Do We Really Need Fully Stochastic Bayesian Neural Networks?

Mrinank Sharma, Sebastian Farquhar, Eric Nalisnick, Tom Rainforth

September 12, 2025

Overview

- Bayesian neural networks (BNNs): posterior and predictive
- Theory: universal conditional distribution approximators (UCDA)
- Bayesian justification?
- Empirical study across inference modalities and datasets
- Practical takeaways

Bayesian Neural Networks

- In standard neural networks, parameters θ are fixed; learning is a point estimate.
- In BNNs, weights and biases are random variables with a prior $p(\theta)$.
- Posterior via Bayes rule: $p(\theta \mid D) \propto p(D \mid \theta)\, p(\theta)$.
- Predictions are made by marginalising over the posterior:

$$p(y \mid x, D) = \int p(y \mid x, \theta)\, p(\theta \mid D)\, d\theta.$$

- Exact inference is typically intractable $\Rightarrow$ approximate inference: variational inference, MCMC, SWAG, etc.

How Are BNN Parameters Optimized?

- **Goal**: approximate the posterior

$$p(\theta \mid D) \propto p(D \mid \theta)p(\theta)$$

  instead of just minimizing loss at a point.

- **Main approaches**:
  - **Variational Inference (VI)**: optimize $q_\phi(\theta)$ via ELBO

$$\mathcal{L}(\phi) = \mathbb{E}_{q_\phi}[\log p(D \mid \theta)] - \mathrm{KL}(q_\phi \| \, p(\theta))$$

    with reparameterization $\theta = \mu + \sigma \odot \epsilon$.
  - **MCMC** (e.g. HMC, SGLD): asymptotically exact, but expensive.
  - **Laplace Approximation**: Gaussian around MAP estimate.
  - **SWAG / Dropout**: approximate posterior via SGD trajectory or dropout masks.

- **Key insight**: Training learns a *distribution over parameters*, not just a single point estimate.

Variational Inference and ELBO

- Approximate the posterior with a tractable family $q_\phi(\theta)$ by minimising $\mathrm{KL}\big(q_\phi \,\|\, p(\theta \mid D)\big)$.
- Evidence lower bound (ELBO):

$$\mathcal{L}(\phi) = \mathbb{E}_{q_\phi}[\log p(D \mid \theta)] - \mathrm{KL}\big(q_\phi(\theta) \,\|\, p(\theta)\big).$$

- The first term encourages fitting the data, the second regularises toward the prior.
- Mean–field VI often underestimates uncertainty but is scalable.

Reparameterization Trick

- **Problem:** In VI we need gradients of

$$\mathbb{E}_{q_\phi(\theta)}[\log p(D \mid \theta)],$$

but sampling $\theta \sim q_\phi(\theta)$ breaks the computational graph. Backpropagation cannot flow through a random draw.

- **Solution:** Rewrite sampling as a deterministic transform of fixed noise:

$$\theta = \mu_\phi + \sigma_\phi \odot \epsilon, \quad \epsilon \sim \mathcal{N}(0, I).$$

  - ϵ: pure randomness, fixed distribution.
  - μ_ϕ, σ_ϕ: trainable, gradients flow.

- **Intuition:** Training often involves "first sample a θ, then compute the loss." Directly differentiating through "sampling" is impossible. Reparameterization rewrites it as "fixed noise + differentiable transform," so we can use automatic differentiation as usual.

BNN: Prediction and Optimization Summary

Prediction pipeline

1. Sample parameters $\theta \sim q_\phi(\theta)$ (approx. posterior).
2. Compute predictive distribution:

$$p(y \mid x, D) \approx \frac{1}{S} \sum_{s=1}^{S} p(y \mid x, \theta^{(s)}).$$

3. Average over S samples $\Rightarrow$ distributional prediction with uncertainty.

Key: BNN does not output a single point prediction, but a distribution.

Optimization process

1. Define approximate posterior $q_\phi(\theta)$.
2. Optimize ELBO:

$$\mathbb{E}_{q_\phi}[\log p(D \mid \theta)] - \mathrm{KL}(q_\phi \,\|\, p(\theta)).$$

3. Use reparameterization trick: $\theta = \mu + \sigma \odot \epsilon$.
4. Train with SGD/Adam until convergence.

Key: Optimization learns a distribution over parameters, not a single point.

Fully vs. Partially Stochastic Networks

- **Fully stochastic**: every parameter is random; the traditional BNN formulation. High memory/compute cost.
- **Partially stochastic**: only a subset of parameters (e.g., certain layers or biases) are random; the rest are point estimates.
- Question: do we need full stochasticity for expressive and reliable posterior predictives?

Lemma 1: Noise Outsourcing

Lemma 1 (Noise Outsourcing Lemma). Let X and Y be random variables in Borel spaces $\mathcal{X}$ and $\mathcal{Y}$. For any given $m \geq 1$, there exists a random variable $\eta \sim \mathcal{N}(0, I_m)$ and a Borel-measurable function $\tilde{f} : \mathbb{R}^m \times \mathcal{X} \to \mathcal{Y}$ such that η is independent of X and

$$(X, Y) = \big(X, \tilde{f}(\eta, X)\big) \quad \text{a.s.}$$

Thus, $\tilde{f}(\eta, x) \sim Y \mid X = x$, for all $x \in \mathcal{X}$.

Interpretation. Conditional distribution estimation can be reduced to learning a deterministic generator $\tilde{f}$ that maps (x, η) to y.

Lemma 2: Universal Approximation (Arbitrary Width)

Lemma 2 (UAT for Arbitrary Width Networks). Let $\mathcal{X} \subset \mathbb{R}^d$ be compact and let $\mathcal{Y} \subseteq \mathbb{R}^n$. Further, let $f_\theta : \mathcal{X} \to \mathcal{Y}$ be a fully connected neural network with *one* hidden layer of arbitrary width and a non-polynomial activation, where $\theta \in \Theta$ are the parameters. Then, for any continuous $g : \mathcal{X} \to \mathcal{Y}$ and all $\varepsilon > 0$,

$$\exists\, \theta \in \Theta : \ \sup_{x \in \mathcal{X}} \|f_\theta(x) - g(x)\| < \varepsilon,$$

provided the network is sufficiently wide.

Interpretation. A wide 1-hidden-layer MLP can approximate any continuous function on a compact domain.

UCDA: Definition and Sources

Definition (UCDA). *Universal Conditional Distribution Approximator*: a model class/architecture that can, in principle, approximate any target conditional distribution $p(y \mid x)$ arbitrarily well.

Sources (two tools).

- **Noise Outsourcing Lemma:** any conditional can be written as $Y = \tilde{f}(\eta, x)$ with η independent noise.
- **Universal Approximation Theorem (UAT):** sufficiently wide NNs can approximate any continuous map.

Consequence. Combining the two: with a *finite* number of noise variables (often $m \geq n$ where n is output dim), partially stochastic networks can act as UCDAs.

Expressivity of Partially Stochastic Networks

Theorem 1 (Universal Conditional Distribution with Finite Stochasticity).
Let $X \in \mathbb{R}^d$, $Y \in \mathbb{R}^n$. There exist architectures with only a *finite* set of Gaussian noise variables $Z = \{Z_1, \ldots, Z_m\}$ ($m \geq n$), independent of X, that can approximate any continuous conditional $Y \,|\, X$ arbitrarily well.

Sufficient architectures (deterministic weights):

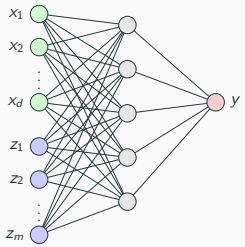
1. **(i) Single-hidden-layer MLP:** input $[Z; X]$, non-polynomial activation, arbitrary width.
2. **(ii) Two-layer MLP (invertible, non-polynomial):** first layer has d deterministic-bias units and m Gaussian-bias units; second layer arbitrary width.
3. **(iii) Two-layer MLP (ReLU):** first layer has $2d$ deterministic-bias units and m Gaussian-bias units; second layer arbitrary width.
4. **(iv) Multi-layer MLP:** each hidden layer has at least $2\max(d+m, n)$ deterministic-bias units; at least one *non-final* layer injects m Gaussian-bias units; later there exists a layer with arbitrarily many units.

Formal statement. If a continuous generator $\tilde{f}(\eta, x)$ exists for $Y \,|\, X$, then for all $\varepsilon > 0$ and $\lambda < \infty$, there exist θ, V, u s.t.

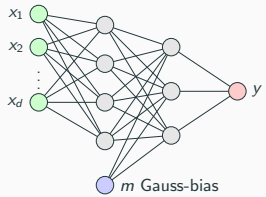
$$\sup_{x,\, \|\eta\| \leq \lambda} \left\| f_\theta(V\eta + u,\ x) - \tilde{f}(\eta, x) \right\| < \varepsilon.$$

Expressivity of Partially Stochastic Networks

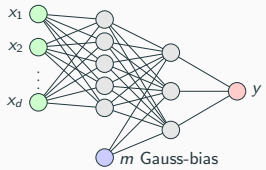
(i) Single hidden layer: input $[Z; X]$



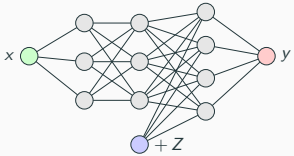
(ii) Two layers (invertible, non-poly)



(iii) Two layers (ReLU)



(iv) Multi-layer MLP with mid-layer noise

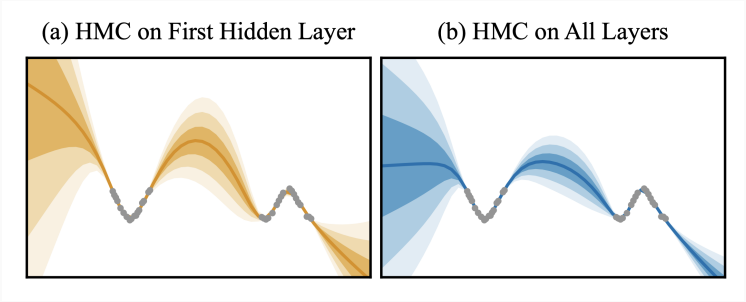


Legend: gray = deterministic units; purple = Gaussian-bias (stochastic) units; green = inputs; red = output.

Discussion

- **Proof sketch:** If a network can represent $[Z; x]$ in some hidden layer and the downstream subnetwork is a universal deterministic approximator (Lemma 2), then the whole model is a UCDA; architectures (i)–(iv) are shown to satisfy these two conditions.
- **How much stochasticity?** Finite m suffices; a practical rule $m \geq n$ is often enough.
- **Architectural scope.** Beyond MLPs—ideas extend when models can expose $[Z; x]$ and keep deterministic capacity downstream.
- **Classification is easier.** Learn class probabilities deterministically; one draw yields a label (only 1D noise needed).
- **Random last-layer $\neq$ UCDA (regression).** Imposes a parametric output family (e.g., Gaussian), limiting $Y \mid X$.
- **Takeaway.** Full stochasticity is *not necessary*: partially stochastic networks already achieve universal conditional expressivity in principle.

Example: Partially vs. Fully Stochastic Inference



This figure contrasts Hamiltonian Monte Carlo inference when only the first hidden layer is treated stochastically versus when all layers are stochastic. Both models capture uncertainty similarly, but the partially stochastic variant trains roughly seven times faster.

Do Bayesian Reasons Support Full Stochasticity?

(a) Priors encode beliefs

Claim: Full stochasticity lets us place priors on all parameters.

Rebuttal: Priors used in practice (e.g., i.i.d. Gaussians) are usually vague/convenience priors and can behave pathologically; they do not confer principled advantages to making *every* weight stochastic.

(b) Averaging improves uncertainty

Claim: Averaging over all parameter hypotheses yields better uncertainty.

Rebuttal: You do not need all parameters to be random. If a hidden layer exposes $[Z; x]$ and the downstream net is a universal deterministic approximator (Lemma 2), the model is already a UCDA. Partially stochastic nets average over hypotheses via a *finite* number of noise dims and can match/beat fully stochastic ones.

(c) Consistent updating with data (Jaynes)

Claim: Bayesian inference provides unique, consistent uncertainty updates.

Rebuttal: This relies on *accurate* posterior inference. In practice—even with high-fidelity methods—posterior *predictive* mixing is poor; hence the theoretical advantage does not materialize.

Experimental Design (Testing the Claims)

Goal. Test whether high-fidelity Bayesian inference actually yields a stable, faithful posterior *predictive* for fully stochastic networks (FSN).

Setup.

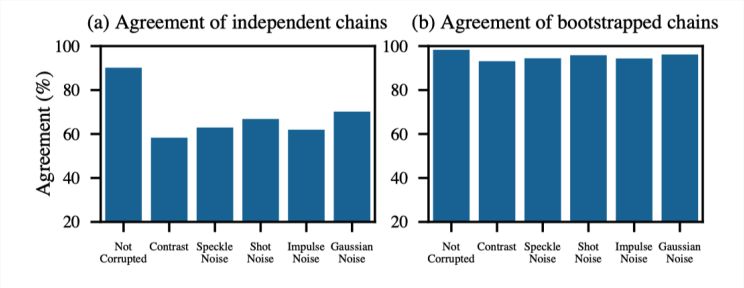
- **Model/Dataset:** ResNet-20(-FRN) on CIFAR-10; also evaluate **CIFAR-10-C** (OOD).
- **Inference:** Use *full-batch HMC* samples from multiple independent chains (publicly released).
- **Predictive construction:** For each chain, average softmax over its posterior samples $\Rightarrow$ one predictive per chain.

Key test (function-space mixing).

- **Chain-to-chain agreement:** fraction of test points on which different HMC chains output the same prediction.
- **Bootstrap (within-chain) baseline:** resample from a *single* chain to estimate finite-sample variability; compare to chain-to-chain.
- **Rationale:** If Bayesian inference is accurate, *between-chain* agreement $\approx$ *within-chain* (bootstrap) agreement, including under OOD shifts.

Link back to (a)(b)(c). Poor mixing undermines (c); if (b) truly required full stochasticity, FSN should excel—evidence tests this.

Results



- **Agreement:** High between-chain agreement in-distribution (ID), but it **drops sharply under OOD**; meanwhile, the bootstrap (within-chain) agreement stays **consistently higher**.
- **Accuracy variability:** Different HMC chains show noticeable gaps in OOD accuracy—evidence of poor exploration of a single posterior predictive.
- **Conclusion on mixing:** Chains do not explore the same posterior *predictive*—even with massive compute and full-batch HMC.

Does Full Stochasticity Improve Predictions In Practice?

Aim. Test whether *fully stochastic networks* (FSN) yield systematically better predictive performance than *partially stochastic networks* (PSN).

Inference method.

- High-fidelity: **HMC** (full-batch or small-scale).
- Local/trajectory posteriors: **Laplace** (diag around MAP), **SWAG** (low-rank Gaussian on SGD).
- Efficient approximation: **MFVI** (mean-field VI).

Tasks/datasets.

- **Regression:** 1D toy (small HMC / larger MFVI); **UCI** (Yacht, Boston, Energy; standard vs. gap splits).
- **Classification:** CIFAR-10 (ID) and CIFAR-10-C (OOD, severity 5), CIFAR-100.

Architectures & PSN placement.

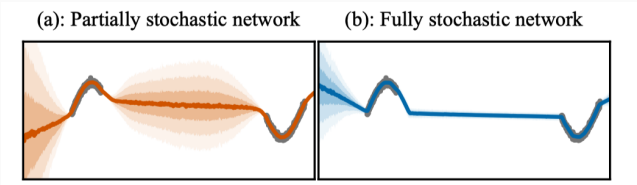
- MLPs for regression; WideResNet/ResNet-FixUp for vision.
- PSN variants: randomize *only* a subset (e.g., input layer, first block, output layer, or a selected weight subset).
- Training styles: *two-stage* (MAP then infer subset) and *joint* (learn Θ_D and $q_\Phi(\Theta_S)$ via ELBO).

Metrics. Accuracy (cls.), Negative Log-Likelihood (NLL), Expected Calibration Error (ECE), plus memory/compute.

1D Regression: HMC (small) and MFVI (larger)

Task. Two-hidden-layer MLP, Gaussian priors.
HMC: small dataset (∼50 pts). Compare **PSN-HMC** (sample only 1st hidden layer) vs **FSN-HMC** (sample all).
MFVI: larger dataset (∼1000 pts). Compare PSN-MFVI (subset) vs FSN-MFVI (all). Prior variance for PSN scaled $\sigma_{PS}^2 = \sigma_{FS}^2 \cdot \frac{|\Theta|}{|\Theta_S|}$.

Results.



- Both PSN and FSN capture uncertainty well in the interpolation region.
- PSN matches FSN under HMC; with MFVI the gap remains small—full stochasticity not required to model predictive spread.

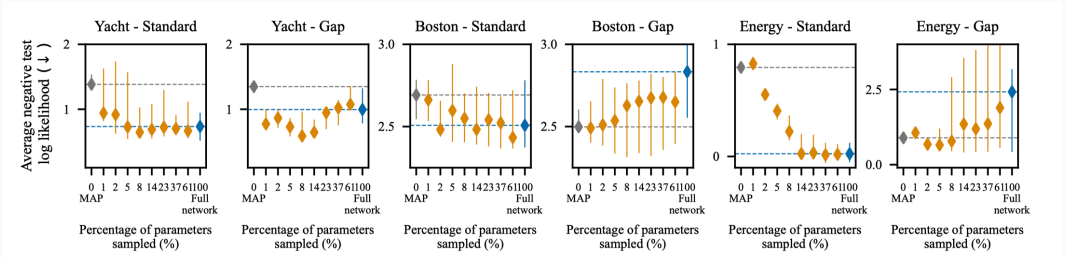
UCI Regression (Yacht, Boston, Energy): HMC

Task. Small MLP; **standard** vs **gap** splits (gap makes OOD-like extrapolation).

PSN-HMC: train MAP, then sample only a subset (largest-magnitude MAP weights), freeze the rest at MAP.

Metric: Median NLL over 15 splits (IQR as error bars).

Results.



- PSN ≈ FSN across datasets/splits; **PSN sometimes better** (e.g., *Yacht-Gap*).
- No systematic benefit of FSN on NLL under high-fidelity HMC.

CIFAR-10 / CIFAR-100: Mean-Field VI

Task. WRN-16-4-FixUp with **MFVI**.
Variants: FSN (all), PSN(*input+output*), deterministic.
Metrics: Acc, NLL, ECE.

Results.

Model	CIFAR10 Acc (%)	CIFAR10 NLL	CIFAR100 Acc (%)	CIFAR100 NLL
Deterministic	95.61 ±0.01	0.187 ±0.001	79.33 ±0.45	0.862 ±0.014
Fully stochastic	94.69 ±0.07	0.214 ±0.002	77.68 ±0.29	0.944 ±0.002
Input layer stochastic	**95.70** ±0.08	0.187 ±0.002	**79.49** ±0.15	0.861 ±0.021
Output layer stochastic	95.60 ±0.05	0.189 ±0.001	78.92 ±0.34	0.933 ±0.010
Output layer and last block stochastic	95.59 ±0.08	**0.168** ±0.0005	79.00 ±0.091	**0.834** ±0.0007

- CIFAR-10: PSN ≈ FSN across metrics.
- CIFAR-100: **PSN slightly better** than FSN.
- **Efficiency:** PSN is more memory/time efficient than FSN.

Conclusions

- Full stochasticity in BNNs is *not* required for expressive and well-calibrated predictive distributions.
- Current approximate inference schemes struggle to realise faithful posterior distributions for fully stochastic networks.
- Partially stochastic networks are theoretically justified (UCDA) and practically superior in many cases.
- Future work may focus on selecting which parameters to randomise and on developing inference schemes targeting the posterior predictive directly.