preserving uncertainty estimation quality.

- **Fully Integrated Quantization:** While this method has the highest overall MSE (3.91×10^{-4}) and slightly worse aleatoric uncertainty estimation (2.42×10^{-4}), it exhibits the worst ELBO performance (-8.94), approximately 41% higher than the baseline model. This highlights that the optimization objectives in varia-

tional inference are particularly sensitive to the combined effects of different quantization strategies.

Our analysis of three quantization approaches on the noisy sine wave regression task reveals that BNNs maintain resilience to quantization, particularly in uncertainty estimation. All methods showed only minor degradation in MSE performance compared to the baseline, with the distribution parameter quantization method achieving the best balance between prediction accuracy and implementation complexity.

A key finding is that all quantization methods significantly underperform the baseline on the ELBO metric, with integrated quantization showing the largest degradation (-8.94 vs. -15.23 for the baseline). This result suggests that the quantization process, while preserving point prediction accuracy and uncertainty estimation capabilities to a reasonable degree, substantially impacts the model's ability to approximate the true posterior distribution within the variational inference framework. The discretization introduced by quantization appears to limit the expressiveness of the approximate posterior, resulting in poorer ELBO scores. This trade-off between computational efficiency through quantization and optimal variational approximation represents an important consideration for deploying BNNs in resource-constrained environments. Additionally, distribution sampling and parameter quantization methods slightly improved aleatoric uncertainty estimation compared to the baseline, suggesting that appropriate quantization can actually enhance certain aspects of BNN performance despite its negative impact on ELBO.

The non-smoothness observed in specific input regions across all quantized models highlights the interaction between quantization errors and the input representation encoding scheme. This preliminary finding on the noisy sine wave dataset provides insights that will be further explored in the subsequent Two Moons and DirtyMNIST experiments.

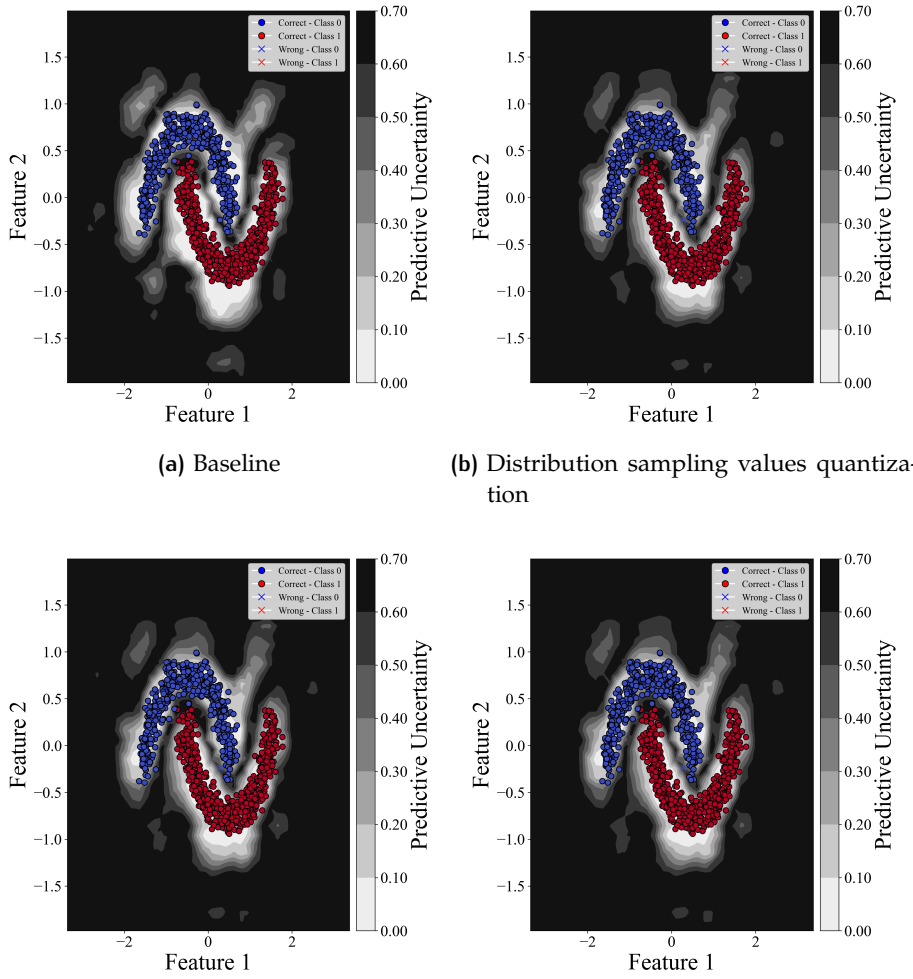
6.4.2 Two Moons

EXPERIMENT SETTING All experiments shared these settings:

- **Dataset:** 2,000 training samples, 2,000 testing samples with noise level 0.1, batch size 64, test range factor 2
- **Model:** MLP with input representation input, two hidden layers of 100 neurons each, alpha-sine activation (a modified sine function facilitating gradient flow in quantized networks, detailed in Section 7.1)

- **Training:** 300 epochs pretraining (lr=0.002), 1,000 epochs main training (lr=0.001), 1 particle for training, random_seed is set to 42
- **KL Annealing:** Maximum weight 0.1, with an end-focused schedule that keeps the KL loss nearly inactive in early epochs and ramps it up only near the end of training.

The model quantization parameters conform to the BSS-2 hardware specifications, with 5-bit unsigned representation for input quantization, 8-bit signed representation for output quantization, and 7-bit signed representation for weight quantization.



(a) Baseline

(b) Distribution sampling values quantization

(c) Distribution parameters Quantization

(d) Integrated Quantization

Figure 6.10: Comparison of uncertainty visualization for different quantization methods on the Two Moons dataset. All quantization approaches (b-d) result in more diffuse decision boundaries compared to the baseline (a), with uncertainty regions becoming less sharply defined as quantization is applied.

EXPERIMENT RESULTS AND ANALYSIS Analysis of Table (6.2) and the visualizations in Figure (6.10) reveals significant patterns in how different quantization methods affect the classification task on the Two Moons dataset.

The baseline model achieves perfect classification accuracy (1.0000) on in-domain data, accompanied by the highest unanimity score (0.9879) and lowest uncertainty (0.0432). This demonstrates that without quantization, the model clearly distinguishes between the two classes and exhibits high certainty in its predictions. As shown in Figure (6.10a), the baseline model displays appropriate uncertainty distribution at class boundaries while maintaining high certainty in intra-class regions.

Method	In-Domain			Out-Domain	
	Acc ↑	Unanimity ↑	Uncertainty	Unanimity ↑	Uncertainty
Baseline	1.0000	0.9879	0.0432	0.5965	0.6590
Distribution sampling values	0.9991	0.9794	0.0812	0.5910	0.6683
Distribution parameters	0.9991	0.9799	0.0804	0.5917	0.6683
Integrated	0.9991	0.9796	0.0807	0.5902	0.6684

Table 6.2: Comparative analysis of quantization techniques on model performance with Two Moons data

All three quantization methods maintain exceptionally high accuracy (0.9991) on in-domain data, only marginally lower than the baseline model, indicating the robust nature of Bayesian Neural Networks when facing quantization. However, these methods lead to slight decreases in unanimity scores (approximately 0.9794-0.9799) and increases in uncertainty (approximately 0.0804-0.0812), consistent with our observations from the noisy sine wave experiment.

For out-of-domain data, all models including the baseline show significantly reduced unanimity (approximately 0.59) and substantially increased uncertainty (approximately 0.66-0.67). This indicates that all models correctly identify out-of-domain data and express appropriately high uncertainty, a crucial characteristic of Bayesian Neural Networks. Notably, compared to the baseline, all quantization methods exhibit slightly higher uncertainty in out-of-domain regions, suggesting that quantization somewhat enhances the model's sensitivity to out-of-domain data.

The visualizations demonstrate that all quantization methods maintain uncertainty boundary shapes similar to the baseline model; however, they display wider transitions with more pronounced shading gradients in boundary regions. Particularly at class interfaces, quantization methods exhibit higher predictive uncertainty, evidenced by the broader and more granular uncertainty bands. This indicates that quantization noise has more significant effects in these complex regions where decision boundaries are less distinct. The uncertainty contours in the quantized models show more intricate patterns, re-

flecting how discretization influences the model's confidence in areas where class separation is most challenging.

Comparing the three quantization approaches:

- **Sample-Based Quantization Method:** Maintains good accuracy and unanimity in-domain, but exhibits slightly higher uncertainty (0.0812) than other methods. Figure (6.10b) shows this method produces wider uncertainty regions at decision boundaries.
- **Parameter-Based Quantization Method:** Performs very similarly to sampling value quantization on both in-domain and out-of-domain metrics, but with marginally higher unanimity (0.9799), suggesting that parameter quantization may provide slightly more stable predictions. Figure (6.10c) illustrates its uncertainty distribution at boundaries, which shows patterns similar to but subtly different from sampling value quantization.
- **Fully Integrated Quantization:** Combining the previous two approaches, this method performs best in terms of out-of-domain uncertainty (0.6684), indicating a slight advantage in detecting anomalous data.

Overall, the Two Moons experiment further confirms BNNs' ability to maintain strong performance under quantization, particularly regarding accuracy. All quantization methods exhibit similar performance characteristics, demonstrating BNNs' adaptability to different quantization strategies. However, quantization does affect the model's uncertainty estimation, making it more uncertain on in-domain data, which may require special consideration in certain applications.

6.4.3 DirtyMNIST

EXPERIMENT SETTING All experiments shared these settings:

- **Dataset:** DirtyMNIST is used for training, MNIST, AmbiguousMNIST, and FashionMNIST are used for testing, and the batch size is set to 100.
- **Model:** MLP with input representation obtained by shifting the scalar input to be non-negative (subtracting the minimum value) (based on our findings in Section 5.5), two hidden layers of 100 neurons each, SoftPlus activation (based on our findings in Section 7.1.3).
- **Training:** 1500 epochs pretraining (lr=0.001), 1,000 epochs main training (lr=0.0001), 1 particle for training, random_seed is set to 42.

- **KL Annealing:** Maximum weight 0.25, with an end-focused schedule that keeps the KL loss nearly inactive in early epochs and ramps it up only near the end of training.

The model quantization parameters conform to the BSS-2 hardware specifications, with 5-bit unsigned representation for input quantization, 8-bit signed representation for output quantization, and 7-bit signed representation for weight quantization. Due to the complexity of the DirtyMNIST dataset, applying input and output quantization to the final linear layer (before computing logits) introduces non-negligible quantization errors that significantly impact model performance. Therefore, in practice, the final linear layer maintains only weight quantization while removing input and output quantization to preserve classification accuracy and maintain a good uncertainty estimation. A detailed analysis of the impact of these quantization parameters on model performance can be found in Appendix (a.2).

EXPERIMENT RESULTS AND ANALYSIS Analysis of Table (7.5) and visualizations in Figure (6.11) reveals significant patterns in how different quantization methods affect classification performance and uncertainty estimation across various test distributions.

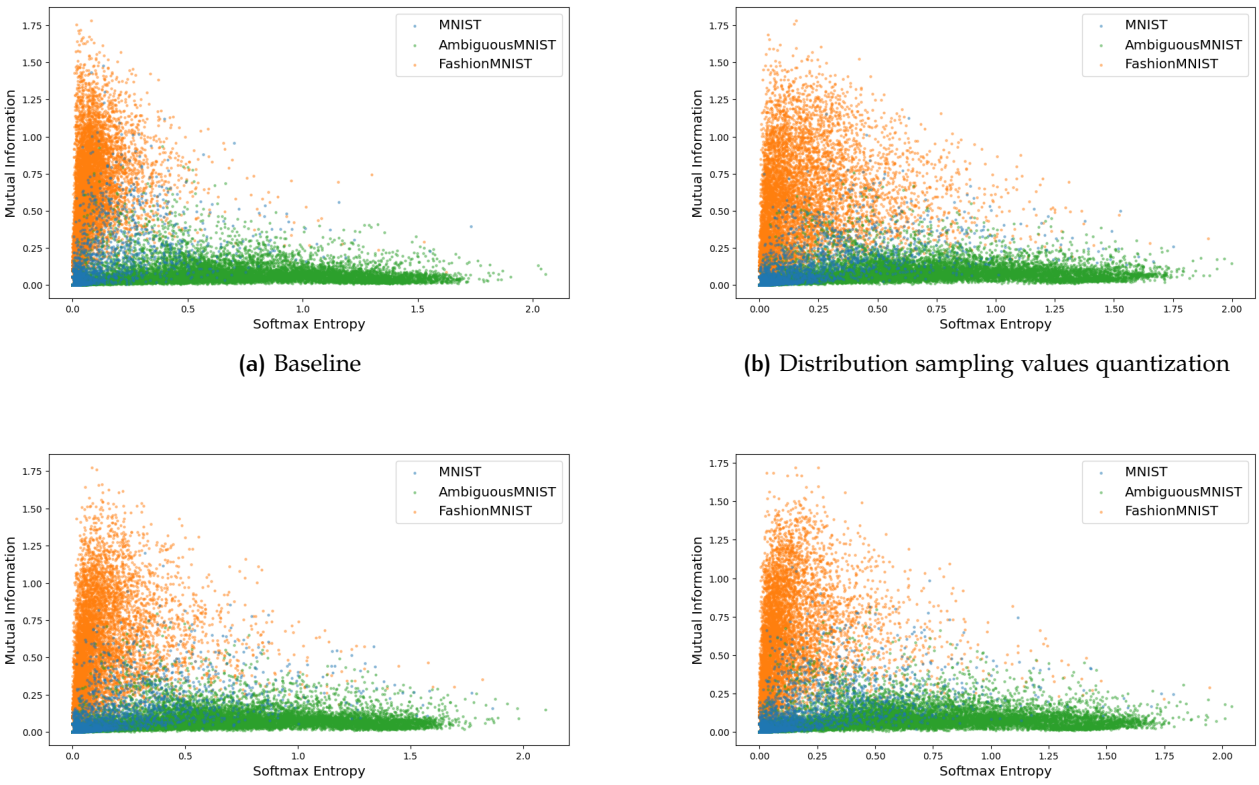
Method	MNIST					AmbiguousMNIST					FashionMNIST				
	A ↑	U ↑	P.U.	S.E.	M.I.	A ↑	U ↑	P.U.	S.E.	M.I.	A ↑	U ↑	P.U.	S.E.	M.I.
Baseline	0.98	0.98	0.05	0.02	0.03	0.51	0.73	0.70	0.63	0.07	0.09	0.77	0.56	0.07	0.49
Distribution sampling values	0.98	0.97	0.07	0.05	0.03	0.51	0.72	0.75	0.66	0.08	0.04	0.78	0.57	0.14	0.43
Distribution parameters	0.97	0.97	0.07	0.05	0.03	0.51	0.72	0.74	0.66	0.08	0.04	0.80	0.51	0.10	0.41
Integrated	0.97	0.97	0.07	0.05	0.03	0.51	0.72	0.75	0.67	0.08	0.05	0.80	0.51	0.10	0.42

Table 6.3: Performance comparison of quantization methods across different MNIST datasets. ↑ indicates that higher values are better. A: Accuracy, U: Unanimity, P.U.: Predictive Uncertainty, S.E.: Softmax Entropy, M.I.: Mutual Information. Results indicate that quantization errors simultaneously affect both aleatoric uncertainty (measured by Softmax Entropy) and epistemic uncertainty (measured by Mutual Information), with the most significant changes observed in the FashionMNIST dataset.

On the standard MNIST test set, the baseline model achieves excellent performance with 98% accuracy and unanimity, while all quantization methods show slight degradation with 97% accuracy but maintain 97% unanimity. This indicates that quantization has minimal impact on model performance for well-structured, in-distribution data. Notably, all quantization methods exhibit slightly higher predictive uncertainty (0.07 vs. 0.05) and softmax entropy (0.05 vs. 0.02) compared to the baseline, which is consistent with patterns observed in previous

experiments and suggests that quantization introduces a small degree of additional uncertainty even in well-defined classification regions.

For the AmbiguousMNIST dataset, all models including the baseline show substantially reduced accuracy (51%), reflecting the inherent classification difficulty of ambiguous samples. Importantly, the uncertainty decomposition reveals that all models correctly handle this challenge: unanimity scores remain relatively high (72-73%), while both predictive uncertainty and softmax entropy are appropriately elevated (0.70-0.75 and 0.63-0.67 respectively). The high aleatoric uncertainty (measured by softmax entropy) combined with relatively low epistemic uncertainty (mutual information 0.07-0.08) confirms that all models correctly attribute the classification difficulty to intrinsic data ambiguity rather than model knowledge limitations. Notably, quantization does not significantly alter this fundamental uncertainty decomposition behavior.



(a) Baseline

(b) Distribution sampling values quantization

(c) Distribution parameters Quantization

(d) Integrated Quantization

Figure 6.11: Scatter plots demonstrating how different quantization methods deteriorate cross-entropy on the Fashion MNIST dataset. The distribution sampling values quantization (b) shows more severe degradation compared to other methods, particularly affecting the relationship between softmax entropy and aleatoric uncertainty.

The most compelling insights emerge from the FashionMNIST results, where accuracy is expectedly low (4-9%) since FashionMNIST represents a genuinely out-of-distribution challenge. The critical finding is that all quantized models show higher mutual information (0.41-0.43) compared to the baseline (0.49), correctly identifying this as an epistemic uncertainty scenario. However, quantized models also exhibit substantially increased aleatoric uncertainty (softmax entropy 0.10-0.14) compared to the baseline (0.07), which is theoretically problematic since FashionMNIST represents an epistemic rather than aleatoric uncertainty challenge.

This inappropriate attribution of aleatoric uncertainty to an epistemic problem represents the most significant limitation of quantization in BNNs. The baseline model's uncertainty decomposition for FashionMNIST data (low softmax entropy 0.07, high mutual information 0.49) clearly distinguishes between the two uncertainty types, while quantized models blur this distinction by exhibiting elevated softmax entropy values. This degradation suggests that quantization impairs the model's ability to precisely attribute uncertainty to its proper source when facing domain-shift scenarios.

Comparing the three quantization approaches:

- **Sample-Based Quantization Method:** Performs well on MNIST and AmbiguousMNIST tests, with successful preservation of uncertainty decomposition for ambiguous data. However, for FashionMNIST, it shows the highest degree of uncertainty diffusion with elevated softmax entropy (0.14) and reduced mutual information (0.43), suggesting this method most significantly compromises uncertainty discrimination for out-of-distribution data.

- **Parameter-Based Quantization Method:** Demonstrates similar performance patterns but with slightly better preservation of epistemic uncertainty differentiation. The mutual information remains relatively high (0.41) for FashionMNIST while exhibiting moderate softmax entropy increase (0.10), indicating more stable uncertainty attribution than sample-based quantization.

- **Fully Integrated Quantization:** Shows the most conservative uncertainty profile, with mutual information (0.42) and moderate softmax entropy (0.10) for FashionMNIST data. This approach appears to strike the best balance between preserving epistemic uncertainty recognition and limiting inappropriate aleatoric uncertainty attribution for out-of-distribution scenarios.

The DirtyMNIST experiment conclusively demonstrates that while quantized BNNs maintain acceptable classification performance on in-distribution data and appropriately elevate uncertainty for both ambiguous and out-of-distribution samples, quantization fundamentally compromises the precision of uncertainty decomposition. This

degradation is most pronounced for out-of-distribution data, where quantized models inappropriately attribute epistemic uncertainty challenges to aleatoric sources. For safety-critical applications requiring reliable uncertainty quantification and clear distinction between knowledge limitations and data ambiguity, this represents a significant limitation that must be carefully considered when deploying quantized BNNs in resource-constrained environments.

6.5 SUMMARY OF QUANTIZATION METHODS IMPACT

Based on the experimental results from three different tasks (noisy sine wave regression, Two Moons classification, and DirtyMNIST classification), several consistent patterns emerge regarding the impact of quantization methods on Bayesian Neural Networks. All quantization approaches demonstrate remarkable resilience in maintaining prediction accuracy while revealing complex effects on uncertainty estimation, highlighting the fundamental tension between computational efficiency and probabilistic inference quality.

The most striking commonality across all experiments is the preservation of prediction accuracy. On the noisy sine wave task, quantization methods showed only modest MSE degradation ($3.86 - 3.9110^{-4}$) compared to the baseline (3.3810^{-4}), representing approximately 14-16% performance reduction. Similarly, in the Two Moons classification task, all quantization methods achieved 99.91% accuracy, virtually identical to the baseline's perfect performance. For DirtyMNIST, standard MNIST classification accuracy remained consistently high at 97-98% across all quantization approaches. This consistent preservation of prediction accuracy suggests that the discrete approximations introduced by quantization do not fundamentally compromise the core predictive capabilities of Bayesian Neural Networks.

A paradoxical finding emerges regarding uncertainty estimation, where quantization appears to enhance certain aspects while potentially degrading others. In regression tasks, both distribution sampling and parameter quantization methods actually improved aleatoric uncertainty modeling compared to the baseline (2.3810^{-4} vs 2.4110^{-4} aleatoric MSE). This suggests that quantization-induced noise can serve as a beneficial implicit regularization mechanism. However, in classification tasks, quantization consistently increased predictive uncertainty, particularly at decision boundaries, indicating a trade-off between prediction confidence and uncertainty calibration.

Perhaps the most significant limitation identified across all experiments is the degradation of uncertainty decomposition capabilities, particularly evident in the DirtyMNIST experiment. While baseline models clearly distinguished between aleatoric uncertainty (data am-

biguity) and epistemic uncertainty (model knowledge limitations), quantized models exhibited diffusion across both uncertainty types. For out-of-distribution FashionMNIST data, which should primarily evoke epistemic uncertainty, quantized models inappropriately elevated aleatoric uncertainty measures (softmax entropy 0.10-0.14 vs baseline 0.07), indicating impaired ability to attribute uncertainty to its proper source.

The impact on variational inference quality represents another consistent finding, with all quantization methods showing substantial ELBO degradation across tasks. In the noisy sine wave experiment, ELBO values deteriorated from -15.23 (baseline) to approximately -9.0 (quantized methods), suggesting that the discrete approximations fundamentally constrain the model's ability to capture the full complexity of the posterior distribution. This degradation appears to be a fundamental consequence of the expressivity limitations imposed by quantization, affecting the theoretical foundation of Bayesian inference rather than just practical performance metrics.

Method-specific characteristics emerged through comparative analysis, revealing distinct advantages and limitations of each quantization approach. Sample-based quantization offered the simplest implementation with reasonable performance trade-offs but exhibited the highest uncertainty diffusion in challenging scenarios. Parameter-based quantization, leveraging logarithmic quantization for standard deviation parameters, achieved the best balance between prediction accuracy and uncertainty preservation while maintaining implementation feasibility. The fully integrated approach, while combining both quantization strategies, showed the most pronounced ELBO degradation despite maintaining competitive practical performance, highlighting the nonlinear interaction effects between different quantization components.

These findings collectively suggest that while quantization enables efficient deployment of Bayesian Neural Networks in resource-constrained environments, it introduces fundamental limitations in uncertainty reasoning capabilities. The choice of quantization method should therefore be carefully aligned with application requirements, particularly when precise uncertainty decomposition is critical for safety-critical applications. Future research directions should focus on developing quantization-aware training strategies specifically designed to preserve uncertainty attribution capabilities while maintaining the computational efficiency advantages of quantized implementations.

7 OPTIMIZATION STRATEGIES FOR QUANTIZED SVI

In Chapter 6, we presented a qualitative performance analysis of Bayesian Neural Networks with different quantization levels based on input representation techniques, using three distinct datasets: Noisy Sine Wave, Two Moons, and DirtyMNIST. The results demonstrated the effectiveness of three different quantization approaches: Distribution Sampling Values quantization, Distribution Parameters Quantization, and Integrated Quantization Methods. While the quantized models achieved impressive performance in those experiments, it is important to note that this success was made possible through extensive optimization strategies that were applied but not detailed in chapter 6.

The promising results observed in Chapter 6 were not achieved without challenges. Quantizing Bayesian Neural Networks inherently introduces various complications, including potential degradation in predictive accuracy and uncertainty estimation quality. To achieve the performance levels reported in the previous chapter, we developed and implemented several specialized optimization techniques that are systematically presented in this chapter.

This chapter focuses on the optimization strategies that enabled the successful quantization results demonstrated previously. We provide a comprehensive examination of the modifications made to various model components, including specialized activation functions, refined loss function formulations, and enhanced training procedures specifically designed for quantized Bayesian Neural Networks. Through detailed analysis of these optimization approaches, we demonstrate how quantization errors can be minimized while maintaining high-quality uncertainty estimates, all within the constraints of reduced computational complexity. The research presented in this chapter not only explains the technical foundation behind the successful results in Chapter 6 but also provides valuable insights into the mechanisms of preserving uncertainty information during the quantization process. These findings offer important theoretical and practical guidance for future implementations of Bayesian inference on neuromorphic hardware.

To thoroughly investigate the impact of various optimization techniques on quantized BNNs, we primarily focus our exploratory work on the Noisy Sine Wave dataset. This dataset offers clear visualization opportunities that allow us to better understand the effects of each optimization strategy. The visualizations provide intuitive insights into how different modifications affect the model's predictive distribution,

uncertainty estimation, and overall performance under quantization constraints. Our comprehensive exploration encompasses several key aspects of BNN quantization, including activation function selection, weight clipping strategies, distribution-aware quantization approaches, outlier preservation techniques, and the stability of aleatoric uncertainty estimation in regression tasks.

7.1 EFFECT OF ACTIVATION FUNCTION ON INPUT REPRESENTATION

Bayesian Neural Networks (BNNs) demonstrate particular sensitivity to the choice of activation functions, which significantly impacts both performance and uncertainty estimation quality. While traditional neural networks also benefit from appropriate activation function selection, this aspect becomes even more critical in BNNs where activation functions must effectively propagate not only point estimates but also uncertainty information through the network.

Recent research, such as the work by Tempczyk et al. (2022), has demonstrated that thoughtful modifications to activation functions can substantially improve BNN performance. Specifically, they found that replacing standard ReLU with Leaky ReLU activations significantly enhanced BNN calibration and performance. Similarly, Fakhfakh and Chaari (2023) explored using Bayesian optimization to find optimal sparse neural network architectures with trainable activation functions, further highlighting the importance of activation function design in probabilistic neural networks.

When implementing quantized BNNs, the selection of appropriate activation functions becomes critically important in relation to input representation methods. Our sectional thermometer encoding transforms low-dimensional inputs into expanded feature representations that, when multiplied by weight matrices during forward propagation, can produce values with significantly increased magnitudes. This transformation creates a fundamental scale mismatch between the typical operating ranges of standard activation functions (often designed for normalized inputs) and the substantially larger values they receive in our quantized network architecture. Consequently, activation functions must be carefully selected or adapted to appropriately handle these amplified signals without losing the ability to capture meaningful non-linearities in the representation space.

Additionally, our experiments with DirtyMNIST revealed that when global minimum offset adjustments are applied to inputs, ReLU-based activation functions struggle to adapt effectively. This observation highlights another critical aspect of activation function selection: maintaining appropriate output properties for subsequent layer input quantization. Non-negative activation outputs are particularly valuable in

this context, as they simplify the quantization process for the next layer's inputs and help maintain representation stability throughout the network.

Standard activation functions that perform well in floating-point precision may exhibit problematic behaviors when operating on these amplified quantized values, including:

- **Input range saturation:** The amplification of quantized inputs pushes values into saturation regions of activation functions like sigmoid and tanh, causing most neurons to operate in regions with minimal gradient sensitivity and limited representational capacity.
- **Representation collapse:** When amplified quantized inputs predominantly fall in activation function saturation regions, the effective output space collapses to a small subset of possible values, significantly reducing the network's expressive power.
- **Uncertainty distortion:** In Bayesian networks, activation functions must preserve uncertainty information, but when inputs are amplified beyond optimal operating ranges, uncertainty propagation becomes distorted, leading to poor calibration and unreliable posterior estimates.
- **Inter-layer quantization challenges:** Activation functions that produce outputs with wide-ranging or negative values create additional complexity for input quantization in subsequent layers, potentially introducing further representation errors.

To address these challenges, we explored several modified activation functions specifically designed to counteract the input amplification effects in quantized BNNs while maintaining beneficial properties for inter-layer quantization. Our approach focused on three main objectives:

1. Rescaling amplified inputs to appropriate operating ranges.
2. Maintaining effective uncertainty representation throughout the network.
3. Generating non-negative outputs that facilitate stable quantization across network layers.

7.1.1 Alpha-Sigmoid

The selection of activation functions in quantized Bayesian Neural Networks (BNNs) significantly impacts both network performance and the quality of uncertainty estimation. Through our research, we have identified that activation functions play a crucial role in input

representation, particularly when dealing with quantized inputs. In quantized BNNs, continuous input values are typically converted to discrete representations, often mapped to ranges such as $[0, 2^k - 1]$ where k represents the bit width. When these quantized inputs interact with weight matrices during forward propagation, the resulting values are frequently amplified by several orders of magnitude, creating significant challenges for standard activation functions.

The standard Sigmoid function, when processing these amplified values from quantized inputs and weight interactions, faces substantial representation limitations. To address this challenge, we propose the Alpha-Sigmoid activation function:

$$\text{Alpha-Sigmoid}(x) = \sigma(\alpha x) = \frac{1}{1 + \exp(-\alpha x)} \tag{7.1}$$

Where α is a scaling parameter less than 1 (e.g., $\alpha = 0.1$). This simple modification yields several significant improvements:

- **Representation Correction**: By reducing input scale through the α parameter, Alpha-Sigmoid effectively counteracts the amplification effect resulting from quantized input-weight interactions, bringing most values back into the sensitive region of the Sigmoid function.
- **Enhanced Representation Capacity**: Alpha-Sigmoid improves the network's ability to distinguish between similar inputs, particularly near decision boundaries where fine discrimination is critical.
- **Quantization Level Redistribution**: The modified function makes more effective use of available quantization levels by distributing them more appropriately across the activation space.

Alpha-Sigmoid, through this input representation mechanism, enables networks to better handle quantized inputs, significantly improving representational power, particularly in decision boundary regions. Experimental results demonstrate that even simple fixed α values can substantially enhance the performance of quantized networks.

LEARNABLE ALPHA PARAMETER: Beyond fixed α values, we also explored making α a learnable parameter, allowing networks to automatically adjust the scaling factor most appropriate for specific quantization schemes. We propose a more generalized form:

$$f(x) = \gamma \cdot \sigma(\alpha \cdot x) + \beta \tag{7.2}$$

where α, β, and γ are all learnable parameters controlling input scaling, horizontal offset, and output scaling, respectively. This formulation enables the activation function to adapt to the specific input representation requirements of different network layers and tasks.

FIXED ALPHA-SIGMOID EXPERIMENT AND ANALYSIS To empirically validate the effectiveness of Alpha-Sigmoid in quantized neural networks, we designed a controlled experiment focused on input representation quality. We chose a noisy sine wave regression task as our benchmark, as it provides a clear metric to evaluate how well different activation functions preserve input information under quantization constraints.

Experiment Settings: All experiments shared these settings:

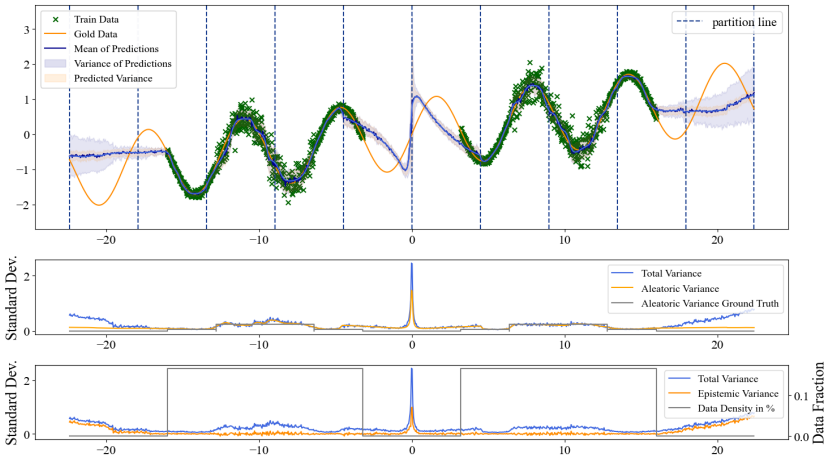
- **Dataset:** 2,000 training samples, 2,000 testing samples, batch size 64
- **Model:** MLP with input representation input, 50 neurons in hidden layer, Sigmoid activation and Alpha-Sigmoid activation($\alpha = 0.1$).
- **Training:** 300 epochs pretraining (lr=0.02), 1,000 epochs main training (lr=0.005), 1 particle for training, random_seed is set to 42.
- **KL Annealing:** Maximum weight 0.025, with an end-focused schedule that keeps the KL loss nearly inactive in early epochs and ramps it up only near the end of training.

Experiment Results and Analysis: Figure (7.1) presents a comparative visualization of model performance using standard Sigmoid and Alpha-Sigmoid activation functions on the noisy sine wave regression task. The visual evidence offers several important insights into the performance characteristics of both activation functions.

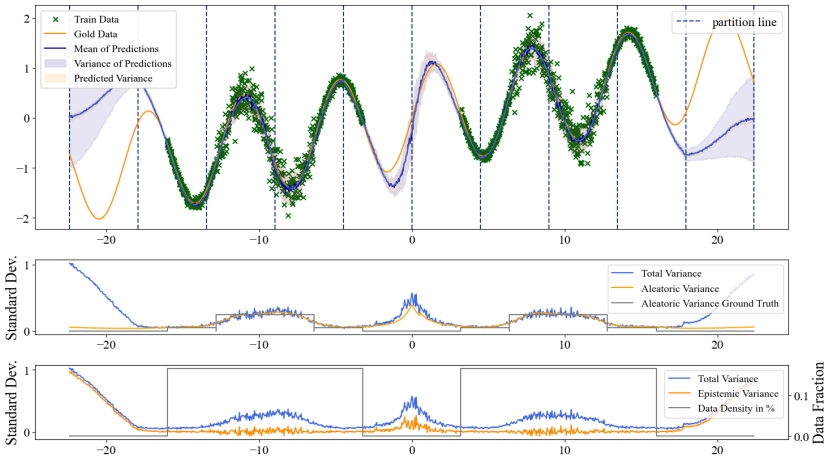
In Figure (7.1a) (standard Sigmoid), we observe that the prediction curve (green) generally follows the ground truth data (blue), but shows notable deviations in regions with complex patterns. The uncertainty estimates displayed in the bottom panels reveal significant variance spikes, particularly at $x \approx 0$, indicating potential instability in uncertainty quantification. Furthermore, in data-sparse regions (near boundaries $x \approx -20$ and $x \approx 20$), the standard Sigmoid configuration produces uncertainty estimates that do not adequately reflect the limited available information, showing a tendency toward overconfident predictions in these regions.

Contrastingly, Figure (7.1b) (Alpha-Sigmoid with $\alpha = 0.1$) demonstrates markedly improved performance characteristics. The prediction curve more closely adheres to the ground truth data across the entire input domain, particularly in regions with complex patterns. More importantly, the uncertainty patterns appear more distributed and consistent across the input range, with smoother variance transitions that better correspond to data density. In data-sparse regions, Alpha-Sigmoid produces appropriately higher uncertainty estimates, correctly reflecting the limited information available for making predictions in these areas.

The visual comparison provides clear evidence that Alpha-Sigmoid enables more reliable uncertainty quantification while maintaining or improving prediction accuracy. The smoother uncertainty transitions and better-calibrated uncertainty in data-sparse regions indicate that Alpha-Sigmoid effectively counteracts the input amplification effects typical in quantized networks, allowing the model to operate in more sensitive regions of the activation function.



(a) Sigmoid



(b) Alpha-Sigmoid

Figure 7.1: Comparison of different activation function configurations on noisy sine wave regression task: (a) Standard Sigmoid activation function; (b) Alpha-Sigmoid activation function. The plots show the prediction curves (green) against ground truth data (blue), along with corresponding uncertainty estimates (bottom panels). The Alpha-Sigmoid configuration provides more reasonable uncertainty estimates while maintaining prediction accuracy, particularly in out of domain regions.

Table (7.1) provides quantitative performance metrics that further substantiate the visual observations. The results reveal consistent and

significant improvements across all evaluation metrics when using Alpha-Sigmoid compared to standard Sigmoid activation.

Mean Squared Error (MSE) shows a substantial reduction from 5.76×10^{-4} with standard Sigmoid to 3.38×10^{-4} with Alpha-Sigmoid, representing a 41.3% improvement in prediction accuracy. This significant error reduction aligns with the visual observation that Alpha-Sigmoid enables the model to better capture the underlying data patterns.

The Negative Log-Likelihood (NLL) improves from -0.01421 to -0.01506, indicating that Alpha-Sigmoid produces better-calibrated probability distributions. This metric is particularly important in Bayesian Neural Networks as it reflects how well the model's predictive distribution matches the true data distribution.

Evidence Lower Bound (ELBO) shows a dramatic improvement, changing from 12.73 with standard Sigmoid to -15.23 with Alpha-Sigmoid. This substantial decrease represents a fundamental improvement in the variational approximation quality, as lower ELBO values (especially negative values) indicate a tighter bound on the marginal likelihood and thus a better model fit. This remarkable ELBO reduction suggests that Alpha-Sigmoid enables the network to achieve a variational approximation that much more closely resembles the true posterior distribution.

The Aleatoric MSE also shows improvement, decreasing from 2.63×10^{-4} to 2.40×10^{-4}, representing an 8.7% reduction. Unlike our previous observation with the preliminary data, this reduction suggests that Alpha-Sigmoid not only improves epistemic uncertainty modeling but also enhances the model's ability to capture intrinsic data noise. This dual improvement in both epistemic and aleatoric uncertainty estimation underscores the comprehensive benefits of the Alpha-Sigmoid activation function in quantized Bayesian Neural Networks.

Method	MSE ($\times 10^{-4}$) ↓	NLL ↓	ELBO ↓	Aleatoric MSE ($\times 10^{-4}$) ↓
Sigmoid	5.76	-0.01421	12.73	2.63
Alpha-Sigmoid($\alpha = 0.1$)	**3.38**	**-0.01506**	**-15.23**	**2.40**

Table 7.1: Comparative Analysis of Activation Functions on Model Performance of Noisy Sine Wave Data (Training Metrics). Alpha-Sigmoid consistently outperforms standard Sigmoid across all metrics, achieving approximately 41% reduction in MSE and significant improvements in both uncertainty representation (NLL, ELBO) and aleatoric uncertainty modeling.

Alpha-Sigmoid achieves performance improvements through three mechanisms:

1. Input rescaling via the α parameter counteracts amplification from quantized input-weight interactions, bringing values into the sensitive region of Sigmoid and addressing input saturation.

2. Operating in the sensitive region enhances representation capacity, improving discrimination between similar inputs at decision boundaries, directly reducing prediction error.

3. More appropriate distribution of quantization levels mitigates representation collapse where standard activations reduce effective output space to a small subset of possible values.

Our analysis demonstrates that Alpha-Sigmoid provides a simple yet effective solution to input representation challenges in quantized Bayesian Neural Networks. By introducing the scaling parameter α to counteract input amplification, Alpha-Sigmoid significantly enhances performance and uncertainty estimation.

LEARNABLE ALPHA-SIGMOID EXPERIMENT AND ANALYSIS Based on the good results of fixed Alpha-Sigmoid, we conducted a series of experiments to investigate whether making the parameters learnable can further improve the performance of quantized neural networks.

Experiment Settings: We maintained the same general settings as in the fixed parameter experiments:

- **Dataset:** 2,000 training samples, 2,000 testing samples, batch size 64.
- **Model:** MLP with input representation input, 50 neurons in hidden layer.
- **Training:** 300 epochs pretraining (lr=0.02), 1,000 epochs main training (lr=0.005), 1 particle for training, random_seed is set to 42.
- **KL Annealing:** Maximum weight 0.025, end-focused schedule.
- **Quantization settings:** 5-bit unsigned representation for input quantization, 8-bit signed representation for output quantization, and 7-bit signed representation for weight quantization. Distribution Sampling Values Quantization.

We systematically explored the following parameter configurations:

- **[alpha,beta,gamma]=fix**: Baseline configuration with all parameters fixed ($\alpha = 0.1$, $\beta = 0.0$, $\gamma = 1.0$).
- **[beta,gamma]=fix, [alpha]=learnable**: Only α is learnable, initialized at 0.1. β and γ are fixed to 0.0 and 1.0.
- **[gamma]=fix, [alpha,beta]=learnable**: Both α and β are learnable (initialized at $\alpha = 0.1$, $\beta = 0.0$). γ is fixed to 1.0.
- **[beta]=fix, [alpha,gamma]=learnable**: Both α and γ are learnable (initialized at $\alpha = 0.1$, $\gamma = 1.0$). β is fixed to 0.0.

- **[alpha,beta,gamma]=learnable**: Fully learnable configuration (initialized at $\alpha = 0.1$, $\beta = 0.0$, $\gamma = 1.0$).
- **[alpha]=fix, [beta,gamma]=learnable**: Both β and γ are learnable (initialized at $\beta = 0.0$, $\gamma = 1.0$. α is fixed to o.o.

Experiment Results and Analysis:

Figure (7.2) presents a comprehensive visual comparison across different Alpha-Sigmoid parameter configurations on the noisy sine wave regression task. The visualization reveals several critical insights into how parameter learnability affects model performance and uncertainty estimation.

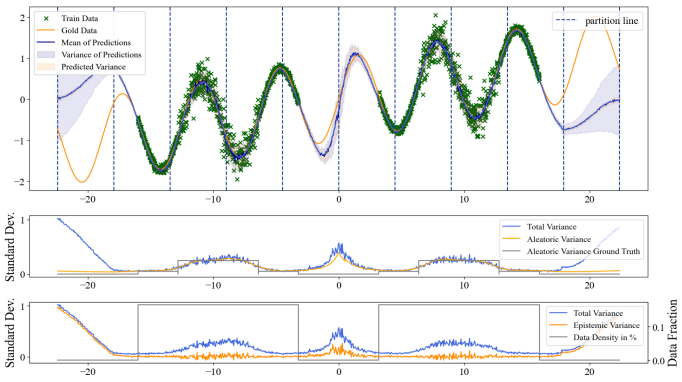
In Figure (7.2a) (all parameters fixed with $\alpha = 0.1$), we observe that the prediction curve closely follows the ground truth data across the entire input domain, demonstrating excellent fitting performance. The uncertainty estimates appear well-distributed, with appropriate variation corresponding to data density. This configuration serves as our baseline for comparison.

Examining Figure (7.2b) (only α learnable), we note comparable prediction accuracy to the fixed configuration, but with slightly different uncertainty characteristics. The uncertainty estimates show a more pronounced response in regions of data complexity, suggesting that the network has adapted the scaling parameter to better capture epistemic uncertainty in these critical regions.

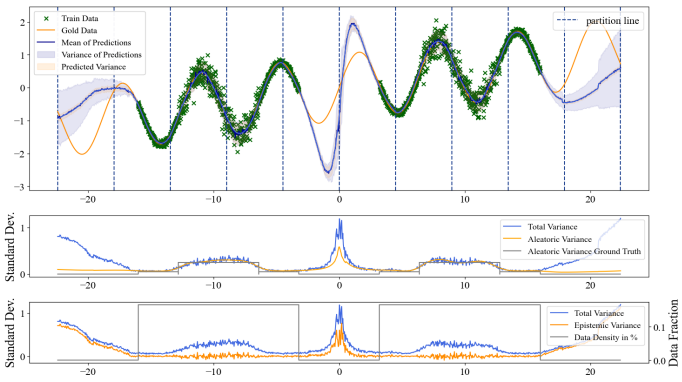
Figure (7.2c) (both α and β learnable) demonstrates that adding horizontal offset learnability results in modified uncertainty patterns. While the prediction curve maintains reasonable accuracy, the uncertainty estimation shows different characteristics, particularly in boundary regions, indicating that the horizontal offset parameter enables the model to adjust its response characteristics based on input position.

In Figure (7.2d) (both α and γ learnable), we observe that output scaling learnability introduces significant changes in uncertainty estimation patterns. This configuration shows distinct uncertainty characteristics compared to other variants, with notably reduced epistemic uncertainty in the middle regions of the input domain. This reduction in central uncertainty suggests that output scaling parameter γ enables the model to reallocate its uncertainty expression, potentially shifting focus from well-represented central regions to more challenging boundary areas, fundamentally altering how the model expresses predictive uncertainty across the input space.

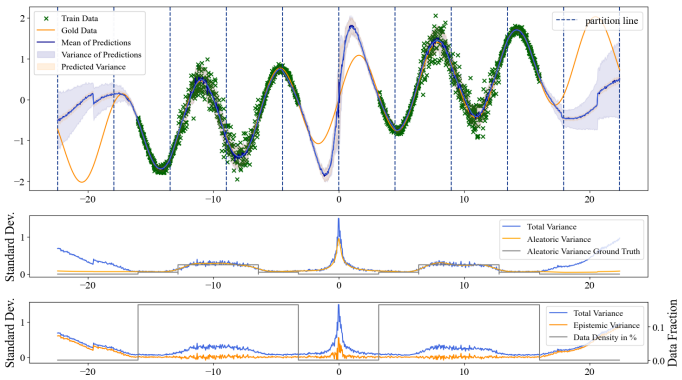
Figure (7.2e) (all parameters learnable) reveals that full parameter learnability leads to distinct prediction and uncertainty characteristics. While maintaining reasonable prediction accuracy overall, this configuration exhibits unique uncertainty patterns with more pronounced peaks in specific regions, indicating that the model has potentially overtrained the activation parameters to specific data patterns.



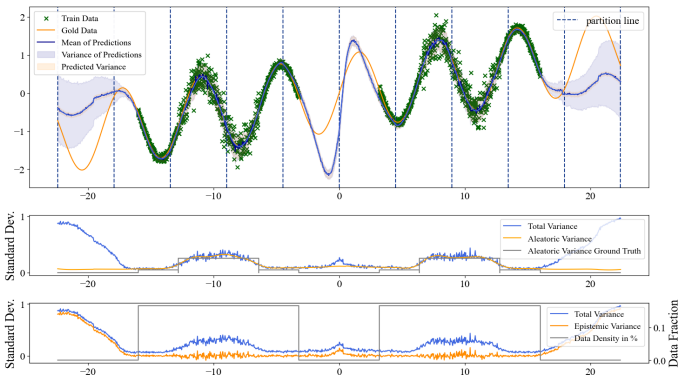
(a) All parameters fixed



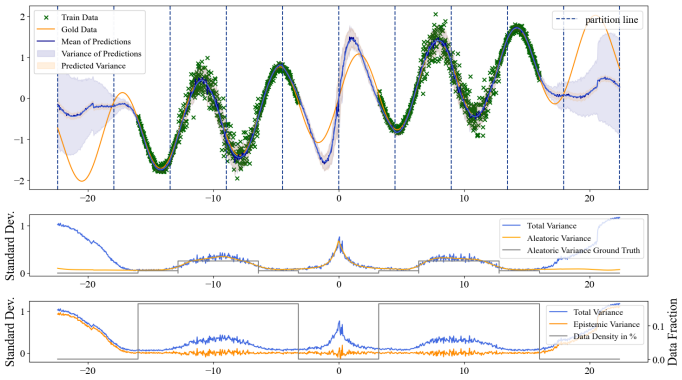
(b) Only Alpha learnable



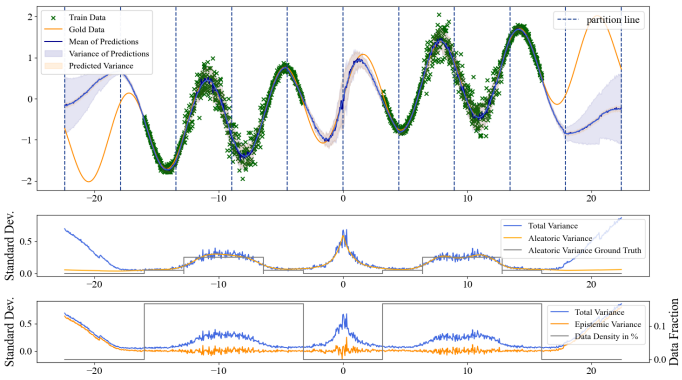
(c) Alpha and Beta learnable



(d) Alpha and Gamma learnable



(e) All parameters learnable



(f) Beta and Gamma learnable

Figure 7.2: Comparison of different Alpha-Sigmoid configurations on noisy sine wave regression task. (a) All parameters fixed; (b) Only Alpha learnable; (c) Alpha and Beta learnable; (d) Alpha and Gamma learnable; (e) All parameters learnable; (f) Beta and Gamma learnable. Results demonstrate that fixed parameters (a) provide smoother fitting across the domain. The Beta and Gamma learnable configuration (f) effectively captures both the curve and uncertainty in middle regions. Other learnable configurations improve uncertainty estimation but introduce larger fluctuations in middle regions. Notably, the Alpha and Gamma learnable configuration (d) results in diminished uncertainty in the middle region. These findings suggest different parameter learning strategies significantly impact model behavior, with specific combinations offering trade-offs between smoothness and uncertainty representation in quantized BNNs.

Figure (7.2f) (both β and γ learnable) shows the most effective capture of both curve characteristics and uncertainty in middle regions. This configuration maintains smoother predictions while still providing meaningful uncertainty estimates, suggesting that this particular combination of learnable parameters offers an advantageous trade-off between smoothness and uncertainty representation.

Method	MSE ($\times 10^{-4}$) ↓	NLL ↓	ELBO ↓	Aleatoric MSE ($\times 10^{-4}$) ↓
All parameters fixed	**3.38**	**-0.0151**	-15.23	**2.41**
Only Alpha learnable	3.91	-0.0119	-13.16	2.42
Alpha and Beta learnable	5.63	-0.0142	-14.23	2.51
Alpha and Gamma learnable	6.24	-0.0123	-28.36	2.71
All parameters learnable	6.26	-0.0130	**-28.95**	2.95
Beta and Gamma learnable	5.77	-0.0140	-12.61	2.60

Table 7.2: Comparative Analysis of Alpha-Sigmoid with Different Learnable Parameter Configurations on Model Performance Using Noisy Sine Wave Data (Training Metrics Evaluation)

Table (7.2) provides quantitative performance metrics that offer further insights into the relative performance of each configuration. Interestingly, the results reveal a complex relationship between parameter learnability and overall performance.

Mean Squared Error (MSE) results show that the fixed parameter configuration achieves the lowest error at 3.38×10^{-4}, outperforming all learnable variants. The fully learnable configuration exhibits an MSE of 6.26×10^{-4}, while the Beta and Gamma learnable configuration shows a moderate performance degradation at 5.77×10^{-4}. This suggests that parameter learnability, while offering theoretical adaptability advantages, may lead to suboptimal solutions for direct prediction accuracy in this specific task.

The Negative Log-Likelihood (NLL) results align with the MSE findings, with the fixed parameter configuration achieving the best score of -0.0151, indicating superior probability distribution calibration. The learnable configurations show varying degrees of performance degradation in this metric, with the only Alpha learnable configuration exhibiting the worst performance at -0.0119.

Evidence Lower Bound (ELBO) results present a contrasting pattern. Here, the fully learnable configuration achieves the best score at -28.95, followed closely by the Alpha and Gamma learnable configuration at -28.36. This significant improvement over the fixed configuration (-15.23) indicates that learnable parameters enable a tighter bound on the marginal likelihood, suggesting a better variational approximation despite worse direct prediction metrics. This apparent contradiction highlights the complex interplay between prediction accuracy and uncertainty representation quality in Bayesian Neural Networks.

Aleatoric MSE results show the fixed parameter configuration maintaining the best performance at 2.41×10^{-4}, with only minor degradation in the only Alpha learnable configuration at 2.42×10^{-4}. The more complex learnable configurations show more substantial increases in aleatoric MSE, with the fully learnable configuration exhibiting the highest value at 2.95×10^{-4}, suggesting that excessive parameter flexibility may compromise the model's ability to accurately capture intrinsic data noise.

Our analysis reveals several important insights regarding learnable activation function parameters in quantized Bayesian Neural Networks:

1. **Prediction-Uncertainty Tradeoff**: While fixed parameter configurations generally achieve better direct prediction metrics (MSE, NLL), learnable parameters significantly improve variational approximation quality (ELBO), suggesting a fundamental tradeoff between prediction accuracy and uncertainty representation.

2. **Configuration-Specific Advantages**: Different learnable configurations exhibit distinct performance characteristics, indicating that the optimal choice depends on whether the primary goal is prediction accuracy or uncertainty quality.

3. **Parameter Interaction Effects**: The complex relationships between performance metrics across configurations suggest significant interaction effects among the parameters, highlighting the importance of considering their combined impact rather than individual contributions.

4. **Task Dependency**: The observation that even empirically determined "optimal" fixed values may underperform suggests strong task dependency, reinforcing the potential value of learnable parameters that can adapt to specific task characteristics.

In summary, our investigation into learnable Alpha-Sigmoid parameters reveals a nuanced picture where parameter learnability offers significant benefits for uncertainty representation quality while potentially compromising direct prediction accuracy. This tradeoff highlights the importance of carefully considering the specific requirements of the application when selecting between fixed and learnable parameter configurations in quantized Bayesian Neural Networks. The findings emphasize that learning activation function parameters alongside the model can achieve more accurate uncertainty representation while maintaining acceptable prediction accuracy, particularly in the challenging context of quantized BNNs' input representation process.

7.1.2 Alpha-Sine

While investigating input representation with Sectional Thermometer Encoding (5.3), we discovered that Sine activation functions face unique challenges in classification tasks, particularly for datasets requiring clear decision boundaries (such as the Two Moons dataset). The standard Sine function presents two major issues when processing quantized inputs:

1. **Periodicity-Induced Representation Confusion**: When quantized inputs are mapped to wider ranges (e.g., 5-bit unsigned quantization's [0,31]), the periodicity of the standard Sine function (repeating every 2π) causes different input values to produce identical or highly similar outputs, creating confusion in the representation space.

2. **Insufficient Discrimination After Quantization**: In quantized environments, the differences between Sine function outputs for adjacent quantization levels may be too small, especially near the function's extremes, reducing the model's ability to distinguish between different inputs.

To address these issues, we introduce the Alpha-Sine activation function:

Where α is a value significantly less than 1 (e.g., $\alpha = 0.1$). This modification provides several key advantages for quantized input representation:

- **Effective Period Extension**: By reducing α, Alpha-Sine effectively extends the function's period from 2π to $2\pi/\alpha$. With $\alpha = 0.1$, the period increases to approximately 20π, meaning the function remains far from completing a full cycle across the entire mapped [0,31] range, greatly reducing representation confusion caused by periodicity.

- **Enhanced Linear Region**: Smaller α values cause the Sine function to approximate linearity over a larger input range, improving the network's ability to maintain topological relationships in input representation—critical for classification tasks like Two Moons that require preserving data topology.

- **Quantization Precision Redistribution**: Alpha-Sine more effectively utilizes available quantization levels by distributing them across a narrower output range, improving discriminability between similar input values.

From a theoretical perspective, Alpha-Sine fundamentally enhances the representational capacity of quantized inputs by altering the input

mapping approach. For the standard Sine function, any inputs differing by $2\pi n$ (where n is an integer) will produce identical outputs:

$$\sin(x) = \sin(x + 2\pi n) \tag{7.4}$$

For Alpha-Sine, this relationship becomes:

$$\sin(\alpha x) = \sin(\alpha x + 2\pi n) = \sin(\alpha(x + \frac{2\pi n}{\alpha})) \tag{7.5}$$

This means that the inputs must differ by $\frac{2\pi}{\alpha}$ to produce the same output. When $\alpha = 0.1$, the inputs need to differ by about 20π to cause representation conflicts, which is far beyond the actual input range used in the Sectional Thermometer Encoding scheme. Experiments on classification datasets such as Two Moons show that even with a fixed small value of α, Alpha-Sine can significantly improve model classification performance under quantization while maintaining computational simplicity.

LEARNABLE ALPHA PARAMETER: Similar to our approach with Alpha-Sigmoid, we also explored a learnable parameter version of Alpha-Sine. The generalized form can be expressed as:

$$f(x) = \gamma \cdot \sin(\alpha \cdot x) + \beta \tag{7.6}$$

where α, β, and γ are learnable parameters that control input scaling, vertical offset, and output scaling, respectively. This parameterization allows the activation function to adaptively adjust to the specific requirements of different network layers and classification tasks, further enhancing the model's representational capacity under quantization constraints.

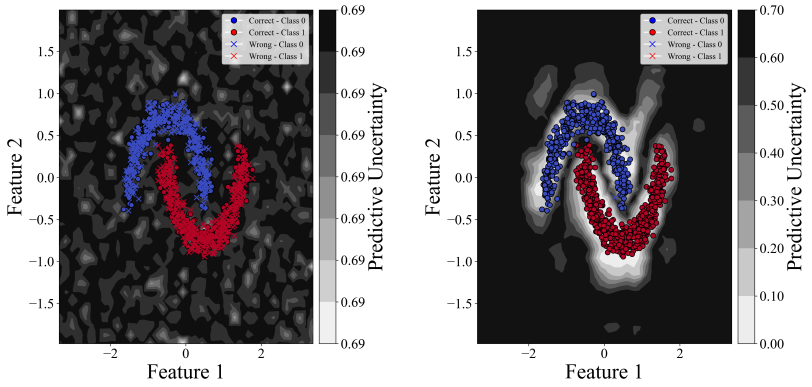
FIXED ALPHA–SINE EXPERIMENTS AND ANALYSIS To empirically verify the effectiveness of Alpha-Sine in quantized neural networks, we designed a controlled experiment focusing on the quality of input representation. We chose a Two Moons task as a benchmark because the sine activation function can achieve good results on the two moon dataset without any special processing.

Experiment Settings: All experiments shared these settings:

- **Dataset:** 2,000 training samples, 2,000 testing samples with noise level 0.1, batch size 64, test range factor 2.
- **Model:** MLP with input representation, two hidden layers of 100 neurons each, Sine and Alpha-Sine activation.
- **Training:** 300 epochs pretraining (lr=0.002), 1,000 epochs main training (lr=0.001), 1 particle for training, random_seed is set to 42.

- **KL Annealing:** Maximum weight 0.1, with an end-focused schedule that keeps the KL loss nearly inactive in early epochs and ramps it up only near the end of training.

Experiment Results and Analysis: Figure (7.3) and Table (7.3) reveal the substantial advantages of Alpha-Sine over standard Sine activation in quantized settings for the Two Moons classification task.



(a) Sine **(b)** Alpha-Sine

Figure 7.3: Comparison between standard sine and Alpha-Sine activation functions for Two Moons classification with input representation (a) Use Sine as activation function. (b) Use Alpha-Sine($\alpha = 0.1$) as the activation function.

Visually, standard Sine (Figure 7.3a) produces uniform uncertainty (≈ 0.69) throughout the feature space, failing to distinguish between decision boundaries and confident regions. In contrast, Alpha-Sine (Figure 7.3b) generates precise decision boundaries with appropriate uncertainty gradients—low in class-dense areas and elevated at boundaries.

Quantitatively, Alpha-Sine achieves perfect in-domain accuracy (1.0000 vs. 0.4703), near-perfect ensemble unanimity (0.9879 vs. 0.5395), and significantly lower uncertainty for known data (0.0432 vs. 0.6926). Crucially, while standard Sine shows identical uncertainty for both in-domain and out-domain data (0.6926), Alpha-Sine appropriately increases uncertainty from 0.0432 to 0.6590 for unfamiliar inputs—demonstrating proper epistemic uncertainty behavior.

Method	In-Domain			Out-Domain	
	Acc ↑	Unanimity ↑	Uncertainty	Unanimity ↑	Uncertainty
Sine	0.4703	0.5395	0.6926	0.5397	0.6926
Alpha-Sine	1.0000	0.9879	0.0432	0.5965	0.6590

Table 7.3: Comparative analysis of activation functions on the performance of models based on input representation for Two Moons data

The performance gap stems from standard Sine's problematic 2π periodicity, which creates representational ambiguity in quantized settings. By applying a scaling factor ($\alpha = 0.1$), Alpha-Sine stretches the function, dramatically improving input separability and quantization efficiency without increasing computational cost—a critical advantage for resource-constrained neuromorphic implementations.

LEARNABLE ALPHA-SINE EXPERIMENT AND ANALYSIS Based on the success of fixed Alpha-Sine and our promising results with learnable Alpha-Sigmoid, we conducted a comprehensive investigation to determine whether making the Alpha-Sine parameters learnable could further enhance the performance of quantized neural networks in classification tasks.

Experiment Settings: All experiments shared these settings:

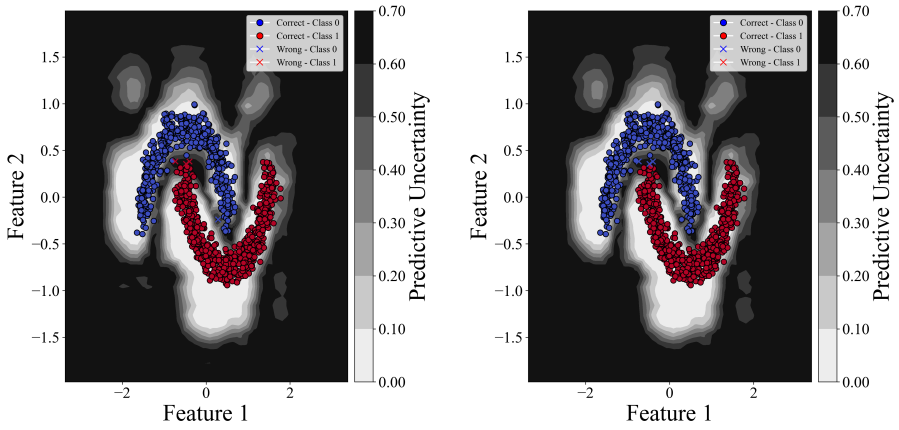
- **Dataset:** 2,000 training samples, 2,000 testing samples with noise level 0.1, batch size 64, test range factor 2
- **Model:** MLP with input representation, two hidden layers of 100 neurons each, Sine and Alpha-Sine activation.
- **Training:** 300 epochs pretraining (lr=0.002), 1,000 epochs main training (lr=0.001), 1 particle for training, random_seed is set to 42
- **KL Annealing:** Maximum weight 0.1, end-focused schedule with increase factor 10.0

We systematically explored the following parameter configurations:

- **[alpha,beta,gamma]=fix**: Baseline configuration with all parameters fixed ($\alpha = 0.1$, $\beta = 0.0$, $\gamma = 1.0$).
- **[beta,gamma]=fix, [alpha]=learnable**: Only α is learnable, initialized at 0.1. β and γ are fixed to 0.0 and 1.0.
- **[gamma]=fix, [alpha,beta]=learnable**: Both α and β are learnable (initialized at $\alpha = 0.1$, $\beta = 0.0$). γ is fixed to 1.0.
- **[beta]=fix, [alpha,gamma]=learnable**: Both α and γ are learnable (initialized at $\alpha = 0.1$, $\gamma = 1.0$). β is fixed to 0.0.
- **[alpha,beta,gamma]=learnable**: Fully learnable configuration (initialized at $\alpha = 0.1$, $\beta = 0.0$, $\gamma = 1.0$).
- **[alpha]=fix, [beta,gamma]=learnable**: Both β and γ are learnable (initialized at $\beta = 0.0$, $\gamma = 1.0$). α is fixed to 0.1.

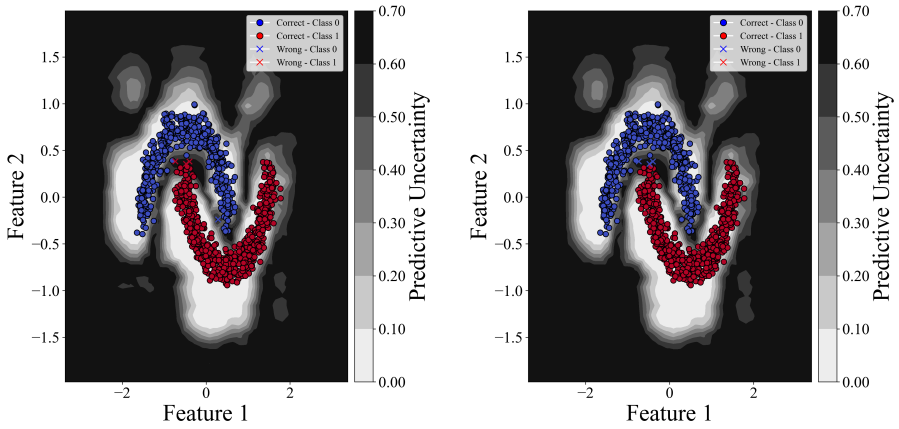
Experiment Results and Analysis:

In Figure (7.4) (all parameters fixed with $\alpha = 0.1$), we observe that the model establishes a clear decision boundary between the



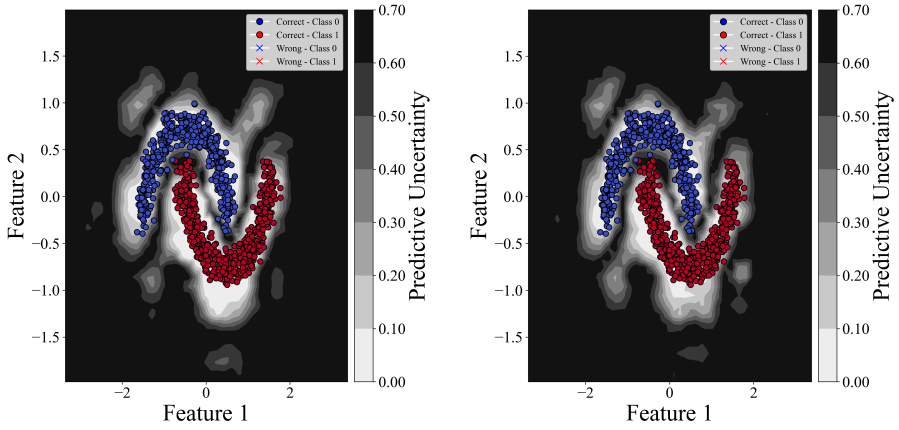
(a) All parameters learnable

(b) Only Alpha learnable



(c) Alpha and Beta learnable

(d) Alpha and Gamma learnable



(e) All parameters fixed

(f) Beta and Gamma learnable

Figure 7.4: Comparison of different Alpha-Sine configurations on Two Moons classification task. (a) All parameters fixed; (b) Only Alpha learnable; (c) Alpha and Beta learnable; (d) Alpha and Gamma learnable; (e) All parameters learnable; (f) Beta and Gamma learnable. All configurations establish clear decision boundaries, with fixed parameters (e) and Beta-Gamma learnable (f) achieving perfect accuracy and well-defined uncertainty gradients. Other learnable configurations (a,b,c,d) show slightly higher in-domain uncertainty while maintaining similar boundary clarity. Alpha-Sine effectively resolves periodicity-induced confusion in all configurations, concentrating uncertainty at class boundaries while minimizing it in class-dense regions.

two classes with minimal uncertainty in class-dense regions. This configuration achieves perfect classification accuracy and exhibits well-defined uncertainty gradients that peak precisely at the class boundaries, demonstrating robust discriminative capability while appropriately expressing uncertainty in ambiguous regions.

Examining Figure (7.4b) (only α learnable), we note a slight increase in in-domain uncertainty compared to the fixed configuration, with classification accuracy remaining high at 99.73%. The learnable scaling parameter allows the model to adapt its activation response to optimize the trade-off between decision confidence and uncertainty representation, resulting in a marginally different uncertainty distribution while maintaining clear class separation.

Figure (7.4c) (both α and β learnable) demonstrates similar classification performance to the alpha-only learnable configuration, with 99.65% accuracy. The addition of horizontal offset learnability introduces minor modifications to the uncertainty landscape, maintaining concentrated uncertainty at class boundaries while showing slightly increased in-domain uncertainty. This configuration suggests that horizontal offset adaptability provides limited additional benefit for this particular classification task.

In Figure (7.4d) (both α and γ learnable), we observe classification performance matching the alpha-only learnable configuration at 99.73% accuracy. The output scaling learnability shows distinct effects on uncertainty distribution, particularly at class boundaries, indicating that output scaling parameter γ enables fine-tuned control over confidence levels at decision boundaries while maintaining classification accuracy.

Figure (7.4e) (all parameters learnable) reveals that full parameter learnability leads to classification performance comparable to other learnable configurations at 99.65% accuracy. This configuration exhibits a balanced uncertainty distribution with appropriate concentration at class boundaries, demonstrating that the model can effectively utilize all three parameters to optimize its decision boundaries and uncertainty representation simultaneously.

Figure (7.4f) (both β and γ learnable) presents a particularly interesting case, achieving perfect classification accuracy (100%) while maintaining well-defined uncertainty gradients. This configuration, along with the fixed parameter baseline, shows superior in-domain performance, suggesting that maintaining a fixed scaling parameter while allowing horizontal and vertical adaptability provides an optimal balance for the Two Moons classification task.

Table (7.4) provides quantitative performance metrics that offer further insights into the relative performance of each configuration across both in-domain and out-of-domain scenarios.

For in-domain performance, the fixed parameter and Beta-Gamma learnable configurations achieve perfect classification accuracy (100%),

Method	In-Domain			Out-Domain	
	Acc ↑	Unanimity ↑	Uncertainty	Unanimity ↑	Uncertainty
All parameters learnable	0.9965	0.9808	0.0673	0.6131	0.6447
Only Alpha learnable	0.9973	0.9811	0.0666	0.6134	0.6443
Alpha and Beta learnable	0.9965	0.9809	0.0666	0.6132	0.6446
Alpha and Gamma learnable	0.9973	0.9807	0.0672	0.6128	0.6443
All parameters fixed	1.0000	0.9879	0.0432	0.5965	0.6590
Beta and Gamma learnable	1.0000	0.9824	0.0641	0.5897	0.6638

Table 7.4: Comparative Analysis of Activation Functions with Different Learnable Parameter Configurations on Model Performance Based on Two Moons Data Input Representation

outperforming all other learnable variants. These configurations also demonstrate superior unanimity scores (98.79% and 98.24% respectively), indicating higher decision consistency across multiple forward passes. However, the fixed parameter configuration shows notably lower in-domain uncertainty (0.0432) compared to all learnable configurations, with the Beta-Gamma learnable configuration exhibiting an uncertainty level (0.0641) more consistent with other learnable variants. This suggests that while the fixed parameter configuration maximizes in-domain confidence, learnable parameters introduce a more conservative uncertainty estimation that may better reflect the inherent ambiguity at decision boundaries.

For out-of-domain performance, a different pattern emerges. The learnable configurations demonstrate higher unanimity scores (approximately 61.3%) compared to fixed parameter and Beta-Gamma learnable configurations (59.65% and 58.97% respectively). Correspondingly, the learnable configurations show slightly lower out-of-domain uncertainty (approximately 64.4%) compared to fixed parameter and Beta-Gamma learnable configurations (65.90% and 66.38% respectively). This inverse relationship between out-of-domain unanimity and uncertainty highlights the fundamental trade-off between decision consistency and uncertainty representation in extrapolation scenarios.

Our analysis reveals several important insights regarding learnable activation function parameters in classification tasks using quantized neural networks:

1. **Accuracy-Uncertainty Balance**: Fixed parameter and Beta-Gamma learnable configurations achieve superior in-domain classification accuracy and unanimity, but at the cost of potentially underestimating uncertainty. Fully learnable configurations maintain excellent accuracy while providing more conservative uncertainty estimates that may better reflect epistemic uncertainty.

2. **Domain Adaptation Trade-offs**: Learnable parameters demonstrate superior out-of-domain unanimity but lower uncertainty compared to fixed configurations, suggesting different calibra-

tion behaviors in extrapolation scenarios. This highlights the challenge of balancing in-domain performance with out-of-domain reliability.

3. **Parameter-Specific Effects**: The Beta-Gamma learnable configuration uniquely combines the high accuracy of fixed parameters with uncertainty characteristics more similar to learnable configurations, suggesting that this specific combination offers an advantageous balance for classification tasks.

4. **Boundary Precision**: All configurations effectively concentrate uncertainty at class boundaries while minimizing it in class-dense regions, with Alpha-Sine successfully resolving the periodicity-induced confusion typical of standard sine activations. This demonstrates the fundamental advantage of the Alpha-Sine formulation regardless of parameter learnability.

In summary, our investigation into learnable Alpha-Sine parameters reveals that while fixed parameters may achieve marginally better in-domain classification metrics, learnable parameters offer valuable advantages in uncertainty representation and potentially out-of-domain robustness. The Beta-Gamma learnable configuration emerges as a particularly promising approach, combining the strengths of both fixed and learnable paradigms to achieve perfect accuracy while maintaining appropriate uncertainty gradients. These findings suggest that selectively learnable activation function parameters can provide an optimal balance between classification performance and uncertainty representation in quantized neural networks.

7.1.3 SoftPlus Activation

While exploring activation functions for quantized BNNs, we identified a critical limitation of ReLU in scenarios involving global minimum offset adjustments. When processing DirtyMNIST data, we observed that applying an offset to ensure non-negative inputs creates a problematic interaction with ReLU activation. Since ReLU simply passes through all positive values linearly, it fails to appropriately rescale these offset-adjusted inputs, effectively transferring the artificial offset directly to subsequent layers without meaningful transformation. This leads to representation distortion throughout the network, as the offset-induced shift remains unaddressed by ReLU's simplistic non-negative passing mechanism.

This limitation prompted us to investigate SoftPlus as an alternative activation function that could better handle offset-adjusted inputs while maintaining the beneficial property of non-negative outputs.

SoftPlus, defined as $\text{SoftPlus}(x) = \ln(1 + e^x)$, offers several advantages specific to quantized BNNs with input representation considerations:

$$\text{SoftPlus}(x) = \ln(1 + e^x) \tag{7.7}$$

Unlike ReLU, which abruptly transitions at $x = 0$ with a non-differentiable point, SoftPlus provides a smooth approximation of the ReLU function while maintaining non-negative outputs. This smoothness is particularly beneficial in quantized networks, where discontinuities can exacerbate representation issues when inputs cluster around transition points. The gradual transition from near-zero to linear behavior allows SoftPlus to meaningfully transform offset-adjusted inputs rather than simply passing them through.

When inputs processed through Sectional Thermometer Encoding undergo matrix multiplication with weight parameters, the resulting amplified values can be effectively processed by SoftPlus without the saturation issues that plague Sigmoid or Tanh functions. SoftPlus maintains a linear-like behavior for large positive inputs while smoothly transitioning to near-zero (but still positive) outputs for negative inputs, providing appropriate scaling across the entire input spectrum.

The non-negative nature of SoftPlus outputs also simplifies the quantization process for subsequent layers, as the quantization scheme need only account for a one-sided distribution. This property becomes especially valuable when global minimum offset adjustments are applied to inputs, as observed in our DirtyMNIST experiments, where maintaining consistent representation across layers is crucial.

In Bayesian Neural Networks, where uncertainty propagation is critical, SoftPlus further demonstrates favorable properties. The function's smooth nature allows for gradual uncertainty diffusion, avoiding the sharp cutoffs in uncertainty representation that can occur with ReLU when values fall exactly at or near zero. This leads to more reliable uncertainty estimates throughout the network, contributing to better overall model calibration. Our empirical evaluation on DirtyMNIST confirmed that SoftPlus significantly outperforms ReLU when global minimum offset adjustments are applied, providing more stable inter-layer representations and better classification accuracy. These results highlight that activation function selection in quantized BNNs must consider not only immediate transformative properties but also how these functions interact with quantization adjustments throughout the entire network.

SOFTPLUS EXPERIMENTS AND ANALYSIS To empirically verify the effectiveness of SoftPlus in quantized neural networks, we designed a controlled experiment focusing on the quality of input representation. Although the Relu activation function can achieve good performance at full precision, it experiences dramatic performance degradation

under any form of quantization. Given this issue, we explore replacing Relu with SoftPlus activation functions in quantized versions, and conduct experimental exploration and analysis specifically for Sample-Based Quantization Method on the DirtyMNIST dataset. **Experiment Settings:** All experiments shared these settings:

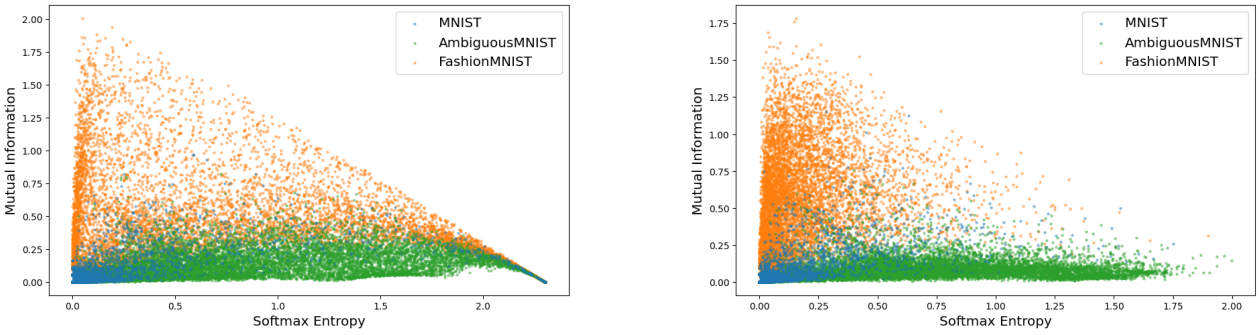
- **Dataset:** DirtyMNIST is used for training, MNIST, AmbiguousMNIST, and FashionMNIST are used for testing, and the batch size is set to 100.
- **Model:** MLP with input representation obtained by shifting the scalar input to be non-negative (subtracting the minimum value), two hidden layers of 100 neurons each, Relu activation and SoftPlus activation.
- **Training:** 1500 epochs pretraining (lr=0.001), 1,000 epochs main training (lr=0.0001), 1 particle for training, random_seed is set to 42.
- **KL Annealing:** Maximum weight 0.25, with an end-focused schedule that keeps the KL loss nearly inactive in early epochs and ramps it up only near the end of training.

The model quantization parameters conform to the BSS-2 hardware specifications, with 5-bit unsigned representation for input quantization, 8-bit signed representation for output quantization, and 7-bit signed representation for weight quantization. Due to the complexity of the DirtyMNIST dataset, applying input and output quantization to the final linear layer (before computing logits) introduces non-negligible quantization errors that significantly impact model performance. Therefore, in practice, the final linear layer maintains only weight quantization while removing input and output quantization to preserve classification accuracy and maintain a good uncertainty estimation. A detailed analysis of the impact of these quantization parameters on model performance can be found in Appendix (a.2).

Experiment Results and Analysis: Figure (7.5) and Table (7.5) present a comprehensive comparative analysis of ReLU and SoftPlus activation functions across three MNIST dataset variants under 7-bit sample-based quantization. The experimental results provide both visual and quantitative evidence of the distinct uncertainty representation capabilities of these activation functions.

The scatter plots in Figure (7.5) illustrate the relationship between softmax entropy and mutual information across the three dataset variants (MNIST in blue, AmbiguousMNIST in green, and FashionMNIST in orange). With ReLU activation (Figure 7.5a), the visualization shows significant overlapping between dataset clusters, particularly between AmbiguousMNIST and FashionMNIST samples. The uncertainty measures exhibit compressed patterns, with mutual information values

failing to adequately distinguish between different levels of uncertainty across datasets, especially for the more complex FashionMNIST samples.



(a) Relu **(b)** SoftPlus

Figure 7.5: Scatter plots comparing the relationship between softmax entropy and mutual information across different MNIST variants (MNIST, AmbiguousMNIST, and FashionMNIST) using different activation functions under 7-bit quantization. SoftPlus (b) demonstrates superior uncertainty calibration compared to ReLU (a) when applied with the sample-based quantization method, showing more distinct clustering patterns for each dataset variant and maintaining clearer separation between different levels of aleatoric and epistemic uncertainty measures.

In marked contrast, SoftPlus activation (Figure 7.5b) produces distinctly more structured uncertainty representations. The separation between dataset variants is considerably clearer, with each dataset occupying more defined regions in the uncertainty space. Most notably, FashionMNIST samples (orange) demonstrate a more pronounced distribution extending into higher mutual information values, indicating more robust epistemic uncertainty quantification for this complex dataset. AmbiguousMNIST samples (green) maintain appropriate uncertainty levels while showing clearer differentiation from the other datasets, and MNIST samples (blue) cluster appropriately in the low-uncertainty region, reflecting high confidence predictions on standard examples.

The quantitative results in Table (7.5) provide comprehensive validation of the visual observations. On the standard MNIST dataset, both activation functions achieve comparable high performance (accuracy: ReLU 0.97, SoftPlus 0.98; unanimity: ReLU 0.96, SoftPlus 0.97), with nearly identical uncertainty measures. However, the advantage of SoftPlus becomes increasingly evident as dataset complexity increases.

For FashionMNIST, the unanimity difference is also substantial (ReLU 0.64 vs. SoftPlus 0.78). More critically, SoftPlus exhibits significantly better uncertainty calibration with much lower softmax entropy

Method	MNIST					AmbiguousMNIST					FashionMNIST				
	A ↑	U ↑	P.U.	S.E.	M.I.	A ↑	U ↑	P.U.	S.E.	M.I.	A ↑	U ↑	P.U.	S.E.	M.I.
Relu	0.97	0.96	0.12	0.09	0.03	0.50	0.69	0.94	0.82	0.13	0.11	0.64	1.09	0.70	0.39
SoftPlus	0.98	0.97	0.07	0.05	0.03	0.51	0.72	0.75	0.66	0.08	0.04	0.78	0.57	0.14	0.43

Table 7.5: Performance comparison of ReLU and SoftPlus activation functions across different MNIST datasets under 7-bit sample-based quantization. ↑ indicates that higher values are better. Metrics include: A: Accuracy, U: Unanimity, P.U.: Predictive Uncertainty, S.E.: Softmax Entropy, M.I.: Mutual Information. SoftPlus consistently outperforms ReLU across all datasets, achieving higher accuracy and unanimity while maintaining better uncertainty calibration, as evidenced by lower softmax entropy and appropriately modulated mutual information values that reflect the varying complexity of each dataset variant.

(0.14 vs. 0.70) and appropriately higher mutual information (0.43 vs. 0.39), indicating superior capability in distinguishing between aleatoric and epistemic uncertainty sources.

On AmbiguousMNIST, the results reveal similar patterns: comparable accuracy (both 0.51) and unanimity (ReLU 0.69, SoftPlus 0.72), but SoftPlus demonstrates improved uncertainty quantification with lower softmax entropy (0.66 vs. 0.82) and lower but more appropriate mutual information (0.08 vs. 0.13), reflecting better uncertainty calibration for inherently ambiguous samples.

These findings comprehensively demonstrate that SoftPlus activation provides superior uncertainty calibration across varying dataset complexity levels, particularly excelling in epistemic uncertainty representation for out-of-distribution and complex datasets. The improved uncertainty decomposition is essential for reliable quantized Bayesian Neural Networks, enabling more accurate confidence estimation when processing novel or ambiguous inputs. These results validate our approach of utilizing smoother activation functions that maintain gradient information across the entire input range, establishing a more robust foundation for subsequent logarithmic quantization strategies aimed at preserving critical variance characteristics in quantized representations.

7.1.4 Summary of the impact of activation functions

Through our comprehensive investigation into activation function selection for quantized Bayesian Neural Networks, we have established that activation function choice profoundly impacts both performance and uncertainty estimation quality in these specialized architectures. Our research addresses the critical challenge posed by the substantial

input amplification effects that occur when quantized inputs interact with weight matrices in BNNs, leading to values that are several orders of magnitude larger than those intended for standard activation functions.

Our experimental findings across multiple datasets and tasks reveal several key insights:

Activation Function Adaptations for Quantized BNNs: We introduced three novel activation function variants specifically designed for quantized BNNs: Alpha-Sigmoid, Alpha-Sine, and SoftPlus. Each addresses distinct challenges in input representation while maintaining the beneficial property of non-negative outputs that facilitate stable quantization across network layers.

Fundamental Insights: Our research establishes that activation function selection in quantized BNNs must address four critical aspects:

1. counteracting input amplification effects through appropriate scaling
2. maintaining effective uncertainty representation throughout the network
3. generating non-negative outputs for simplified inter-layer quantization
4. providing smooth transitions to preserve gradient information and uncertainty propagation.

The systematic exploration of both fixed and learnable parameter configurations across different tasks demonstrates that optimal activation function design requires task-specific considerations. While fixed parameters often provide superior direct performance metrics, learnable parameters enable adaptive responses to quantization challenges and improved uncertainty representation quality.

Our findings establish a foundation for activation function selection in quantized BNNs, demonstrating that thoughtful modifications to standard activation functions can substantially enhance model performance and uncertainty estimation. As neuromorphic hardware implementations continue to advance, these activation function adaptations provide crucial strategies for maintaining model fidelity while navigating the constraints imposed by quantized representations. The insights gained from this research contribute to the broader development of reliable and efficient Bayesian Neural Networks for resource-constrained deployment scenarios.

7.2 EFFECT OF WEIGHT CLIPPING ON QUANTIZATION EFFICIENCY WITHOUT BIAS

Traditional neural networks rely heavily on bias terms to shift activation functions and enable effective learning across the input space. However, in quantized Bayesian Neural Networks, bias terms introduce additional complexities to the quantization process. This section investigates how weight clipping can address the challenges in bias-free quantized networks with rigorous analysis of experimental results.

WHY BIAS FREE To simplify the quantization process and better isolate the effects of weight distribution quantization on model accuracy and uncertainty estimation, we propose and experimentally validate a bias-free quantization approach. By uniformly eliminating bias terms across network layers, we achieve several theoretical advantages:

- **Unified Quantization Pipeline:** Eliminating bias terms allows the quantization process to focus exclusively on the weight matrix W, simplifying the design and optimization of the quantization function $Q(\cdot)$.
- **Reduced Parameter Space:** For a network layer with n output neurons, the bias-free design reduces n parameters, thereby decreasing memory requirements on resource-constrained devices.
- **Cleaner Quantization Effect Analysis:** Removing bias terms eliminates parameter interaction effects, making the impact function of weight quantization $f(Q(W))$ more amenable to analysis.

KEY MECHANISMS OF WEIGHT CLIPPING To address the convergence challenges in bias-free quantized networks, we develop a framework for weight clipping that constrains weight values to a predefined range. Formally, weight clipping applies the following transformation to weights during training:

$$W_{clipped} = \text{clip}(W, W_{min}, W_{max}) = \max(W_{min}, \min(W, W_{max})) \quad (7.8)$$

where W_{min} and W_{max} are the lower and upper bounds for weight values, respectively.

Weight clipping operates through several key mechanisms in bias-free quantized networks:

1. **Range Limitation:** Clipping constrains weights to a predefined range, effectively truncating the weight matrix and preventing extreme values. This operation significantly transforms the weight distribution.

2. **Distribution Reshaping:** Through clipping, the normally Gaussian-distributed weights are transformed into distributions with significant mass concentration at the boundaries. This reshaping likely helps the network find a more stable parameter space.

3. **Implicit Regularization:** By limiting model capacity, clipping may lead to better generalization performance. The network is forced to learn within a constrained parameter space, which can be mathematically expressed as a bound on the model norm: $|W|_F \leq \sqrt{n}W_{max}$ (where n is the number of weights).

These mechanisms provide a foundation for understanding the effectiveness of weight clipping in bias-free quantized neural networks.

EXPERIMENTS AND ANALYSIS To empirically verify the effectiveness of weight clipping in quantized neural networks, we designed a controlled experiment focusing on the quality of input representation. We chose a noisy sine wave regression task as a benchmark because it provides a clear metric to evaluate how well different activation functions preserve input information under quantization constraints.

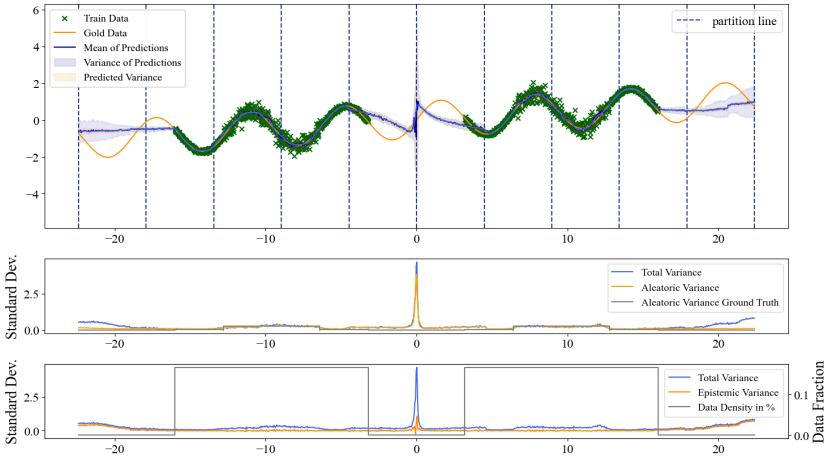
Experiment Settings: All experiments shared these settings:

- **Dataset:** 2,000 training samples, 2,000 testing samples, batch size 64
- **Model:** MLP with input representation input, 50 neurons in hidden layer, alpha-sigmoid activation, bias free and weight clipping($W_{min} = -1, W_{max} = 1$, which corresponds to approximately the $[-\sigma, \sigma]$ interval of a standard normal distribution, capturing about 68.3% of the probability mass).
- **Training:** 300 epochs pretraining (lr=0.02), 1,000 epochs main training (lr=0.005), 1 particle for training, random_seed is set to 42.
- **KL Annealing:** Maximum weight 0.025, with an end-focused schedule that keeps the KL loss nearly inactive in early epochs and ramps it up only near the end of training.

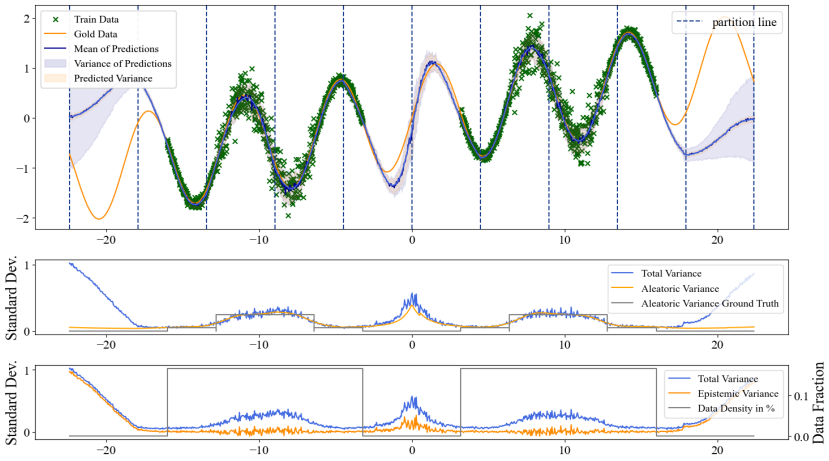
Experiment Results and Analysis Figure (7.6) presents a comparative visualization of model performance with and without weight clipping on the noisy sine wave regression task. The visual evidence provides several important insights into the impact of weight clipping on model performance and uncertainty estimation.

In Figure (7.6a) (bias-free BNNs without weight clipping), we observe that the prediction curve (green) follows the general trend of the ground truth data, but exhibits significant deviations and instability across the input domain. The uncertainty estimates displayed in the middle and bottom panels reveal pronounced variance spikes,

particularly at $x \approx 0$, indicating substantial instability in uncertainty quantification. Moreover, in the boundary regions ($x \approx -20$ and $x \approx 20$), the model without weight clipping produces inadequate uncertainty estimates that fail to properly reflect the limited information available in these areas, demonstrating a tendency toward inconsistent uncertainty representation.



(a) Bias-free BNNs without weight clipping



(b) Bias-free BNNs with weight clipping

Figure 7.6: Comparison of bias-free Bayesian Neural Networks on the noisy sine wave regression task: (a) Network without weight clipping shows poor uncertainty calibration and prediction instability; (b) Network with weight clipping demonstrates improved predictive accuracy and better-calibrated uncertainty estimates. The plots display prediction means (green lines) against training data (black dots), with shaded regions representing the predictive variance. Middle and bottom panels show the standard deviation of predictions across different regions, highlighting significant improvements in uncertainty quantification with weight clipping, particularly in out of domain regions.

In contrast, Figure (7.6b) (bias-free BNNs with weight clipping) demonstrates markedly improved performance characteristics. The prediction curve adheres more closely to the ground truth data across the entire input domain, showing enhanced stability and accuracy. More importantly, the uncertainty patterns appear better distributed and more consistent across the input range, with smoother variance transitions that correspond more appropriately to data density patterns. In the boundary regions, the weight-clipped model produces higher and more consistent uncertainty estimates, correctly reflecting the limited information available for making predictions in these areas.

The visual comparison provides clear evidence that weight clipping enables more reliable uncertainty quantification while significantly improving prediction accuracy. The smoother uncertainty transitions and better-calibrated uncertainty in data-sparse regions indicate that weight clipping effectively constrains the weight distribution, preventing extreme values that could lead to unstable predictions and poor uncertainty representation.

Table (7.6) provides quantitative performance metrics that further substantiate the visual observations. The results reveal significant improvements in most evaluation metrics when employing weight clipping compared to the standard bias-free configuration.

Mean Squared Error (MSE) shows a substantial reduction from 7.07×10^{-4} with the bias-free model to 3.89×10^{-4} with weight clipping, representing a 45.0% improvement in prediction accuracy. This significant error reduction aligns with the visual observation that weight clipping enables the model to better capture underlying data patterns while maintaining stability.

The Negative Log-Likelihood (NLL) improves from -0.0109 to -0.0117, indicating that weight clipping produces better-calibrated probability distributions. While this improvement appears modest, it represents an important enhancement in how well the model's predictive distribution matches the true data distribution, a critical aspect of Bayesian Neural Networks.

Interestingly, the Evidence Lower Bound (ELBO) shows a contrasting pattern, with the bias-free configuration achieving -15.76 compared to -9.45 with weight clipping. This increase in ELBO suggests that while weight clipping improves direct predictive metrics, it may introduce constraints that affect the tightness of the variational approximation. This trade-off highlights the complex interplay between prediction accuracy and the fidelity of the variational approximation in Bayesian Neural Networks.

The Aleatoric MSE also shows improvement, decreasing from 2.62×10^{-4} to 2.38×10^{-4}, representing a 9.2% reduction. This improvement suggests that weight clipping enhances the model's ability to capture intrinsic data noise, complementing its benefits for epistemic uncertainty modeling.

Figures (7.7) and (7.8) provide deeper insights into how weight clipping transforms the weight distributions across different network layers. Figure (7.7) displays density distributions comparing original weights (orange) with clipped weights (blue). The plots reveal that weight clipping produces sharper boundaries in the weight distributions while preserving the overall distribution shape. This transformation is particularly evident in the hidden_layer_0 and output_var layers, where the broader tails of the original distribution are effectively constrained.

Figure (7.8) presents scatter plots examining the relationship between original weights and their clipped counterparts. The high Pearson correlation coefficients ($r > 0.93$) across all layers demonstrate that weight clipping preserves the essential characteristics of the weight distributions while effectively constraining extreme values. The horizontal clustering patterns at the boundaries of the weight range visually confirm this effect, showing how outlier weights are systematically constrained to the specified range of [-1, 1].

Method	MSE ($\times 10^{-4}$) ↓	NLL ↓	ELBO ↓	Aleatoric MSE ($\times 10^{-4}$) ↓
Bias-Free	7.07	-0.0109	**-15.76**	2.62
Weight Clipping	**3.89**	**-0.0117**	-9.45	**2.38**

Table 7.6: Comparative Analysis of Wight Clipping on Model Performance of Noisy Sine Wave Data (Training Metrics). The table shows that the Weight Clipping method achieves better MSE, NLL, and Aleatoric MSE training indicators on noisy sine wave data than the Bias-Free method, but performs poorly on the ELBO indicator, which indicates that Weight Clipping may improve the prediction accuracy and uncertainty estimation of the model, but has a certain impact on the lower bound of the model's evidence.

Detailed analysis of the clipping statistics reveals significant differences in how weight clipping affects each layer:

- **hidden_layer_0:** With a shape of (50, 10), this layer shows the most extreme weight values, ranging from -3.107 to 2.644 before clipping. Approximately 30.60% of weights required clipping, with an average clipping magnitude of 0.171. This higher percentage indicates that the first hidden layer tends to develop more extreme weights during training.

- **output_mean:** With a shape of (1, 50), this layer has a more moderate original weight range of -2.575 to 1.795. Only 16.00% of weights required clipping, with a lower average clipping magnitude of 0.112. This suggests that the output mean layer naturally develops more constrained weights during training.

- **output_var:** With a shape of (1, 50), this layer shows intermediate original weight values ranging from -1.730 to 1.961. However, it

has the highest percentage of clipped weights at 36.00%, with an average clipping magnitude of 0.191. This high clipping percentage in the uncertainty estimation layer is particularly notable, suggesting that constraining extreme weights in this layer is especially important for proper uncertainty calibration.

Weight clipping achieves performance improvements through several mechanisms:

- **Stability Enhancement:** By constraining weights to a predefined range [-1, 1], weight clipping prevents extreme weight values that could cause numerical instability or gradient explosion during training, resulting in more consistent model behavior.
- **Improved Signal Propagation:** Constrained weights ensure more balanced signal propagation through the network, preventing individual neurons from dominating the network's output and allowing for more nuanced representation of input patterns.
- **Better Uncertainty Calibration:** By limiting the magnitude of weights, weight clipping reduces the model's tendency to produce overconfident predictions in regions with limited data, resulting in more realistic uncertainty estimates that better reflect the available information.
- **Enhanced Generalization:** The constraint on weight values acts as an implicit regularization mechanism, reducing model complexity and improving generalization to unseen data by preventing overfitting to specific training examples.

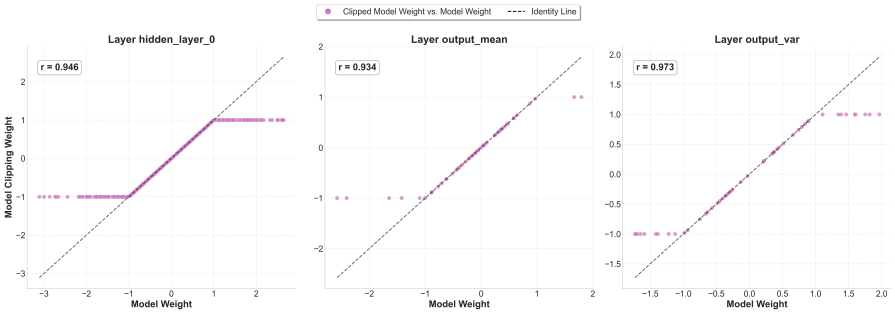


Figure 7.7: Comparison between original model weights and clipped model weights across three key layers (hidden_layer_0, output_mean, and output_var). The scatter plots show the relationship between original model weights (x-axis) and their corresponding clipped values (y-axis), with Pearson correlation coefficients (r) indicating the degree of linear relationship preservation. The high correlation values ($r > 0.93$) demonstrate that the clipping operation preserves the essential weight distribution characteristics while effectively constraining extreme values, as evidenced by the horizontal clustering patterns at the boundaries of the weight range.

The analysis demonstrates that weight clipping provides a simple yet effective solution to improve both predictive accuracy and uncertainty quantification in quantized Bayesian Neural Networks. By constraining weights to a carefully selected range that approximates the standard normal distribution's central region, weight clipping significantly enhances model performance without requiring complex architectural changes or additional parameters.

The observed trade-off between ELBO and direct predictive metrics highlights an important consideration in Bayesian Neural Network design: improvements in practical prediction tasks may sometimes come at the cost of theoretical optimality in variational approximation. This insight suggests that practitioners should carefully consider their specific application requirements when deciding whether to implement weight clipping in their models.

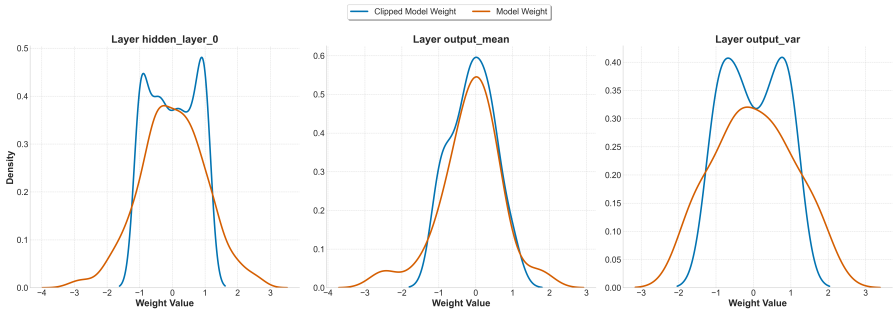


Figure 7.8: Density distribution comparison between original model weights (orange) and clipped model weights (blue) across three key network layers (hidden_layer_0, output_mean, and output_var). The density plots illustrate how weight clipping affects the probability distribution of weights in each layer. Notable differences include sharper boundaries in the clipped weight distributions compared to the broader tails of the original weights, particularly evident in the hidden_layer_0 and output_var layers. This visualization highlights the effect of the clipping operation in concentrating weights within a constrained range while maintaining the overall distribution shape.

7.3 STANDARD DEVIATION PRESERVATION FOR LOGARITHMIC QUANTIZATION IN BNNS

Bayesian Neural Networks (BNNs) provide a principled framework for uncertainty quantification in deep learning, capturing both aleatoric and epistemic uncertainty through probabilistic weight distributions. However, the effective representation and processing of these uncertainties under quantization constraints presents significant challenges, particularly regarding standard deviation characteristics of uncertainty distributions.

A critical observation from our preliminary experiments reveals that uncertainty information in BNNs exhibits highly skewed standard deviation distributions. The scale parameters of these distributions typically span an exceptionally wide range, from approximately 10^{-6} to 10^{2}, creating a substantial challenge for uniform quantization approaches [1]. This extreme standard deviation range presents a fundamental representation problem: uniform quantization schemes inevitably sacrifice either precision for smaller values or representation capacity for larger values.

Further investigation revealed that the primary sources of meaningful uncertainty information are concentrated in the higher magnitude standard deviation values rather than the lower ones. This finding carries significant implications for quantization strategies, as traditional uniform quantization approaches would disproportionately truncate these critical high standard deviation values while allocating unnecessary precision to less informative low standard deviation regions.

From a theoretical perspective, this significant disparity in the information content between high and low standard deviation components can be explained through the lens of Bayesian inference principles. In BNNs, posterior weight distributions approximate the true posterior $p(\mathbf{w}|\mathcal{D})$ given training data $\mathcal{D}$. The standard deviation of these distributions directly relates to predictive uncertainty through the posterior predictive distribution:

$$p(y|\mathbf{x}, \mathcal{D}) = \int p(y|\mathbf{x}, \mathbf{w}) p(\mathbf{w}|\mathcal{D}) d\mathbf{w} \tag{7.9}$$

Higher standard deviation values in weight distributions contribute disproportionately to the overall uncertainty in predictions because:

1. **Uncertainty propagation:** When computing the posterior predictive distribution, the propagation of uncertainty follows a nonlinear relationship with standard deviation magnitude. Weight components with larger standard deviations contribute exponentially more to output uncertainty than those with smaller standard deviations.

2. **Information-theoretic perspective:** From an information theory standpoint, the entropy of a Gaussian distribution scales logarithmically with standard deviation ($H = \frac{1}{2}\ln(2\pi e\sigma^2)$). Thus, high standard deviation components carry substantially more uncertainty information than low standard deviation components.

1 This range was roughly estimated from statistical analysis of standard deviation values observed across our experimental BNN models.

3. **Posterior concentration:** Well-identified parameters in BNNs tend to have concentrated posteriors with small standard deviations, while parameters with limited data support or complex interactions exhibit larger standard deviations. The latter are precisely those that should dominate uncertainty estimation in robust Bayesian inference.

These theoretical insights suggest that preserving high-standard deviation components should be prioritized in any efficient quantization scheme for BNNs. Furthermore, they indicate that logarithmic quantization would provide a natural solution to this representation challenge by allocating quantization levels proportionally to value magnitude, effectively compressing the representation of the wide-ranging standard deviation scale into a more manageable form while preserving the relative relationships between standard deviation values. This approach inherently prioritizes relative precision rather than absolute precision, aligning well with the logarithmic nature of uncertainty information processing in probabilistic inference.

EXPERIMENTS AND ANALYSIS To validate our theoretical analysis and systematically investigate the impact of standard deviation magnitude on BNN performance, we designed a series of experiments specifically targeting standard deviation distribution characteristics.

Experiment Settings: All experiments shared these settings:

- **Dataset:** 2,000 training samples, 2,000 testing samples, batch size 64
- **Model:** MLP with input representation input, 50 neurons in hidden layer, Alpha-Sigmoid activation($\alpha = 0.1$).
- **Training:** 300 epochs pretraining (lr=0.02), 1,000 epochs main training (lr=0.005), 1 particle for training, random_seed is set to 42.
- **KL Annealing:** Maximum weight 0.025, with an end-focused schedule that keeps the KL loss nearly inactive in early epochs and ramps it up only near the end of training.

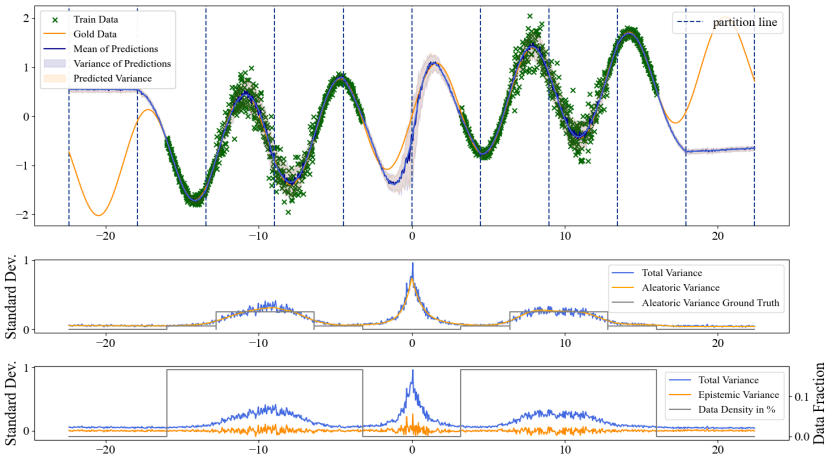
We conducted two complementary experiments to isolate the effects of high and low standard deviation components:

- **Low Standard Deviation Preservation:** In this experiment, we retained all standard deviation values below the layer-wise mean while setting all standard deviation values above the mean to a minimal constant (10^{-6}). This allowed us to evaluate model performance when high-standard deviation components are effectively eliminated.

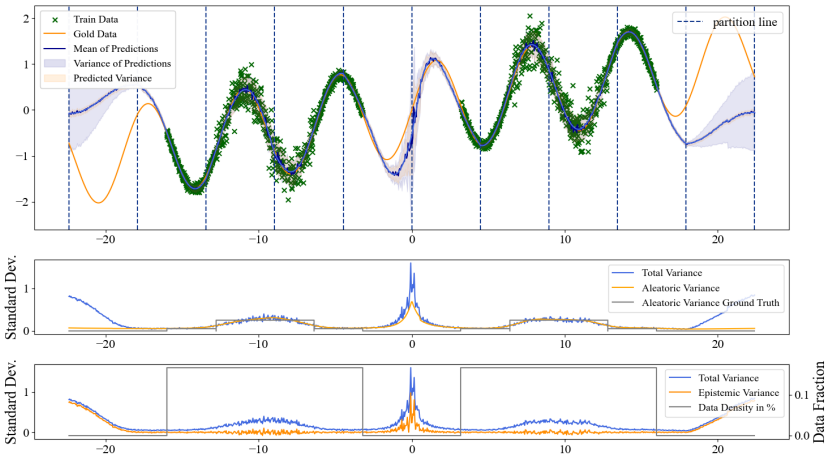
- **High Standard Deviation Preservation:** Conversely, we retained all standard deviation values above the layer-wise mean while setting all standard deviation values below the mean to a minimal constant (10^{-6}). This experiment assessed model performance when only high-standard deviation components contribute to uncertainty representation.

Experiment Results and Analysis:

Figure (7.9) compares Low and High Standard Deviation Preservation strategies on the noisy sine wave regression task. The visual results reveal important differences in their uncertainty quantification capabilities.



(a) Low Standard Deviation Preservation:



(b) High Standard Deviation Preservation:

Figure 7.9: Comparison of Standard Deviation preservation strategies on the noisy sine wave regression task: (a) Low Standard Deviation Preservation approach; (b) High Standard Deviation Preservation approach. The upper panels display prediction curves (blue) against training data points (green crosses) and ground truth functions (orange), along with predicted standard deviation (shaded blue). The middle and lower panels show decomposition of uncertainty into epistemic, aleatoric, and total standard deviation components. The High Standard Deviation Preservation strategy demonstrates improved uncertainty calibration particularly in out of domain regions (e.g., x < -20 and x > 20), with more appropriate widening of confidence bands and better distinction between epistemic and aleatoric uncertainty sources.

In the Low Standard Deviation Preservation approach (Figure 7.9a), while predictions generally follow the ground truth, uncertainty estimates in out of domain regions ($x < -20$ and $x > 20$) appear inadequate, showing overconfidence where data is scarce. The uncertainty decomposition panels demonstrate insufficient standard deviation

estimation, particularly for epistemic uncertainty in these boundary regions.

By contrast, the High Standard Deviation Preservation strategy (Figure 7.9b) maintains similar prediction accuracy while providing better-calibrated uncertainty. This approach demonstrates appropriately higher uncertainty in out of domain regions, with epistemic uncertainty properly increasing where training data is absent, which aligns with expected Bayesian behavior.

Table (7.7) confirms these visual observations with quantitative improvements across all metrics. The High Standard Deviation Preservation strategy achieves modest enhancements in prediction accuracy (MSE reduction of 2.0%) and aleatoric uncertainty modeling (Aleatoric MSE reduction of 1.2%). More significantly, it improves probabilistic calibration (NLL improvement from -0.01485 to -0.0150) and variational approximation quality (ELBO improvement of 17.1%, from -10.58 to -12.39).

These findings provide theoretical guidance for scale quantizer selection in Bayesian Neural Networks. The superior performance of High Standard Deviation Preservation directly supports logarithmic quantization schemes, which inherently allocate more quantization levels to larger values. This non-uniform approach naturally preserves the most significant uncertainty information while applying more aggressive quantization to smaller standard deviation values with less impact. For resource-constrained hardware implementations, logarithmic quantization offers an effective solution that maintains well-calibrated uncertainty estimates—particularly in out of domain regions—while achieving competitive prediction accuracy.

Method	MSE ($\times 10^{-4}$) ↓	NLL ↓	ELBO ↓	Aleatoric MSE ($\times 10^{-4}$) ↓
Low Standard Deviation Preservation:	5.46	-0.01485	-10.58	2.53
High Standard Deviation Preservation:	**5.35**	**-0.0150**	**-12.39**	**2.50**

Table 7.7: Comparative Analysis of Standard Deviation Preservation Strategies on Model Performance of Noisy Sine Wave Data (Training Metrics). The table shows that the High Standard Deviation Preservation strategy outperforms the Low Standard Deviation Preservation approach across all metrics, demonstrating modest improvements in MSE (2.0%), NLL (1.0%), ELBO (17.1%), and Aleatoric MSE (1.2%). These results suggest that preserving higher standard deviation during training leads to both better prediction accuracy and more well-calibrated uncertainty estimates.

7.4 SUMMARY OF OPTIMIZATIONS FOR QUANTIZED SVI

This summary provides concrete recommendations for improving quantized Bayesian Neural Networks (QBNNs) based on our experimental analysis, highlighting effective approaches and practices to avoid.

ACTIVATION FUNCTION SELECTION. Activation function selection in quantized BNNs significantly impacts both performance and uncertainty estimation. When quantized inputs interact with weight matrices, values are frequently amplified by several orders of magnitude, creating substantial challenges. We recommend modified activation functions specifically designed to counteract these effects: **Alpha-Sigmoid** ($\sigma(\alpha x)$ with $\alpha = 0.1$) for regression tasks, which reduced MSE by 41.3% and improved ELBO from 12.73 to -15.23; **Alpha-Sine** ($\sin(\alpha x)$ with $\alpha = 0.1$) for classification tasks, which improved Two Moons accuracy from 47.03% to 100%; and **SoftPlus** ($\ln(1 + e^x)$) when global minimum offset adjustments are applied, significantly improving uncertainty calibration in datasets like DirtyMNIST. Practitioners should carefully adapt standard activation functions or risk significant performance degradation. Effective activation functions for quantized BNNs must balance three critical properties: appropriate input scaling to counter amplification effects, smoothness to preserve gradient information, and output characteristics that facilitate subsequent layer quantization.

WEIGHT CLIPPING FOR BIAS-FREE NETWORKS. To simplify the quantization process, we propose a bias-free approach with weight clipping. By eliminating bias terms and constraining weights to a predefined range (e.g., $[-1, 1]$), this technique improved MSE by 45.0% and NLL from -0.0109 to -0.0117 in regression tasks. Weight clipping operates through several key mechanisms: stability enhancement by preventing extreme values, improved signal propagation, better uncertainty calibration in data-sparse regions, and enhanced generalization through implicit regularization. Different layers show varying sensitivity to clipping, with uncertainty estimation layers showing particularly high clipping percentages (36%). Without clipping, bias-free networks show significant instability and degraded performance.

STANDARD DEVIATION PRESERVATION FOR UNCERTAINTY QUANTIZATION. Uncertainty information in BNNs exhibits highly skewed standard deviation distributions spanning an extremely wide range (10^{-6} to 10^2), creating substantial challenges for uniform quantization. Our experiments reveal that meaningful uncertainty information is concentrated in higher magnitude standard deviation values. We

recommend prioritizing high standard deviation preservation and implementing logarithmic quantization schemes that allocate more quantization levels to larger values. This approach aligns with Bayesian inference principles: uncertainty propagation follows non-linear relationships with standard deviation magnitude, information content scales logarithmically with standard deviation, and well-identified parameters typically have small standard deviations while parameters with limited data support exhibit more informative larger standard deviations. Uniform quantization should be avoided as it leads to overconfidence in out-of-distribution regions and poorer uncertainty calibration.

IMPLEMENTATION RECOMMENDATIONS. For optimal performance in quantized SVI Bayesian Neural Networks, we recommend an integrated approach combining:

1. Activation function adaptation using scaled variants or smooth alternatives based on specific task requirements
2. Weight constraint implementation in bias-free networks, with particular attention to layers responsible for uncertainty estimation
3. Non-uniform standard deviation quantization employing logarithmic schemes that preserve high-magnitude components critical for proper uncertainty representation.

These strategies significantly improve both predictive accuracy and uncertainty estimation quality while maintaining computational efficiency for resource-constrained neuromorphic hardware implementations.

8 BIT-WIDTH SENSITIVITY ANALYSIS

In the preceding chapters, we introduced different quantization methods for Bayesian Neural Networks and explored various optimization strategies to enhance their performance under quantization constraints. Chapter 6 presented three distinct quantization approaches—Distribution Sampling Values Quantization, Distribution Parameters Quantization, and Integrated Quantization—and demonstrated their effectiveness across multiple datasets. Building upon these foundations, Chapter 7 detailed specialized optimization techniques, including activation function modifications and weight clipping strategies, which significantly improved the performance of quantized BNNs.

The relationship between bit-width allocation and model performance represents a fundamental trade-off in hardware-efficient deep learning. In traditional neural networks, numerous studies have examined this relationship, establishing various precision thresholds for different architectures and tasks (Jacob et al., 2017; Gholami et al., 2021; Zhuang et al., 2017). However, Bayesian Neural Networks present unique challenges due to their dual objectives of maintaining both predictive accuracy and uncertainty estimation quality under quantization constraints.

Having demonstrated the feasibility of BNN quantization in our previous experiments, we now face an important limitation: those studies primarily utilized fixed bit-width configurations dictated by neuromorphic hardware constraints. A critical next step in this research is to understand the relationship between various bit-width allocations and model performance, which is essential for effectively deploying BNNs across different resource-constrained environments. This chapter therefore undertakes a systematic investigation into how quantized BNNs respond to different bit-width configurations, offering valuable insights into the minimum precision requirements needed to preserve both performance and uncertainty estimation quality.

The bit-width sensitivity analysis presented in this chapter serves several important purposes:

- **Identifying Precision Thresholds:** Determining the minimum bit-width requirements below which performance and uncertainty estimation significantly deteriorate for each quantization method.
- **Comparing Quantization Method Resilience:** Assessing which of the three quantization approaches demonstrates superior resilience to aggressive bit-width reduction.

- **Guiding Hardware-Specific Implementations:** Providing empirical evidence to inform bit-width allocation decisions in custom hardware implementations of BNNs.
- **Understanding Uncertainty Preservation:** Analyzing how bit-width reduction differentially affects aleatoric and epistemic uncertainty estimation.

We focus our analysis on the DirtyMNIST dataset, which presents an ideal testbed for bit-width sensitivity exploration due to its complexity and clear distinction between in-distribution data (MNIST), ambiguous data (AmbiguousMNIST), and out-of-distribution data (FashionMNIST). This diversity enables comprehensive evaluation of bit-width reduction effects. It allows us to examine impacts on both predictive accuracy and uncertainty estimation across varying data complexity levels.

Our investigation begins with a systematic examination of performance across a spectrum of precision levels: 2-bit, 4-bit, 8-bit, and 16-bit quantization, while also including 32-bit floating-point precision as a reference baseline and 7-bit integer quantization for direct comparison with our previous experiments. We apply these precision configurations to each of the three quantization methods previously introduced, maintaining consistent network architectures and training procedures to isolate the effects of bit-width variation.

Through this comprehensive bit-width sensitivity analysis, we aim to establish empirical guidelines for precision requirements in quantized BNNs, providing practical insights for researchers and practitioners implementing these models in resource-constrained environments.

8.1 THEORETICAL CONSIDERATIONS FOR BIT-WIDTH SELECTION

Before presenting our experimental results, it is important to establish the theoretical framework that informs bit-width sensitivity in Bayesian Neural Nnetworks. Three key considerations shape our understanding of how precision reduction affects BNN performance:

INFORMATION PRESERVATION IN WEIGHT DISTRIBUTIONS Unlike deterministic neural networks, BNNs represent each weight as a probability distribution typically parameterized by mean and variance. When quantizing these distributions, we must consider how precision reduction affects the information content of the entire distribution rather than just point estimates. Theoretically, the number of bits required to represent a distribution with a given fidelity can be related

to its differential entropy. For Gaussian distributions with variance σ^2, the differential entropy is given by:

$$h(X) = \frac{1}{2}\ln(2\pi e\sigma^2) \tag{8.1}$$

This relationship suggests that weight distributions with higher variances (encoding greater uncertainty) require more bits to represent accurately compared to distributions with lower variances. Consequently, aggressive bit-width reduction may disproportionately affect the representation of uncertain weights, potentially degrading epistemic uncertainty estimation more severely than point prediction accuracy.

ERROR PROPAGATION EFFECTS Quantization introduces representational errors that propagate through the network, with potentially compounding effects in deeper architectures. In BNNs, this error propagation mechanism is particularly complex due to the uncertainty propagation inherent in Bayesian inference. The forward pass in a BNN can be represented as:

$$p(y|\mathbf{x}) = \int p(y|\mathbf{x},\mathbf{w})p(\mathbf{w})d\mathbf{w} \tag{8.2}$$

When weights are quantized, the approximate posterior $q_\phi(\mathbf{w})$ includes additional quantization errors, potentially leading to misaligned uncertainty estimates. Theoretically, these errors accumulate quadratically with network depth in the worst case, suggesting that deeper BNNs may require higher precision to maintain performance under quantization.

ALEATORIC VS. EPISTEMIC UNCERTAINTY SEPARATION One of the key advantages of BNNs is their ability to distinguish between aleatoric uncertainty (data inherent noise) and epistemic uncertainty (model uncertainty). This separation relies on the precise representation of weight distribution parameters. With limited bit-width, the model's capacity to distinguish between these uncertainty types may degrade, particularly affecting the representation of epistemic uncertainty which depends on higher-order moments of the weight distributions.

The minimum theoretical bit-width required to maintain this distinction can be approximated using rate-distortion theory, which suggests that the number of bits needed scales with the logarithm of the desired precision in uncertainty estimates. For classification tasks with C classes, distinguishing between uncertainty types with confidence threshold τ requires approximately $\log_2(C/\tau)$ bits per weight, providing a theoretical lower bound for uncertainty-preserving quantization.

These theoretical considerations guide our experimental investigation, informing our analysis of the observed performance patterns across different precision levels.

8.2 EXPERIMENTAL SETUP FOR BIT-WIDTH SENSITIVITY

To systematically evaluate bit-width sensitivity, we designed a comprehensive experimental framework using the DirtyMNIST dataset. This dataset was selected due to its multi-faceted evaluation capabilities: standard MNIST samples assess in-distribution performance, AmbiguousMNIST samples evaluate aleatoric uncertainty estimation, and FashionMNIST samples test epistemic uncertainty for out-of-distribution data.

For each combination of quantization method and bit-width configuration, we maintained consistent model architecture, training procedures, and evaluation metrics to isolate the effects of precision variation. All bit-width configurations in our experiments utilize integer quantization rather than floating-point formats. To establish a comprehensive precision spectrum and facilitate direct comparison with our previous experiments, we examined the following integer bit-width configurations for weight quantization:

- **32-bit (Float):** Full floating-point precision as reference, providing the theoretical upper bound for performance
- **16-bit:** High-precision integer quantization, establishing a baseline close to floating-point performance
- **8-bit:** Standard low-precision integer quantization representing common hardware implementations
- **7-bit:** Integer quantization equivalent to our previous experiments in Chapter 6, enabling direct comparison
- **4-bit:** Aggressive integer quantization challenging representation capacity
- **2-bit:** Extreme integer quantization testing fundamental limits of BNN quantization

Experiment Settings: All experiments shared these settings:

- **Dataset:** DirtyMNIST is used for training, MNIST, AmbiguousMNIST, and FashionMNIST are used for testing, and the batch size is set to 100.
- **Model:** MLP with input representation obtained by shifting the scalar input to be non-negative (subtracting the minimum value) (based on our findings in Section 5.5), two hidden layers of 100 neurons each, SoftPlus activation (based on our findings in Section 7.1).

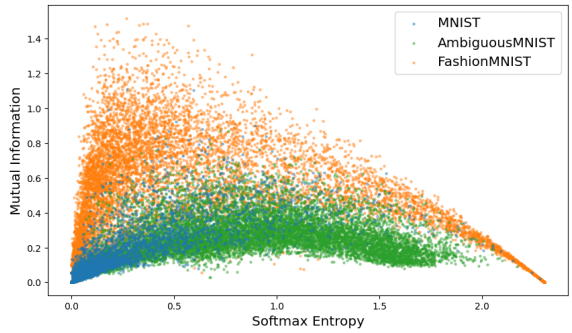
- **Training:** 1500 epochs pretraining (lr=0.001), 1,000 epochs main training (lr=0.0001), 1 particle for training, random_seed is set to 42.
- **KL Annealing:** Maximum weight 0.25, with an end-focused schedule that keeps the KL loss nearly inactive in early epochs and ramps it up only near the end of training.
- **Additional Quantization Parameters:** Consistent with our previous experiments and BSS-2 hardware constraints, we maintained 5-bit unsigned representation for input quantization and 8-bit signed representation for output quantization across all experiments, varying only the weight quantization bit-width according to the configurations listed above. Due to the complexity of the DirtyMNIST dataset, applying input and output quantization to the final linear layer (before computing logits) introduces non-negligible quantization errors that significantly impact model performance. Therefore, in practice, the final linear layer maintains only weight quantization while removing input and output quantization to preserve classification accuracy and maintain a good uncertainty estimation. A detailed analysis of the impact of these quantization parameters on model performance can be found in Appendix (a.2).

This experimental design isolates the effects of weight precision reduction while maintaining consistent activation quantization parameters. For each quantization method (Sample-Based Quantization Method, Parameter-Based Quantization Method, and Fully Integrated Quantization Method), we separately investigated the impact of precision reduction on performance metrics including accuracy, uncertainty calibration, and uncertainty decomposition.

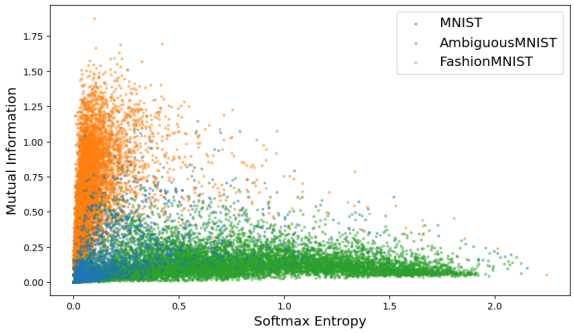
This expanded precision range of integer quantization enables us to thoroughly map the relationship between bit-width and performance while maintaining continuity with our previous findings.

8.3 SAMPLE-BASED QUANTIZATION METHOD

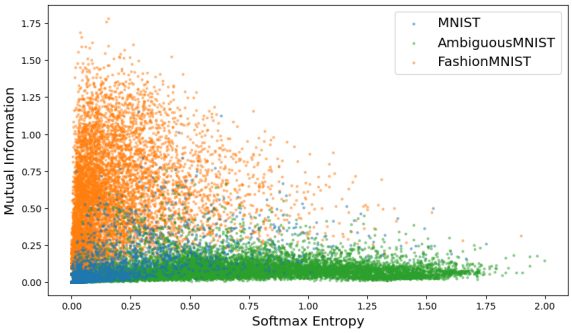
The experimental results for the Sample-Based Quantization Method across different bit-width configurations reveal several distinctive characteristics that differentiate this approach from other quantization strategies. Figure (8.1) presents scatter plots visualizing the relationship between softmax entropy and mutual information across different MNIST dataset variants, while Table (8.1) provides the corresponding quantitative metrics.



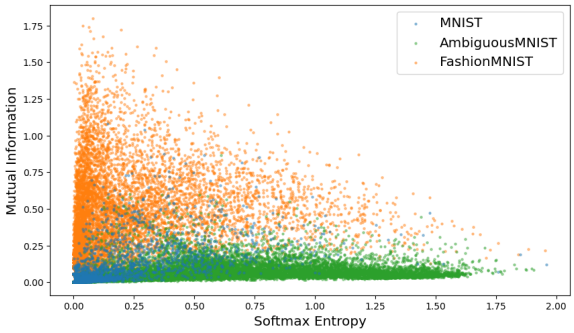
(a) 2-bit



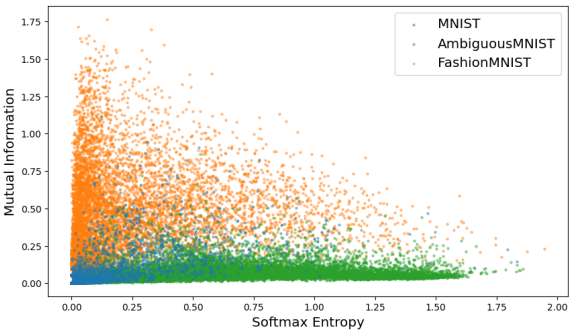
(b) 4-bit



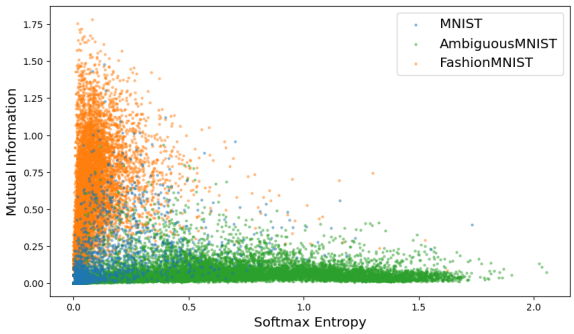
(c) 7-bit



(d) 8-bit



(e) 16-bit



(f) 32-bit (Float)

Figure 8.1: Scatter plots of softmax entropy versus mutual information across different MNIST dataset variants using Sample-Based Quantization Method. Each subplot represents a different bit-width configuration. Interestingly, 4-bit through 8-bit configurations demonstrate uncertainty calibration performance closest to the 32-bit floating-point baseline, while both 2-bit and 16-bit show more pronounced degradation in dataset separation, particularly for Fashion MNIST. This non-monotonic relationship between bit-width and uncertainty discrimination capability suggests an optimal precision range exists for Sample-Based Quantization in BNNs.

At the lowest precision level (2-bit), Sample-Based Quantization demonstrates remarkable robustness compared to alternative methods. Despite the extreme quantization, the method maintains high accuracy (0.97) and unanimity (0.94) on standard MNIST data. This resilience to severe precision constraints represents a significant advantage in resource-limited deployment scenarios. However, the uncertainty metrics at 2-bit precision show noticeable degradation, with elevated predictive uncertainty (0.18), softmax entropy (0.11), and mutual information (0.07) compared to higher precision configurations. This indicates that while classification performance remains strong, uncertainty calibration suffers to some extent.

The transition from 2-bit to 4-bit precision yields substantial improvements in uncertainty calibration, with predictive uncertainty decreasing from 0.18 to 0.08 and softmax entropy from 0.11 to 0.04 on MNIST. Interestingly, performance metrics stabilize at 4-bit precision, with minimal additional improvements observed at higher bit-widths. This "performance plateau" suggests that 4-bit precision captures the essential characteristics required for both accurate classification and well-calibrated uncertainty estimates using the Sample-Based approach.

Across ambiguous data (AmbiguousMNIST), the method appropriately maintains reduced accuracy (0.49-0.51) across all bit-widths, correctly reflecting the inherent ambiguity in these samples. However, uncertainty metrics show significant variation with bit-width, with 2-bit configuration exhibiting substantially higher predictive uncertainty (1.09) and softmax entropy (0.84) compared to higher precision configurations (0.70-0.81 for predictive uncertainty and 0.63-0.68 for softmax entropy, respectively). This suggests that while the method correctly identifies these samples as ambiguous regardless of precision, the quantification of uncertainty becomes more calibrated as precision increases.

For out-of-distribution data (FashionMNIST), Sample-Based Quantization shows appropriate uncertainty behavior with consistently low accuracy (0.04-0.09) across bit-widths. Notably, the method demonstrates a non-monotonic relationship between bit-width and uncertainty decomposition. The 4-bit configuration achieves the most balanced combination of low softmax entropy (0.07) and high mutual information (0.45), suggesting optimal epistemic uncertainty quantification at this precision level.

The scatter plots in Figure (8.1) visually confirm these observations, showing that 4-bit through 8-bit configurations demonstrate uncertainty calibration performance closest to the 32-bit floating-point baseline. Both 2-bit and 16-bit configurations show more pronounced degradation in dataset separation, particularly for FashionMNIST samples. This non-monotonic relationship between bit-width and un-

certainty discrimination capability suggests an optimal precision range exists for Sample-Based Quantization in BNNs.

Bit-Width	MNIST					AmbiguousMNIST					FashionMNIST				
	A ↑	U ↑	P.U.	S.E.	M.I.	A ↑	U ↑	P.U.	S.E.	M.I.	A ↑	U ↑	P.U.	S.E.	M.I.
2-bit	0.97	0.94	0.18	0.11	0.07	0.49	0.62	1.09	0.84	0.25	0.06	0.62	1.06	0.58	0.48
4-bit	0.97	0.97	0.08	0.04	0.04	0.50	0.71	0.81	0.68	0.10	0.07	0.79	0.52	0.07	0.45
7-bit	0.98	0.97	0.07	0.05	0.03	0.51	0.72	0.75	0.66	0.08	0.04	0.78	0.57	0.14	0.43
8-bit	0.98	0.98	0.07	0.05	0.03	0.51	0.73	0.74	0.66	0.08	0.07	0.78	0.58	0.20	0.38
16-bit	0.98	0.97	0.07	0.05	0.03	0.51	0.73	0.73	0.66	0.07	0.07	0.78	0.58	0.20	0.38
32-bit (Float)	0.98	0.98	0.05	0.02	0.03	0.51	0.73	0.70	0.63	0.07	0.09	0.77	0.56	0.07	0.49

Table 8.1: Performance metrics of Sample-Based Quantization Method across different bit-width configurations. ↑ indicates higher values are better. A: Accuracy, U: Unanimity, P.U.: Predictive Uncertainty, S.E.: Softmax Entropy, M.I.: Mutual Information. Results reveal a non-monotonic relationship between bit-width and performance, with 4-bit configurations and 32-bit floating-point achieving the best uncertainty calibration. The 2-bit configuration shows significant degradation in uncertainty metrics, particularly for ambiguous and out-of-distribution data. Notably, intermediate bit-widths (8-bit and 16-bit) unexpectedly exhibit poorer FashionMNIST uncertainty decomposition than both lower (4-bit) and higher (32-bit) precision configurations, suggesting that optimal uncertainty quantification occurs at specific precision points rather than continuously improving with increased bit-width.

In summary, Sample-Based Quantization demonstrates exceptional robustness at extremely low precision while achieving optimal uncertainty calibration at moderate precision levels (4-8 bits) and at the full 32-bit floating-point precision. The method's ability to maintain high classification performance even at 2-bit precision represents a significant advantage for deployment in severely resource-constrained environments, while the balanced uncertainty decomposition at 4-bit and 32-bit precisions offers valuable options for applications requiring well-calibrated uncertainty estimates.

8.4 PARAMETER-BASED QUANTIZATION EXPERIMENT RESULT AND ANALYSIS

The Parameter-Based Quantization Method exhibits dramatically different performance characteristics across bit-width configurations compared to the Sample-Based approach. Figure (8.2) visualizes the relationship between softmax entropy and mutual information for this

method, while Table (8.2) provides the corresponding performance metrics.

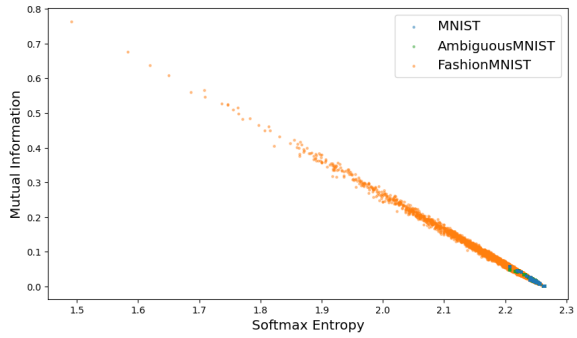
The most striking observation is the catastrophic failure of Parameter-Based Quantization at 2-bit precision. Across all datasets, the 2-bit configuration results in near-random accuracy (0.10 on MNIST, 0.16 on AmbiguousMNIST, 0.10 on FashionMNIST) and extremely low unanimity (0.18). The uncertainty metrics reveal a complete collapse of the model's discriminative capabilities, with extraordinarily high predictive uncertainty (2.26) and softmax entropy (2.22-2.26) combined with minimal mutual information (0.01-0.04). This failure mode manifests visually in Figure (8.2a) as a nearly perfect line in the uncertainty space, indicating complete loss of the model's ability to differentiate between samples.

However, the method demonstrates remarkable recovery when transitioning from 2-bit to 4-bit precision. At 4-bit, accuracy on MNIST jumps to 0.97, unanimity to 0.97, and uncertainty metrics normalize to levels comparable with higher precision configurations. This sharp performance threshold between 2-bit and 4-bit precision suggests a critical minimum information capacity required for the Parameter-Based approach to function effectively.

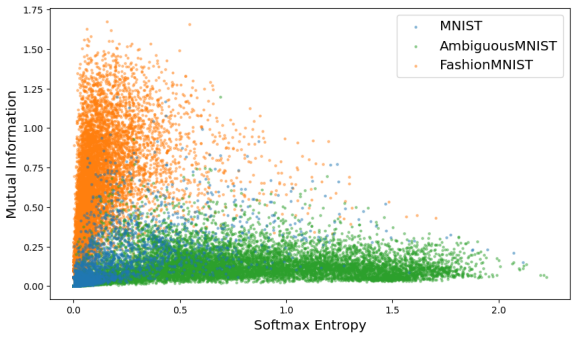
Bit-Width	MNIST					AmbiguousMNIST					FashionMNIST				
	A ↑	U ↑	P.U.	S.E.	M.I.	A ↑	U ↑	P.U.	S.E.	M.I.	A ↑	U ↑	P.U.	S.E.	M.I.
2-bit	0.10	0.18	2.26	2.25	0.01	0.16	0.18	2.26	2.26	0.01	0.10	0.18	2.26	2.22	0.04
4-bit	0.97	0.97	0.09	0.05	0.04	0.50	0.70	0.83	0.71	0.12	0.07	0.75	0.65	0.11	0.54
7-bit	0.97	0.97	0.07	0.05	0.03	0.51	0.72	0.74	0.66	0.08	0.04	0.80	0.51	0.10	0.41
8-bit	0.97	0.97	0.07	0.04	0.03	0.51	0.73	0.73	0.66	0.08	0.07	0.77	0.58	0.16	0.42
16-bit	0.97	0.97	0.07	0.05	0.03	0.51	0.72	0.75	0.67	0.08	0.07	0.80	0.52	0.14	0.38
32-bit (Float)	0.98	0.98	0.05	0.02	0.03	0.51	0.73	0.70	0.63	0.07	0.09	0.77	0.56	0.07	0.49

Table 8.2: Performance metrics of Parameter-Based Quantization Method across different bit-width configurations. ↑ indicates higher values are better. A: Accuracy, U: Unanimity, P.U.: Predictive Uncertainty, S.E.: Softmax Entropy, M.I.: Mutual Information. The table reveals a critical precision threshold between 2-bit and 4-bit configurations. At 2-bit, the method completely fails with near-random accuracy (0.1) and extremely high uncertainty (P.U. 2.26), indicating total loss of model effectiveness. Performance dramatically recovers at 4-bit precision and remains stable across higher bit-widths, with 4-bit exhibiting unexpectedly high mutual information for FashionMNIST (0.54) compared to higher precision configurations.

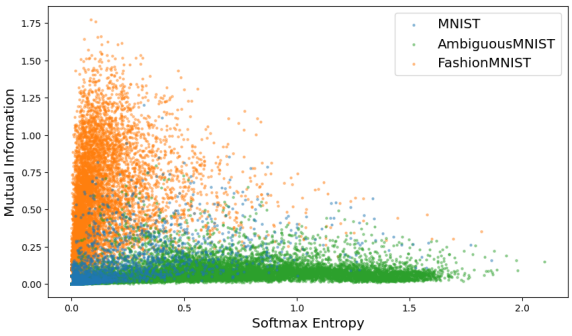
Once this 4-bit threshold is crossed, the method performs consistently across higher precision levels on standard MNIST data, with minimal variations in performance metrics from 4-bit to 32-bit. For ambiguous data (AmbiguousMNIST), the method appropriately main-



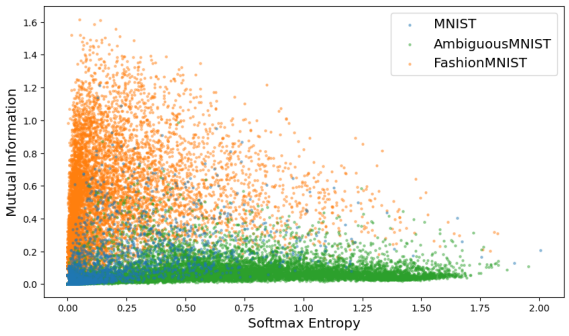
(a) 2-bit



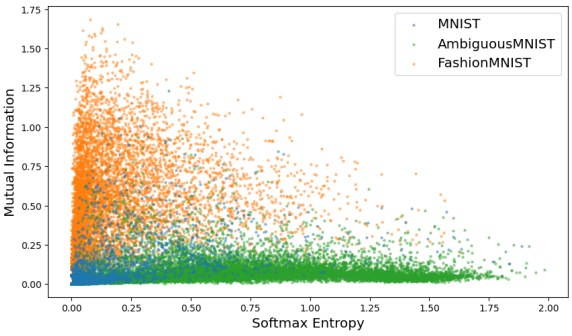
(b) 4-bit



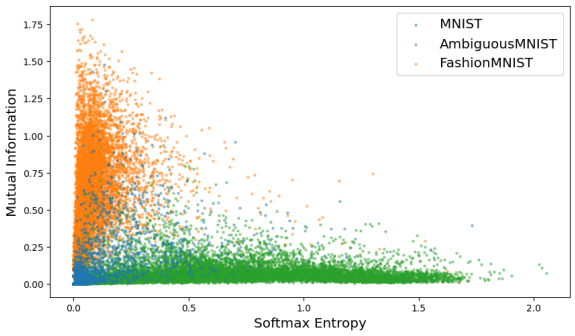
(c) 7-bit



(d) 8-bit



(e) 16-bit



(f) 32-bit (Float)

Figure 8.2: Scatter plots of softmax entropy versus mutual information across different MNIST dataset variants using Parameter-Based Quantization Method. Each subplot represents a different bit-width configuration from 2-bit to 32-bit floating-point. Results show catastrophic failure at 2-bit precision where all dataset classes collapse into a nearly perfect line, indicating complete loss of discriminative uncertainty. Performance rapidly recovers at 4-bit precision and remains relatively stable across higher bit-widths, suggesting Parameter-Based Quantization requires a minimum threshold of 4-bit precision to maintain effective uncertainty characterization.

tains reduced accuracy ($\approx 0.50 - 0.51$) and elevated uncertainty metrics across 4-bit and higher precisions, correctly reflecting the inherent ambiguity in these samples.

For out-of-distribution data (FashionMNIST), Parameter-Based Quantization displays an intriguing precision-dependent behavior in uncertainty decomposition. Notably, the 4-bit configuration exhibits unexpectedly high mutual information (0.54) compared to higher precision configurations (0.38-0.42), suggesting potentially superior epistemic uncertainty quantification for out-of-distribution data at this specific precision level.

The scatter plots in Figure (8.2) confirm the dramatic transition from the completely collapsed uncertainty space at 2-bit precision to well-formed uncertainty representations at 4-bit and above. From 4-bit to 32-bit, the uncertainty distributions remain relatively stable, with subtle variations in the separation between dataset clusters.

In summary, Parameter-Based Quantization exhibits a critical precision threshold between 2-bit and 4-bit configurations, below which the method completely fails and above which it performs comparably to full precision. The unexpectedly strong epistemic uncertainty quantification at 4-bit precision for out-of-distribution data suggests potential advantages of moderate quantization for specific uncertainty-aware applications.

8.5 FULLY INTEGRATED QUANTIZATION EXPERIMENT RESULT AND ANALYSIS

The Fully Integrated Quantization Method, which combines quantization across all model components, demonstrates distinctive performance characteristics and failure modes across different bit-width configurations. Figure (8.3) presents the visualization of uncertainty relationships for this method, while Table (8.3) provides the corresponding performance metrics.

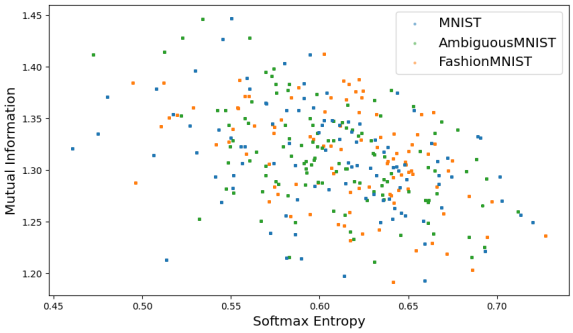
At 2-bit precision, Fully Integrated Quantization exhibits a unique and particularly interesting failure pattern that differs fundamentally from both Sample-Based and Parameter-Based methods. While accuracy collapses to near-random levels (0.06-0.10) across all datasets, similar to Parameter-Based Quantization, the uncertainty metrics reveal a paradoxical combination of moderate softmax entropy (0.61) and high mutual information (1.31). This distinctive uncertainty profile differs significantly from both other methods at the same bit-width, highlighting the complex interactions between aggressive quantization and uncertainty decomposition.

Bit-Width	MNIST					AmbiguousMNIST					FashionMNIST				
	A ↑	U ↑	P.U.	S.E.	M.I.	A ↑	U ↑	P.U.	S.E.	M.I.	A ↑	U ↑	P.U.	S.E.	M.I.
2-bit	0.10	0.27	1.92	0.61	1.31	0.06	0.27	1.92	0.61	1.31	0.10	0.27	1.93	0.61	1.31
4-bit	0.97	0.97	0.08	0.05	0.03	0.50	0.71	0.81	0.71	0.11	0.07	0.73	0.67	0.10	0.57
7-bit	0.97	0.97	0.07	0.05	0.03	0.51	0.72	0.75	0.67	0.08	0.05	0.80	0.51	0.10	0.42
8-bit	0.97	0.98	0.07	0.04	0.03	0.51	0.73	0.73	0.65	0.07	0.07	0.79	0.55	0.15	0.41
16-bit	0.97	0.97	0.07	0.05	0.03	0.51	0.72	0.75	0.67	0.08	0.07	0.79	0.54	0.14	0.40
32-bit (Float)	0.98	0.98	0.05	0.02	0.03	0.51	0.73	0.70	0.63	0.07	0.09	0.77	0.56	0.07	0.49

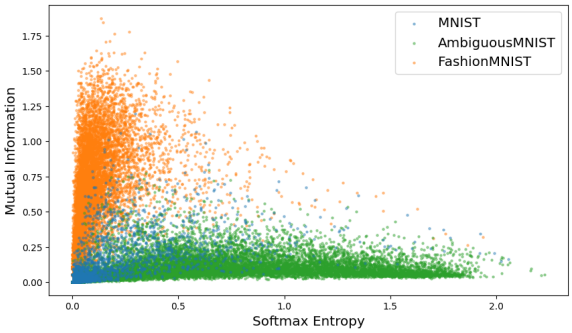
Table 8.3: Performance metrics of Fully Integrated Quantization Method across different bit-width configurations. ↑ indicates higher values are better. A: Accuracy, U: Unanimity, P.U.: Predictive Uncertainty, S.E.: Softmax Entropy, M.I.: Mutual Information. The 2-bit configuration demonstrates a distinctive failure pattern with extremely low accuracy (0.06-0.10) and unanimity (0.27), combined with extraordinarily high predictive uncertainty (1.92-1.93), moderate softmax entropy (0.61), and high mutual information (1.31) across all datasets, indicating poor uncertainty calibration. Performance recovers dramatically at 4-bit precision, with accuracy jumping to 0.97 for MNIST and uncertainty metrics normalizing to levels comparable with higher precision configurations. The 4-bit configuration exhibits particularly strong epistemic uncertainty quantification for FashionMNIST with the highest mutual information (0.57) among all tested bit-widths. From 4-bit to 32-bit, performance remains relatively stable, with the 32-bit floating-point configuration also demonstrating excellent uncertainty quantification, suggesting Fully Integrated Quantization achieves effective performance with minimal precision requirements.

This unique failure mode is clearly visible in Figure (8.3a), where samples across all datasets cluster in a narrow band of high mutual information (1.30-1.45) with limited entropy differentiation (0.45-0.65). This pattern suggests that at 2-bit precision, the quantization errors in the Fully Integrated approach fundamentally disrupt the model's ability to appropriately balance aleatoric and epistemic uncertainty sources.

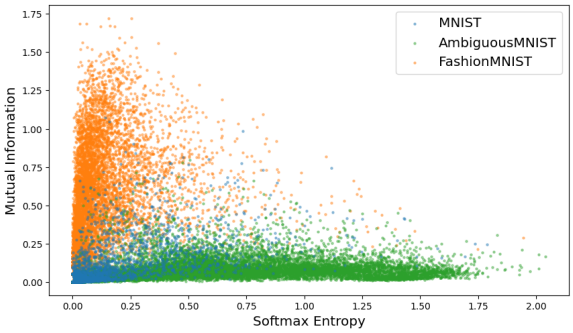
Similar to Parameter-Based Quantization, the Fully Integrated method demonstrates dramatic recovery when transitioning from 2-bit to 4-bit precision. At 4-bit, accuracy on MNIST jumps to 0.97, unanimity to 0.97, and uncertainty metrics normalize to levels comparable with higher precision configurations. This confirms the existence of a critical precision threshold for effective functioning of the method.



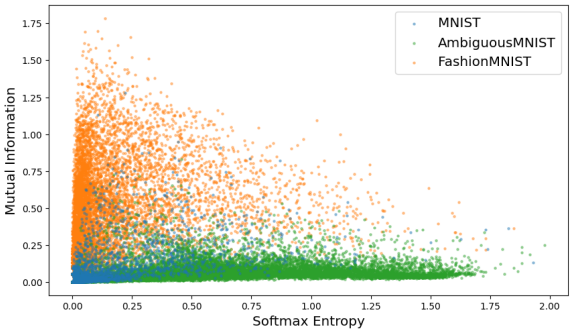
(a) 2-bit



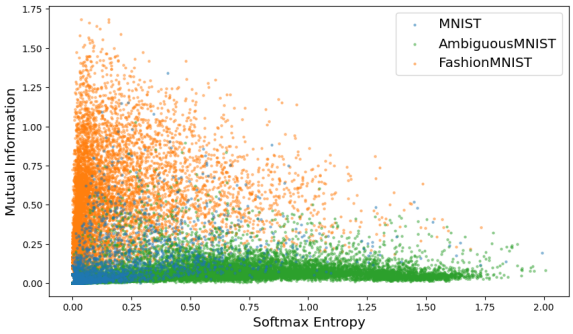
(b) 4-bit



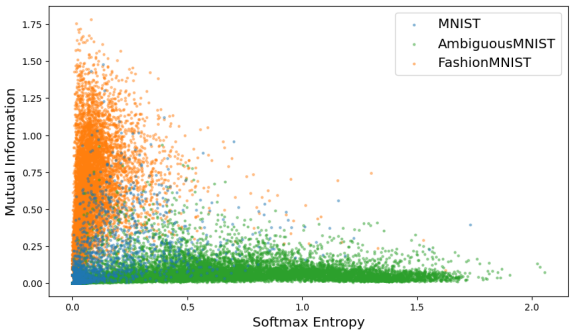
(c) 7-bit



(d) 8-bit



(e) 16-bit



(f) 32-bit (Float)

Figure 8.3: Scatter plots of softmax entropy versus mutual information across different MNIST dataset variants using Fully Integrated Quantization Method. Each subplot represents a different bit-width configuration from 2-bit to 32-bit floating-point. At 2-bit precision, the method exhibits a unique failure mode with all datasets clustered in a narrow band of high mutual information (1.30-1.45) with limited entropy differentiation (0.45-0.65), indicating poor uncertainty calibration. Performance recovers dramatically at 4-bit and higher precisions, where the three datasets form distinct clusters with proper uncertainty decomposition. From 4-bit to 32-bit, the uncertainty distributions remain relatively stable with excellent separation between in-distribution (MNIST), ambiguous (AmbiguousMNIST), and out-of-distribution (FashionMNIST) samples, suggesting Fully Integrated Quantization requires only 4-bit precision to achieve effective uncertainty quantification comparable to full floating-point precision.

Once the 4-bit threshold is crossed, performance stabilizes across higher bit-widths for standard MNIST data. For ambiguous data (AmbiguousMNIST), the method appropriately maintains reduced accuracy ($\approx 0.50 - 0.51$) and elevated uncertainty metrics across 4-bit and higher precisions. The uncertainty decomposition remains relatively consistent across these higher precision levels, suggesting stable uncertainty calibration once sufficient precision is available.

For out-of-distribution data (FashionMNIST), the method shows precision-dependent variations in uncertainty decomposition. The 4-bit configuration exhibits the highest mutual information (0.57) compared to other precision configurations, suggesting superior epistemic uncertainty quantification at this specific precision level.

From 4-bit to 32-bit, the uncertainty distributions remain relatively stable with excellent separation between in-distribution (MNIST), ambiguous (AmbiguousMNIST), and out-of-distribution (FashionMNIST) samples. This suggests that Fully Integrated Quantization requires only 4-bit precision to achieve effective uncertainty quantification comparable to full floating-point precision.

In summary, Fully Integrated Quantization exhibits a distinctive failure mode at extremely low precision characterized by poor uncertainty calibration across all datasets. Once sufficient precision is available (4-bit and above), the method performs comparably to full precision, with potentially advantageous epistemic uncertainty quantification at 4-bit precision for out-of-distribution data, while maintaining stable performance across higher bit-widths.

8.6 PERFORMANCE COMPARISON OF DIFFERENT QUANTIZATION METHODS

Our comprehensive bit-width sensitivity analysis across three distinct quantization methods reveals intriguing patterns that provide valuable insights for deploying BNNs in resource-constrained environments. At the lowest precision level (2-bit), we observe fundamentally different failure modes among the methods. Sample-Based Quantization demonstrates remarkable resilience, maintaining high accuracy (0.97) on MNIST while suffering moderate uncertainty calibration degradation. In stark contrast, Parameter-Based Quantization experiences catastrophic failure with near-random accuracy (0.10) and complete collapse of uncertainty discrimination. Fully Integrated Quantization exhibits a unique intermediate failure pattern with near-random accuracy but a distinctive uncertainty profile combining moderate softmax entropy (0.61) with unusually high mutual information (1.31).

The transition from 2-bit to 4-bit precision represents a critical threshold for all methods. While Sample-Based Quantization shows incremental improvements, both Parameter-Based and Fully Integrated

methods demonstrate dramatic recovery, with performance metrics jumping from near-random to levels comparable with full precision. This sharp transition suggests that 4-bit precision represents a minimum viable threshold for effective uncertainty-aware quantized BNNs when using parameter-based approaches, while sample-based approaches can function effectively even at 2-bit precision if some uncertainty calibration degradation is acceptable.

Across higher precision levels (4-bit to 32-bit), all three methods demonstrate relatively stable performance on standard MNIST data, with minimal variations in accuracy and uncertainty metrics. For ambiguous data (AmbiguousMNIST), all methods appropriately maintain reduced accuracy (≈ 0.50) and elevated uncertainty metrics, correctly reflecting the inherent ambiguity in these samples. The most significant differentiation occurs in out-of-distribution data (FashionMNIST), where interestingly, all three methods exhibit non-monotonic relationships between bit-width and uncertainty decomposition. The 4-bit configurations consistently demonstrate superior epistemic uncertainty quantification with higher mutual information compared to intermediate bit-widths (8-bit, 16-bit).

This non-monotonic relationship between precision and uncertainty calibration represents a significant finding with important implications for BNN deployment. It suggests that simply maximizing precision does not necessarily optimize uncertainty estimation, and targeted precision selection based on specific application requirements may be more effective. The consistent pattern of 4-bit precision offering advantages for epistemic uncertainty quantification across all methods is particularly noteworthy and warrants further investigation into the theoretical underpinnings of this phenomenon.

In terms of practical deployment recommendations, our analysis suggests that Sample-Based Quantization offers the best overall resilience to aggressive quantization, making it ideal for extremely resource-constrained environments. For applications requiring well-calibrated uncertainty estimates, 4-bit precision represents an optimal trade-off point across all methods, offering uncertainty calibration comparable or sometimes superior to full precision while significantly reducing memory and computational requirements.

This comparative analysis demonstrates that effective BNN deployment in resource-constrained environments requires careful consideration of both the quantization method and specific bit-width configuration, with precision requirements potentially differing based on whether predictive accuracy or specific aspects of uncertainty estimation are prioritized. These findings establish empirical guidelines for precision selection in quantized BNNs and highlight the complex interplay between quantization strategy, bit-width allocation, and uncertainty calibration in Bayesian Deep Learning.

9 POTENTIAL FUTURE WORK

DISTINGUISHING QUANTIZATION EFFECTS ON UNCERTAINTY TYPES A critical research direction worth exploring is the development of methodologies to better differentiate how quantization errors specifically impact epistemic (model) uncertainty versus aleatoric (data) uncertainty. Current approaches do not adequately address how discretization effects might disproportionately influence these distinct uncertainty types.

Our empirical results reveal significant performance degradation under 2-bit constraints across Sample-Based Quantization Methods, Parameter-Based Quantization Methods, and Fully Integrated Quantization Methods. However, their impact on uncertainty estimation varies considerably. The quantization errors introduced when discretizing the distribution parameters fundamentally differ from those arising during the quantization of distribution samples. Furthermore, our analysis indicates that quantization errors manifest differently across uncertainty types, necessitating a more nuanced approach to uncertainty-aware quantization.

Future work should investigate theoretical frameworks and empirical methodologies to isolate and measure these effects separately. This could involve developing specialized test cases with controlled aleatoric and epistemic uncertainty components, along with appropriate metrics to quantify how different quantization strategies affect each uncertainty type. Understanding these relationships would enable the development of targeted quantization strategies that preserve critical uncertainty information while maintaining computational efficiency.

PARAMETER–SPECIFIC QUANTIZATION STRATEGIES FOR BAYESIAN DISTRIBUTIONS Our analysis reveals that different parameters of weight distributions exhibit varying sensitivity to bit-width constraints, with distribution parameters demonstrating significantly higher sensitivity to low bit-width quantization than weight samples. Future research should investigate decomposed quantization approaches that treat distribution means (μ) and standard deviations (σ) with distinct quantization strategies optimized for their specific properties.

The mean parameters, which directly influence model predictions, require stable representation across bit-width reductions, while standard deviation parameters, which govern uncertainty estimation, are more sensitive to discretization artifacts. A promising direction involves developing heterogeneous quantization schemes where μ and σ utilize different bit allocations and quantization functions. For in-

stance, μ could employ signed asymmetric quantization with higher bit-width allocation to maintain prediction accuracy, while σ might benefit from logarithmic quantization schemes that better preserve the exponential relationships inherent in uncertainty calculations.

Additionally, exploiting the inherent correlations between μ and σ in Bayesian weight distributions could enable joint quantization strategies that optimize uncertainty-accuracy trade-offs more effectively than current uniform approaches. This could include developing learned quantization functions that adapt to the joint distribution characteristics of these parameters, or implementing dynamic scaling factors that adjust based on the local properties of the weight distribution in parameter space.

QUANTITATIVE ASSESSMENT OF UNCERTAINTY QUALITY While the Noisy Sine Wave dataset provides excellent visualization opportunities for intuitive understanding of uncertainty estimation quality, our work highlights the need for more rigorous quantitative metrics specifically designed to evaluate in-distribution versus out-of-distribution uncertainty in regression tasks. Future work should focus on developing standardized metrics that quantify the calibration of predictive uncertainty across the input domain, particularly at decision boundaries and extrapolation regions. Creating domain-specific evaluation frameworks that assess the practical utility of uncertainty estimates in downstream decision-making tasks would also be valuable.

Unlike classification tasks where metrics such as Softmax Entropy and Mutual Information are well-established for uncertainty evaluation, regression tasks currently lack standardized metrics for comprehensively assessing uncertainty quality, especially for distinguishing between in-distribution and out-of-distribution uncertainties. Such advancements would facilitate more objective comparison between different quantization methods and their effects on uncertainty estimation quality in regression settings.

MIXED-PRECISION STRATEGIES FOR THE FINAL LINEAR LAYER Our experimental findings highlight the critical bottleneck imposed by input-output quantization in the final linear layer, where logit distributions exhibit significant variations across different datasets. The dramatic dynamic range differences observed—particularly FashionMNIST's 1,649-unit range versus AmbiguousMNIST's 535-unit range (as detailed in Appendix a.2, Table a.2)—necessitate more sophisticated quantization approaches that preserve fine-grained probability distinctions essential for uncertainty estimation.

Future work should explore mixed-precision architectures that dynamically allocate bit-width based on layer-specific requirements and data characteristics. For the final linear layer, this could involve implementing adaptive precision allocation schemes that increase bit-

width for output computations when detecting high-dynamic-range inputs, while maintaining standard precision for typical cases. Such approaches might leverage statistical analysis of activation distributions to predict optimal bit-width allocations on a per-sample or per-batch basis.

Furthermore, developing uncertainty-aware quantization policies that prioritize preserving probability mass in high-uncertainty regions could significantly improve calibration. This could include implementing gradient-sensitive quantization that allocates higher precision to inputs and outputs with larger gradient magnitudes, or entropy-guided quantization that increases precision in regions where the model exhibits higher predictive uncertainty. Additionally, investigating hardware-software co-design approaches that enable efficient dynamic precision switching on neuromorphic devices would be crucial for practical deployment of these advanced quantization strategies.

ADAPTIVE ENCODING STRATEGIES FOR INPUT REPRESENTATION While our sectional Thermometer Encoding method demonstrates significant efficacy in representing low-dimensional continuous data with preserved non-negative mapping and relative numerical relationships, our findings indicate that encoding strategies substantially influence the quality of uncertainty in Bayesian Neural Networks. The current fixed partitioning approach, though effective, still exhibits limitations in adapting to the underlying data distribution.

The optimal encoding strategy remains an open research question worthy of further investigation. A promising direction involves developing adaptive encoding methodologies that dynamically adjust sectioning parameters based on data characteristics. Such adaptive approaches could utilize density-based partitioning that allocates more sections to densely populated regions of the input space, thereby enhancing resolution where uncertainty quantification is most critical. Additionally, employing information-theoretic principles to maximize mutual information between input encodings and posterior predictive distributions could potentially lead to more calibrated uncertainty estimates. Furthermore, addressing the sudden transitions between sections that currently exist would resolve discontinuities in the encoded representation that obscure the relationship between encoding strategies and uncertainty distribution patterns.

These adaptive strategies could significantly enhance the models capacity to represent uncertainty accurately across varying data distributions without requiring manual tuning of encoding parameters for each application domain.

INTEGRATION OF KOLMOGOROV–ARNOLD NETWORKS WITH BAYESIAN FRAMEWORKS Our exploration of activation function optimization for quantized models revealed persistent sensitivity issues in Bayesian

Neural Networks. Despite improvements achieved through Alpha-Sigmoid and Alpha-Sine functions, fundamental limitations remain in the fixed functional forms of traditional activation functions.

Kolmogorov-Arnold Networks (KANs) present an intriguing alternative worthy of investigation (Liu et al., 2025). By learning the activation functions themselves rather than merely optimizing parameters of predefined functions, KANs offer potential advantages for integration with Bayesian frameworks. The adaptive nature of KANs could potentially mitigate quantization errors by discovering activation functions that are inherently robust to discretization effects. A Bayesian formulation of KANs would enable uncertainty quantification over both network weights and activation function parameters, potentially yielding more comprehensive uncertainty estimates. Furthermore, the mathematical foundations of KANs, based on the Kolmogorov-Arnold representation theorem, may provide theoretical guarantees regarding approximation capabilities even under quantization constraints.

Implementation challenges include developing efficient variational inference methods compatible with the KAN architecture and ensuring computational tractability when sampling from the posterior distribution of both weights and activation parameters.

HARDWARE IMPLEMENTATION AND EFFICIENCY ANALYSIS A significant limitation of our current work is the absence of hardware simulation and evaluation of quantized Bayesian Neural Networks. Future research should address this gap by implementing the proposed quantization methods on specialized hardware accelerators to measure actual energy consumption and computational overhead. Analyzing the trade-offs between precision, uncertainty quality, and hardware efficiency across different deployment scenarios would provide crucial insights. Additionally, exploring hardware-aware training techniques that jointly optimize model architecture and quantization parameters for specific hardware constraints could lead to more efficient implementations.

This analysis would provide valuable insights into the practical applicability of quantized Bayesian Neural Networks in resource-constrained environments, such as edge devices and embedded systems, where uncertainty-aware decision making is increasingly important. By addressing these research directions, future work can build upon our foundations to develop more efficient, accurate, and reliable quantized Bayesian Neural Networks suitable for deployment in real-world applications requiring robust uncertainty quantification.

10 CONCLUSION

This thesis has investigated the integration of Bayesian Neural Networks with quantization techniques for uncertainty-aware deep learning in resource-constrained environments. Our comprehensive exploration across multiple levels of analysis has yielded several significant contributions to the field.

We have demonstrated that effective input representation is critical for uncertainty preservation in quantized models, with our proposed sectional Thermometer Encoding showing particular promise for low-dimensional continuous data. This encoding strategy successfully maintains the relative numerical relationships while supporting efficient implementation in quantized architectures.

Our investigation into various quantization approaches demonstrated that all three quantization levels—Sample-Based, Parameter-Based, and Fully Integrated Quantization—can achieve comparable performance under specific hardware architecture bit-width constraints. Rather than one approach consistently outperforming others, each method exhibits unique strengths depending on the implementation context and specific uncertainty requirements. These findings provide practical guidance for selecting appropriate quantization strategies based on particular hardware constraints and application needs.

We developed optimization strategies specifically for quantized SVI, including the Alpha-Sigmoid and Alpha-Sine activation functions along with specialized weight clipping techniques. These strategies significantly improved the robustness of uncertainty estimates under quantization constraints. Our bit-width sensitivity analysis established that 4-bit precision represents the effective minimum for maintaining acceptable uncertainty quality with our optimized approaches. Notably, we identified that the larger standard deviation values in weight distributions play a crucial role in preserving uncertainty information, providing partial theoretical justification for logarithmic quantization approaches in Bayesian Neural Networks.

Through empirical evaluation on diverse datasets, we have shown that quantized Bayesian Neural Networks can successfully capture both aleatoric and epistemic uncertainty, even in challenging out-of-distribution scenarios. The practical implications of this work extend to numerous applications where reliable uncertainty estimation must coexist with computational efficiency requirements.

In conclusion, this research advances the state of the art in efficient uncertainty quantification for deep learning. By bridging the

gap between Bayesian methods and quantization techniques, we provided a foundation for deploying uncertainty-aware models in edge computing, embedded systems, and other constrained environments. Future research directions, including distinguishing quantization effects on uncertainty types, parameter-specific quantization strategies for Bayesian distributions, quantitative assessment of uncertainty quality, and mixed-precision strategies for the final linear layer, promise to further enhance this emerging field at the intersection of Bayesian Inference and neural network quantization.

****

a APPENDIX

A.1 ERRORS IN DECIMALS OF DIFFERENT LENGTHS

decimal length	$error_{average}$	$error_{max}$
5	0.01595	0.03123
15	0.0000149	0.00003
23	0.0	0.0
25	0.0	0.0

Table a.1: Error analysis for different decimal length representations in fixed-point number format

Table (a.1) presents an analysis of errors introduced during the conversion process of decimal numbers with different precision levels to fixed-point representation.

The error values were calculated by measuring the absolute difference between the original floating-point values and their corresponding values after conversion to fixed-point format and back to floating-point. This allows us to quantify the precision loss during the conversion process.

For our error metrics, we computed both the average error ($error_{average}$) across all test samples and the maximum error ($error_{max}$) observed in the dataset.

The results show a clear pattern: as the decimal length increases, the conversion error decreases significantly. With 5 decimal places, we observe an average error of 0.01595, which diminishes to just 0.0000149 when using 15 decimal places.

Interestingly, when we extend the decimal representation to 23 places or beyond, the error effectively becomes zero. This phenomenon can be explained by the underlying storage format of our dataset. Since our data is stored in float32 format, which provides approximately 23 bits of precision for the mantissa, representing decimals beyond this limit does not capture any additional information that can be stored in the IEEE 754 single-precision format.

Therefore, the zero error values for 23 and 25 decimal places indicate that we have reached the inherent precision limit of the float32 format. At this point, the neural network would process identical values whether we use the original floating-point representation or the fixed-point conversion, making the two formats functionally equivalent for training purposes.

A.2 INPUT-OUTPUT QUANTITATIVE IMPACT ANALYSIS

While complete quantization achieved near full-precision performance on the Noisy Sine Wave and Two Moons datasets, the complexity of the Dirty MNIST dataset presents significant challenges when both input and output quantization are applied to the final linear layer before computing logits. The successful deployment of quantized neural networks on neuromorphic hardware depends fundamentally on understanding the differential impacts of quantization across model layers. For Bayesian Neural Networks, the logit computation preceding uncertainty estimation is particularly sensitive to quantization errors.

Our analysis employs the Sample-Based Quantization method described in Section (6.1), which applies quantization after sampling from weight distributions. As illustrated in Figure (6.7), this approach implements 5-bit unsigned input quantization, 7-bit signed weight quantization, and 8-bit signed output quantization to ensure compatibility with BSS-2 hardware specifications. The quantization function is defined as:

$$x_q = s \cdot \text{clamp}\left(\left\lfloor \frac{x}{s} \right\rceil + z, n_{min}, n_{max}\right) \tag{a.1}$$

where s represents the scaling factor determined through statistical analysis of activation distributions, and z denotes the zero-point offset. For output quantization, these parameters critically influence the precision of logit representation in the final layer.

The core challenge arises from the sensitivity of uncertainty calculations to quantization errors. The softmax operation required for entropy computation exhibits extreme sensitivity to small variations in logit values:

$$p_i = \frac{e^{z_{i,q}}}{\sum_{j=1}^{10} e^{z_{j,q}}} \tag{a.2}$$

where $z_{i,q}$ represents the quantized logit for class i. When operating under 8-bit signed output constraints, the limited effective dynamic range must accommodate logit distributions spanning different characteristics across MNIST, AmbiguousMNIST, and FashionMNIST datasets. This creates a fundamental tension between preserving fine-grained probability differences and avoiding overflow at distribution extremes.

We examined four distinct quantization strategies for the final linear layer: (1) complete quantization implementing all specified bit-widths for all layers, (2) weight-only quantization where all linear layers maintain full precision inputs and outputs while retaining weight quantization, (3) elimination of output quantization in the final layer only while implementing full quantization (input, weight, and output)

for all other layers, and (4) removal of both input and output quantization in the final layer while implementing full quantization for all other layers.

EXPERIMENT SETTING All experiments shared these settings:

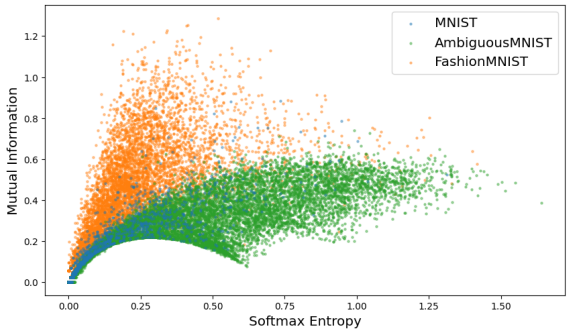
- **Dataset:** DirtyMNIST is used for training, MNIST, AmbiguousMNIST, and FashionMNIST are used for testing, and the batch size is set to 100.
- **Model:** MLP with input representation obtained by shifting the scalar input to be non-negative (subtracting the minimum value) (based on our findings in Section 5.5), two hidden layers of 100 neurons each, SoftPlus activation (based on our findings in Section 7.1).
- **Training:** 1500 epochs pretraining (lr=0.001), 1,000 epochs main training (lr=0.0001), 1 particle for training, random_seed is set to 42.
- **KL Annealing:** Maximum weight 0.25, with an end-focused schedule that keeps the KL loss nearly inactive in early epochs and ramps it up only near the end of training.

The weight quantization of each model uses the same 7-bit signed Sample-Based quantization method. The key distinction lies in the configuration of input and output quantization for the final linear layer.

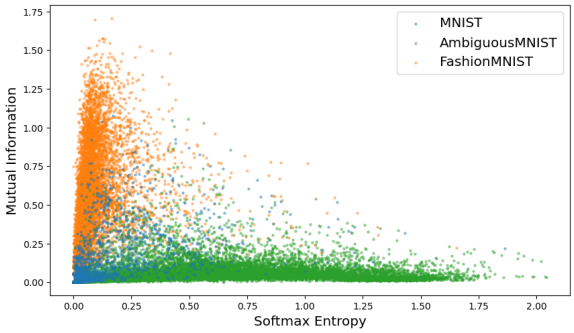
EXPERIMENT RESULTS AND ANALYSIS Figure (a.1) and Table (a.3) present a comprehensive evaluation of how different input-output quantization strategies affect Bayesian Neural Network performance on uncertainty estimation tasks. The complete quantization scenario (Figure a.1a) reveals severe degradation in uncertainty decomposition, with FashionMNIST samples exhibiting artificially elevated epistemic uncertainty (Mutual Information reaching 1.3) and the complete absence of the characteristic vertical separation pattern expected for well-calibrated uncertainty. The quantization artifacts manifest as increased dispersion across both uncertainty dimensions, fundamentally compromising the model's ability to discriminate between aleatoric and epistemic uncertainty sources.

Weight-only quantization (Figure a.1b) dramatically restores uncertainty calibration by eliminating input-output quantization while preserving weight quantization. The distinct clustering pattern emerges with clear vertical separation, indicating preserved capability to distinguish between uncertainty types. FashionMNIST samples appropriately cluster in the high epistemic uncertainty region (maximal values around 1.7), while MNIST samples maintain their expected low-uncertainty positioning. This configuration demonstrates that

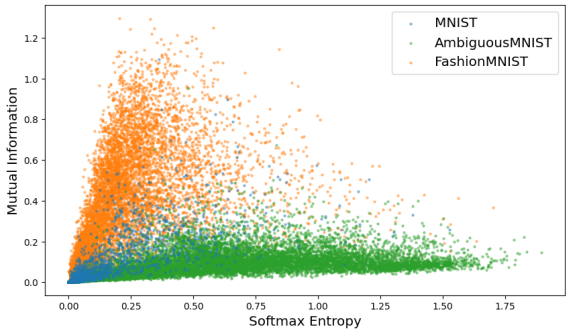
7-bit weight quantization proves sufficient for maintaining model performance while achieving computational efficiency.



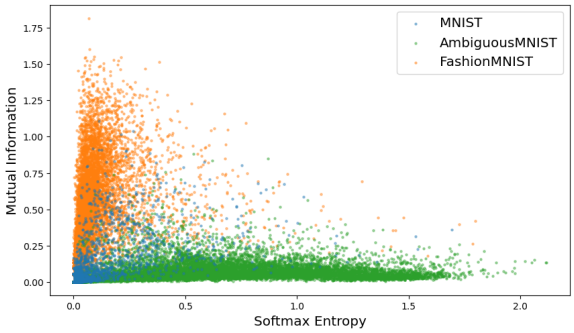
(a) Complete Quantization (All Layers)



(b) Weight-Only Quantization (All Layers)



(c) Final Layer: No Output Quantization



(d) Final Layer: No Input-Output Quantization

Figure a.1: Uncertainty decomposition analysis under different input-output quantization strategies: The scatter plots illustrate the relationship between epistemic uncertainty (Mutual Information) and aleatoric uncertainty (Softmax Entropy) across MNIST, AmbiguousMNIST, and FashionMNIST test sets. While configurations (b) and (d), which eliminate both input and output quantization for the entire network or final layer respectively, achieve excellent cluster definition with clear dataset separation, configuration (c), which preserves output quantization in the final layer, fails to establish the distinctive vertical separation pattern, indicating that output quantization presents a more critical bottleneck for uncertainty estimation accuracy than input quantization.

The final layer no output quantization configuration (Figure a.1c) shows only partial recovery from complete quantization degradation. While cluster separation improves compared to complete quantization, the vertical separation pattern remains diffuse, particularly for FashionMNIST samples. The persistence of 5-bit input quantization in the final layer introduces discernible artifacts, resulting in elevated softmax entropy (0.19 vs baseline 0.04) and reduced epistemic uncertainty (0.32 vs baseline 0.37), suggesting that input quantization, though

less critical than output quantization, still contributes to uncertainty estimation degradation.

The final layer no input quantization and no output quantization configuration (Figure a.1d) yields results nearly indistinguishable from weight-only quantization, with optimal uncertainty decomposition showing clear dataset separation and well-defined vertical alignment of uncertainty values. The quantitative metrics confirm this similarity, with epistemic uncertainty for FashionMNIST at 0.39 compared to 0.40 for weight-only quantization and 0.37 for the baseline. This near-perfect correspondence demonstrates that activation precision in the final layer is critical for maintaining uncertainty hierarchy across diverse datasets.

Dataset	Range	Standard Deviation	99.9% Percentile Range	Dynamic Range
MNIST	[-604.40, 765.30]	63.53	[-315.72, 261.86]	1,369.7
AmbiguousMNIST	[-342.82, 192.07]	18.08	[-117.21, 67.40]	534.9
FashionMNIST	[-1,007.72, 641.52]	96.76	[-439.06, 368.14]	1,649.3

Table a.2: Logit distribution statistics across different datasets. The dramatic variations in dynamic range create a fundamental quantization challenge when constrained to 8-bit signed output representation. FashionMNIST presents the most extreme quantization requirements, with a dynamic range exceeding 1,649 units.

The logit distribution analysis (Table a.2) reveals the fundamental challenge underlying these experimental observations. FashionMNIST's dynamic range of 1,649.3 units requires a quantization scale factor of approximately 6.46 units per level when constrained to 8-bit signed output—catastrophically coarse for preserving the subtle probability distinctions necessary for reliable uncertainty estimation. In contrast, AmbiguousMNIST's compressed distribution (range $\approx$ 535 units) reflects the dataset's inherent ambiguity, while MNIST's wide range indicates the model's high confidence in its predictions. This differential in logit characteristics across datasets explains why complete quantization particularly degrades FashionMNIST performance, as the coarse quantization steps cannot adequately represent the fine-grained probability differences required for proper uncertainty decomposition.

The quantitative metrics (Table a.3) corroborate these visual and theoretical findings with precise measurements. Complete quantization causes substantial degradation in uncertainty metrics, most notably for AmbiguousMNIST where softmax entropy drops from 0.64 to 0.44, and FashionMNIST where epistemic uncertainty (M.I.) decreases from 0.37 to 0.30 while aleatoric uncertainty inappropriately increases. This paradoxical result—where quantization simultaneously reduces perceived model uncertainty while increasing data-related uncertainty—indicates fundamental calibration failure. Configurations preserving activation precision (rows 3-5) maintain baseline-like per-

formance across all metrics, with the remarkable similarity between weight-only quantization and final layer no input-output quantization (M.I. values of 0.40 and 0.39 respectively for FashionMNIST) conclusively demonstrating that output quantization represents the primary bottleneck for uncertainty estimation accuracy.

Method	MNIST					AmbiguousMNIST					FashionMNIST				
	A ↑	U ↑	P.U.	S.E.	M.I.	A ↑	U ↑	P.U.	S.E.	M.I.	A ↑	U ↑	P.U.	S.E.	M.I.
Full Precision (Baseline)	0.98	0.98	0.05	0.02	0.03	0.51	0.73	0.72	0.64	0.07	0.07	0.84	0.41	0.04	0.37
Complete Quantization (All Layers)	0.97	0.98	0.05	0.02	0.03	0.51	0.72	0.68	0.44	0.24	0.05	0.84	0.39	0.09	0.30
Weight-Only Quantization (All Layers)	0.97	0.98	0.05	0.03	0.03	0.51	0.74	0.71	0.64	0.07	0.04	0.81	0.47	0.07	0.40
Final Layer: No Output Quantization	0.98	0.98	0.05	0.03	0.02	0.51	0.73	0.72	0.63	0.09	0.05	0.80	0.50	0.19	0.32
Final Layer: No Input-Output Quantization	0.98	0.98	0.05	0.03	0.03	0.51	0.73	0.72	0.65	0.07	0.04	0.81	0.47	0.08	0.39

Table a.3: Performance comparison of different input-output quantization strategies across MNIST test datasets. ↑ indicates that higher values are better. A: Accuracy, U: Unanimity, P.U.: Predictive Uncertainty, S.E.: Softmax Entropy, M.I.: Mutual Information. Results demonstrate that complete input-output quantization (row 2) causes significant degradation in uncertainty metrics, particularly for FashionMNIST where epistemic uncertainty (M.I.) drops from 0.37 to 0.30. Configurations eliminating input-output quantization (rows 3-5) maintain performance closer to the baseline, with row 3 (Weight-Only) and row 5 (Final Layer: No Input-Output) achieving nearly identical metrics, confirming output quantization as the primary bottleneck for uncertainty estimation accuracy.

BIBLIOGRAPHY

Akram, Mohsin, Muhammad Adnan, Syed Ali, Jameel Ahmad, Amr Yousef, Tagrid Alshalali, and Zaffar Shaikh (Jan. 2025). "Uncertainty-aware diabetic retinopathy detection using deep learning enhanced by Bayesian approaches." In: *Scientific Reports* 15. DOI: 10.1038/s41598-024-84478-x.

Attias, Hagai (1999). "A Variational Baysian Framework for Graphical Models." In: *Advances in Neural Information Processing Systems*. Ed. by S. Solla, T. Leen, and K. Müller. Vol. 12. MIT Press. URL: https://proceedings.neurips.cc/paper_files/paper/1999/file/74563ba21a90da13dacf2a73e3ddefa7-Paper.pdf.

Banner, Ron, Yury Nahshan, Elad Hoffer, and Daniel Soudry (2019). *Post-training 4-bit quantization of convolution networks for rapid-deployment*. arXiv: 1810.05723 [cs.CV]. URL: https://arxiv.org/abs/1810.05723.

Bengio, Yoshua, Nicholas Léonard, and Aaron Courville (2013). *Estimating or Propagating Gradients Through Stochastic Neurons for Conditional Computation*. arXiv: 1308.3432 [cs.LG]. URL: https://arxiv.org/abs/1308.3432.

Bingham, Eli, Jonathan P. Chen, Martin Jankowiak, Fritz Obermeyer, Neeraj Pradhan, Theofanis Karaletsos, Rohit Singh, Paul Szerlip, Paul Horsfall, and Noah D. Goodman (2018). *Pyro: Deep Universal Probabilistic Programming*. arXiv: 1810.09538 [cs.LG]. URL: https://arxiv.org/abs/1810.09538.

Bishop, Christopher M. (2006). *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Berlin, Heidelberg: Springer-Verlag. ISBN: 0387310738.

Blei, David M., Alp Kucukelbir, and Jon D. McAuliffe (Apr. 2017). "Variational Inference: A Review for Statisticians." In: *Journal of the American Statistical Association* 112.518, pp. 859–877. ISSN: 1537-274X. DOI: 10.1080/01621459.2017.1285773. URL: http://dx.doi.org/10.1080/01621459.2017.1285773.

Blott, Michaela, Thomas Preusser, Nicholas Fraser, Giulio Gambardella, Kenneth O'Brien, and Yaman Umuroglu (2018). *FINN-R: An End-to-End Deep-Learning Framework for Fast Exploration of Quantized Neural Networks*. arXiv: 1809.04570 [cs.AR]. URL: https://arxiv.org/abs/1809.04570.

Blundell, Charles, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra (2015). *Weight Uncertainty in Neural Networks*. arXiv: 1505.05424 [stat.ML]. URL: https://arxiv.org/abs/1505.05424.

Bowman, Samuel R., Luke Vilnis, Oriol Vinyals, Andrew M. Dai, Rafal Jozefowicz, and Samy Bengio (2016). *Generating Sentences*

from a Continuous Space. arXiv: 1511.06349 [cs.LG]. URL: https://arxiv.org/abs/1511.06349.

Buckman, Jacob, Aurko Roy, Colin Raffel, and Ian Goodfellow (2018). "Thermometer Encoding: One Hot Way To Resist Adversarial Examples." In: *International Conference on Learning Representations*. URL: https://openreview.net/forum?id=S18Su--CW.

Carpenter, Bob, Andrew Gelman, Matthew D. Hoffman, Daniel Lee, Ben Goodrich, Michael Betancourt, Marcus Brubaker, Jiqiang Guo, Peter Li, and Allen Riddell (2017). "Stan: A Probabilistic Programming Language." In: *Journal of Statistical Software* 76.1, pp. 1–32. DOI: 10.18637/jss.v076.i01. URL: https://www.jstatsoft.org/index.php/jss/article/view/v076i01.

Chen, Xi, Diederik P. Kingma, Tim Salimans, Yan Duan, Prafulla Dhariwal, John Schulman, Ilya Sutskever, and Pieter Abbeel (2017). *Variational Lossy Autoencoder*. arXiv: 1611.02731 [cs.LG]. URL: https://arxiv.org/abs/1611.02731.

Depeweg, Stefan, José Miguel Hernández-Lobato, Finale Doshi-Velez, and Steffen Udluft (2018). *Decomposition of Uncertainty in Bayesian Deep Learning for Efficient and Risk-sensitive Learning*. arXiv: 1710.07283 [stat.ML]. URL: https://arxiv.org/abs/1710.07283.

Dettmers, Tim, Mike Lewis, Younes Belkada, and Luke Zettlemoyer (2022). *LLM.int8(): 8-bit Matrix Multiplication for Transformers at Scale*. arXiv: 2208.07339 [cs.LG]. URL: https://arxiv.org/abs/2208.07339.

Edelsbrunner, H., D. Kirkpatrick, and R. Seidel (1983). "On the shape of a set of points in the plane." In: *IEEE Transactions on Information Theory* 29.4, pp. 551–559. DOI: 10.1109/TIT.1983.1056714.

Fakhfakh, Mohamed and Lotfi Chaari (2023). *Bayesian optimization for sparse neural networks with trainable activation functions*. arXiv: 2304.04455 [cs.LG]. URL: https://arxiv.org/abs/2304.04455.

Figurnov, Michael, Shakir Mohamed, and Andriy Mnih (2019). *Implicit Reparameterization Gradients*. arXiv: 1805.08498 [cs.LG]. URL: https://arxiv.org/abs/1805.08498.

Foong, Andrew Y. K., David R. Burt, Yingzhen Li, and Richard E. Turner (2020). *On the Expressiveness of Approximate Inference in Bayesian Neural Networks*. arXiv: 1909.00719 [stat.ML]. URL: https://arxiv.org/abs/1909.00719.

Frantar, Elias, Saleh Ashkboos, Torsten Hoefler, and Dan Alistarh (2023). *GPTQ: Accurate Post-Training Quantization for Generative Pre-trained Transformers*. arXiv: 2210.17323 [cs.LG]. URL: https://arxiv.org/abs/2210.17323.

Fu, Hao, Chunyuan Li, Xiaodong Liu, Jianfeng Gao, Asli Celikyilmaz, and Lawrence Carin (2019). *Cyclical Annealing Schedule: A Simple Approach to Mitigating KL Vanishing*. arXiv: 1903.10145 [cs.LG]. URL: https://arxiv.org/abs/1903.10145.

Gal, Yarin and Zoubin Ghahramani (20–22 Jun 2016). "Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning." In: *Proceedings of The 33rd International Conference on Machine Learning*. Ed. by Maria Florina Balcan and Kilian Q. Weinberger. Vol. 48. Proceedings of Machine Learning Research. New York, New York, USA: PMLR, pp. 1050–1059. URL: https://proceedings.mlr.press/v48/gal16.html.

Gawlikowski, Jakob et al. (2022). *A Survey of Uncertainty in Deep Neural Networks*. arXiv: 2107.03342 [cs.LG]. URL: https://arxiv.org/abs/2107.03342.

Gholami, Amir, Sehoon Kim, Zhen Dong, Zhewei Yao, Michael W. Mahoney, and Kurt Keutzer (2021). *A Survey of Quantization Methods for Efficient Neural Network Inference*. arXiv: 2103.13630 [cs.CV]. URL: https://arxiv.org/abs/2103.13630.

Goodman, Noah, Vikash Mansinghka, Daniel M. Roy, Keith Bonawitz, and Joshua B. Tenenbaum (2014). *Church: a language for generative models*. arXiv: 1206.3255 [cs.PL]. URL: https://arxiv.org/abs/1206.3255.

Graves, Alex (2011). "Practical Variational Inference for Neural Networks." In: *Advances in Neural Information Processing Systems*. Ed. by J. Shawe-Taylor, R. Zemel, P. Bartlett, F. Pereira, and K.Q. Weinberger. Vol. 24. Curran Associates, Inc. URL: https://proceedings.neurips.cc/paper_files/paper/2011/file/7eb3c8be3d411e8ebfab08eba5f49632-Paper.pdf.

Guo, Chuan, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger (2017). *On Calibration of Modern Neural Networks*. arXiv: 1706.04599 [cs.LG]. URL: https://arxiv.org/abs/1706.04599.

Gustafsson, Fredrik K., Martin Danelljan, and Thomas B. Schön (2020). *Evaluating Scalable Bayesian Deep Learning Methods for Robust Computer Vision*. arXiv: 1906.01620 [cs.LG]. URL: https://arxiv.org/abs/1906.01620.

Higgins, Irina, Loïc Matthey, Arka Pal, Christopher P. Burgess, Xavier Glorot, Matthew M. Botvinick, Shakir Mohamed, and Alexander Lerchner (2017). "beta-VAE: Learning Basic Visual Concepts with a Constrained Variational Framework." In: *ICLR (Poster)*. OpenReview.net. URL: http://dblp.uni-trier.de/db/conf/iclr/iclr2017.html#HigginsMPBGBML17.

Hoffman, Matt, David M. Blei, Chong Wang, and John Paisley (2013). *Stochastic Variational Inference*. arXiv: 1206.7051 [stat.ML]. URL: https://arxiv.org/abs/1206.7051.

Homan, Matthew D. and Andrew Gelman (Jan. 2014). "The No-U-turn sampler: adaptively setting path lengths in Hamiltonian Monte Carlo." In: *J. Mach. Learn. Res.* 15.1, pp. 1593–1623. ISSN: 1532-4435.

Hubara, Itay, Matthieu Courbariaux, Daniel Soudry, Ran El-Yaniv, and Yoshua Bengio (2016). *Quantized Neural Networks: Training Neural*

Networks with Low Precision Weights and Activations. arXiv: 1609.07061 [cs.NE]. URL: https://arxiv.org/abs/1609.07061.

Hüllermeier, Eyke and Willem Waegeman (Mar. 2021). "Aleatoric and epistemic uncertainty in machine learning: an introduction to concepts and methods." In: *Machine Learning* 110.3, pp. 457–506. ISSN: 1573-0565. DOI: 10.1007/s10994-021-05946-3. URL: http://dx.doi.org/10.1007/s10994-021-05946-3.

Izmailov, Pavel, Sharad Vikram, Matthew D. Hoffman, and Andrew Gordon Wilson (2021). *What Are Bayesian Neural Network Posteriors Really Like?* arXiv: 2104.14421 [cs.LG]. URL: https://arxiv.org/abs/2104.14421.

Jacob, Benoit, Skirmantas Kligys, Bo Chen, Menglong Zhu, Matthew Tang, Andrew Howard, Hartwig Adam, and Dmitry Kalenichenko (2017). *Quantization and Training of Neural Networks for Efficient Integer-Arithmetic-Only Inference*. arXiv: 1712.05877 [cs.LG]. URL: https://arxiv.org/abs/1712.05877.

Jain, Sambhav R., Albert Gural, Michael Wu, and Chris H. Dick (2020). *Trained Quantization Thresholds for Accurate and Efficient Fixed-Point Inference of Deep Neural Networks*. arXiv: 1903.08066 [cs.CV]. URL: https://arxiv.org/abs/1903.08066.

Jordan, Michael I., Zoubin Ghahramani, Tommi S. Jaakkola, and Lawrence K. Saul (1998). "An Introduction to Variational Methods for Graphical Models." In: *Learning in Graphical Models*. Ed. by Michael I. Jordan. Dordrecht: Springer Netherlands, pp. 105–161. ISBN: 978-94-011-5014-9. DOI: 10.1007/978-94-011-5014-9_5. URL: https://doi.org/10.1007/978-94-011-5014-9_5.

Joshaghani, Mohammad, Amirabbas Davari, Faezeh Nejati Hatamian, Andreas Maier, and Christian Riess (2023). "Bayesian Convolutional Neural Networks for Limited Data Hyperspectral Remote Sensing Image Classification." In: *IEEE Geoscience and Remote Sensing Letters* 20, pp. 1–5. DOI: 10.1109/LGRS.2023.3287504.

Jospin, Laurent Valentin, Hamid Laga, Farid Boussaid, Wray Buntine, and Mohammed Bennamoun (May 2022). "Hands-On Bayesian Neural Networks—A Tutorial for Deep Learning Users." In: *IEEE Computational Intelligence Magazine* 17.2, pp. 29–48. ISSN: 1556-6048. DOI: 10.1109/mci.2022.3155327. URL: http://dx.doi.org/10.1109/MCI.2022.3155327.

Kendall, Alex and Yarin Gal (2017). *What Uncertainties Do We Need in Bayesian Deep Learning for Computer Vision?* arXiv: 1703.04977 [cs.CV]. URL: https://arxiv.org/abs/1703.04977.

Kingma, Diederik P and Max Welling (2022). *Auto-Encoding Variational Bayes*. arXiv: 1312.6114 [stat.ML]. URL: https://arxiv.org/abs/1312.6114.

Kingma, Diederik P., Tim Salimans, and Max Welling (2015). *Variational Dropout and the Local Reparameterization Trick*. arXiv: 1506.02557 [stat.ML]. URL: https://arxiv.org/abs/1506.02557.

Klein, Natalie, Adra Carr, Zigfried Hampel-Arias, Amanda Ziemann, Eric Flynn, and Kevin Mitchell (May 2022). "Quantifying uncertainty in machine learning for hyperspectral target detection and identification." In: p. 11. DOI: 10.1117/12.2622926.

Krishnamoorthi, Raghuraman (2018). *Quantizing deep convolutional networks for efficient inference: A whitepaper*. arXiv: 1806.08342 [cs.LG]. URL: https://arxiv.org/abs/1806.08342.

Lakshminarayanan, Balaji, Alexander Pritzel, and Charles Blundell (2017). *Simple and Scalable Predictive Uncertainty Estimation using Deep Ensembles*. arXiv: 1612.01474 [stat.ML]. URL: https://arxiv.org/abs/1612.01474.

Lecun, Y., L. Bottou, Y. Bengio, and P. Haffner (1998). "Gradient-based learning applied to document recognition." In: *Proceedings of the IEEE* 86.11, pp. 2278–2324. DOI: 10.1109/5.726791.

Li, Yucen Lily, Tim G. J. Rudner, and Andrew Gordon Wilson (2024). *A Study of Bayesian Neural Network Surrogates for Bayesian Optimization*. arXiv: 2305.20028 [cs.LG]. URL: https://arxiv.org/abs/2305.20028.

Lin, Ji, Chuang Gan, and Song Han (2019). *Defensive Quantization: When Efficiency Meets Robustness*. arXiv: 1904.08444 [cs.LG]. URL: https://arxiv.org/abs/1904.08444.

Liu, Ziming, Yixuan Wang, Sachin Vaidya, Fabian Ruehle, James Halverson, Marin Soljačić, Thomas Y. Hou, and Max Tegmark (2025). *KAN: Kolmogorov-Arnold Networks*. arXiv: 2404.19756 [cs.LG]. URL: https://arxiv.org/abs/2404.19756.

Louizos, Christos and Max Welling (2017). *Multiplicative Normalizing Flows for Variational Bayesian Neural Networks*. arXiv: 1703.01961 [stat.ML]. URL: https://arxiv.org/abs/1703.01961.

M, Beevi Fathima, N. Santhi, and N. Ramasamy (2024). "Unlocking the Future of Stroke Diagnosis-Bayesian CNN and MRI Fusion." In: *2024 International Conference on E-mobility, Power Control and Smart Systems (ICEMPS)*, pp. 1–5. DOI: 10.1109/ICEMPS60684.2024.10559315.

Micikevicius, Paulius et al. (2022). *FP8 Formats for Deep Learning*. arXiv: 2209.05433 [cs.LG]. URL: https://arxiv.org/abs/2209.05433.

Mitros, John and Brian Mac Namee (2019). *On the Validity of Bayesian Neural Networks for Uncertainty Estimation*. arXiv: 1912.01530 [stat.ML]. URL: https://arxiv.org/abs/1912.01530.

Mohamad, Saad, Abdelhamid Bouchachia, and Moamar Sayed-Mouchaweh (2018). *Asynchronous Stochastic Variational Inference*. arXiv: 1801.04289 [stat.ML]. URL: https://arxiv.org/abs/1801.04289.

Mukhoti, Jishnu, Andreas Kirsch, Joost Amersfoort, Philip Torr, and Yarin Gal (Feb. 2021). *Deterministic Neural Networks with Appropriate Inductive Biases Capture Epistemic and Aleatoric Uncertainty*. DOI: 10.48550/arXiv.2102.11582.

Mullachery, Vikram, Aniruddh Khera, and Amir Husain (2018). *Bayesian Neural Networks*. arXiv: 1801.07710 [cs.LG]. URL: https://arxiv.org/abs/1801.07710.

Nagel, Markus, Rana Ali Amjad, Mart van Baalen, Christos Louizos, and Tijmen Blankevoort (2020). *Up or Down? Adaptive Rounding for Post-Training Quantization*. arXiv: 2004.10568 [cs.LG]. URL: https://arxiv.org/abs/2004.10568.

Ovadia, Yaniv, Emily Fertig, Jie Ren, Zachary Nado, D Sculley, Sebastian Nowozin, Joshua V. Dillon, Balaji Lakshminarayanan, and Jasper Snoek (2019). *Can You Trust Your Model's Uncertainty? Evaluating Predictive Uncertainty Under Dataset Shift*. arXiv: 1906.02530 [stat.ML]. URL: https://arxiv.org/abs/1906.02530.

Pappalardo, Alessandro (2023). *Xilinx/brevitas*. DOI: 10.5281/zenodo.3333552. URL: https://doi.org/10.5281/zenodo.3333552.

Paszke, Adam et al. (2019). *PyTorch: An Imperative Style, High-Performance Deep Learning Library*. arXiv: 1912.01703 [cs.LG]. URL: https://arxiv.org/abs/1912.01703.

Pehle, Christian, Sebastian Billaudelle, Benjamin Cramer, Jakob Kaiser, Korbinian Schreiber, Yannik Stradmann, Johannes Weis, Aron Leibfried, Eric Müller, and Johannes Schemmel (2022). *The BrainScaleS-2 accelerated neuromorphic system with hybrid plasticity*. arXiv: 2201.11063 [cs.NE]. URL: https://arxiv.org/abs/2201.11063.

Pretnar, Matija (2015). "An Introduction to Algebraic Effects and Handlers. Invited tutorial paper." In: *Electronic Notes in Theoretical Computer Science* 319. The 31st Conference on the Mathematical Foundations of Programming Semantics (MFPS XXXI)., pp. 19–35. ISSN: 1571-0661. DOI: https://doi.org/10.1016/j.entcs.2015.12.003. URL: https://www.sciencedirect.com/science/article/pii/S1571066115000705.

Ranganath, Rajesh, Sean Gerrish, and David M. Blei (2013). *Black Box Variational Inference*. arXiv: 1401.0118 [stat.ML]. URL: https://arxiv.org/abs/1401.0118.

Rezende, Danilo Jimenez and Shakir Mohamed (2016). *Variational Inference with Normalizing Flows*. arXiv: 1505.05770 [stat.ML]. URL: https://arxiv.org/abs/1505.05770.

Ries, Daniel, Jason Adams, and Joshua Zollweg (2023). *Target Detection on Hyperspectral Images Using MCMC and VI Trained Bayesian Neural Networks*. arXiv: 2308.06293 [eess.IV]. URL: https://arxiv.org/abs/2308.06293.

Ritchie, Daniel, Paul Horsfall, and Noah D. Goodman (2016). *Deep Amortized Inference for Probabilistic Programs*. arXiv: 1610.05735 [cs.AI]. URL: https://arxiv.org/abs/1610.05735.

Seitzer, Maximilian, Arash Tavakoli, Dimitrije Antic, and Georg Martius (2022). *On the Pitfalls of Heteroscedastic Uncertainty Estimation with Probabilistic Neural Networks*. arXiv: 2203.09168 [cs.LG]. URL: https://arxiv.org/abs/2203.09168.

Shannon, C. E. (1948). "A mathematical theory of communication." In: *The Bell System Technical Journal* 27.3, pp. 379–423. DOI: 10.1002/j.1538-7305.1948.tb01338.x.

Sharma, Himanshu and Elise Jennings (Sept. 2020). "Bayesian neural networks at scale: a performance analysis and pruning study." In: *The Journal of Supercomputing* 77.4, pp. 3811–3839. ISSN: 1573-0484. DOI: 10.1007/s11227-020-03401-z. URL: http://dx.doi.org/10.1007/s11227-020-03401-z.

Shlens, Jonathon (2014). *Notes on Kullback-Leibler Divergence and Likelihood*. arXiv: 1404.2000 [cs.IT]. URL: https://arxiv.org/abs/1404.2000.

Silva, Rafael, Boxuan Zhong, Yuhan Chen, and Edgar Lobaton (Oct. 2021). *Improving Performance and Quantifying Uncertainty of Body-Rocking Detection using Bayesian Neural Networks*. DOI: 10.36227/techrxiv.16779301.v1.

Sønderby, Casper Kaae, Tapani Raiko, Lars Maaløe, Søren Kaae Sønderby, and Ole Winther (2016). *Ladder Variational Autoencoders*. arXiv: 1602.02282 [stat.ML]. URL: https://arxiv.org/abs/1602.02282.

Staber, Brian and Sébastien Da Veiga (2023). *Benchmarking Bayesian neural networks and evaluation metrics for regression tasks*. arXiv: 2206.06779 [cs.LG]. URL: https://arxiv.org/abs/2206.06779.

Tempczyk, Piotr, Ksawery Smoczyński, Philip Smolenski-Jensen, and Marek Cygan (2022). *One Simple Trick to Fix Your Bayesian Neural Network*. arXiv: 2207.13167 [stat.ML]. URL: https://arxiv.org/abs/2207.13167.

Titsias, M.K. and M. Lázaro-Gredilla (Jan. 2014). "Doubly stochastic variational bayes for non-conjugate inference." In: *31st International Conference on Machine Learning, ICML 2014* 5, pp. 4056–4069.

Tran, Dustin, Matthew D. Hoffman, Rif A. Saurous, Eugene Brevdo, Kevin Murphy, and David M. Blei (2017). *Deep Probabilistic Programming*. arXiv: 1701.03757 [stat.ML]. URL: https://arxiv.org/abs/1701.03757.

Umuroglu, Yaman, Nicholas J. Fraser, Giulio Gambardella, Michaela Blott, Philip Leong, Magnus Jahre, and Kees Vissers (Feb. 2017). "FINN: A Framework for Fast, Scalable Binarized Neural Network Inference." In: *Proceedings of the 2017 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays*. FPGA '17. ACM, pp. 65–74. DOI: 10.1145/3020078.3021744. URL: http://dx.doi.org/10.1145/3020078.3021744.

Valdenegro-Toro, Matias and Daniel Saromo (2022). *A Deeper Look into Aleatoric and Epistemic Uncertainty Disentanglement*. arXiv: 2204.09308 [cs.LG]. URL: https://arxiv.org/abs/2204.09308.

Varshney, Kush R. and Homa Alemzadeh (2017). *On the Safety of Machine Learning: Cyber-Physical Systems, Decision Sciences, and Data Products*. arXiv: 1610.01256 [cs.CY]. URL: https://arxiv.org/abs/1610.01256.

Wang, Kuan, Zhijian Liu, Yujun Lin, Ji Lin, and Song Han (2019). *HAQ: Hardware-Aware Automated Quantization with Mixed Precision*. arXiv: 1811.08886 [cs.CV]. URL: https://arxiv.org/abs/1811.08886.

Wiegerinck, Wim (2013). *Variational Approximations between Mean Field Theory and the Junction Tree Algorithm*. arXiv: 1301.3901 [cs.LG]. URL: https://arxiv.org/abs/1301.3901.

Wilson, Andrew Gordon and Pavel Izmailov (2022). *Bayesian Deep Learning and a Probabilistic Perspective of Generalization*. arXiv: 2002.08791 [cs.LG]. URL: https://arxiv.org/abs/2002.08791.

Wimmer, Lisa, Yusuf Sale, Paul Hofman, Bern Bischl, and Eyke Hüllermeier (2023). *Quantifying Aleatoric and Epistemic Uncertainty in Machine Learning: Are Conditional Entropy and Mutual Information Appropriate Measures?* arXiv: 2209.03302 [cs.LG]. URL: https://arxiv.org/abs/2209.03302.

Xiao, Han, Kashif Rasul, and Roland Vollgraf (Aug. 2017). "Fashion-MNIST: a Novel Image Dataset for Benchmarking Machine Learning Algorithms." In: DOI: 10.48550/arXiv.1708.07747.

Yao, Jiayu, Weiwei Pan, Soumya Ghosh, and Finale Doshi-Velez (2019). *Quality of Uncertainty Quantification for Bayesian Neural Network Inference*. arXiv: 1906.09686 [cs.LG]. URL: https://arxiv.org/abs/1906.09686.

Zadeh, Ali Hadi, Isak Edo, Omar Mohamed Awad, and Andreas Moshovos (Oct. 2020). "GOBO: Quantizing Attention-Based NLP Models for Low Latency and Energy Efficient Inference." In: *2020 53rd Annual IEEE/ACM International Symposium on Microarchitecture (MICRO)*. IEEE, pp. 811–824. DOI: 10.1109/micro50266.2020.00071. URL: http://dx.doi.org/10.1109/MICRO50266.2020.00071.

Zhang, Cheng, Judith Butepage, Hedvig Kjellstrom, and Stephan Mandt (2018). *Advances in Variational Inference*. arXiv: 1711.05597 [cs.LG]. URL: https://arxiv.org/abs/1711.05597.

Zhou, Shuchang, Yuxin Wu, Zekun Ni, Xinyu Zhou, He Wen, and Yuheng Zou (2018). *DoReFa-Net: Training Low Bitwidth Convolutional Neural Networks with Low Bitwidth Gradients*. arXiv: 1606.06160 [cs.NE]. URL: https://arxiv.org/abs/1606.06160.

Zhuang, Bohan, Chunhua Shen, Mingkui Tan, Lingqiao Liu, and Ian Reid (2017). *Towards Effective Low-bitwidth Convolutional Neural Networks*. arXiv: 1711.00205 [cs.CV]. URL: https://arxiv.org/abs/1711.00205.

ERKLÄRUNG

Ich versichere, dass ich diese Arbeit selbständig verfasst habe und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe.

Heidelberg, den 14. Mai 2025

Yong Wu