Uncertainty-Preserving QBNNs: Multi-Level Quantization of SVI-Based Bayesian Neural Networks for Image Classification

Hendrik Borras*, Yong Wu*, Bernhard Klein, Holger Fröning
Hardware and Artificial Intelligence (HAWAII) Lab, Heidelberg University, Germany
{hendrik.borras, bernhard.klein, holger.froening}@ziti.uni-heidelberg.de, yongwu_cs@outlook.com

Abstract—**Bayesian Neural Networks (BNNs) provide principled uncertainty quantification but suffer from substantial computational and memory overhead compared to deterministic networks. While quantization techniques have successfully reduced resource requirements in standard deep learning models, their application to probabilistic models remains largely unexplored. We introduce a systematic multi-level quantization framework for Stochastic Variational Inference based BNNs that distinguishes between three quantization strategies: Variational Parameter Quantization (VPQ), Sampled Parameter Quantization (SPQ), and Joint Quantization (JQ). Our logarithmic quantization for variance parameters, and specialized activation functions to preserve the distributional structure are essential for calibrated uncertainty estimation. Through comprehensive experiments on Dirty-MNIST, we demonstrate that BNNs can be quantized down to 4-bit precision while maintaining both classification accuracy and uncertainty disentanglement. At 4 bits, Joint Quantization achieves up to 8× memory reduction compared to floating-point implementations with minimal degradation in epistemic and aleatoric uncertainty estimation. These results enable deployment of BNNs on resource-constrained edge devices and provide design guidelines for future analog "Bayesian Machines" operating at inherently low precision.**

Index Terms—**Robustness, Quantization, Bayesian Neural Networks**

I. Introduction

Deep Neural Networks (DNNs) have become the dominant modeling paradigm across vision, language, and scientific domains, yet they still lack a principled mechanism to quantify uncertainty in their predictions. Standard architectures produce point estimates—often interpreted as probabilities through functions such as softmax—even though these outputs do not reflect true probabilities [1]. Consequently, neural networks may remain overconfident when confronted with inputs that lie outside their domain of competence (out-of-domain data).

Bayesian Neural Networks (BNNs) [2], [3] address this limitation by placing probability distributions over model parameters, allowing them to quantify uncertainty in their predictions and to differentiate between uncertainty inherent in the data (aleatoric uncertainty) and uncertainty that reflects the model's lack of knowledge or capacity (epistemic uncertainty) [4], [5]. This probabilistic formulation provides a mathematically grounded way for models to express limited knowledge, which is crucial under distribution shift, scarce training data, or safety-critical decision-making [5]. Among approximate inference techniques for BNNs, Stochastic Variational Inference (SVI) has emerged as a practical and scalable approach due to its compatibility with modern deep learning frameworks [6], [7].

Despite these advantages, BNNs suffer from substantial computational and memory overhead. The need to maintain and update distributions over parameters—for SVI typically represented via floating-point distributional parameters such as means and variances of Gaussians—results in significantly higher memory and compute requirements than in standard deterministic networks. This mismatch limits deployment on resource-constrained hardware, where power, throughput, and memory bandwidth are critical constraints.

Neural network quantization provides a powerful way to reduce model size and compute cost by lowering numerical precision [8]. However, while quantization is well understood for deterministic models, applying it directly to probabilistic models is non-trivial: quantization may distort the distributional structure essential for calibrated uncertainty estimates. Recent works have begun to explore low-precision Bayesian neural networks, including post-training quantization of variational BNNs [9]–[11]. However, these approaches treat quantization as a single, post-hoc compression step and do not analyze where and how quantization should be applied within the SVI pipeline to preserve predictive uncertainty. In particular, we are not aware of a multi-level, SVI-integrated quantization framework that studies bit-width vs. uncertainty fidelity (aleatoric and epistemic) across inputs, variational parameters, and stochastic samples. Three fundamental gaps persist:

1) No multi-level view of quantization in Bayesian inference: existing work treats quantization as a single operation, ignoring the different stochastic levels in SVI (inputs, distributions, samples).
2) Lack of uncertainty-preserving precision reduction methods: there are no quantization approaches explicitly designed to preserve the statistical semantics of μ, σ, and sampled weights in BNNs.
3) No systematic evaluation of bit-width vs. uncertainty fidelity: prior studies mainly examine accuracy vs. bit-width, but not the impact on aleatoric and epistemic uncertainty, nor on uncertainty calibration.

[*]These authors contributed equally to this work.

Addressing these gaps is essential to enable hardware-efficient Bayesian inference that maintains the main benefit of BNNs: reliable uncertainty estimation.

The objective of this work is to develop and experimentally validate a hardware-aware, multi-level quantization framework for SVI-based Bayesian Neural Networks that reduces computational cost while preserving accuracy and calibrated predictive uncertainty. Concretely, our contributions are:

1) **Multi-Level Quantization Framework for SVI-BNN Classifiers**: we introduce a systematic decomposition of quantization locations in SVI-based Bayesian image classifiers—covering inputs, sampled weights, and variational distribution parameters—and analyze how each level affects classification accuracy and predictive uncertainty (aleatoric and epistemic) on Dirty-MNIST [12].
2) **Quantization-Aware SVI Optimization**: we design and empirically study quantization-robust SVI configurations, including specialized activation functions, as well as magnitude-based clipping and log-quantization strategies that preserve the variance structure of latent distributions under low precision.
3) **Comprehensive Bit-Width Sensitivity Study**: we perform an extensive analysis of classification accuracy, and aleatoric/epistemic uncertainty decomposition across multiple bit-widths. This yields practical guidelines for selecting precision levels that balance efficiency and uncertainty fidelity.

The ability to deploy Bayesian models on constrained hardware unlocks applications where both low latency and calibrated uncertainty are required, such as embedded medical devices, autonomous robots, and safety-critical control systems. This work bridges the gap between reliable probabilistic modeling and low-precision hardware efficiency, and provides a first step toward principled design rules for quantized Bayesian inference. The insights are directly relevant for future BNN accelerators (called "Bayesian Machines"), for example based on probabilistic photonic computing [13]–[15], FPGA implementations, and mixed-signal/neuromorphic platforms that naturally operate at low bit-widths or with stochastic hardware primitives.

II. Background

Bayesian Neural Networks are widely regarded as a principled approach to uncertainty-aware learning [7], [16]. They aim to approximate the posterior distribution $p(\omega|D) \propto p(D|\omega)\,p(\omega)$ over parameters ω. Since exact posterior inference is intractable for modern neural architectures, a variety of approximate methods have been developed, including Variational Inference (VI) [6], [7], Markov chain Monte Carlo (MCMC), Deep Ensembles [17], and Monte Carlo dropout [18]. In VI, one introduces a parametrized family $q_\Theta(\omega)$ over the model parameters ω (see also Figure 1), where Θ denotes the variational parameters, and optimizes the evidence lower bound (ELBO) to minimize the Kullback–Leibler divergence between $q_\Theta(\omega)$ and the true posterior $p(\omega|\mathcal{D})$.

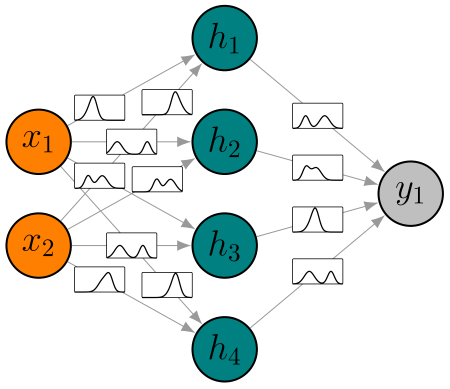


Fig. 1: A Bayesian Neural Network can be viewed as a straightforward extension of a standard neural network, where the deterministic parameters are replaced by probability distributions.

In contrast to sampling-based methods such as Markov chain Monte Carlo, which are computationally prohibitive for large-scale data, VI casts posterior estimation as an optimization problem that can be solved efficiently using stochastic gradient updates [19]. In practice, one often assumes independent Gaussian distributions parametrized by mean μ and variance σ^2 for $q_\Theta(\omega)$, following the mean-field assumption [5].

A. BNN Quality Metrics

While general uncertainty assessments are useful, BNNs often benefit from a more fine-grained view, separating uncertainty into *aleatoric* and *epistemic* components. Aleatoric uncertainty refers to variability arising from noise in the underlying data-generating process. Because this uncertainty is inherent to the observations, it cannot be reduced even when additional data are collected. Epistemic uncertainty, in contrast, reflects the modeling error and typically decreases as more data become available. It captures aspects of model capacity and learned knowledge, and is commonly assessed by evaluating the model's ability to detect out-of-domain (OOD) data.

This decomposition of aleatoric and epistemic uncertainty separates total uncertainty into two distinct scores:

- Softmax Entropy ($\mathbb{E}_{p(w|\mathcal{D})}[H[y|x,w]]$) represents the aleatoric uncertainty component. It quantifies the average predictive ambiguity conditioned on a specific model, capturing the inherent noise or class overlap present in the input.
- Mutual Information ($I[y,w|x,\mathcal{D}]$) quantifies the epistemic uncertainty. It quantifies how much the predictive distribution varies across posterior weight samples, capturing uncertainty arising from insufficient data or imperfectly learned models.

In practice, these uncertainty components can be estimated using Monte Carlo sampling from the posterior distribution, i.e., executing multiple forward passes of the Bayesian Neural

Network to obtain multiple predictions (samples) $p(y = c|x, w_n)$. For brevity, let p denote $p(y = c|x, w_n)$. With N weight samples $\{w_n\}_{n=1}^{N}$ from the approximate posterior $q(w)$, the *Softmax Entropy* (aleatoric) is computed as:

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

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:

$$H(y|x,\mathcal{D}) \approx -\sum_{c=1}^{C}\left(\left(\frac{1}{N}\sum_{n=1}^{N} p\right) \cdot \log\left(\frac{1}{N}\sum_{n=1}^{N} p\right)\right) \tag{2}$$

Finally, the *Mutual Information* (epistemic) is derived by subtracting the aleatoric component from the total uncertainty:

$$I[y, w|x, \mathcal{D}] = H(y|x,\mathcal{D}) - \mathbb{E}_{p(w|\mathcal{D})}[H[y|x,w]] \tag{3}$$

B. BNN Evaluation Dataset

The primary community standard for evaluating a BNN for epistemic and aleatoric uncertainty is Dirty-MNIST [12], where the whole dataset is composed of a set of three sub-datasets, as illustrated in Figure 2. The first set being standard

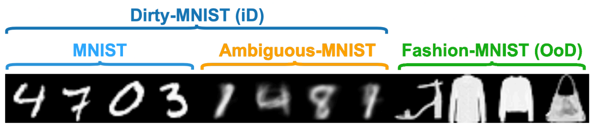


Fig. 2: Examples from the Dirty-MNIST dataset as used for assessment of in-domain prediction quality (MNIST), aleatoric uncertainty (Ambiguous-MNIST) and epistemic uncertainty (Fashion-MNIST), respectively.

MNIST [20] and Fashion-MNIST [21], where Fashion-MNIST is used as the OOD test case. Additionally, Dirty-MNIST introduces Ambiguous-MNIST [12], which is constructed such that each image resembles two digits and thus classes simultaneously. Meaning, that it can be used to evaluate aleatoric uncertainty, as the distinction between the two classes is irreducible.

Training is then done on the training sets of both MNIST and Ambiguous-MNIST, while evaluation is done using the test splits of all three datasets. The resulting prediction quality is judged using multiple metrics:

- In-domain prediction quality is evaluated using accuracy evaluation on MNIST.
- Aleatoric uncertainty estimation quality is evaluated using the Area Under the Receiver Operating Characteristic Curve (AUROC) of MNIST against Ambiguous-MNIST.
- Epistemic uncertainty estimation quality is similarly evaluated using the AUROC of MNIST and Ambiguous-MNIST against Fashion-MNIST.
- To investigate the disentanglement of aleatoric and epistemic uncertainty, we visually investigate how the three datasets separate in a scatter plot of Mutual Information over Softmax Entropy.

Using these metrics and factors one is able to concisely investigate the prediction quality not only of the standard quality metric for classification (accuracy), but also evaluate how well uncertainty estimation and disentanglement work. In a prototypical scatter plot (e.g. Figure 4), all samples from Ambiguous-MNIST should cluster on the x-axis (increased Softmax Entropy score but Mutual Information close to zero), while the samples from Fashion-MNIST should cluster around the y-axis (increased Mutual Information score but Softmax Entropy close to zero). Samples from MNIST should have low scores for both Mutual Information and Softmax Entropy, thus should cluster in the bottom left of the figure.

III. Related Work

Although there is a vast body of work on quantization for standard Deep Neural Networks, BNNs have received surprisingly little dedicated attention. While one might expect that prediction accuracy under quantization behaves similarly for BNNs and deterministic DNNs, the crucial open question is how quantization impacts *uncertainty estimates*—the very core reason for using BNNs in the first place.

Quantization for standard Deep Neural Networks has been extensively studied in the context of resource-efficient inference on embedded and general-purpose hardware. Surveys and system studies such as Roth et al. [8] and Gholami et al. [22] provide overviews of compression techniques—especially Post-Training Quantization (PTQ), Quantization-Aware Training (QAT), and mixed-precision schemes—and document that 4–8 bit inference is often achievable with minimal accuracy loss on standard benchmarks. Foundational work by Jacob et al. [23] introduced the now-standard *integer-arithmetic-only 8-bit quantization* with scale and zero-point, forming the basis of many mobile and edge deployments, while Banner et al. [24] demonstrated that, with careful activation clipping and bias correction, even *4-bit PTQ* can be practical for convolutional networks without retraining. Beyond uniform bit-widths, methods such as HAQ [25] or Galen [26] treat quantization as a *hardware-aware optimization problem*, automatically selecting layer-wise bit-widths under latency, energy, or model-size constraints for specific accelerators. System-focused frameworks like DeepChip [27] integrate low-precision arithmetic with sparsity and optimized kernels to achieve multi-$\times$ speedups and substantial memory savings on constrained hardware. Together, these works underscore the maturity and effectiveness of low-bit quantization for deterministic DNNs.

Ferianc et al. [9] conduct a broad empirical investigation into how low-precision quantization affects both predictive accuracy and uncertainty quality in BNNs. They evaluate several Bayesian inference schemes—including Bayes-by-Backprop, MC Dropout, and SGHMC—under uniform quantization of weights and activations down to sub-8-bit precision across a range of datasets and architectures. Their main finding is that BNNs are surprisingly robust to standard low-bit quantization:

predictive performance, calibration, and uncertainty measures remain largely stable even at aggressive bit-width reductions. However, their study focuses on generic uniform PTQ and does not analyze how quantization interacts with the internal structure of variational inference or task-specific pipelines, leaving open questions about where quantization most strongly affects uncertainty propagation in SVI-based Bayesian classifiers.

Subedar et al. [10] present one of the first empirical investigations into the quantization of SVI-BNNs, focusing on PTQ of pretrained mean-field variational models on MNIST and CIFAR-10. Their method quantizes the learned posterior parameters by applying INT8 PTQ to the mean and a uniform low-bit quantizer to the standard deviation, while also quantizing sampled noise for Monte Carlo inference. Their study demonstrates that even aggressive variance quantization (down to 1 bit) preserves predictive accuracy and calibration, and does not substantially degrade uncertainty quality, even under dataset shift. However, their approach is limited to a single-level, post-hoc quantization of the trained posterior and does not modify the SVI training process, the stochastic sampling pathway, or the input representation.

Lin et al. [11] extend the line of work on quantized Bayesian deep learning by proposing a PTQ workflow for BNNs built on Bayesian-Torch and targeting INT8 inference on 4th Gen Intel Xeon (Sapphire Rapids). Their framework mirrors standard *PyTorch* static PTQ: observers are inserted, calibration is performed on representative data, and full-precision Bayesian models (e.g., variational ResNet-50 on ImageNet) are converted into quantized BNNs with INT8 weights and activations that exploit Intel AMX instructions. They then characterize low-precision BNN workloads at system level, reporting up to $6.9\times$ inference throughput speedup and $4\times$ memory reduction over FP32 BNNs while preserving top-1 accuracy and uncertainty calibration on ImageNet. Beyond ImageNet, they evaluate a medical histology classifier and OOD detection (Camelyon17-WILDS), showing that robustness to data drift and the quality of predictive uncertainty are essentially unaffected by INT8 quantization.

Taken together, the studies by Ferianc et al., Subedar et al., and Lin et al. demonstrate that Post-Training Quantization of pretrained Bayesian models—including INT8 quantization of weights and activations and even sub-8-bit quantization of σ—can preserve accuracy and aggregate uncertainty metrics on standard vision benchmarks. These methods, however, all treat quantization as a single-step, post-hoc compression procedure applied to a fixed Bayesian model, and primarily consider uniform or INT8 schemes. In contrast, our work performs a multi-level, intra-SVI quantization analysis for Bayesian image classification on Dirty-MNIST. Rather than quantizing μ and σ alone, we explicitly quantize distribution parameters, and stochastic samples, while additionally introduce quantization-aware SVI adjustments to examine how different quantization locations and bit-widths affect predictive uncertainty (aleatoric and epistemic). Furthermore, unlike Lin et al., we do not target one specific INT8 hardware pipeline but instead develop hardware-aware yet hardware-agnostic guidelines for sub-8-bit Bayesian classification.

IV. Multi-level quantization of Bayesian Neural Networks

Compared to the quantization of deterministic Deep Neural Networks, Bayesian Neural Networks require more than just straightforward quantization of weights and activations. For these one must distinguish between values which are sampled from probability distributions and those which are not. Since probability distributions in SVI are always parametrizable distributions, there are two points at which quantization can occur, resulting in three types of possible quantization strategies:

1) **Variational Parameter Quantization (*VPQ*):** Quantizing the parameters of the variational distribution, such as the mean and variance in a Gaussian, reduces the memory footprint of the model. The sampled values themselves, however, remain in floating-point format, meaning that the resulting computations are not inherently more efficient than those of standard variational inference.
2) **Sampled Parameter Quantization (*SPQ*):** Instead of quantizing the variational parameters, this approach quantizes the samples drawn from the variational distribution, enabling integer computations when activations are quantized as well.
3) **Joint Quantization (*JQ*):** This approach combines both *VPQ* and *SPQ* to simultaneously reduce memory consumption and computational cost. It therefore constitutes the primary method investigated in this work.

For the standard implementation of BNNs, in which weights are probabilistic this means the following: activations can be quantized just as in deterministic DNNs, but weights need to be taken into special consideration, given the points above.

For the implementation of the investigated BNN we use *Pyro* [28], a probabilistic programming language (PPL), based on the widely-used deep learning framework *PyTorch* [29]. *Pyro* splits an SVI model into two parts:

- **Model:** it contains the main computational graph, of the probabilistic model, for BNNs these are layers like fully-connected layers or activation functions. It additionally specifies in which places probabilistic distributions are sampled. For each distribution, a prior in the sense of Bayes' theorem needs to be given.
- **Guide:** it represents the variational distributions of SVI, i.e. the approximated Bayesian posterior. This means that the guide contains information about the structure of all probability distributions, whose parameters are learned during training.

During training this means that special attention needs to be paid to the guide in particular, as it contains the primary parameters of interest. While some PPLs, such as *Bayesian-Torch* and *TensorFlow Probability*, do not do this explicit model/guide splitting. All PPLs implement a representation of the Bayesian prior and posterior, meaning that conclusions drawn from our *Pyro* implementation apply similarly.

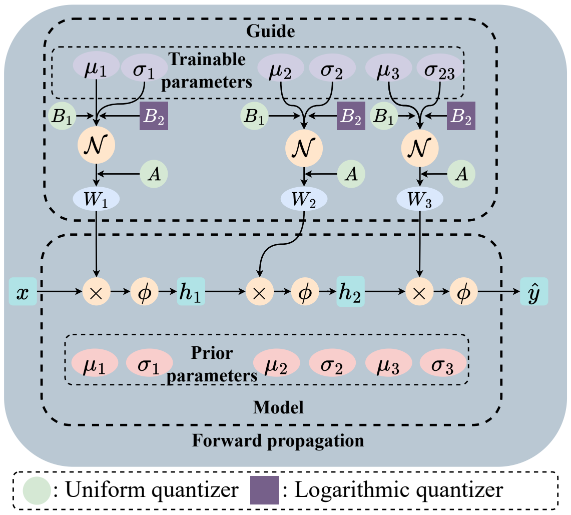

Guide
Trainable parameters
μ_1 σ_1 μ_2 σ_2 μ_3 σ_{23}
B_1 B_2
$\mathcal{N}$
A
W_1 W_2 W_3
x $\times$ ϕ h_1 h_2 $\hat{y}$
Prior parameters
μ_1 σ_1 μ_2 σ_2 μ_3 σ_3
Model
Forward propagation
: Uniform quantizer
: Logarithmic quantizer

Fig. 3: *JQ* as implemented in *Pyro*: uniform quantization is applied in general, while logarithmic quantization is applied specifically to standard deviation parameters. This approach allows for the more precise representation of a wide range of standard deviation values.

Figure 3 shows the quantization process used in this work for both the guide and model. Notably we do not quantize activations, as we are primarily interested in the extension of quantization to BNNs, less the general interaction with DNN components. As all parameters of the model live in the guide, the quantization is accordingly implemented there. Each variational distribution is then equipped with three distinct quantizers:

- Uniform quantizer A (*SPQ*): Quantizes the results of the sampling process
- Quantizers B (*VPQ*): One for each parameter of the given variational distribution, here a Gaussian
  - Uniform quantizer B_1: quantizes the mean parameter of the Gaussian distribution
  - Logarithmic quantizer B_2: quantizes the variance parameter of the Gaussian distribution

All quantizers are implemented as straight-through-gradient estimators, allowing for high-precision quantization-aware training. From an implementation standpoint all uniform quantizers were implemented using the quantization library *Brevitas* [30], while the logarithmic quantizers are a custom implementation using *PyTorch*.

V. Challenges of quantized Bayesian Neural Networks

As already touched upon in section IV: Quantization approaches in BNNs differ significantly from deterministic DNNs, due to the introduction of distributions over weights instead of point estimates. To further complicate things, BNNs are even in their non quantized form very sensitive to hyper-parameters.

A. Activation functions for BNNs

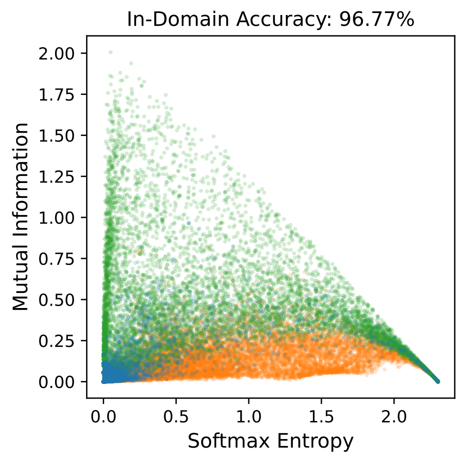
In particular activation function choice is a significant concern for BNNs [31], [32]. This is in stark contrast to deterministic networks, where especially for tasks such as classification only small changes, if any, have been made to the well-established baseline of the ReLU activation function. To highlight this issue, Figure 4 shows the same BNN from section VI with different activation functions, while being otherwise quantized to seven bit, using *SPQ* quantization. At this level of quantization a good separation of aleatoric and epistemic uncertainty is still expected, meaning that the three datasets should cluster into distinct regions of the scatter plot (also see Section II-B): MNIST on the bottom left, Fashion-MNIST on the top left and Ambiguous-MNIST on the bottom right. While this works as expected for the SoftPlus activation function (Fig. 4b), the standard ReLU shows strong deficiencies in Fig. 4a. Notably, ReLU can still separate MNIST from the other datasets, but Fashion-MNIST and Ambiguous-MNIST become strongly entangled with each other. Resulting in substandard uncertainty separation. An additional benefit of SoftPlus is that the in-domain accuracy also slightly increases.

B. Quantization of probability distributions

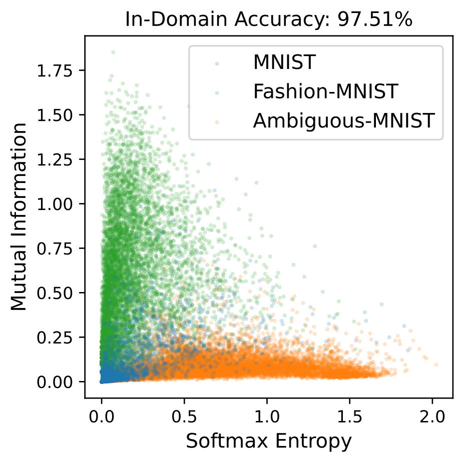
When switching from simple *SPQ* to include the variational parameters with *VPQ* and *JQ*, further challenges become apparent. While uniform quantization works well for parameters mirroring the point estimates of deterministic NNs, such as mean for a Gaussian distribution, the same is not true for more complex parameters, such as the standard deviation of a Gaussian. In probabilistic programming languages, these parameters are represented in log-space to avoid numerical instabilities. As a result, uniform quantization can discard substantial information when the parameter values are small. Figure 5 illustrates this issue visually. We thus opted for a custom logarithmic quantization for these parameters, as highlighted in Fig. 3.

Additionally, we find that clipping the weights to a pre-defined range (we recommend -1 to 1) can significantly help in producing high quality uncertainty estimates. This is in particular noticeable, when a model contains no bias in their linear or convolutional layers. Here, it leads to improved in-domain classification, as well as better uncertainty calibration. The optimization has particular importance for quantized BNNs, as the quantization of the bias is in some cases tricky.

All in all, we identify three key points to watch out for when quantizing BNNs: The activation function must be well-chosen, we recommend SoftPlus for classification tasks. Parameters that are typically handled in log-space—such as the standard deviation of a Gaussian—should be quantized in log-space as well. For BNNs without bias terms, the weights should be clipped to a fixed range.



(a) ReLU



(b) SoftPlus

Fig. 4: Scatter plots comparing the relationship between Softmax Entropy and Mutual Information for Dirty-MNIST using different activation functions under 7-bit *SPQ* quantization. SoftPlus (b) demonstrates superior uncertainty calibration compared to ReLU (a), showing more distinct clustering patterns for each dataset variant and maintaining clearer separation between different levels of aleatoric and epistemic uncertainty.

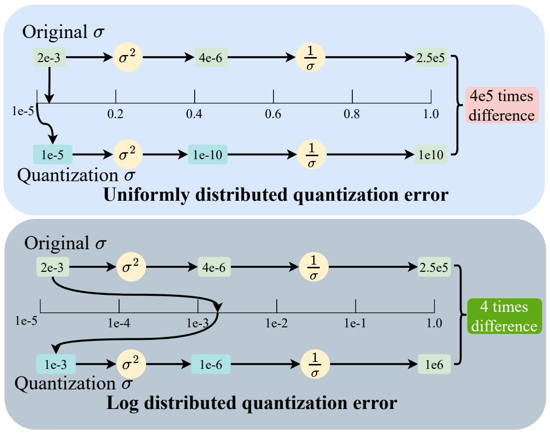


Fig. 5: Comparison of quantization error between linear (uniform) quantization and logarithmic quantization of the standard deviation parameter at different scales when running backward propagation. Due to the involvement of $1/\sigma^2$ term in log-probability calculations, quantization errors are significantly amplified for small values. Linear quantization shows substantially higher relative errors in this scenario.

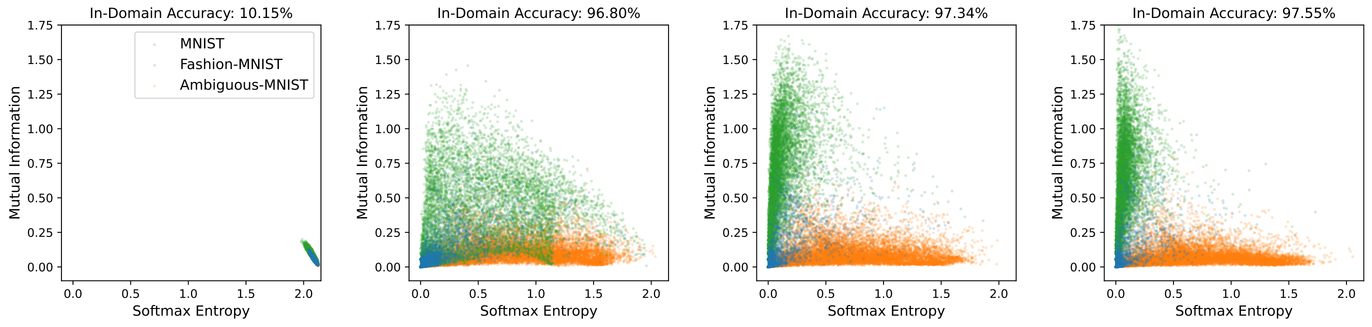
VI. Experimental results

So far we have set up a comprehensive framework for the quantization of BNNs (Section IV), with optimizations for practical deployments (Section V). We will now investigate how a BNN reacts to increasingly aggressive bitwidths. The results shown here are for the Joint Quantization of both variational parameters and sampling results, as this is the most complex and difficult case. We expand on the result of *VPQ* and *SPQ* further below.

The evaluation is done using the Dirty-MNIST benchmark. We train an MLP with two hidden layers of 100 neurons each and SoftPlus activation functions. The model is first pretrained as a deterministic floating point model for 1500 epochs and learning rate of 10^{-3}. The resulting parameters are then transferred into the quantized BNN model of same structure, as a starting point for the Gaussian means (μ). Finally, the quantized BNN is then trained for 1000 epochs. We additionally regularize the loss using KL-annealing using a linearly increasing schedule from 0 to 0.25 over the BNN training, as described in [32]. Final evaluation is done with 100 samples per test image.

Figure 6 highlights four different quantization bit-widths. Starting with the floating point baseline in Fig. 6d, one can observe that the BNN starts out with very strong capability to distinguish ID and OOD data, as well being able to successfully disentangle aleatoric and epistemic uncertainty, while exhibiting good in-domain accuracy. This performance is reasonably well maintained down to 4 bits, as can be seen in Fig. 6c. At 3 bits the BNNs performance starts deteriorating significantly. While the in-domain accuracy is only marginally affected, this is not the case for the uncertainty estimation. In general one can still observe high uncertainties for both Fashion-MNIST and Ambiguous-MNIST, however the two types of uncertainties are now partially entangled with each other and no clear separation is possible anymore. At 2 bit quantization the BNN breaks down completely. Predictions collapse to a very narrow space of MI and SE and in-domain accuracy deteriorates to random guessing. These results are mirrored in both the Accuracy and AUROC results presented in Table I.

For *VPQ* these observations are generally the same, as shown in table I. Both ID-Accuracy and uncertainty disentanglement degrade from four to three bit and collapse at two bit. *SQ* however is more resistant. Here the degradation of



(a) 2-bit (b) 3-bit (c) 4-bit (d) 32-bit (floating point)

Fig. 6: Scatter plots of Softmax Entropy versus Mutual Information across multiple MNIST variants using the Joint Quantization method. Each subplot corresponds to a different bit-width configuration, ordered from left to right such that quantization artifacts are strongest on the left and full 32-bit precision is shown on the right. While both in-domain accuracy and uncertainty disentanglement degrade under more aggressive quantization, uncertainty disentanglement already deteriorates substantially at 3-bit precision.

TABLE I: Quantization performance across different bit-widths and quantization methods. Crossed out cells indicate, that prediction performance had collapsed.

Metric [%]	2-bits	3-bits	4-bits	Full-precision
Joint Quantization (JQ)				
Accuracy	~~10.15~~	96.80	97.34	97.55
AUROC: F-MNIST	~~93.34~~	74.66	86.41	84.70
AUROC: A-MNIST	~~78.10~~	94.94	96.16	97.01
Variational Parameter Quantizaiton (VPQ)				
Accuracy	~~10.61~~	96.51	97.39	97.55
AUROC: F-MNIST	~~62.77~~	93.12	84.55	84.70
AUROC: A-MNIST	~~53.34~~	94.14	96.50	97.01
Sample Quantization (SQ)				
Accuracy	96.35	97.26	97.66	97.55
AUROC: F-MNIST	59.91	77.93	75.36	84.70
AUROC: A-MNIST	94.70	95.91	96.20	97.01

both metrics continues down to two bits, where uncertainties mix similarly to 6b, without fully collapsing. We attribute this increased resistance to the fact, that the actual variational parameters are still represented at high-precision and are thus able to approximate a complex posterior distribution.

While these experiments were done only on Dirty-MNIST, they show a general trend and guideline, for initial experiments on more complex datasets.

VII. Summary and Outlook

We proposed a multi-level quantization framework for SVI–based Bayesian Neural Networks and instantiated it with Variational Parameter Quantization (VPQ), Sampled Parameter Quantization (SPQ), and Joint Quantization (JQ).

This separation of quantization locations shows that BNNs can tolerate aggressive precision reduction while maintaining useful uncertainty estimates: 4-bit joint quantization largely preserves accuracy and the separation of aleatoric and epistemic uncertainty on Dirty-MNIST, whereas at 3 bits this separation deteriorates and at 2 bits both accuracy and uncertainty collapse. These findings complement prior work on Post-Training Quantization of Bayesian models by revealing that the location and structure of quantization within the SVI pipeline are as important as the nominal bit-width. As such this work, as a first, enables multi-level quantization using Quantization-Aware-Training.

Our results yield compact design rules for low-precision BNNs: (i) smooth activations such as SoftPlus improve uncertainty disentanglement compared to ReLU under quantization; (ii) parameters represented in log-space, in particular standard deviations, should use logarithmic rather than uniform quantizers; and (iii) simple magnitude clipping stabilizes training and improves calibration in bias-free architectures. These choices together enable uncertainty-aware BNNs at 4-bit precision.

From a systems perspective, the observed robustness suggests that SVI-based BNNs can be compressed by roughly an order of magnitude in parameter memory with modest loss in quality, making Bayesian inference more attractive for edge and accelerator-based platforms, including analog Bayesian machines [13]–[15] that naturally operate at low precision and exploit stochastic primitives.

Future work includes extending the study to convolutional and transformer-based BNNs, integrating activation quantization and full-integer inference, and validating the approach on FPGA or analog prototypes, ideally coupled with automated, hardware-aware bit-width allocation, forming a promising path toward truly resource-efficient, uncertainty-aware learning.

Acknowledgment

The authors acknowledge support by the state of Baden-Württemberg through bwHPC and the German Research Foundation (DFG) through grant INST 35/1597-1 FUGG.

References

[1] C. Guo, G. Pleiss, Y. Sun, and K. Q. Weinberger, "On calibration of modern neural networks," *CoRR*, vol. abs/1706.04599, 2017. [Online]. Available: https://arxiv.org/abs/1706.04599
[2] D. J. MacKay, "A practical Bayesian framework for backpropagation networks," *Neural computation*, vol. 4, no. 3, pp. 448–472, 1992.

[3] R. M. Neal, *Bayesian Learning for Neural Networks*. Springer Science & Business Media, 1996, vol. 118. [Online]. Available: https://doi.org/10.1007/978-1-4612-0745-0

[4] A. Kendall and Y. Gal, "What uncertainties do we need in bayesian deep learning for computer vision?" *CoRR*, vol. abs/1703.04977, 2017. [Online]. Available: https://arxiv.org/abs/1703.04977

[5] L. V. Jospin, H. Laga, F. Boussaid, W. Buntine, and M. Bennamoun, "Hands-on bayesian neural networks—a tutorial for deep learning users," *IEEE Computational Intelligence Magazine*, vol. 17, no. 2, pp. 29–48, May 2022. [Online]. Available: http://dx.doi.org/10.1109/MCI.2022.3155327

[6] D. M. Blei, A. Kucukelbir, and J. D. McAuliffe, "Variational inference: A review for statisticians," *Journal of the American Statistical Association*, vol. 112, no. 518, pp. 859–877, Apr. 2017. [Online]. Available: http://dx.doi.org/10.1080/01621459.2017.1285773

[7] C. Blundell, J. Cornebise, K. Kavukcuoglu, and D. Wierstra, "Weight uncertainty in neural networks," in *32nd International Conference on Machine Learning*, ser. ICML, 2015, pp. 1613–1622. [Online]. Available: https://arxiv.org/abs/1505.05424

[8] W. Roth, G. Schindler, B. Klein, R. Peharz, S. Tschiatschek, H. Fröning, F. Pernkopf, and Z. Ghahramani, "Resource-efficient neural networks for embedded systems," *Journal of Machine Learning Research*, vol. 25, no. 50, pp. 1–51, 2024. [Online]. Available: http://jmlr.org/papers/v25/18-566.html

[9] M. Ferianc, J. Gamper, C.-Y. Chen, Q. Delaunay, Y. Li, and I. Murray, "On the effects of quantisation on model uncertainty in Bayesian neural networks," in *37th Conference on Uncertainty in Artificial Intelligence*, ser. UAI, C. de Campos and M. H. Maathuis, Eds., vol. 161. PMLR, 2021, pp. 514–524. [Online]. Available: https://proceedings.mlr.press/v161/ferianc21a.html

[10] M. Subedar, R. Krishnan, S. N. Kashyap, and O. Tickoo, "Quantization of bayesian neural networks and its effect on quality of uncertainty," in *ICML 2021 Workshop on Uncertainty and Robustness in Deep Learning*, 2021. [Online]. Available: https://www.gatsby.ucl.ac.uk/~balaji/udl2021/accepted-papers/UDL2021-paper-039.pdf

[11] J.-L. Lin, R. Krishnan, K. R. Ranipa, M. Subedar, V. Sanghavi, M. Arunachalam, O. Tickoo, R. Iyer, and M. T. Kandemir, "Quantization for bayesian deep learning: Low-precision characterization and robustness," in *IEEE International Symposium on Workload Characterization*, ser. IISWC, 2023, pp. 180–192. [Online]. Available: https://doi.org/10.1109/IISWC59245.2023.00020

[12] J. Mukhoti, A. Kirsch, J. van Amersfoort, P. H. S. Torr, and Y. Gal, "Deep deterministic uncertainty: A simple baseline," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, ser. CVPR, 2022. [Online]. Available: https://arxiv.org/abs/2102.11582

[13] F. Brückerhoff-Plückelmann, H. Borras, B. Klein, A. Varri, M. Becker, J. Dijkstra, M. Brückerhoff, C. D. Wright, M. Salinga, H. Bhaskaran, and et al., "Probabilistic photonic computing with chaotic light," *Nature Communications*, vol. 15, no. 1, Dec 2024. [Online]. Available: https://www.nature.com/articles/s41467-024-54931-6

[14] F. Brückerhoff-Plückelmann, A. P. Ovvyan, A. Varri, H. Borras, B. Klein, L. Meyer, C. D. Wright, H. Bhaskaran, G. S. Syed, A. Sebastian, and et al., "Probabilistic photonic computing for AI," *Nature Computational Science*, vol. 5, no. 5, p. 377–387, May 2025. [Online]. Available: https://www.nature.com/articles/s43588-025-00800-1

[15] F. Brückerhoff-Plückelmann, H. Borras, S. U. Hulyal, L. Meyer, X. Ji, J. Hu, J. Sun, B. Klein, F. Ebert, J. Dijkstra, L. McRae, P. Schmidt, T. J. Kippenberg, H. Fröning, and W. Pernice, "Uncertainty reasoning with Photonic Bayesian Machines," *CoRR*, vol. abs/2512.02217, 2025. [Online]. Available: https://arxiv.org/abs/2512.02217

[16] Z. Ghahramani, "Probabilistic machine learning and artificial intelligence," *Nature*, vol. 521, no. 7553, pp. 452–459, May 2015. [Online]. Available: https://doi.org/10.1038/nature14541

[17] B. Lakshminarayanan, A. Pritzel, and C. Blundell, "Simple and scalable predictive uncertainty estimation using deep ensembles," in *31st International Conference on Neural Information Processing Systems*, ser. NeurIPS. Red Hook, NY, USA: Curran Associates Inc., 2017, pp. 6405–6416. [Online]. Available: https://arxiv.org/abs/1612.01474

[18] Y. Gal and Z. Ghahramani, "Dropout as a bayesian approximation: Representing model uncertainty in deep learning," in *33rd International Conference on Machine Learning*, ser. ICML, vol. 48, 2016, pp. 1050–1059. [Online]. Available: https://proceedings.mlr.press/v48/gal16.html

[19] M. D. Hoffman, D. M. Blei, C. Wang, and J. Paisley, "Stochastic variational inference," *Journal of Machine Learning Research*, vol. 14, pp. 1303–1347, 2013. [Online]. Available: https://www.jmlr.org/papers/volume14/hoffman13a/hoffman13a.pdf

[20] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998. [Online]. Available: https://doi.org/10.1109/5.726791

[21] H. Xiao, K. Rasul, and R. Vollgraf, "Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms," *CoRR*, vol. abs/1708.07747, 2017. [Online]. Available: http://arxiv.org/abs/1708.07747

[22] A. Gholami, S. Kim, Z. Dong, Z. Yao, M. W. Mahoney, and K. Keutzer, "A survey of quantization methods for efficient neural network inference," *CoRR*, vol. abs/2103.13630, 2021. [Online]. Available: https://arxiv.org/abs/2103.13630

[23] B. Jacob, S. Kligys, B. Chen, M. Zhu, M. Tang, A. Howard, H. Adam, and D. Kalenichenko, "Quantization and training of neural networks for efficient integer-arithmetic-only inference," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, ser. CVPR, 2018, pp. 2704–2713. [Online]. Available: https://doi.org/10.1109/CVPR.2018.00286

[24] R. Banner, Y. Nahshan, and D. Soudry, "Post training 4-bit quantization of convolutional networks for rapid-deployment," in *33rd International Conference on Neural Information Processing Systems*, ser. NeurIPS. Red Hook, NY, USA: Curran Associates Inc., 2019. [Online]. Available: https://arxiv.org/abs/1810.05723

[25] K. Wang, Z. Liu, Y. Lin, J. Lin, and S. Han, "Haq: Hardware-aware automated quantization with mixed precision," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, ser. CVPR, 2019, pp. 8604–8612. [Online]. Available: https://doi.org/10.1109/CVPR.2019.00881

[26] T. Krieger, B. Klein, and H. Fröning, "Towards Hardware-Specific Automatic Compression of Neural Networks," *AAAI Conference on Artificial Intelligence, International Workshop on Practical Deep Learning in the Wild*, Feb. 2023. [Online]. Available: http://arxiv.org/abs/2212.07818

[27] G. Schindler, M. Zöhrer, F. Pernkopf, and H. Fröning, "Towards efficient forward propagation on resource-constrained systems," in *Machine Learning and Knowledge Discovery in Databases - European Conference, ECML PKDD 2018, Dublin, Ireland, September 10-14, 2018, Proceedings, Part I*, ser. Lecture Notes in Computer Science, M. Berlingerio, F. Bonchi, T. Gärtner, N. Hurley, and G. Ifrim, Eds., vol. 11051. Springer, 2018, pp. 426–442. [Online]. Available: https://doi.org/10.1007/978-3-030-10925-7_26

[28] E. Bingham, J. P. Chen, M. Jankowiak, F. Obermeyer, N. Pradhan, T. Karaletsos, R. Singh, P. Szerlip, P. Horsfall, and N. D. Goodman, "Pyro: Deep universal probabilistic programming," *CoRR*, vol. abs/1810.09538, 2018. [Online]. Available: https://arxiv.org/abs/1810.09538

[29] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Köpf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "Pytorch: An imperative style, high-performance deep learning library," *CoRR*, vol. abs/1912.01703, 2019. [Online]. Available: https://arxiv.org/abs/1912.01703

[30] A. Pappalardo, "Xilinx/brevitas," 2023. [Online]. Available: https://doi.org/10.5281/zenodo.3333552

[31] P. Tempczyk, K. Smoczyński, P. Smolenski-Jensen, and M. Cygan, "One simple trick to fix your bayesian neural network," *CoRR*, vol. abs/2207.13167, 2022. [Online]. Available: https://arxiv.org/abs/2207.13167

[32] B. Klein, "Resource-efficient and robust inference of deep and bayesian neural networks on embedded and analog computing platforms," Ph.D. dissertation, Heidelberg University, 2025. [Online]. Available: https://arxiv.org/abs/2510.24951