Heteroscedastic Variational Last Layers

James Harrison[1], **John Willes**[2], **Mikkel Jordahn**[3], **Paul Brunzema**[4], **Jasper Snoek**[1]
[1]Google DeepMind, [2]Vector Institute,
[3]Technical University of Denmark, [4]RWTH Aachen University
Contact: jamesharrison@google.com

Abstract

We present a simple, inexpensive, and effective method for heteroscedastic uncertainty quantification in neural networks. We build on Variational Bayesian Last Layers (VBLL), wherein deterministic training objectives are developed for variational inference of the network last layer. In particular, we (1) Introduce t-VBLL layers, which perform variational inference for the aleatoric noise covariance, and (2) Introduce Het-VBLL, a Bayesian last layer scheme to model heteroscedastic noise. These methods are based on novel, analytically tractable evidence lower bounds. We further discuss parameterization and initialization within these models. We show that these novel design elements enable effective uncertainty modeling at minimal additional cost, and substantially improve performance over similar methods such as VBLLs.

1 Introduction

Effective uncertainty quantification (UQ) in neural networks is crucial for a wide range of downstream applications, such as active learning, exploration, decision making and enhancing model robustness (Neal, 1995; Kendall & Gal, 2017; Wilson & Izmailov, 2020). The scale of modern neural networks, however, necessitates computationally inexpensive UQ methods, particularly when compared to computationally intensive approaches like Markov Chain Monte Carlo (MCMC) (Izmailov et al., 2021) or full-network Bayes-by-Backprop (Blundell et al., 2015). Among these inexpensive methods, last-layer uncertainty methods——which maintain a distribution over the parameters of only the final layer, rather than a point estimate——have emerged as a promising strategy, balancing strong predictive performance with low computational overhead during training and inference (Daxberger et al., 2021; Kristiadi et al., 2021; Harrison et al., 2018, 2024; Liu et al., 2022; Watson et al., 2020, 2021). Variational Bayesian Last Layers (VBLLs) in particular offer a lightweight, easy-to-train, and effective approach to UQ (Harrison et al., 2024; Brunzema et al., 2025; Watson et al., 2021).

VBLLs replace point estimation of neural network last layers with variational inference, and exploit analytically tractable evidence lower bounds (ELBOs) to yield an easy-to-train, inexpensive approach to Bayesian deep learning. However, the quality of uncertainty quantification in VBLLs—in particular, of epistemic (reducible) uncertainty (Der Kiureghian & Ditlevsen, 2009)—depends on a model of the aleatoric (irreducible) noise. For standard VBLLs, a point estimate of the aleatoric noise variance is computed via MAP estimation, due to the simplicity of this procedure in combination with neural network training. Although the noise term may appear relatively unimportant, it is essential for weighing the predictive accuracy and uncertainty terms, thus playing a vital role in accurate uncertainty quantification.

In this work, we investigate and substantially improve the modeling of aleatoric uncertainty in VBLLs. Specifically, we present two methods for more expressive noise modeling. First, we develop t-VBLLs: a variational approach to noise inference that maintains a full variational posterior over the noise variance. In the regression case, this leads to the familiar Student-t distributed predictive distribution.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

For classification, we introduce a novel sampling-free lower bound on the standard ELBO, enabling effective training with low variance. This approach allows us to characterize the epistemic uncertainty over the aleatoric noise, leading to a more complete model of uncertainty in last layer methods. Second, we introduce Het-VBLLs: a method for variational inference of heteroscedastic (input-dependent) noise (Le et al., 2005; Hayashi, 2011). This builds upon a second VBLL model to characterize noise and leverages the novel training ELBOs developed in the first part of the paper.

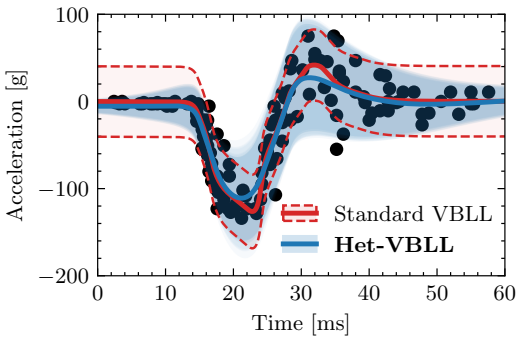


Figure 1: Het-VBLLs vs. VBLLs on motorcycle-impact data from Silverman (1985).

These approaches at most double the size of the last layer, a small change in total parameters, making them highly scalable while substantially improving the calibration and uncertainty quantification of neural networks. In particular, our contributions are:

- We introduce t-VBLL and Het-VBLL, two novel last layer variational methods for the homoscedastic and heteroscedastic settings respectively. These are based on the development of novel lower bounds on the standard ELBOs which enable sampling-free training.
- We evaluate these models on a wide variety of supervised regression and classification, as well as Bayesian optimization settings. We show that our methods consistently result in better predictive likelihood, calibration, better active learning, and better robustness to outliers than baseline methods.
- We release easy-to-use implementations of both t-VBLL and Het-VBLL for the regression and classification setting under `https://github.com/VectorInstitute/vbll`.

2 Related Work

While variance prediction is conceptually simple and capable of expressive representation of aleatoric uncertainty, it is limited in its ability to quantify epistemic uncertainty (Der Kiureghian & Ditlevsen, 2009; Kendall & Gal, 2017). Similarly, many approaches toward epistemic uncertainty quantification rely on assumed likelihoods that inaccurately model aleatoric uncertainty. Kendall & Gal (2017) made the most direct case for the value of characterizing both aleatoric and epistemic uncertainty, and introduced relatively simple models combining Bayesian dropout for epistemic uncertainty with variance prediction for aleatoric.

2.1 Variance Prediction

Variance prediction networks aim to, beyond standard point predictions, predict variance terms capturing (aleatoric) uncertainty (Nix & Weigend, 1994; Bishop & Quazaz, 1996). In regression, this typically corresponds to predicting the mean and variance of a Gaussian likelihood, whilst in classification, this typically corresponds to modeling logit noise (Collier et al., 2020, 2021). Concretely, these networks typically parameterize the variance via

$$\log \Sigma_i = \boldsymbol{m}_i^\top \boldsymbol{\phi}(\boldsymbol{x}) \tag{1}$$

where $\boldsymbol{m}_i$ is a (point estimate) last layer. These networks are typically trained via maximum likelihood. Variance prediction networks broadly fall into the set of heteroscedastic models, although there is often a conflation of epistemic and aleatoric uncertainty in the predicted variance (Kendall & Gal, 2017). Skafte et al. (2019) proposed to improve variance prediction by (instead of predicting the log variance) predicting the parameters of an inverse Gamma distribution, and marginalize over these parameters to yield a Student-t predictive distribution. This parameterization has strong similarities to the approaches in this work, and similarly aims to address the shortcomings of aleatoric uncertainty prediction via epistemic noise modeling.

2.2 Epistemic Uncertainty in Bayesian Deep Learning

Variance prediction networks are limited in their ability to quantify epistemic uncertainty. Like standard networks, predictive behavior far from data is hard to anticipate, and thus predictive uncertainty may be smaller (or larger) than desired. Thus, epistemic uncertainty quantification—even for models that characterize aleatoric uncertainty such as variance prediction models—are necessary. Moreover, the posterior inferred by approximate Bayesian methods relies on the chosen likelihood. Thus, model misspecification in the form of assumed homoscedasticity can substantially harm posterior inference and the predictive performance of the model (Le et al., 2005; Kersting et al., 2007; Lazaro-Gredilla & Titsias, 2011).

A wide variety of epistemic uncertainty quantification methods for neural networks have been developed including variational methods (Blundell et al., 2015), Dropout-based methods (Gal & Ghahramani, 2016), ensembling (Lakshminarayanan et al., 2017), randomized priors (Osband et al., 2018, 2023), and many others. However, these methods are typically expensive and scale poorly to larger models. In this work, we build on variational Bayesian last layers (VBLLs) as a scalable and effective Bayesian approach (Harrison et al., 2024). These models fit into a class of similar last layer uncertainty quantification approaches, including SNGP (Liu et al., 2022) and last layer Laplace approximation methods (Daxberger et al., 2021). For these methods, the last layer inference strategy relies on computing a last layer via approximate Bayesian linear regression after each epoch, and thus incorporating variance prediction (and epistemic uncertainty for this term) is challenging.

3 Preliminaries

We parameterize a neural network by separating its last layer weights, W, from the feature extractor (all preceding layers), denoted by $\boldsymbol{\phi}(\cdot)$. The network output, $\boldsymbol{z}$, for an input $\boldsymbol{x}$ are modeled as

$$\boldsymbol{z} = W\boldsymbol{\phi}(\boldsymbol{x}) + \boldsymbol{\varepsilon} \tag{2}$$

where $\boldsymbol{\varepsilon}$ is zero-mean Gaussian noise. For regression tasks, the output is $\boldsymbol{y} = \boldsymbol{z}$. For classification[1], the class probabilities are given by $p(\boldsymbol{y} \mid \boldsymbol{x}) = \text{softmax}(\boldsymbol{z})$. We denote the dimensionalities of the inputs, outputs, and features as N_x, N_y, and N_ϕ, respectively, such that $W \in \mathbb{R}^{N_y \times N_\phi}$, $\boldsymbol{\phi}(\boldsymbol{x}) \in \mathbb{R}^{N_\phi}$, and $\boldsymbol{y} \in \mathbb{R}^{N_y}$ (with one-hot for classification).

Uncertainty quantification in neural networks has been an active area of research since the early 1990s (MacKay, 1992; Nix & Weigend, 1994), gaining renewed attention with the field's resurgence in the early 2010s (Blundell et al., 2015; Gal & Ghahramani, 2016; Lakshminarayanan et al., 2017; Papamarkou et al., 2024). Two prominent approaches have emerged. The first is variance prediction, where the model directly outputs predictive uncertainty, typically by making the noise term $\boldsymbol{\varepsilon}$ input-dependent (Nix & Weigend, 1994). The second involves quantifying uncertainty over the network parameters themselves, with Bayesian inference being the most common method (MacKay, 1992; Blundell et al., 2015). Modern Bayesian neural networks typically use variational inference, optimizing a lower bound on the marginal likelihood by parameterizing distributions over the network weights.

In this work, we build on Variational Bayesian Last Layers (VBLLs), a minimal and low-variance approach to variational deep learning (Harrison et al., 2024). The VBLL method places a prior $p(W)$ and a variational posterior $q(W)$ only on the weights of the final layer, W, while the weights of the feature extractor $\boldsymbol{\phi}$ are learned as point estimates, as in standard neural network training.

Harrison et al. (2024) derive a tractable objective by lower-bounding the log marginal likelihood:

$$\log p(Y \mid X, \Sigma) \geq \sum_{(\boldsymbol{x},\boldsymbol{y})\in\mathcal{D}} \mathbb{E}_{q(W)}[\log p(\boldsymbol{y} \mid \boldsymbol{x}, W, \Sigma)] - \text{KL}(q(W)|p(W)) \tag{3}$$

This bound leverages the exact computation of the expected log-likelihood (or a lower bound thereof for classification), resulting in an inexpensive, sampling-free training objective. The model is trained by jointly optimizing the feature extractor weights and the parameters of the variational posterior $q(\cdot)$. Throughout the original work, the noise covariance $\boldsymbol{\varepsilon} \sim \mathcal{N}(0, \Sigma)$ is assumed to have a known, fixed covariance Σ, or a point estimate for Σ is learned via maximum likelihood or MAP estimation.

[1] Choosing noise identity or zero noise variance recovers standard loss functions for regression and classification, respectively. The additive aleatoric noise term is not required for classification, as aleatoric noise is captured by logit scale.

This paper extends the VBLL framework to perform variational inference directly on the noise covariance Σ, drawing on techniques from the variance prediction literature. We introduce a factorized variational posterior over the diagonal elements of the noise covariance, $q(\Sigma) = \prod_i^{N_y} q(\Sigma_i)$. We explore both homoscedastic posteriors of the form $q(\Sigma)$ and input-dependent heteroscedastic posteriors of the form $q(\Sigma \mid \boldsymbol{x})$.

4 Approach

In this section we introduce two models: t-VBLLs and Het-VBLLs (see toy experimental results in Fig. 1). We first discuss the noise parameterization that defines each model. We introduce a set of variational bounds used as training objectives, and these general results apply to both homoscedastic and heteroscedastic models. A full discussion of the technical details is provided in the Appendix.

4.1 Parameterizing the Noise Distribution

We consider independent priors (and variational posteriors) over rows of W and fix a diagonal Σ, and so it is sufficient to consider each row independently, of the form

$$\boldsymbol{z}_i = \boldsymbol{w}_i^\top \boldsymbol{\phi}(\boldsymbol{x}) + \boldsymbol{\varepsilon}_i \tag{4}$$

with $\boldsymbol{\varepsilon}_i \sim \mathcal{N}(0, \Sigma_i)$. We fix a prior over the last layer parameters for each row $\boldsymbol{w}_i$ and the covariance element Σ_i

$$p(W, \Sigma \mid \boldsymbol{x}) = \prod_{i=1}^{N_y} p(\boldsymbol{w}_i \mid \Sigma_i, \boldsymbol{x}) p(\Sigma_i \mid \boldsymbol{x}) \tag{5}$$

with[2] $p(\boldsymbol{w}_i \mid \Sigma_i, \boldsymbol{x}) = \mathcal{N}(\underline{\boldsymbol{w}}_i, \Sigma_i(\boldsymbol{x})\underline{S}_i)$. This corresponds to the same choice of prior as Harrison et al. (2024), although with the covariance parameterization being scaled by Σ_i. This scaling is standard in Bayesian linear regression to enable recursive updating of sufficient statistics of the posterior. Here, it yields easy-to-evaluate variational objectives. We similarly structure our variational posterior as

$$q(W, \Sigma \mid \boldsymbol{x}) = \prod_{i=1}^{N_y} q(\boldsymbol{w}_i \mid \Sigma_i, \boldsymbol{x}) q(\Sigma_i \mid \boldsymbol{x}) \tag{6}$$

following standard results in Bayesian regression, with $q(\boldsymbol{w}_i \mid \Sigma_i) = \mathcal{N}(\bar{\boldsymbol{w}}_i, \Sigma_i(\boldsymbol{x})S_i)$.

We will consider two variational families for the noise covariance, corresponding to the homoscedastic case (in which Σ does not depend on $\boldsymbol{x}$) and the heteroscedastic case respectively.

Homoscedastic. We fix an inverse Gamma variational posterior. This is the canonical (conjugate) prior for the noise covariance in Bayesian linear regression. This choice of prior results in a Student t-distributed posterior predictive distribution (Box & Tiao, 2011), which we exploit in our prediction, giving rise to t-VBLLs.

Heteroscedastic. In the heteroscedastic setting, we parameterize the noise covariance with a VBLL as

$$\log \Sigma_i = \boldsymbol{m}_i^\top \boldsymbol{\phi}(\boldsymbol{x}) \tag{7}$$

with $\boldsymbol{m}_i \sim q(\boldsymbol{m}_i) = \mathcal{N}(\bar{\boldsymbol{m}}_i, Z_i)$ (and similarly choose as prior $\boldsymbol{m}_i \sim \mathcal{N}(\underline{\boldsymbol{m}}_i, \underline{Z}_i)$). This yields a log-Normal variational posterior for Σ. We refer to the approach of using a second VBLL for the noise covariance as Het-VBLL. While this does not result in a closed-form predictive distribution, it yields convenient evaluation of the training objective.

For both the inverse Gamma and log-Normal variational posteriors, each of $\mathbb{E}[\Sigma_i], \mathbb{E}[\Sigma_i^{-1}]$, and $\mathbb{E}[\log \Sigma_i]$ (where the expectation is taken with respect to the variational posterior) are analytically tractable. These results are presented in Appendix A. In the next section, we develop lower bounds on the log marginal likelihood for both regression and classification containing (only) these terms, yielding an analytically tractable training objective.

[2]Throughout, we use overbars to denote mean parameters and underbars to denote prior parameters

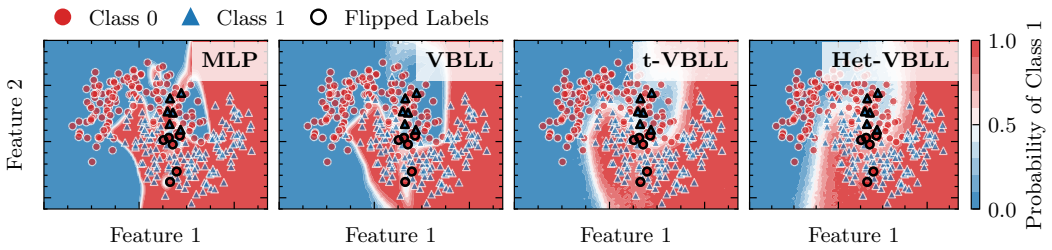


Figure 2: Classification on a noisy two-moon dataset under label noise. A standard MLP struggles on this complex data set. VBLLs show an improved decision boundary but still display slight overfitting. Our models perform best, with Het-VBLL providing the best qualitative results.

4.2 Variational Lower Bounds

The variational lower bound under the variational posterior is

$$\log p(Y \mid X) \geq \sum_{(\boldsymbol{x},\boldsymbol{y})\in\mathcal{D}} \mathbb{E}_{q(W,\Sigma|\boldsymbol{x})}[\log p(\boldsymbol{y} \mid \boldsymbol{x}, W, \Sigma)] - \mathrm{KL}(q(W,\Sigma \mid \boldsymbol{x}) \| p(W,\Sigma \mid \boldsymbol{x})), \tag{8}$$

where the expected log-likelihood term is

$$\mathrm{ELL} := \mathbb{E}_{q(W,\Sigma|\boldsymbol{x})}[\log p(\boldsymbol{y} \mid \boldsymbol{x}, W, \Sigma)] = \sum_{i=1}^{N_y} \mathbb{E}_{q(\Sigma_i|\boldsymbol{x})}[\mathbb{E}_{q(\boldsymbol{w}_i|\Sigma_i,\boldsymbol{x})}[\log p(\boldsymbol{y}_i \mid \boldsymbol{x}, \boldsymbol{w}_i, \Sigma_i)]].$$

With the chosen variational posterior in (6), we now discuss the bounds that we use as training objectives in both the homoscedastic and heteroscedastic case. Full development of results is presented in Appendix B for the homoscedastic case and Appendix C for heteroscedastic.

Regression. In the regression case, the inner expectation over $\boldsymbol{w}_i$ evaluates to

$$\mathrm{ELL} = -\frac{1}{2}\sum_{i=1}^{N_y} \mathbb{E}_{q(\Sigma|\boldsymbol{x})}[\Sigma_i^{-1}(\boldsymbol{y}_i - \bar{\boldsymbol{w}}_i^\top \boldsymbol{\phi})^2 + \log \Sigma_i] + \boldsymbol{\phi}^\top S_i \boldsymbol{\phi} \tag{9}$$

following results from Harrison et al. (2024). Due to linearity with respect to Σ_i^{-1} and $\log \Sigma_i$, this expectation is analytically tractable for both the homoscedastic and the heteroscedastic case.

Classification. For classification, we develop the bound

$$\mathrm{ELL} \;\geq\; \bar{\boldsymbol{w}}_{\boldsymbol{y}}^\top \boldsymbol{\phi} - \mathrm{LSE}_i\big(\bar{\boldsymbol{w}}_i^\top \boldsymbol{\phi} + \alpha_i(\boldsymbol{\phi}^\top S_i \boldsymbol{\phi} + 1)\big) - \frac{1}{2}\sum_i \mathbb{E}\left[\frac{1}{4}\Sigma_i - \alpha_i + \alpha_i^2 \Sigma_i^{-1}\right](\boldsymbol{\phi}^\top S_i \boldsymbol{\phi} + 1). \tag{10}$$

where α_i are variational parameters and $\mathrm{LSE}_i(v_i) = \log \sum_i \exp(v_i)$ denotes the log-sum-exp over the output index. The expectation in (10) is with respect to $q(\Sigma \mid \boldsymbol{x})$. This result was derived to yield linearity in Σ_i, Σ_i^{-1} which in turn yields analytical tractability of this objective. This result builds upon the variational multivariate logistic regression ELBO developed by Knowles & Minka (2011). For the full derivation, see Appendix B.

4.3 Training and Prediction

Both the regression and classification models are trained in the same way as standard neural networks: the weights of the variational posteriors and the neural network features $\boldsymbol{\phi}$ are jointly trained via minibatch gradient descent methods. In our experiments we use AdamW (Kingma et al., 2015; Loshchilov & Hutter, 2017), but other standard optimizers also work well. We find gradient clipping to substantially improve performance, as noted by Harrison et al. (2024).

Following Blundell et al. (2015), we adjust the weight on the KL terms in the training objective to yield unbiased minibatch estimation. In particular, we divide (8) by the number of data points (which we write $|\mathcal{D}|$), and thus estimation of the likelihood term can be done by the mean of a minibatch, and the KL terms have a weight of $1/|\mathcal{D}|$. In the heteroscedastic setting, the KL term in (8) decomposes

into

$$\mathrm{KL}(q(W,\Sigma \mid \boldsymbol{x}) \| p(W,\Sigma \mid \boldsymbol{x})) = \sum_{i=1}^{N_y} (\mathrm{KL}(q(\boldsymbol{m}_i) \| p(\boldsymbol{m}_i)) + \sum_{\boldsymbol{x} \in \mathcal{D}} \mathbb{E}[\mathrm{KL}(q(\boldsymbol{w}_i \mid \Sigma_i) \| p(\boldsymbol{w}_i \mid \Sigma_i))])$$

where the expectation is with respect to $q(\Sigma_i \mid \boldsymbol{x})$. If we correct for minibatch training, we see that the first term has a $1/|\mathcal{D}|$ weight whereas the second term has a weight of 1 (this is discussed further in Appendix C). Intuitively, this means that the posterior will not concentrate as more data is observed. Therefore, in practice we also use a $1/|\mathcal{D}|$ weight for this term which substantially improves performance.

Training Procedure. The models presented here can both be used in full model training, or can be used in a multi-step training procedure. We investigate three approaches to using t- and Het-VBLLs in this work:

1. Train a network with a t/Het-VBLL head from scratch.
2. Use a t/Het-VBLL head on frozen, pretrained features.
3. Replace the standard last layer of a pretrained network and jointly train the last layer and fine-tune the features.

All three of these methods perform well, although they are suitable to different settings. For smaller models (such as those we use in regression and Bayesian optimization) we train from scratch. For very large models, t/Het-VBLL models can require training for as much as 50-100% more total steps[3]. Therefore, we generally advocate for using our last layers with pretrained features for large models, either via freezing features or fine-tuning the model. For the case in which we use t/Het-VBLL on top of pretrained features, we initialize the mean of the VBLL layer to match the last layer of the pretrained model. We use methods for improved gradient estimation in heteroscedastic models, following (Skafte et al., 2019; Seitzer et al., 2022; Stirn et al., 2023). Details on gradient estimation and the complexity of training/prediction are provided in the appendix.

Hyperparameters. Most hyperparameters for t/Het-VBLLs match those from the standard VBLL model. The primary hyperparameters are the last layer priors, which we chose to be simple zero-mean isotropic Gaussians. Thus, the only hyperparameter that has to be tuned is the variance scale. For Het-VBLL models, the use of two VBLL heads leads to two prior variance scale parameters. For t-VBLL models, the prior for the noise introduces hyperparameters in the form of the inverse Gamma prior parameters. We share parameters across output dimensions, reducing the number of hyperparams to two. For classification models, the bound in (10) introduces variational parameters α_i for each output dimension i. These can be learned jointly with the network weights (which automatically tightens the bound) or can be fixed and treated as a hyperparameter. Moreover, these parameters can be shared across output dimensions or treated independently for each dimension.

Prediction. Prediction for regression in the homoscedastic case is analytically tractable via a Student t-distributed posterior predictive. For t-VBLL classification, it is necessary to sample noise realizations via first sampling a realization of Σ from the variational posterior, and then sampling each $z_i \mid \boldsymbol{x}, \Sigma_i \sim \mathcal{N}(\bar{\boldsymbol{w}}_i^\top \boldsymbol{\phi}(\boldsymbol{x}), \Sigma_i \boldsymbol{\phi}(\boldsymbol{x})^\top S_i \boldsymbol{\phi}(\boldsymbol{x}))$. Given this logit sampling procedure, Monte Carlo marginalization over the variational posterior can be performed by generating multiple samples. For the heteroscedastic case, sampling Σ is required for prediction for both regression and classification. For regression, given a realization of Σ, W can be analytically marginalized (matching standard VBLLs). For classification, sampling both Σ and logits is required.

5 Experiments

We investigate the performance of our t-VBLL and Het-VBLL in supervised regression and classification, segmentation, and Bayesian optimization. Due to space constraints, we present only a minimal set of results in the paper body, with experiment details and further results deferred to Appendix E.

[3]We note that this is similar to SNGP (Liu et al., 2022) and is substantially faster than many other Bayesian deep learning methods

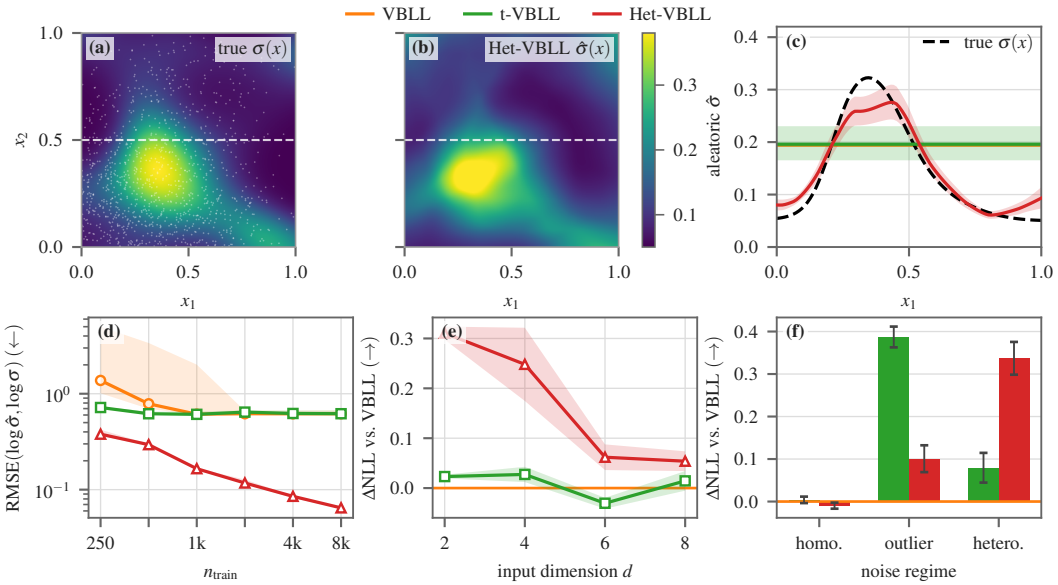


Figure 3: Recovering a known heteroscedastic noise field. **(a, b)** True and recovered noise field at $d = 2$, and **(c)** a transect through them, along which VBLL and t-VBLL are constant by construction. **(d)** Recovery against dataset size, **(e)** paired NLL improvement over VBLL against input dimension, and **(f)** the same quantity under three noise regimes. Setup and further results in Appendix E.1.2.

5.1 Supervised Regression

Toy Data. To demonstrate the Het-VBLLs ability to capture heteroscedatic noise, we compare them against standard VBLLs in Figure 1 on motorcycle-impact data (Silverman, 1985; Kersting et al., 2007). Het-VBLLs are able to capture the heteroscedasticity in the data whereas standard VBLLs with an MAP estimate do not.

Synthetic Data with Known Noise. On real data, the ground-truth aleatoric noise is unknown, so it is difficult to reveal whether a heteroscedastic model has recovered the underlying noise structure or simply fits the data better for unrelated reasons. We therefore construct a benchmark on $[0, 1]^d$ with observations $y = f(\boldsymbol{x}) + \sigma(\boldsymbol{x})\varepsilon$, where the mean function f and the noise field σ are drawn independently as GP sample paths. Both are known analytically at every input, allowing the estimated $\sigma(\boldsymbol{x})$ to be compared directly against the ground truth; the full setup, including a decomposition into mean, aleatoric and epistemic components, is described in Appendix E.1.2. Het-VBLL closely recovers the noise field (Figure 3(a–c)), whereas VBLL and t-VBLL are homoscedastic by construction and therefore cannot represent input-dependent noise, regardless of the amount of available data (Figure 3(d)). This advantage also translates into improved predictive performance, as shown in Figure 3(e), where we fix the dataset size and vary the input dimension: Het-VBLL outperforms both baselines at every dimension we consider. Finally, this controlled setting shows that the two proposed model classes address complementary regimes rather than competing with one another. Figure 3(f) demonstrates that t-VBLL is preferable under outlier contamination, Het-VBLL when the noise varies with the input, and both perform similarly to VBLL when the noise is genuinely homoscedastic.

UCI Datasets. We also benchmark the t-VBLLs and Het-VBLLs on the standard UCI datasets (Dua & Graff, 2017) in Tables 1 and 4 (Appendix). Both models exhibit strong performance, matching or surpassing standard VBLLs (and other baselines) across all tasks. Our experimental procedure and baselines match Harrison et al. (2024) exactly, and experimental details are provided in Appendix E.1.3. Both models show strong performance across all tasks.

QM9. We run experiments on the QM9 property prediction dataset (Wu et al., 2018) (Table 2) using the zero-point energy U_0 as the target. We use the PaiNN message passing backbone from Schütt et al. (2021) with a modified readout function that can accommodate last layer approximations with

Table 1: Results for UCI regression tasks.

	BOSTON		CONCRETE		ENERGY	
	NLL (↓)	RMSE (↓)	NLL (↓)	RMSE (↓)	NLL (↓)	RMSE (↓)
t-VBLL	2.51 ± 0.10	$\mathbf{2.87 \pm 0.28}$	$\mathbf{2.95 \pm 0.08}$	$\mathbf{4.70 \pm 0.36}$	1.19 ± 0.13	0.85 ± 0.11
Het-VBLL	$\mathbf{2.35 \pm 0.14}$	3.14 ± 0.53	3.05 ± 0.13	5.20 ± 0.35	1.17 ± 0.11	1.53 ± 0.23
VBLL	2.55 ± 0.06	$\mathbf{2.92 \pm 0.12}$	3.22 ± 0.07	5.09 ± 0.13	1.37 ± 0.08	0.87 ± 0.04
GBLL	2.90 ± 0.05	4.19 ± 0.17	3.09 ± 0.03	5.01 ± 0.18	$\mathbf{0.69 \pm 0.03}$	$\mathbf{0.46 \pm 0.02}$
LDGBLL	2.60 ± 0.04	3.38 ± 0.18	$\mathbf{2.97 \pm 0.03}$	$\mathbf{4.80 \pm 0.18}$	0.81 ± 0.03	0.50 ± 0.02
MAP	2.60 ± 0.07	3.02 ± 0.17	3.04 ± 0.04	$\mathbf{4.75 \pm 0.12}$	1.44 ± 0.09	0.53 ± 0.01
RBF GP	$\mathbf{2.41 \pm 0.06}$	$\mathbf{2.83 \pm 0.16}$	3.08 ± 0.02	5.62 ± 0.13	$\mathbf{0.66 \pm 0.04}$	$\mathbf{0.47 \pm 0.01}$
Dropout	$\mathbf{2.36 \pm 0.04}$	$\mathbf{2.78 \pm 0.16}$	$\mathbf{2.90 \pm 0.02}$	$\mathbf{4.45 \pm 0.11}$	1.33 ± 0.00	0.53 ± 0.01
Ensemble	2.48 ± 0.09	$\mathbf{2.79 \pm 0.17}$	3.04 ± 0.08	4.55 ± 0.12	$\mathbf{0.58 \pm 0.07}$	$\mathbf{0.41 \pm 0.02}$
SWAG	2.64 ± 0.16	3.08 ± 0.35	3.19 ± 0.05	5.50 ± 0.16	1.23 ± 0.08	0.93 ± 0.09
BBB	$\mathbf{2.39 \pm 0.04}$	$\mathbf{2.74 \pm 0.16}$	2.97 ± 0.03	4.80 ± 0.13	$\mathbf{0.63 \pm 0.05}$	0.43 ± 0.01

Table 2: Results on QM9 property prediction dataset.

			W/ Hold-Out		W/O Hold-Out	
	MAE (↓)	RMSE (↓)	NLL (↓)	ECE (↓)	NLL (↓)	ECE (↓)
DNN	6.10 ± 0.07	14.69 ± 0.60	-2.78 ± 0.04	0.10 ± 0.01	24.35 ± 6.86	0.15 ± 0.00
VBLL	$\mathbf{5.64 \pm 0.16}$	$\mathbf{13.23 \pm 1.17}$	-3.01 ± 0.08	0.09 ± 0.00	9.28 ± 2.15	0.13 ± 0.00
t-VBLL	$\mathbf{5.68 \pm 0.08}$	$\mathbf{13.13 \pm 0.57}$	-3.01 ± 0.04	0.09 ± 0.00	-0.85 ± 0.15	0.08 ± 0.00
Het-VBLL	6.09 ± 0.06	14.71 ± 0.35	-2.79 ± 0.04	0.11 ± 0.00	$\mathbf{-1.24 \pm 0.00}$	0.23 ± 0.00
Var-NN	8.99 ± 0.12	34.04 ± 1.30	$\mathbf{-3.09 \pm 0.02}$	$\mathbf{0.05 \pm 0.00}$	-	-

Table 3: Results on CIFAR-100

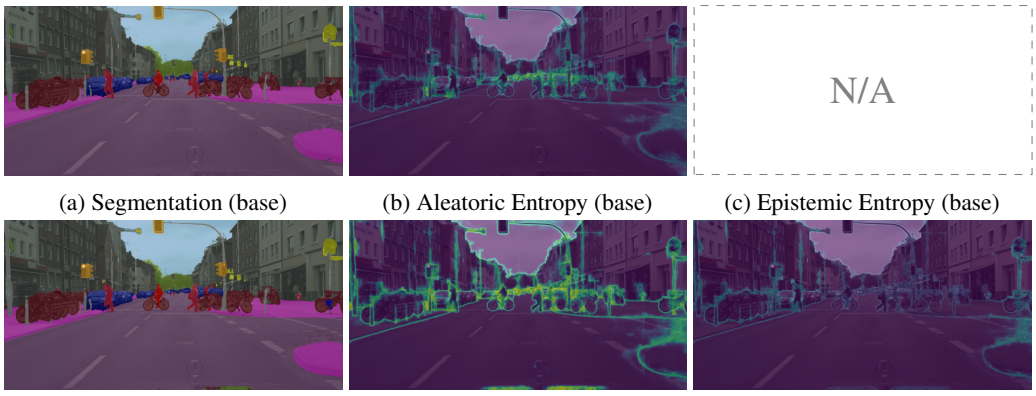
Method	Feature Weights	Accuracy (↑)	ECE (↓)	NLL (↓)	SVHN OOD (↑)	CIFAR-10 OOD (↑)
DNN	Trained	80.4 ± 0.29	0.107 ± 0.004	0.941 ± 0.016	0.799 ± 0.020	0.795 ± 0.001
SNGP	Trained	80.3 ± 0.23	$\mathbf{0.030 \pm 0.004}$	$\mathbf{0.761 \pm 0.007}$	$\mathbf{0.846 \pm 0.019}$	0.798 ± 0.001
VBLL	Trained	80.7 ± 0.03	0.040 ± 0.002	0.913 ± 0.011	$\mathbf{0.849 \pm 0.006}$	0.791 ± 0.003
VBLL	Frozen	80.7 ± 0.02	0.063 ± 0.000	0.831 ± 0.005	0.843 ± 0.001	0.804 ± 0.001
t-VBLL	Frozen	$\mathbf{81.3 \pm 0.10}$	0.039 ± 0.001	0.782 ± 0.004	0.681 ± 0.017	$\mathbf{0.811 \pm 0.003}$
Het-VBLL	Frozen	$\mathbf{81.2 \pm 0.07}$	0.039 ± 0.001	$\mathbf{0.777 \pm 0.005}$	0.685 ± 0.001	$\mathbf{0.811 \pm 0.001}$
LLLA	Frozen	80.4 ± 0.29	0.210 ± 0.018	1.048 ± 0.014	0.834 ± 0.014	$\mathbf{0.811 \pm 0.002}$
VBLL	Fine-Tuned	80.8 ± 0.11	0.047 ± 0.001	0.783 ± 0.002	0.779 ± 0.008	0.794 ± 0.003
t-VBLL	Fine-Tuned	80.2 ± 0.25	0.035 ± 0.001	0.810 ± 0.006	0.815 ± 0.032	0.791 ± 0.003
Het-VBLL	Fine-Tuned	80.2 ± 0.12	0.048 ± 0.002	0.823 ± 0.003	0.838 ± 0.006	0.805 ± 0.003

any predictive distribution without removal of the inductive biases present in the PaiNN network. See Appendix E.1.4 for more details. For the MAP estimated PaiNN network, we post-hoc estimate homoscedastic output noise based on a hold-out dataset, allowing us to obtain ECE and NLL metrics. For all variations of VBLL models we utilize standard deviation scaling using the same hold-out dataset as proposed in Levi et al. (2020) to calibrate the predictive uncertainties of these models (see *W/ Hold-Out* in Table 2). Note that this retains rankings of predictive uncertainties, but simply scales their magnitudes. For some applications (such as BO), hold-out datasets may not be available and we therefore also present results where temperature scaling and noise estimation only can be done based on the training set (see *W/O Hold-Out* in Table 2).

In Table 2 we note how the VBLL and t-VBLL are the best performing both in terms of MAE and RMSE and the Var-NNs from Jordahn et al. (2025) have the best NLL and ECE. However, in the case where no hold-out dataset is used, the Het-VBLLs are the best in terms of NLL and ECE, and have significantly better precision than the Var-NNs. Further results on calibration are provided in the Appendix.

5.2 Supervised Classification

Toy Data. We test our models on a noisy version of the two-moon data set in Figure 2. We further introduce label noise by flipping 20% of the labels for points within a specific range of the first



(a) Segmentation (base) (b) Aleatoric Entropy (base) (c) Epistemic Entropy (base)

(d) Segmentation (Het) (e) Aleatoric Entropy (Het) (f) Epistemic Entropy (Het)

Figure 4: Comparison of the base model (top row) and the proposed Het-VBLL model (bottom row). The models are evaluated on semantic segmentation and uncertainty estimation. Note that the baseline model does not yield an epistemic uncertainty prediction.

feature (circled). A standard MLP overfits significantly, and even standard VBLLs struggle with these outliers. In contrast, our proposed t-VBLL and Het-VBLL models demonstrate robustness, achieving a significantly improved decision boundary.

CIFAR-10/100. We also evaluate the proposed models on CIFAR-10 (Appendix E) and CIFAR-100 (Table 3), comparing t-VBLL and Het-VBLL against standard DNNs, VBLLs (Harrison et al., 2024), and LLLA (Daxberger et al., 2021) baselines. Our models show substantial improvements in accuracy with a substantial reduction in ECE and NLL. The results indicate that incorporating heavy-tailed and heteroscedastic modeling improves uncertainty quantification.

5.3 Semantic Segmentation

Semantic segmentation inherently presents a heteroscedastic problem, with per-pixel uncertainty varying significantly across diverse scenes (Kendall & Gal, 2017). To validate our models in this setting, we evaluate on the CityScapes semantic segmentation benchmark validation set (Cordts et al., 2016). We use the SegFormer model with an MIT-B0 backbone (Xie et al., 2021) as our base architecture. We compare this baseline to two Bayesian extensions: a standard VBLL (VBLL SegFormer) and its heteroscedastic counterpart (Het-VBLL SegFormer).

Het-VBLL SegFormer reaches the highest mIoU (0.769, compared to 0.768 for VBLL and 0.765 for the baseline; see Table 6), i.e., segmentation quality is on par, while additionally providing aleatoric and epistemic uncertainty estimates. Improvements over the baseline are most pronounced for rare classes such as Truck, Bus and Rider, with Het-VBLL improving on 12 of 19 classes in total, including better disambiguation of drivable-space classes such as road and sidewalk. These results show that explicit heteroscedastic noise modeling can be added without loss in segmentation quality. A visualization of the predictive uncertainty is provided in Figure 4, showing the segmentation for a frame and the aleatoric and epistemic entropy for the baseline model and the Het-VBLL model. Aleatoric entropy is elevated around the central horizon, where distant objects are small and ambiguous at the given resolution.

5.4 Bayesian Optimization

We evaluate t-VBLL and Het-VBLL networks as surrogates in Bayesian optimization. As baselines, we use standard VBLLs (Harrison et al., 2024; Brunzema et al., 2025), Gaussian processes (GPs) with box constraints on the lengthscales, GPs with D-scaled priors on the lengthscales (D-GP) (Hvarfner et al., 2024), GPs with a Yeo-Johnson outcome transformation (YJ-GP) which is popular for dealing with heteroscedatic noise (Cowen-Rivers et al., 2022), as well as a variance predictive network (Var-NN). All experimental details and surrogate configurations are listed in Appendix E.4. We compare all models using an upper confidence bound acquisition function (Srinivas et al., 2010) with

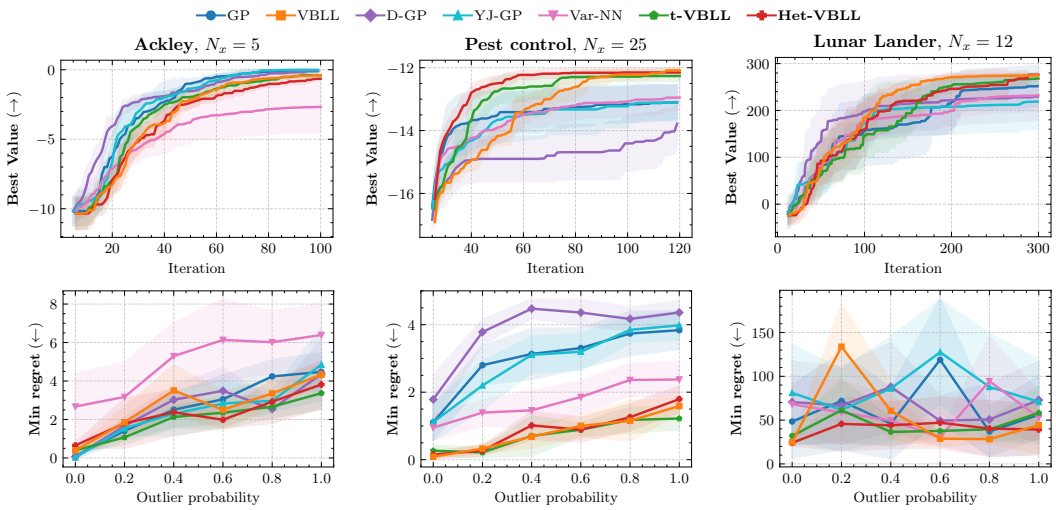


Figure 5: *Top row:* Best value without outliers. *Bottom row:* Minimum regret obtained under heavy-tail distributed outliers on Ackley, Pest Control, and Lunar Lander. Our models demonstrate more robustness to outliers as well as competitive performance regarding best value for no outliers.

the same hyperparameter $\beta = 2$ such that the difference in performance is only due to differences in the model.

As a benchmark experiments, we focus on the classic Ackley objective ($N_x = 5$), as well as pest control with $N_x = 25$ (Oh et al., 2019) and Lunar Lander (Eriksson et al., 2019), where automatic feature learning in Bayesian neural networks has been shown to outperform standard GPs (Li et al., 2024; Brunzema et al., 2025). To test our models, we add heavy-tailed noise from a zero-mean Laplace distribution to the outcome of an experiment, controlled by an outlier probability. In Fig. 5, we show the performance of all surrogates over this outlier probability for 10 random seeds each. VBLL-based models clearly outperform GPs. We can further see, that as the outlier probability increases also the minimum regret obtained by a surrogate increases. Our proposed models perform on par with VBLLs for an outlier probability of zero are more robust to outliers.

6 Discussion and Conclusion

In this paper we have introduced two novel methods for scalable Bayesian deep learning, each applicable to both regression and classification. Development of these methods relied on the design of novel training objectives and model architectures that enabled sampling-free computation of a lower bound on the (standard) ELBO. Both t-VBLL and Het-VBLL show strong performance relatively to both similar last layer methods and considerably more expensive methods. Several questions remain with these models. First, we have investigated basic approaches described in the literature to stabilize the heteroscedastic models, but they still show inconsistent performance (as in the QM9 experiments). Second, training of VBLL models is still slower than vanilla models, and optimization schemes to accelerate training are likely possible to accelerate training.

Finally, several promising avenues remain for future work. While our current formulation relies on independent priors and assumes a diagonal noise covariance for computational efficiency, extending this framework to capture full covariance structures—potentially via inverse-Wishart variational posteriors for t-VBLL and non-diagonal log-normal-type parameterizations for Het-VBLL—could allow the model to learn complex correlations between output dimensions. Moreover, there are several possible next steps for improving training and inference costs. Het-VBLL and classification t-VBLL models currently rely on Monte Carlo sampling for prediction, developing closed-form approximations or variance reduction schemes could reduce inference cost. Both models typically take more training steps to train from scratch and to fine-tune compared to vanilla models, and novel optimizers or initialization schemes may accelerate this training.

Acknowledgments

The authors thank James Allingham and David Stenger for their helpful comments and discussions. Paul Brunzema is partially funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation)–RTG 2236/2 (UnRAVeL). Simulations were performed in part with resources from the NHR Center NHR4CES at RWTH Aachen University (project number p0022034). M. Jordahn acknowledges support from the Novo Nordisk Foundation under grant no NNF22OC0076658 (Bayesian neural networks for molecular discovery).

References

Balandat, M., Karrer, B., Jiang, D. R., Daulton, S., Letham, B., Wilson, A. G., and Bakshy, E. BoTorch: A Framework for Efficient Monte-Carlo Bayesian Optimization. In *Neural Information Processing Systems (NeurIPS)*, 2020.

Betancourt, M. A conceptual introduction to hamiltonian monte carlo. *arXiv:1701.02434*, 2017.

Bishop, C. and Quazaz, C. Regression with input-dependent noise: A bayesian treatment. *Neural Information Processing Systems (NeurIPS)*, 9, 1996.

Blei, D. M. and Lafferty, J. D. A correlated topic model of science. *The annals of applied statistics*, 2007.

Blundell, C., Cornebise, J., Kavukcuoglu, K., and Wierstra, D. Weight uncertainty in neural network. In *International Conference on Machine Learning (ICML)*, 2015.

Box, G. E. and Tiao, G. C. *Bayesian inference in statistical analysis*, volume 40. John Wiley & Sons, 2011.

Brunzema, P., Jordahn, M., Willes, J., Trimpe, S., Snoek, J., and Harrison, J. Bayesian optimization via continual variational last layer training. *International Conference on Learning Representations (ICLR)*, 2025.

Collier, M., Mustafa, B., Kokiopoulou, E., Jenatton, R., and Berent, J. A simple probabilistic method for deep classification under input-dependent label noise. *arXiv:2003.06778*, 2020.

Collier, M., Mustafa, B., Kokiopoulou, E., Jenatton, R., and Berent, J. Correlated input-dependent label noise in large-scale image classification. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

Cordts, M., Omran, M., Ramos, S., Rehfeld, T., Enzweiler, M., Benenson, R., Franke, U., Roth, S., and Schiele, B. The cityscapes dataset for semantic urban scene understanding. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

Cowen-Rivers, A. I., Lyu, W., Tutunov, R., Wang, Z., Grosnit, A., Griffiths, R. R., Maraval, A. M., Jianye, H., Wang, J., Peters, J., et al. Hebo: Pushing the limits of sample-efficient hyper-parameter optimisation. *Journal of Artificial Intelligence Research (JAIR)*, 74:1269–1349, 2022.

Daxberger, E., Kristiadi, A., Immer, A., Eschenhagen, R., Bauer, M., and Hennig, P. Laplace redux-effortless Bayesian deep learning. *Neural Information Processing Systems (NeurIPS)*, 2021.

Der Kiureghian, A. and Ditlevsen, O. Aleatory or epistemic? does it matter? *Structural safety*, 2009.

Dua, D. and Graff, C. UCI machine learning repository, 2017. URL http://archive.ics.uci.edu/ml.

Eriksson, D., Pearce, M., Gardner, J., Turner, R. D., and Poloczek, M. Scalable global optimization via local Bayesian optimization. In *Neural Information Processing Systems (NeurIPS)*, 2019.

Gal, Y. and Ghahramani, Z. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *International Conference on Machine Learning (ICML)*, 2016.

Gardner, J. R., Pleiss, G., Bindel, D., Weinberger, K. Q., and Wilson, A. G. Gpytorch: Blackbox matrix-matrix Gaussian process inference with gpu acceleration. In *Neural Information Processing Systems (NeurIPS)*, 2018.

Girolami, M. and Calderhead, B. Riemann manifold langevin and hamiltonian monte carlo methods. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 2011.

Harrison, J., Sharma, A., and Pavone, M. Meta-learning priors for efficient online Bayesian regression. *Workshop on the Algorithmic Foundations of Robotics (WAFR)*, 2018.

Harrison, J., Willes, J., and Snoek, J. Variational Bayesian last layers. *International Conference on Learning Representations (ICLR)*, 2024.

Hayashi, F. *Econometrics*. Princeton University Press, 2011.

Hvarfner, C., Hellsten, E. O., and Nardi, L. Vanilla Bayesian optimization performs great in high dimensions. In *International Conference on Machine Learning (ICML)*, 2024.

Izmailov, P., Vikram, S., Hoffman, M. D., and Wilson, A. G. G. What are bayesian neural network posteriors really like? In *International Conference on Machine Learning (ICML)*, 2021.

Jankowiak, M., Pleiss, G., and Gardner, J. Parametric gaussian process regressors. In *International Conference on Machine Learning (ICML)*, 2020.

Jordahn, M., Jensen, J. V., Schmidt, M. N., and Andersen, M. R. On local posterior structure in deep ensembles, 2025.

Kendall, A. and Gal, Y. What uncertainties do we need in bayesian deep learning for computer vision? *Neural Information Processing Systems (NeurIPS)*, 30, 2017.

Kersting, K., Plagemann, C., Pfaff, P., and Burgard, W. Most likely heteroscedastic gaussian process regression. In *International Conference on Machine Learning (ICML)*, 2007.

Kingma, D. P., Salimans, T., and Welling, M. Variational dropout and the local reparameterization trick. *Neural Information Processing Systems (NeurIPS)*, 28, 2015.

Knowles, D. and Minka, T. Non-conjugate variational message passing for multinomial and binary regression. In *Neural Information Processing Systems (NeurIPS)*, 2011.

Kristiadi, A., Hein, M., and Hennig, P. Learnable uncertainty under laplace approximations. In *Uncertainty in Artificial Intelligence (UAI)*, 2021.

Lakshminarayanan, B., Pritzel, A., and Blundell, C. Simple and scalable predictive uncertainty estimation using deep ensembles. *Neural Information Processing Systems (NeurIPS)*, 2017.

Lazaro-Gredilla, M. and Titsias, M. K. Variational heteroscedastic gaussian process regression. In *International Conference on Machine Learning (ICML)*, 2011.

Le, Q. V., Smola, A. J., and Canu, S. Heteroscedastic gaussian process regression. In *International Conference on Machine Learning (ICML)*, pp. 489–496, 2005.

Levi, D., Gispan, L., Giladi, N., and Fetaya, E. Evaluating and calibrating uncertainty prediction in regression tasks. *arXiv:1905.11659*, 2020.

Li, Y. L., Rudner, T. G. J., and Wilson, A. G. A study of Bayesian neural network surrogates for Bayesian optimization. In *International Conference on Learning Representations (ICLR)*, 2024.

Liu, J., Padhy, S., Ren, J., Lin, Z., Wen, Y., Jerfel, G., Nado, Z., Snoek, J., Tran, D., and Lakshminarayanan, B. A simple approach to improve single-model deep uncertainty via distance-awareness. *Journal of Machine Learning Research*, 2022.

Loshchilov, I. and Hutter, F. Decoupled weight decay regularization. *arXiv:1711.05101*, 2017.

MacKay, D. J. A practical bayesian framework for backpropagation networks. *Neural Computation*, 1992.

MMSegmentation. MMSegmentation: Openmmlab semantic segmentation toolbox and benchmark. 2020.

Neal, R. M. *Bayesian Learning for Neural Networks*. PhD thesis, University of Toronto, 1995.

Netzer, Y., Wang, T., Coates, A., Bissacco, A., Wu, B., and Ng, A. Y. Reading digits in natural images with unsupervised feature learning. In *NIPS Workshop on Deep Learning and Unsupervised Feature Learning*, 2011.

Nix, D. A. and Weigend, A. S. Estimating the mean and variance of the target probability distribution. In *International conference on neural networks (ICNN)*, 1994.

Oh, C., Tomczak, J., Gavves, E., and Welling, M. Combinatorial Bayesian optimization using the graph cartesian product. *Neural Information Processing Systems (NeurIPS)*, 2019.

Osband, I., Aslanides, J., and Cassirer, A. Randomized prior functions for deep reinforcement learning. *Neural Information Processing Systems (NeurIPS)*, 2018.

Osband, I., Wen, Z., Asghari, S. M., Dwaracherla, V., Ibrahimi, M., Lu, X., and Van Roy, B. Epistemic neural networks. In *Neural Information Processing Systems (NeurIPS)*, 2023.

Papamarkou, T., Skoularidou, M., Palla, K., Aitchison, L., Arbel, J., Dunson, D., Filippone, M., Fortuin, V., Hennig, P., Hernández-Lobato, J. M., et al. Position: Bayesian deep learning is needed in the age of large-scale ai. In *International Conference on Machine Learning (ICML)*, 2024.

Rahimi, A. and Recht, B. Random features for large-scale kernel machines. *Neural Information Processing Systems (NeurIPS)*, 2008.

Schütt, K., Unke, O., and Gastegger, M. Equivariant message passing for the prediction of tensorial properties and molecular spectra. In *International Conference on Machine Learning*, 2021.

Seitzer, M., Tavakoli, A., Antic, D., and Martius, G. On the pitfalls of heteroscedastic uncertainty estimation with probabilistic neural networks. In *International Conference on Learning Representations (ICLR)*, 2022.

Silverman, B. W. Some aspects of the spline smoothing approach to non-parametric regression curve fitting. *Journal of the Royal Statistical Society: Series B (Methodological)*, 47, 1985.

Skafte, N., Jorgensen, M., and Hauberg, S. Reliable training and estimation of variance networks. *Neural Information Processing Systems (NeurIPS)*, 2019.

Srinivas, N., Krause, A., Kakade, S., and Seeger, M. Gaussian process optimization in the bandit setting: No regret and experimental design. In *International Conference on Machine Learning (ICML)*, 2010.

Stirn, A., Wessels, H., Schertzer, M., Pereira, L., Sanjana, N., and Knowles, D. Faithful heteroscedastic regression with neural networks. In *Artificial Intelligence and Statistics (AISTATS)*, 2023.

Watson, J., Lin, J. A., Klink, P., and Peters, J. Neural linear models with functional gaussian process priors. In *Advances in Approximate Bayesian Inference (AABI)*, 2020.

Watson, J., Lin, J. A., Klink, P., Pajarinen, J., and Peters, J. Latent derivative Bayesian last layer networks. In *Artificial Intelligence and Statistics (AISTATS)*, 2021.

Wilson, A. G. and Izmailov, P. Bayesian deep learning and a probabilistic perspective of generalization. In *Neural Information Processing Systems (NeurIPS)*, volume 33, 2020.

Wu, Z., Ramsundar, B., Feinberg, E. N., Gomes, J., Geniesse, C., Pappu, A. S., Leswing, K., and Pande, V. Moleculenet: a benchmark for molecular machine learning. *Chemical science*, 9(2): 513–530, 2018.

Xie, E., Wang, W., Yu, Z., Anandkumar, A., Alvarez, J. M., and Luo, P. Segformer: Simple and efficient design for semantic segmentation with transformers. In *Neural Information Processing Systems (NeurIPS)*, 2021.

A Background **15**
A.1 Inverse Gamma . . . 15
A.2 Log-Normal . . . 15

B Variational Inference for Homoscedastic Noise **15**
B.1 Variational Lower Bound and Approach . . . 16
B.2 Regression . . . 16
B.2.1 Training Objective . . . 16
B.2.2 Prediction . . . 17
B.3 Classification . . . 17
B.3.1 Training Objective . . . 17
B.3.2 Prediction . . . 18
B.4 Complexity and Parameterization . . . 18

C Heteroscedastic Noise Modeling **18**
C.1 Bayesian Last Layer Methods for Heteroscedastic Noise . . . 19
C.2 Variational Lower Bound for VBLL-Variance Networks . . . 19
C.2.1 Regression . . . 20
C.2.2 Classification . . . 20
C.3 Prediction . . . 20
C.4 Complexity and Parameterization . . . 20

D Training and Algorithmic Details **20**
D.1 Training Methodology . . . 20
D.2 Improving Gradient Estimation in Heteroscedastic Models . . . 21

E Experiment Details and Further Results **21**
E.1 Supervised Regression . . . 21
E.1.1 Motorcycle Dataset . . . 21
E.1.2 GP Sample-Path Regression with Known Aleatoric Noise . . . 21
E.1.3 UCI Datasets . . . 24
E.1.4 QM9 Property Prediction . . . 24
E.2 Supervised Classification . . . 25
E.2.1 Half Moon Dataset . . . 25
E.2.2 CIFAR 10 and CIFAR 100 . . . 26
E.3 Semantic Segmentation . . . 26
E.4 Bayesian Optimization . . . 27

A Background

In this section we discuss the two variational posteriors used in this paper for modeling noise covariance.

A.1 Inverse Gamma

Our first approach is an inverse Gamma distribution,

$$q(\Sigma_i) = \mathcal{IG}(\nu_i, \Psi_i) \tag{11}$$

where $\nu_i > 1, \Psi_i > 0$ are shape and scale parameters. This distribution is equivalent to a Gamma distribution for the inverse covariance. This distribution has several desirable properties. First, it is conjugate: in the Bayesian linear regression (BLR) model (with appropriately chosen priors), an inverse Gamma prior yields an inverse Gamma posterior. Additionally, within the BLR model, an inverse-Gamma posterior yields a t-distributed posterior predictive. While this approach is desirable for several reasons in the classical BLR setting, it is less suited to the heteroscedastic setting as we discuss in Section C. The (inverse) Gamma posterior also has a straightforward generalization for non-diagonal covariances in the (inverse) Wishart distribution. We note a few useful identities, which will be useful in developing our main training objectives:

$$\mathbb{E}[\Sigma_i^{-1}] = \nu_i \Psi_i^{-1} \tag{12}$$

$$\mathbb{E}[\log \Sigma_i] = \log(\Psi_i) - \psi(\nu_i) \tag{13}$$

where $\psi(\cdot)$ denotes the digamma function.

For this parameterization, we write our prior as $p(\Sigma_i) = \mathcal{IG}(\underline{\nu}_i, \underline{\Psi}_i)$. The KL divergence between inverse Gammas is a standard result, with

$$\mathrm{KL}(q(\Sigma_i) \| p(\Sigma_i)) = \underline{\nu}_i \log \frac{\Psi_i}{\underline{\Psi}_i} - \log \frac{\Gamma(\nu_i)}{\Gamma(\underline{\nu}_i)} + (\nu_i - \underline{\nu}_i)\psi(\nu_i) - (\Psi_i - \underline{\Psi}_i)\frac{\nu_i}{\Psi_i} \tag{14}$$

where $\Gamma(\cdot)$ is the gamma function.

A.2 Log-Normal

A second approach to the variational posterior is a log-Normal[4]

$$q(\Sigma_i) = \log \mathcal{N}(\mu_i, C_i) \tag{15}$$

with mean μ_i and variance $C_i > 0$. This choice of variational posterior may initially seem like an odd one: we lose the favorable conjugacy properties of the inverse Gamma posterior, and gain little in return. However, as we see in Section C, the log-Normal approach has substantial benefits for heteroscedastic modeling. We have the same identities,

$$\mathbb{E}[\Sigma_i^{-1}] = \exp(-\mu_i + \frac{1}{2}C_i) \tag{16}$$

$$\mathbb{E}[\log \Sigma_i] = \mu_i. \tag{17}$$

For this posterior specification, the prior is also log-Normal and written as $p(\Sigma_i) = \log \mathcal{N}(\underline{\mu}_i, \underline{C}_i)$. The KL divergence between log-Normals is equivalent to the KL divergence between their corresponding Normal distributions.

B Variational Inference for Homoscedastic Noise

In this section, we derive a variational last layer objective with noise inference in the homoscedastic case. First, we will lay out the structure of the variational lower bounds we develop throughout the paper. We then describe two variational posterior design options, and prior choices. We will then define the training objectives and the resulting posterior predictive distribution.

[4]Note that if $z \sim \mathcal{N}(\mu, \sigma^2)$ then $\exp(z) \sim \log \mathcal{N}(\mu, \sigma^2)$.

B.1 Variational Lower Bound and Approach

We begin by writing the lower bound on the marginal likelihood for arbitrarily specified noise covariance Σ, and then discuss outcomes for inverse Gamma priors/variational posteriors. We exclude dependence on the features $\boldsymbol{\phi}$, which we assume fixed in the derivation. Point estimates for the feature weights are learned via stochastic gradient descent on the ELBO. Generally, we will choose noise priors with distributions that match the variational posterior for tractability, although this is not strictly necessary.

We structure our variational posterior as

$$q(W, \Sigma) = \prod_{i=1}^{N_y} q(\boldsymbol{w}_i \mid \Sigma_i) q(\Sigma_i) \tag{18}$$

with $q(\boldsymbol{w}_i \mid \Sigma_i) = \mathcal{N}(\bar{\boldsymbol{w}}_i, S_i \Sigma_i)$, following standard results in Bayesian regression. The variational lower bound is

$$\log p(Y \mid X) \geq \sum_{(\boldsymbol{x}, \boldsymbol{y}) \in \mathcal{D}} \mathbb{E}_{q(W,\Sigma)}[\log p(\boldsymbol{y} \mid \boldsymbol{x}, W, \Sigma)] - \mathrm{KL}(q(W, \Sigma) \| p(W, \Sigma)) \tag{19}$$

By independence assumptions in the prior and variational posterior, we have

$$\mathrm{KL}(q(W, \Sigma) \| p(W, \Sigma)) = \sum_{i=1}^{N_y} (\mathrm{KL}(q(\Sigma_i) \| p(\Sigma_i)) + \mathbb{E}_{q(\Sigma_i)}[\mathrm{KL}(q(\boldsymbol{w}_i \mid \Sigma_i) \| p(\boldsymbol{w}_i \mid \Sigma_i))]). \tag{20}$$

The expectation of the second KL term is straightforward, and is

$$\mathbb{E}_{q(\Sigma_i)}[\mathrm{KL}(q(\boldsymbol{w}_i \mid \Sigma_i) \| p(\boldsymbol{w}_i \mid \Sigma_i))] = \frac{1}{2} \left(\mathrm{logdet} S_i - \mathrm{logdet} S_i - N_\phi + \mathrm{tr}(S_i^{-1} S_i) \right) \tag{21}$$

$$+ \frac{1}{2} \left(\mathbb{E}_{q(\Sigma_i)}[\Sigma_i^{-1}] (\boldsymbol{w}_i - \bar{\boldsymbol{w}}_i) S_i^{-1} (\boldsymbol{w}_i - \bar{\boldsymbol{w}}_i) \right) \tag{22}$$

The expectation of Σ_i^{-1} is given in Eq. (12), and the KL divergence between the prior and variational posterior of Σ_i (the first term of the decomposition above) is given in Eq. (14). Thus, the unresolved aspect in evaluating the variational lower bound is computing the expected likelihood $\mathbb{E}_{q(W,\Sigma)}[\log p(\boldsymbol{y} \mid \boldsymbol{x}, W, \Sigma)]$.

B.2 Regression

We first discuss the likelihood term and prediction in the regression setting.

B.2.1 Training Objective

The predictive likelihood term in the ELBO is

$$\mathbb{E}_{q(W,\Sigma)}[\log p(\boldsymbol{y} \mid \boldsymbol{x}, W, \Sigma)] = \sum_{i=1}^{N_y} \mathbb{E}_{q(\Sigma_i)}[\mathbb{E}_{q(\boldsymbol{w}_i \mid \Sigma_i)}[\log p(\boldsymbol{y}_i \mid \boldsymbol{x}, \boldsymbol{w}_i, \Sigma_i)]] \tag{23}$$

where the inner expectation is computed as in Harrison et al. (2024, Theorem 1). Since $q(\boldsymbol{w}_i \mid \Sigma_i) = \mathcal{N}(\bar{\boldsymbol{w}}_i, S_i \Sigma_i)$, we have $\mathbb{E}[(\boldsymbol{y}_i - \boldsymbol{w}_i^\top \boldsymbol{\phi})^2] = (\boldsymbol{y}_i - \bar{\boldsymbol{w}}_i^\top \boldsymbol{\phi})^2 + \Sigma_i \, \boldsymbol{\phi}^\top S_i \boldsymbol{\phi}$, so that Σ_i cancels Σ_i^{-1} in the second term:

$$\mathbb{E}_{q(\boldsymbol{w}_i \mid \Sigma_i)}[\log p(\boldsymbol{y}_i \mid \boldsymbol{x}, \boldsymbol{w}_i, \Sigma_i)] = \log \mathcal{N}(\boldsymbol{y}_i \mid \bar{\boldsymbol{w}}_i^\top \boldsymbol{\phi}, \Sigma_i) - \frac{1}{2} \boldsymbol{\phi}^\top S_i \boldsymbol{\phi} \tag{24}$$

$$= -\frac{1}{2} (\Sigma_i^{-1} (\boldsymbol{y}_i - \bar{\boldsymbol{w}}_i^\top \boldsymbol{\phi})^2 + \log \Sigma_i + \boldsymbol{\phi}^\top S_i \boldsymbol{\phi}), \tag{25}$$

where the last line drops the additive constant $-\frac{1}{2} \log 2\pi$ per output. To evaluate the outer expectation,

$$\mathbb{E}_{q(W,\Sigma)}[\log p(\boldsymbol{y} \mid \boldsymbol{x}, W, \Sigma)] = -\frac{1}{2} \sum_{i=1}^{N_y} \mathbb{E}_{q(\Sigma)}[\Sigma_i^{-1} (\boldsymbol{y}_i - \bar{\boldsymbol{w}}_i^\top \boldsymbol{\phi})^2 + \log \Sigma_i + \boldsymbol{\phi}^\top S_i \boldsymbol{\phi}] \tag{26}$$

we leverage the identities for $\mathbb{E}[\Sigma_i^{-1}]$ and $\mathbb{E}[\log \Sigma_i]$ described in the previous section.

B.2.2 Prediction

We now discuss computing the predictive distribution

$$p(\boldsymbol{y}_i \mid \boldsymbol{x}) = \mathbb{E}_{q(\boldsymbol{w}_i, \Sigma_i)}[p(\boldsymbol{y}_i \mid \boldsymbol{x}, \boldsymbol{w}_i, \Sigma_i)]. \tag{27}$$

Inverse Gamma. For the inverse Gamma variational posterior, we can exploit standard conjugacy results. In particular, the posterior predictive for each row i is multivariate t-distributed,

$$p(\boldsymbol{y}_i \mid \boldsymbol{x}) = t_{2\nu_i}(\bar{\boldsymbol{w}}_i^\top \boldsymbol{\phi}, \frac{\Psi}{\nu_i}(1 + \boldsymbol{\phi}^\top S \boldsymbol{\phi})) \tag{28}$$

Log-Normal. For the log-Normal variational posterior, we must turn instead to a Monte Carlo approximation. We will sample

$$\hat{\Sigma}_i \sim q(\Sigma_i) \tag{29}$$

for all i, and for which sampling is straight-forward by simply sampling from a normal and exponentiating. Given this realized sample, we can marginalize over the last layer yielding predictive

$$p(\boldsymbol{y}_i \mid \boldsymbol{x}) = \mathcal{N}(\bar{\boldsymbol{w}}_i^\top \boldsymbol{\phi}, \hat{\Sigma}_i(\boldsymbol{\phi}^\top S_i \boldsymbol{\phi} + 1)) \tag{30}$$

B.3 Classification

We consider a classification model of the form

$$p(\boldsymbol{y} \mid \boldsymbol{x}) = \text{softmax}(W \boldsymbol{\phi}(\boldsymbol{x}) + \boldsymbol{\varepsilon}) \tag{31}$$

where the addition of the aleatoric noise $\boldsymbol{\varepsilon}$ is optional. Our model exactly matches the regression model, with the only difference being the softmax. Critically, we again assume Σ is diagonal, with inverse-Gamma distributed diagonal entries.

B.3.1 Training Objective

We write the likelihood

$$\mathbb{E}_{q(W,\Sigma)}[\log p(\boldsymbol{y} \mid \boldsymbol{x}, W, \Sigma)] = \boldsymbol{y}^\top \bar{W} \boldsymbol{\phi} - \mathbb{E}_{q(\Sigma)} \mathbb{E}_{q(W \mid \Sigma)}[\text{LSE}_i(\boldsymbol{w}_i^\top \boldsymbol{\phi} + \boldsymbol{\varepsilon}_i)] \tag{32}$$

where we assume $\boldsymbol{y}$ is a one-hot encoding of the class labels, and lower bound the $\text{LSE}(\cdot)$ term. Note that this is actually a bound for the expectation with respect to the $\boldsymbol{\varepsilon}$ as well; it is thus looser than necessary tractable in this formulation. However because the additive aleatoric noise is not necessary, this term can be removed in practice. It may be tempting to exploit the relatively standard (Blei & Lafferty, 2007) approach for variational multinominal logistic regression to construct a lower bound on the log-sum-exp term for the inner expectation (as is used in Harrison et al. (2024)), yielding

$$-\mathbb{E}_{q(W \mid \Sigma)}[\text{LSE}_i(\boldsymbol{w}_i^\top \boldsymbol{\phi} + \boldsymbol{\varepsilon}_i)] \geq -\mathbb{E}_{q(\Sigma)}[\log \sum_i \exp(\bar{\boldsymbol{w}}_i^\top \boldsymbol{\phi} + \frac{\Sigma_i}{2}(1 + \boldsymbol{\phi}^\top S_i \boldsymbol{\phi})]. \tag{33}$$

It is possible to further exchange the expectation and the negative log/sum terms, yielding

$$-\mathbb{E}_{q(\Sigma)}[\text{LSE}_i(\bar{\boldsymbol{w}}_i^\top \boldsymbol{\phi} + \frac{\Sigma_i}{2}(1 + \boldsymbol{\phi}^\top S_i \boldsymbol{\phi})] \geq -\log \sum_i \mathbb{E}_{q(\Sigma)}[\exp(\bar{\boldsymbol{w}}_i^\top \boldsymbol{\phi} + \frac{\Sigma_i}{2}(1 + \boldsymbol{\phi}^\top S_i \boldsymbol{\phi})]. \tag{34}$$

However, the expectation on the RHS generally does not exist[5]. Thus, we propose two possible approaches, each of which we discuss below.

Semi-Monte Carlo. First, we may turn to sampling. We sample (via reparameterization trick, available in standard automatic differentiation packages) gamma random variables to compute a Monte Carlo approximation to the expectation in (33).

Reduced Knowles-Minka. Instead of directly using the bound on the log-sum-exp used in Harrison et al. (2024), we can instead use the main result from Knowles & Minka (2011), where (applying their result to our chosen parameterization)

$$-\mathbb{E}_{q(W \mid \Sigma)}[\text{LSE}_i(\boldsymbol{w}_i^\top \boldsymbol{\phi} + \boldsymbol{\varepsilon})] \geq -\frac{1}{2} \sum_i a_i^2 (\boldsymbol{\phi}^\top S_i \boldsymbol{\phi} + 1)\Sigma_i - \text{LSE}_i(\bar{\boldsymbol{w}}_i^\top \boldsymbol{\phi} + (\frac{1}{2} - a_i)(\boldsymbol{\phi}^\top S_i \boldsymbol{\phi} + 1)\Sigma_i) \tag{35}$$

[5]This can be seen by the non-existence of the moment generating function of the inverse gamma distribution.

where a_i are variational parameters that are typically optimized to be maximally tight. Note that choosing $a_i = 0$ for all i exactly recovers (33). To yield a tractable bound for the outer expectation with respect to Σ, it is necessary to remove it from inside the log-sum-exp term. So, we choose

$$a_i = \frac{1}{2} - \frac{\alpha_i}{\Sigma_i} \tag{36}$$

and plugging in yields

$$-\mathbb{E}[\text{LSE}_i(\boldsymbol{w}_i^\top \boldsymbol{\phi} + \boldsymbol{\varepsilon})] \geq -\frac{1}{2}\sum_i (\frac{1}{2} - \frac{\alpha_i}{\Sigma_i})^2 (\boldsymbol{\phi}^\top S_i \boldsymbol{\phi} + 1)\Sigma_i - \text{LSE}_i(\bar{\boldsymbol{w}}_i^\top \boldsymbol{\phi} + \alpha_i(\boldsymbol{\phi}^\top S_i \boldsymbol{\phi} + 1)) \tag{37}$$

$$= -\frac{1}{2}\sum_i (\frac{\Sigma_i}{4} - \alpha_i + \frac{\alpha_i^2}{\Sigma_i})(\boldsymbol{\phi}^\top S_i \boldsymbol{\phi} + 1) - \text{LSE}_i(\bar{\boldsymbol{w}}_i^\top \boldsymbol{\phi} + \alpha_i(\boldsymbol{\phi}^\top S_i \boldsymbol{\phi} + 1)) \tag{38}$$

which is analytically tractable for the outer expectation over Σ, yielding (for the inverse Gamma variational posterior)

$$\mathbb{E}_{q(W,\Sigma)}[\log p(\boldsymbol{y} \mid \boldsymbol{x}, W, \Sigma)] \geq \boldsymbol{y}^\top \bar{W}\boldsymbol{\phi} - \text{LSE}_i(\bar{\boldsymbol{w}}_i^\top \boldsymbol{\phi} + \alpha_i(\boldsymbol{\phi}^\top S_i \boldsymbol{\phi} + 1)) \tag{39}$$

$$- \frac{1}{2}\sum_i \left(\frac{\Psi_i}{4(\nu_i - 1)} - \alpha_i + \frac{\alpha_i^2 \nu_i}{\Psi_i}\right)(\boldsymbol{\phi}^\top S_i \boldsymbol{\phi} + 1). \tag{40}$$

We have several options with α_i; setting $\alpha_i = \frac{1}{2}$ results in the first two terms exactly matching the standard VBLL objective (Harrison et al., 2024). Choosing $\alpha_i = 0$ yields the remarkably simple overall bound

$$-\mathbb{E}_{q(W|\Sigma)}[\text{LSE}_i(\boldsymbol{w}_i^\top \boldsymbol{\phi} + \boldsymbol{\varepsilon})] \geq -\frac{1}{8}\sum_i (\boldsymbol{\phi}^\top S_i \boldsymbol{\phi} + 1)\Sigma_i - \text{LSE}_i(\bar{\boldsymbol{w}}_i^\top \boldsymbol{\phi}). \tag{41}$$

Other options can be chosen for learning α_i's. We can treat them as hyperparameters, in which it is convenient to set $\alpha_i = \alpha$ to reduced the number of parameters, and it can be swept over. Alternatively, because the bound holds for any α_i, they can be learned as model parameters together with the other model parameters. Knowles & Minka (2011) propose an iterative update that exploits convexity with respect to α_i's—while better optimization schemes exploiting convexity are possible, we will not investigate them in this paper.

B.3.2 Prediction

For prediction, we again turn to Monte Carlo approximation within the hierarchical model, combined with local reparameterization, and sample $\hat{\Sigma}_i$ from the variational posterior, and compute

$$\hat{\boldsymbol{z}}_i \sim \mathcal{N}(\bar{\boldsymbol{w}}_i^\top \boldsymbol{\phi}, \hat{\Sigma}_i(1 + \boldsymbol{\phi}^\top S_i \boldsymbol{\phi})) \tag{42}$$

for each logit element. Variance reduction schemes are possible for prediction, but they are beyond the scope of this paper.

B.4 Complexity and Parameterization

We follow the parameterization presented in Harrison et al. (2024), which proposes to parameterize the variational posterior $q(\bar{\boldsymbol{w}}_i, \Sigma_i S_i)$ via a simple unconstrained tensor for $\bar{\boldsymbol{w}}_i$ and with either a strictly positive diagonal or Cholesky-decomposed representation for S_i. Recall the inverse Gamma variational posterior is $q(\Sigma_i) = \mathcal{IG}(\nu_i, \Psi_i)$ with $\nu_i > 1, \Psi_i > 0$. Enforcing strict positivity is done via exponentiating tensors (that are unconstrained). Lower bounds on parameter values are similarly accomplished by adding the offset.

The addition of variational inference in the homoscedastic case adds minimal additional complexity. The only additional tensors added are those of the variational posterior. Thus, we add $2N_y$ parameters but do not estimate a point estimate for Σ_i, and thus add only N_y parameters.

C Heteroscedastic Noise Modeling

In this section we discuss approaches to modeling heteroscedastic noise that also quantifies the epistemic uncertainty associated with aleatoric noise prediction.

C.1 Bayesian Last Layer Methods for Heteroscedastic Noise

Our approach to aleatoric noise modeling in this work builds on the standard VBLL model. In particular, we use a VBLL variational posterior for the last layer in a covariance predictive model of the form

$$\log \Sigma_i = \boldsymbol{m}_i^\top \boldsymbol{\phi}(\boldsymbol{x}) \tag{43}$$

with $\boldsymbol{m}_i \sim q(\boldsymbol{m}_i) = \mathcal{N}(\bar{\boldsymbol{m}}, Z)$ (and similarly choose a Normal prior $\boldsymbol{m}_i \sim \mathcal{N}(\bar{\underline{\boldsymbol{m}}}, \underline{Z})$).

Given this structure, $\log \Sigma_i$ is Normally distributed and Σ_i is log-Normally distributed, as

$$\Sigma_i \sim \log \mathcal{N}(\bar{\boldsymbol{m}}_i^\top \boldsymbol{\phi}(\boldsymbol{x}), \boldsymbol{\phi}(\boldsymbol{x})^\top Z \boldsymbol{\phi}(\boldsymbol{x})) \tag{44}$$

We choose a prior of the same structure over $\boldsymbol{m}$, which we write $p(\boldsymbol{m})$. We established in Appendix A that the log-Normal covariance distribution is a reasonable one. While it allows for tractable variational objectives, it does not allow analytical marginalization for the predictive distribution. We will note identities which are critical in our development, which build upon those presented in Section 3:

$$\mathbb{E}_{q(\boldsymbol{m}_i)}[\Sigma_i] = \exp(\bar{\boldsymbol{m}}_i^\top \boldsymbol{\phi}(x) + \frac{1}{2}\boldsymbol{\phi}(x)^\top Z_i \boldsymbol{\phi}(x)) \tag{45}$$

$$\mathbb{E}_{q(\boldsymbol{m}_i)}[\Sigma_i^{-1}] = \exp(-\bar{\boldsymbol{m}}_i^\top \boldsymbol{\phi}(x) + \frac{1}{2}\boldsymbol{\phi}(x)^\top Z_i \boldsymbol{\phi}(x)) \tag{46}$$

$$\mathbb{E}_{q(\boldsymbol{m}_i)}[\log \Sigma_i] = \bar{\boldsymbol{m}}_i^\top \boldsymbol{\phi}(x) \tag{47}$$

We can compare this modeling approach to learning point estimates of Σ in the standard VBLL model, or to learning a standard variance prediction network. If we set $N_\phi = 1$ and set $\boldsymbol{\phi} = 1$, we exactly recover the VBLL noise estimation scheme under a log-Normal prior. Thus, this heteroscedastic noise scheme represents a strict generalization of the standard VBLL model. If we replace the variational posterior over $\boldsymbol{m}_i$ with a point estimate, we recover a standard variance prediction model.

C.2 Variational Lower Bound for VBLL-Variance Networks

We can obtain tractable variational objectives by combining our variance parameterization with the variational objectives obtained in the last section. Note that each input $\boldsymbol{x}$ induces a Σ and W in our generative model. For T training examples, our variational posterior is

$$q(W_{1:T}, M \mid X) = q(M) \prod_{t=1}^{T} q(W_t \mid M, \boldsymbol{x}_t) \tag{48}$$

where t indexes training data. Following the previous section and indexing rows with i (and dropping data indexing), the terms in this factorized variational posterior can be written

$$q(W \mid M, \boldsymbol{x}) = \prod_{i=1}^{N_y} q(\boldsymbol{w}_i \mid \boldsymbol{m}_i, \boldsymbol{x}) \tag{49}$$

with

$$q(\boldsymbol{w}_i \mid \boldsymbol{m}_i, \boldsymbol{x}) = \mathcal{N}(\bar{\boldsymbol{w}}_i, \Sigma_i(\boldsymbol{m}_i, \boldsymbol{x}) S_i) \tag{50}$$

and

$$q(M) = \prod_{i=1}^{N_y} q(\boldsymbol{m}_i). \tag{51}$$

With this variational posterior, we have

$$\log p(Y \mid X) \geq \mathbb{E}_{q(M)}[\log p(Y \mid X, M)] - \mathrm{KL}(q(M) \| p(M)) \tag{52}$$

$$\geq \sum_{(\boldsymbol{x}, \boldsymbol{y}) \in \mathcal{D}} \mathbb{E}_{q(W \mid M, \boldsymbol{x}) q(M)}[\log p(\boldsymbol{y} \mid \boldsymbol{x}, W, M)] - \mathrm{KL}(q(M) \| p(M)) \tag{53}$$

$$- \sum_{\boldsymbol{x} \in \mathcal{D}} \mathbb{E}_{q(M)}[\mathrm{KL}(q(W \mid M, \boldsymbol{x}) \| p(W \mid M, \boldsymbol{x}))].$$

There are two KL terms in (53). The term for $q(M)$ is a straightforward KL between Gaussians, and factorizes over the dimensionality of $\boldsymbol{y}$. The second KL term is

$$\sum_{\boldsymbol{x}\in\mathcal{D}}\sum_{i=1}^{N_y}\frac{1}{2}(\mathbb{E}_{q(\boldsymbol{m}_i)}[\Sigma_i^{-1}](\bar{\boldsymbol{w}}-\underline{\bar{\boldsymbol{w}}})^\top \underline{S}_i^{-1}(\bar{\boldsymbol{w}}-\underline{\bar{\boldsymbol{w}}})+\mathrm{tr}(\underline{S}_i^{-1}S_i)-\log\frac{\det S_i}{\det \underline{S}_i}-N_\phi) \tag{54}$$

which is tractable via (46).

Practically, the relative weight of the second KL term is much larger as it is the sum of T terms. In practice, we will scale this second KL term by $1/T$ (to match the relative weight of the first KL term), which improves performance. This larger weighting factor results from sharing the parameters of the variational posterior for each W (specifically $\bar{W}, S$).

C.2.1 Regression

To compute the likelihood term in (53) in the regression case, we build upon the objective as written in (26), and apply the developed identities for $\mathbb{E}[\Sigma_i^{-1}]$ and $\mathbb{E}[\log\Sigma_i]$.

C.2.2 Classification

To compute the likelihood term for the classification case, we use the previously developed objective in (38) and apply the expressions for $\mathbb{E}[\Sigma_i^{-1}]$ and $\mathbb{E}[\Sigma_i]$, yielding an analytically tractable heteroscedastic classification model.

C.3 Prediction

Because the variational posterior for the noise covariance is log-Normal, we lose conjugacy for prediction in the regression case. Thus, for both regression and classification, we turn to sampling. For the regression case, we sample Σ from the variational posterior and then marginalize $\boldsymbol{w}$ as in standard VBLL regression models. In classification, we are forced to sample the noise covariance Σ, and then sample from the conditional distribution over logits under the variational posterior.

C.4 Complexity and Parameterization

The approach to heteroscedastic modeling fundamentally relies on parameterizing two VBLL heads: one for the mean, and one for the noise covariance. Both are parameterized as with standard VBLLs, which can have a dense, diagonal, or low-rank covariance structure, and thus the complexity of this layer is twice the complexity of a standard VBLL layer. However, we note that VBLLs (with appropriately chosen covariance parameterizations) have comparable complexity to standard neural network layers, and thus the added computational cost of adding heteroscedasticity is comparable to adding an additional layer to a neural network. For a diagonal covariance, the last layer thus has twice as many parameters as a standard VBLL through a second head for the log noise variance, while t-VBLL adds only the inverse-Gamma parameters (see Section B.4). As the last layer is a small part of the network, the total parameter count changes little, with effects on wall-clock time and memory being task dependent. Denser noise covariances would increase this cost, although a diagonal-plus-low-rank structure would be only a mild increase.

D Training and Algorithmic Details

We train our t-VBLL and Het-VBLL models with standard neural network optimization strategies. Concretely, we jointly train the parameters of the variational posterior together with the the neural network features ϕ. In this work we only train feature point estimates, but it is also possible to train variational posteriors for features via Bayes-by-backprop (as discussed in Harrison et al. (2024)), or train ensembles of models with t-VBLL or Het-VBLL heads.

D.1 Training Methodology

We train via minibatch optimization, with the sums over the data replaced with (re-weighted) expectations over minibatches as in Blundell et al. (2015); Harrison et al. (2024). Note that this

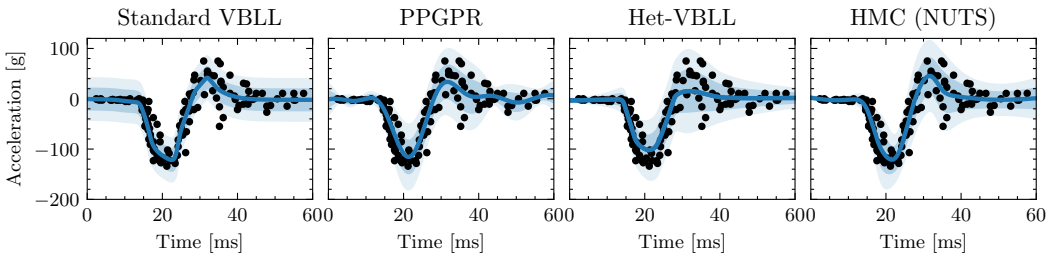


Figure 6: Comparison of VBLL (Harrison et al., 2024), Parametric Predictive GP Regression (Jankowiak et al., 2020), Het-VBLL, and Hamiltonian Monte Carlo (Girolami & Calderhead, 2011; Betancourt, 2017) on the Motorcycle data used in Figure 1

implies that, in the heteroscedastic model, the KL term for $q(M)$ should be weighted by one over the dataset size, whereas the other KL term and the likelihood term are averaged over the dataset.

The models presented in this paper can be used for full model training, or for a phase of training in more complex network training pipelines. For example, a t- or Het-VBLL head can be trained as a linear probe on pre-trained features, or may be used as a head for a fine-tuned model. Practically, VBLL-based models train slightly more slowly than standard neural networks due to (typically) being more heavily regularized, and thus training VBLL-based heads in a second phase of training often accelerates training versus training from scratch with VBLL heads.

D.2 Improving Gradient Estimation in Heteroscedastic Models

The faithful heteroscedastic regression method proposed in Stirn et al. (2023) has been reported to improve heteroscedastic variance output networks and can easily be applied on top of the proposed Het-VBLL models. We implement the faithful regression method by detaching the gradients from the variance output, such that they do not affect the learning of the features $\boldsymbol{\phi}(\boldsymbol{x})$, and scaling the gradient accordingly to obtain a Newton step rather than a gradient step. In regression, the detaching can be done specifically by detaching the features $\boldsymbol{\phi}(\boldsymbol{x})$ before computing the log-noise covariance term. Generally we observe minimal improvements when using faithful regression.

E Experiment Details and Further Results

E.1 Supervised Regression

E.1.1 Motorcycle Dataset

For the experiment in Figure 1 on the data set provided by Silverman (1985), we use for both the VBLL and three Het-VBLL three hidden layers with a feature dimension of $N_\phi = 64$, ELU activation functions, and a prior scale of one. As optimizer, we use AdamW (Loshchilov & Hutter, 2017). For the Het-VBLL, we choose a learning rate of $3e{-}4$ and a weight decay of $1e{-}5$. For the standard VBLLs, we choose a learning rate of $3e{-}3$ and a weight decay of 0. For both, we use gradient clipping at 1. We train both surrogates for 2000 epochs with a batch size of 32. We also include comparison models in Figure 6.

E.1.2 GP Sample-Path Regression with Known Aleatoric Noise

We construct a synthetic regression problem in which both the mean function and the noise field are known analytically at every input, which lets us ask separately *(i)* whether Het-VBLL outperforms the constant-noise variants in this controlled setting and *(ii)* whether its estimate of $\sigma(\boldsymbol{x})$ is accurate.

Data-Generating Process. On the domain $[0, 1]^d$ we draw the mean function f as a sample path from a zero-mean GP with an RBF kernel, using 1024 random Fourier features (Rahimi & Recht, 2008) and a lengthscale of $0.15\sqrt{d}$, so that the complexity of f per unit volume stays comparable as d grows. The path is normalised to zero mean and unit variance over the domain, which fixes the signal scale across seeds and dimensions. The noise field is a second, *independent* sample path g

of the same smoothness, pushed through the standard normal CDF Φ and mapped onto a controlled range,

$$y = f(x) + \sigma(\boldsymbol{x})\,\varepsilon, \qquad \log \sigma^2(\boldsymbol{x}) = \lambda_{\min} + (\lambda_{\max} - \lambda_{\min})\,\Phi(g(\boldsymbol{x})), \qquad \varepsilon \sim \mathcal{N}(0,1), \quad (55)$$

with $\lambda_{\min} = 2\log\sigma_{\min}$ and $\lambda_{\max} = 2\log\sigma_{\max}$ for $\sigma_{\min} = 0.05$ and $\sigma_{\max} = 0.5$. The noise standard deviation therefore varies by a factor of ten across the domain.

Besides this heteroscedastic setting we use two controls with the same mean function: a *homoscedastic* one with $\sigma(\boldsymbol{x}) \equiv 0.15$, which is the noise model standard VBLL assumes, and an *outlier* one that adds sparse zero-mean Laplace contamination of scale 1 to that constant noise with probability 0.05 per observation, which is the outlier model of our Bayesian optimization experiments in Appendix E.4 and the regime t-VBLL is designed for.

Input Distribution and Splits. Training inputs are drawn either uniformly over the domain or, where stated, from a mixture of $2d$ Gaussians so that the domain contains data-rich and data-poor regions. Because the noise field is independent of the input density, the clustered setting makes aleatoric and epistemic uncertainty spatially separable: a model that conflates the two cannot match both fields at once. The validation set contains $\max(500, n_{\text{train}}/2)$ points from the same distribution as the training inputs.

Models and Training. All models share the same backbone, i.e., , three hidden layers with $N_\phi = 64$ and ELU activations, and differ only in the last layer, so any difference in the results is attributable to the noise model. We compare standard VBLL (Harrison et al., 2024), which uses a point estimate of a constant Σ, against t-VBLL and Het-VBLL, all with a dense last-layer covariance. We train with AdamW with a weight decay of $1e{-}4$, a batch size of 128, gradient clipping at 1, and cosine annealing of the learning rate. Each model receives an identically sized hyperparameter search: two learning rates $\{3e{-}3, 1e{-}3\}$ crossed with two settings of its own noise-prior hyperparameter, namely the Wishart scale $\{1e{-}2, 1e2\}$ for VBLL, the prior and Wishart scales $\{(100, 100), (2, 0.5)\}$ for t-VBLL, and the noise prior scale $\{0.01, 0.1\}$ for Het-VBLL. Both settings of each pair are values used elsewhere in this paper in applications. The configuration is selected on validation NLL and the weights of the best-validation epoch are restored. For Het-VBLL we report the NLL of the marginal predictive density, estimated with 200 samples of Σ from the variational posterior.

Alongside test NLL and the RMSE of the predicted mean we report four measures of how well the aleatoric noise itself is recovered, all computed against the known $\sigma(\boldsymbol{x})$ on held-out inputs: the Spearman rank correlation ρ_s between $\hat\sigma(\boldsymbol{x})$ and $\sigma(\boldsymbol{x})$; the $\mathrm{RMSE}(\log\hat\sigma, \log\sigma)$, which is scale-free; the median ratio $\hat\sigma/\sigma$, which is 1 for an unbiased estimate; and the fraction of test inputs at which $\hat\sigma$ lies within $\pm 25\%$ of the truth. We take the posterior median of Σ as the point estimate of the noise variance, so that $\hat\sigma(\boldsymbol{x})$ denotes its square root. Following the parameterization of Section B.2, the predictive variance of t-VBLL and Het-VBLL factorises as $\mathrm{Var}[\boldsymbol{y} \mid \boldsymbol{x}] = (1 + \boldsymbol{\phi}(\boldsymbol{x})^\top S \boldsymbol{\phi}(\boldsymbol{x}))\,\mathbb{E}[\Sigma]$, so we report $\mathbb{E}[\Sigma]$ as the aleatoric component and $\boldsymbol{\phi}(\boldsymbol{x})^\top S \boldsymbol{\phi}(\boldsymbol{x})$ as the epistemic one.

Recovery of the Noise Field. Figure 3(a–c) shows the clustered setting at $d = 2$ with $n_{\text{train}} = 1000$ for a single representative seed; the numbers quoted below are averaged over five seeds. Het-VBLL recovers the noise field with a rank correlation of $\rho_s = 0.95$, a median $\hat\sigma/\sigma$ of 1.03, and an estimate within $\pm 25\%$ of the truth at 80% of held-out inputs. VBLL and t-VBLL are constant by construction and land inside that band at 20% and 21% of inputs respectively, with constant estimates inflated by roughly a third (1.35 and 1.33 in the median). In the uniformly sampled regime at $n_{\text{train}} = 8000$, where the noise field is fully identifiable, the recovery is essentially exact: $\rho_s = 0.99$, a median ratio of 1.00, and 99.7% of inputs within $\pm 25\%$, against 17% for both baselines. Across all of these runs the RMSE of the predicted mean is statistically indistinguishable between the three models.

Figure 9 varies the training-set size at $d = 2$. The recovery of Het-VBLL improves monotonically with data while the constant-noise models saturate. Figure 10 varies the input dimension at a fixed $n_{\text{train}} = 8000$, so that the dimension is the only quantity that changes. Recovery degrades with d, as expected once the data become scarce relative to the volume of the domain, but ρ_s remains well above zero and the ordering in NLL does not change. This does however also showcase that a sufficient coverage may be necessary to fully learn the true noise field.

Performance Under Different Noise Models. Het-VBLL is not intended to be preferable in all settings and in Figure 3(f) we showed that t-VBLL is preferable in settings with outliers. We observe

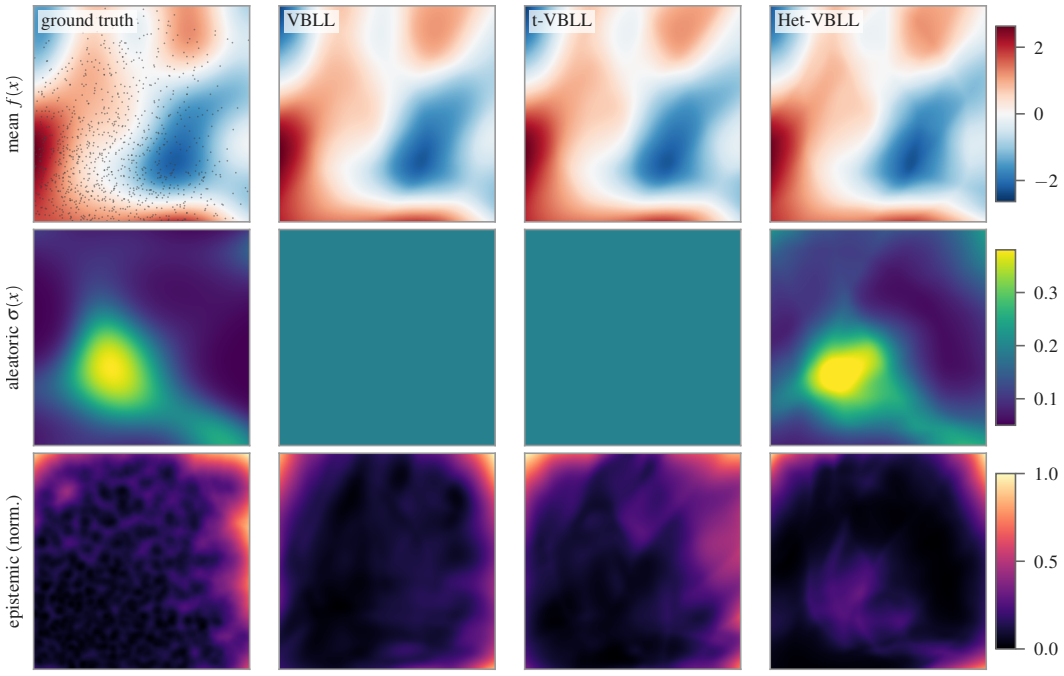


Figure 7: Full decomposition on the synthetic benchmark ($d = 2$, $n_{\text{train}} = 1000$, clustered inputs). Columns, from left to right: ground truth, VBLL, t-VBLL, Het-VBLL. The mean fit is statistically indistinguishable across models (top row), as it should be, since the models differ only in their noise model. The bottom row shows the epistemic term $\boldsymbol{\phi}(\boldsymbol{x})^\top S \boldsymbol{\phi}(\boldsymbol{x})$, normalised to $[0, 1]$.

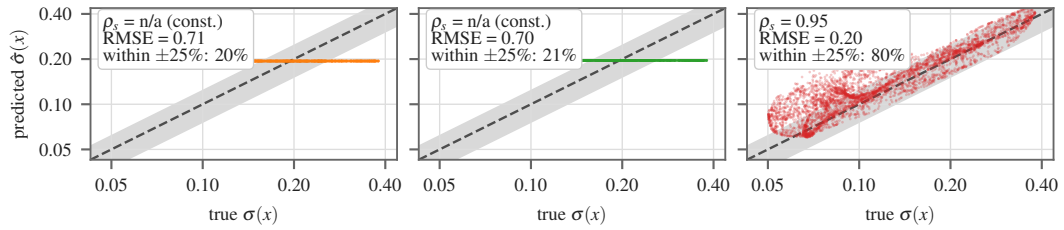


Figure 8: Predicted against true aleatoric noise on 4000 held-out inputs ($d = 2$, $n_{\text{train}} = 1000$, clustered inputs), for VBLL, t-VBLL and Het-VBLL. The shaded band marks $\pm 25\%$. Het-VBLL is unbiased to within 3% in the median and lands inside the band for four fifths of the domain; the constant-noise models are correct only where the true field happens to cross their estimate.

that each noise model wins in the regime it was designed for. The heavy-tailed predictive of t-VBLL is clearly preferable under outlier contamination, Het-VBLL as soon as the noise varies with the input, and all three models perform similar other under clean homoscedastic noise. The homoscedastic control also validates the recovery metrics themselves, i.e., , when the true noise field is constant, VBLL recovers it accurately, so the large errors it makes on the heteroscedastic problem are a property of the model class and not of the estimator.

Computational Resources. Because every seed draws a different mean function and noise field, the absolute metrics vary considerably across seeds while the comparison within a seed does not; we therefore report paired differences, i.e., NLL differences computed per seed and then averaged, wherever we compare models. We use five seeds for the clustered runs at $d = 2$ and three seeds elsewhere, and train for 2000 epochs at $n_{\text{train}} = 1000$ and 500 to 600 epochs in the larger-n_{train} runs and return the model with the lowest NLL. The number of epochs were chosen such that all models clearly converged. All runs were performed on the CPU of a 2021 MacBook Pro with an Apple M1 Pro chip and 16 GB of RAM.

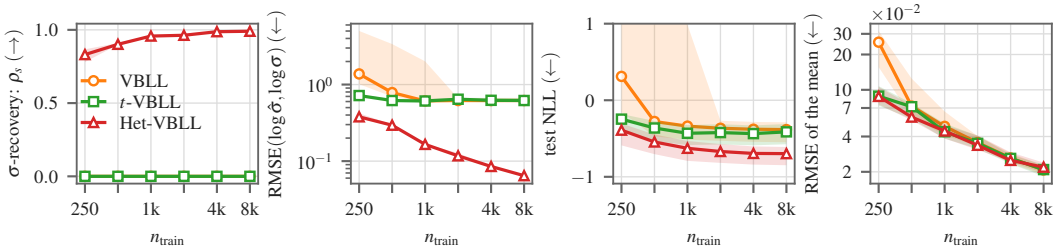


Figure 9: Ablation: increasing the dataset size at $d = 2$ (uniform inputs, median and IQR over seeds). The Spearman correlation is undefined for a constant estimate and is plotted as zero. Het-VBLL can recover the noise field as opposed to the other baselines.

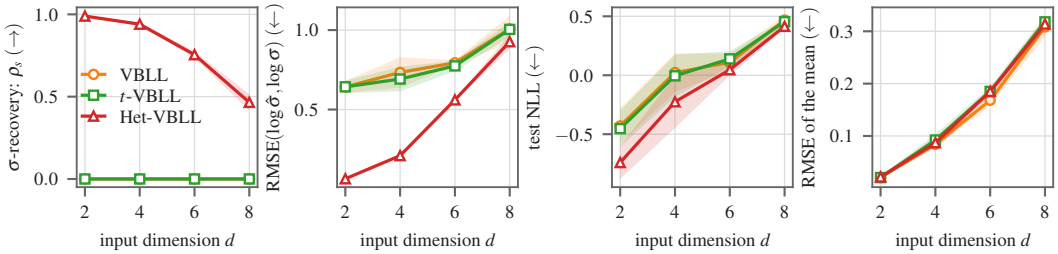


Figure 10: Ablation: increasing the input dimension at a fixed dataset size (uniform inputs, mean $\pm$ s.e. over seeds), so that the dimension is the only quantity that changes. Recovery degrades with d, as expected once the data become scarce relative to the volume of the domain, but it remains well above zero and the ordering in NLL does not change.

E.1.3 UCI Datasets

Results on UCI datasets (Dua & Graff, 2017) are shown in Tables 1 and 4. These experiments match the setting used in Watson et al. (2021) and Harrison et al. (2024), and we compare against their baselines. In particular, we use two hidden layer MLPs of width 50, and use a batch size of 32 (except for POWER in which we use batch size 256). We use AdamW (Loshchilov & Hutter, 2017) with a learning rate of $1e-3$ and weight decay of $1e-2$ on the hidden layers. We ran 10 seeds for each dataset. For more details on the experimental setting, we refer the reader to Harrison et al. (2024).

E.1.4 QM9 Property Prediction

We also run experiments on the QM9 property prediction dataset (Wu et al., 2018) using the zero-point energy U_0 as the target. We use the PaiNN message passing backbone from Schütt et al. (2021) with a modified readout function. The readout function has been modified such that full molecule energy is predicted based on *aggregate features* from all atoms in the molecule, rather than as a sum of energies from individual atoms in the molecule. This modification allows us to directly apply VBLL layers with any predictive distribution, which the original read-out function would not. To retain the inductive bias that total molecule energy scales linearly in the number of atoms in the molecule, we use summation as our aggregation function. See Figure 12 for illustration of original and modified readout functions. Finally, we use the reference atom energy scaling as provided in QM9, and perform a *per atom* standardization and scaling of energy values thus retaining the inductive bias of energy scaling with number of atoms in the molecule. We refer to the code for details.

In Figure 11 we highlight the calibrated ranking of the predictive uncertainty of the VBLL models by plotting confidence graphs that illustrate that the uncertainty estimates of the models generally have good rankings. The oracle lines indicate the theoretically optimal rankings. We illustrate that the predictive uncertainty rankings of the VBLL models, where we note that the t-VBLL models have more predictions with high predictive uncertainties, which we hypothesize is an affect of the heavier tailed predictive distributions. Finally, we find that even though the Het-VBLL models appear better calibrated without post-hoc calibration, their predictive uncertainty rankings are very similar to the other VBLL models.

Table 4: Further results for UCI regression tasks.

	Power		Wine		Yacht	
	NLL (↓)	RMSE (↓)	NLL (↓)	RMSE (↓)	NLL (↓)	RMSE (↓)
t-VBLL	2.75 ± 0.02	3.83 ± 0.07	0.91 ± 0.05	0.62 ± 0.03	0.99 ± 0.34	0.87 ± 0.21
Het-VBLL	$\mathbf{2.73 \pm 0.03}$	3.75 ± 0.09	0.92 ± 0.07	$\mathbf{0.61 \pm 0.03}$	0.74 ± 0.50	1.94 ± 1.03
VBLL	$\mathbf{2.73 \pm 0.01}$	$\mathbf{3.68 \pm 0.03}$	1.02 ± 0.03	0.65 ± 0.01	1.29 ± 0.17	0.86 ± 0.17
GBLL	2.77 ± 0.01	3.85 ± 0.03	1.02 ± 0.01	0.64 ± 0.01	1.67 ± 0.11	1.09 ± 0.09
LDGBLL	2.77 ± 0.01	3.85 ± 0.04	1.02 ± 0.01	0.64 ± 0.01	1.13 ± 0.06	0.75 ± 0.10
MAP	2.77 ± 0.01	3.81 ± 0.04	0.96 ± 0.01	0.63 ± 0.01	5.14 ± 1.62	0.94 ± 0.09
RBF GP	2.76 ± 0.01	3.72 ± 0.04	$\mathbf{0.45 \pm 0.01}$	$\mathbf{0.56 \pm 0.05}$	$\mathbf{0.17 \pm 0.03}$	$\mathbf{0.40 \pm 0.03}$
Dropout	2.80 ± 0.01	3.90 ± 0.04	0.93 ± 0.01	0.61 ± 0.01	1.82 ± 0.01	1.21 ± 0.13
Ensemble	$\mathbf{2.70 \pm 0.01}$	$\mathbf{3.59 \pm 0.04}$	0.95 ± 0.01	0.63 ± 0.01	0.35 ± 0.07	0.83 ± 0.08
SWAG	2.77 ± 0.02	3.85 ± 0.05	0.96 ± 0.03	0.63 ± 0.01	1.11 ± 0.05	1.13 ± 0.20
BBB	2.77 ± 0.01	3.86 ± 0.04	0.95 ± 0.01	0.63 ± 0.01	1.43 ± 0.17	1.10 ± 0.11

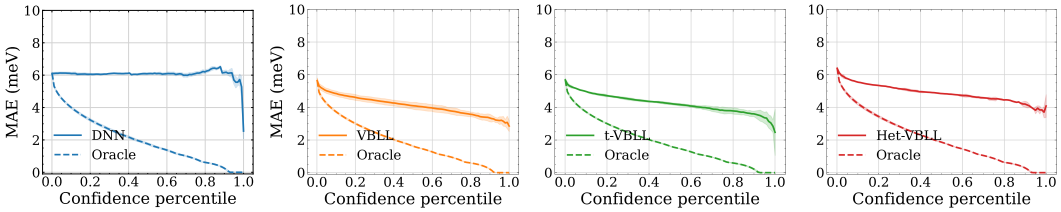


Figure 11: Calibration curves from models trained on QM9 where MAE is plotted as a function of confidence threshold. Oracle lines indicate the theoretically optimal ranking of predictive uncertainties.

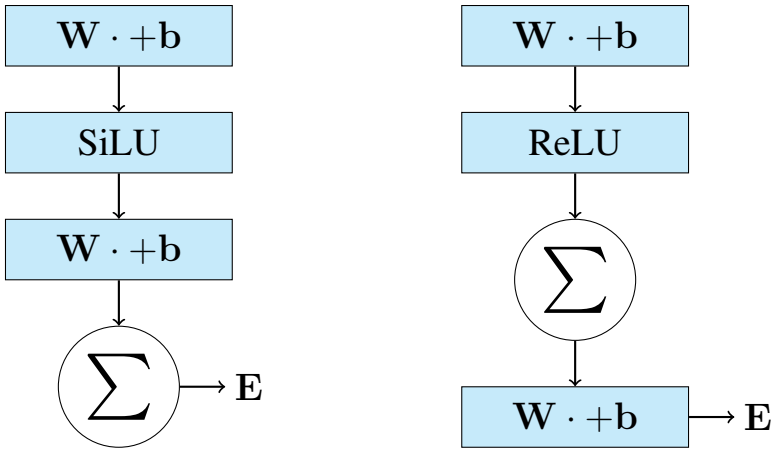
We train 3 seeds for all configurations and report mean and ± 2 standard error of the mean in all QM9 metrics and plots. For the DNN models we use all the same training parameters as in Schütt et al. (2021) except we set our weight decay to $3e-3$. In the VBLL models, we use a dense covariance for the last layer, a Wishart scale of $1e2$ and a prior scale of 1. For the t-VBLL models, we use a dense covariance, with degrees of freedom prior of 1, Wishart scale of 0.5 and prior scale of 2.0. For the Het-VBLL models we use a prior scale of 1, noise prior scale of 0.1 and grad correction of 1. We also use gradient clipping in the Het-VBLL models for stability. All models are trained with AdamW optimizer - as in Schütt et al. (2021) we use a reduce on plateau learning rate scheduler for the MAP estimated PaiNN network but use cosine annealing with VBLL, t-VBLL and Het-VBLL due to the validation loss being more volatile in the VBLL models.

We train the DNN, VBLL and t-VBLL models for 650 epochs and the Het-VBLL models for 1000 epochs, within which they all have converged on MAE, which is the accuracy metric of interest for molecular property prediction datasets. The models were each trained (with random allocation) on a single GPU using an internal CUDA HPC cluster. The hardware contained in the cluster includes: RTX A4000, RTX 2080 Ti, RTX A5000, GTX 1080 Ti, RTX 4070 Ti SUPER, Titan Xp, Titan V, L40S. Although training time varies with allocated GPU, running a single seed for any of the aforementioned methods takes approximately 15 hours. Based on this, reproducing the full QM9 results would take approximately 180 GPU hours. We do however note that tuning prior values and weight decay parameters cost additional compute.

E.2 Supervised Classification

E.2.1 Half Moon Dataset

For the two-moon data set in Figure 2, we use `sklearn` to generate the data and set the noise level to 0.25. We introduce input-dependent label noise by flipping 20% of the labels between $[0.5, 1]$ for feature 1. For all baselines, we use the same backbone configuration consisting of two hidden layers with 128 neurons ($N_\phi = 128$) and ELU activations. Further, we choose a learning rate of $1e-3$ and a weight decay of $1e-4$ and train all models for 1000 epochs. For the standard VBLL and the t-VBLL, we use a prior scale of 1 and a Wishart scale of 1. For the Het-VBLL, we choose a noise prior scale of 0.1 and a prior scale of 1. We trained each model on an M1 MacBook Pro, resulting in training times of 3.51s (MLP), 5.13s (VBLL), 5.39s (t-VBLL), and 7.37s (Het-VBLL).



(a) Original Readout Function (b) Modified Readout Function

Figure 12: Original (a) and modified (b) readout functions for PaiNN network

Table 5: Results on CIFAR-10

Method	Feature Weights	Accuracy (↑)	ECE (↓)	NLL (↓)	SVHN OOD (↑)	CIFAR-100 OOD (↑)
DNN	Trained	95.8 ± 0.19	0.028 ± 0.028	0.183 ± 0.007	0.946 ± 0.005	0.893 ± 0.001
SNGP	Trained	95.7 ± 0.14	0.017 ± 0.003	0.149 ± 0.005	0.960 ± 0.004	$\mathbf{0.902 \pm 0.003}$
VBLL	Trained	$\mathbf{96.4 \pm 0.12}$	0.022 ± 0.001	0.160 ± 0.001	0.969 ± 0.004	$\mathbf{0.900 \pm 0.004}$
VBLL	Frozen	$\mathbf{96.4 \pm 0.01}$	0.024 ± 0.000	0.176 ± 0.000	0.943 ± 0.002	0.895 ± 0.000
t-VBLL	Frozen	96.3 ± 0.03	$\mathbf{0.006 \pm 0.000}$	$\mathbf{0.133 \pm 0.001}$	$\mathbf{0.975 \pm 0.000}$	0.892 ± 0.001
Het-VBLL	Frozen	96.2 ± 0.02	0.009 ± 0.000	0.135 ± 0.000	$\mathbf{0.975 \pm 0.001}$	0.894 ± 0.001
LLLA	Frozen	96.3 ± 0.03	0.010 ± 0.001	$\mathbf{0.133 \pm 0.003}$	0.965 ± 0.010	$\mathbf{0.898 \pm 0.001}$
VBLL	Fine-Tuned	95.6 ± 0.02	0.025 ± 0.000	0.168 ± 0.000	0.955 ± 0.005	$\mathbf{0.900 \pm 0.002}$
t-VBLL	Fine-Tuned	95.6 ± 0.01	0.014 ± 0.001	0.175 ± 0.002	0.950 ± 0.013	0.834 ± 0.007
Het-VBLL	Fine-Tuned	95.7 ± 0.03	0.111 ± 0.000	0.273 ± 0.003	0.916 ± 0.020	0.772 ± 0.001

E.2.2 CIFAR 10 and CIFAR 100

For the CIFAR 10 and CIFAR 100 datasets, we compare our models against other last-layer methods, swapping out the classification head of a pretrained and frozen Wide ResNet-28-10 network following Liu et al. (2022). These experiments match the settings used in Harrison et al. (2024), and we compare against their reported baselines. In particular we compare against standard VBLL (Harrison et al., 2024) and last-layer Laplace (Daxberger et al., 2021) methods, as well as the results for the standard (vanilla) network. Performance is evaluated through accuracy, calibration error (ECE), negative log-likelihood (NLL) and out-of-distribution AUROC for both near and far OOD detection capabilities. For CIFAR-10, we assess near OOD performance using CIFAR-100 as the out-of-distribution dataset and vice versa for the CIFAR-100 OOD evaluation. In both cases, we utilize Street View House Numbers (SVHN) (Netzer et al., 2011) as a far OOD dataset. For t-VBLL we set a prior scale of 1 and an inverse Gamma scale parameter of 10. For Het-VBLL we select a noise prior scale of 0.01 and prior scale of 10. All models are trained using the AdamW optimizer and a learning rate of 1e-3 with a linear warmup. During fine-tuning, the classification head is first trained for two warmup epochs, after which the encoder unfrozen the entire network is jointly optimized. We train 3 seeds for all configurations and report mean and ±1 standard error of the mean for all metrics. On CIFAR-10 (Table 5), both t-VBLL and Het-VBLL achieved competitive accuracy while significantly reducing Expected Calibration Error and Negative Log-Likelihood, with improved out-of-distribution detection performance.

E.3 Semantic Segmentation

VBLL SegFormer and Het-VBLL SegFormer were trained and implemented using the MMSegmentation (MMSegmentation, 2020) framework. We employed an AdamW optimizer with a learning rate of 6e-5 for 160,000 iterations. A linear learning rate warm up period was followed by polynomial

Table 6: Results on CityScapes Semantic Segmentation - Validation

IoU	SegFormer-B0	VBLL SegFormer-B0	Het-VBLL SegFormer-B0
Road	0.980	0.980	**0.982**
Sidewalk	0.840	0.843	**0.847**
Building	0.922	0.922	**0.923**
Wall	0.588	**0.628**	0.587
Fence	**0.568**	0.564	0.552
Pole	0.626	0.613	**0.630**
Traffic Light	**0.698**	0.685	0.698
Traffic Sign	0.777	0.764	**0.782**
Vegetation	**0.926**	0.924	**0.926**
Terrain	0.641	**0.642**	0.639
Sky	**0.950**	0.948	**0.950**
Person	0.807	0.799	**0.808**
Rider	0.574	0.574	**0.587**
Car	0.943	0.942	**0.945**
Truck	0.694	**0.778**	0.719
Bus	0.839	0.841	**0.856**
Train	0.757	0.759	**0.762**
Motorcycle	0.649	0.631	**0.650**
Bicycle	**0.763**	0.754	0.761
mIoU	0.765	0.768	**0.769**

learning rate decay schedule, according to the standard training protocol for this architecture on CityScapes implemented by MMSegmentation. For Het-VBLL Segformer we select a noise prior scale of 0.01 and prior scale of 1.0. VBLL Segformer utilizes a prior scale of 1.0 and a Wishart scale of 10.0. Due to the training cost of the SegFormer models, we report a single run per model and cannot provide standard errors across seeds.

E.4 Bayesian Optimization

All models are implemented using GPyTorch (Gardner et al., 2018) and BoTorch (Balandat et al., 2020). For the Student-t predictive of the t-VBLLs, we directly use the standard deviation of the predictive in UCB. For the Het-VBLL model, we construct this standard deviation by sampling ten Σ from the variational posterior as discussed in Sec. C.3 and then use their mean in the acquisition function. For the outliers, we add heavy-tailed noise from a zero mean Laplace distribution with a standard deviation of 0.5 (Pest Control), 1 (Ackley), or 5 (Lunar Lander) to the output of an experiment according to the outlier probability of the experiment. To calculate the minimum instantiations regret, we approximate the optimal value of Pest Control as 12 and for Lunar Lander as 300. For Ackley, the known optimal value is 0.

VBLL Baselines. For all VBLL baselines, we use 3 hidden layers with 64 neurons and ELU activations and set the maximum training epochs to 10000. For all models but the Het-VBLL, we use a patience of 100 (Brunzema et al., 2025). For the Het-VBLL, we further set the gradient correction scale to 1 as in Appendix E.1.4. All models are trained using AdamW (Loshchilov & Hutter, 2017) and we use a gradient clipping of 1.

Variance Predictive Baseline. For the variance predictive baseline, we use the same backbone as for the VBLL baselines, i.e., 3 hidden layers with 64 neurons and ELU activations, and further adopt the same optimization routine above.

GP Baselines. For all the GP baselines, we choose a Matérn kernel with $\nu = 2.5$ and use individual lengthscales ℓ_i for all input dimensions. Our standard GP baseline, `GP`, optimizes these individual lengthscales within the box constraints $\ell_i \in [0.005, 4]$ as in (Eriksson et al., 2019). Furthermore, we add a GP with D-scaled priors (Hvarfner et al., 2024), `D-GP`, which have demonstrated increased sample efficiency on some tasks in previous work. Lastly, we add a GP with a Yeo-Johnson (YJ) outcome transformation to cope with heteroskedasticity as proposed in (Cowen-Rivers et al., 2022), `YJ-GP`, as a baseline. We refit the YJ transformation at each time step and use the same lengthscale

configuration as for the base GP baseline. For all GP baselines, we learn the hyperparameter of the GP at each iteration through maximum likelihood estimation.

Computational Resources. All BO simulations were performed on an HPC cluster with Intel Xeon 8468 Sapphire at 2.1 GHz using 4 cores. The average wall clock time for Lunar Lander was for GP 1.85 hours, for VBLL 9.02 hours, for t-VBLL 7.2 hours, and for het-VBLL 20 hours. It is however important to note that surrogate fit time is not of high importance in the context of BO as the experiment is usually assumed to be the time consuming bottleneck.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: They do.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Contained in discussion section.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

   Justification: Contained in Appendices B, C.

   Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: See Appendices D, E.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: All datasets are public, and all experimental details are included.

   Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (https://neurips.cc/public/guides/CodeSubmissionPolicy) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (https://neurips.cc/public/guides/CodeSubmissionPolicy) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: It does.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: It does.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
   - If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Contained in Appendix E.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics https://neurips.cc/public/EthicsGuidelines?

   Answer: [Yes]

   Justification: It does.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: The paper is quite narrowly technical; it discusses the applications of uncertainty quantification which generally covers the societal implications.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.

- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: All data is already public.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: All datasets cited.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [N/A]

    Justification: No new assets introduced in the paper.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.

- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: No research with human subjects in this paper.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: No research with human subjects in this paper.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: No LLM work in this paper.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.