Mitigating Model Non-Identifiability in BNN with Latent Variables

Yaniv Yacoby [*1] **Weiwei Pan** [*1] **Finale Doshi-Velez** [1]

Abstract

Bayesian Neural Networks with Latent Variables (BNN+LV) is a deep model that provides estimates of predictive uncertainties through priors on the network parameters as well as a latent input noise variable. However, BNN+LV suffers from model non-identifiability: there are many sets of parameter and latent input values that are equally plausible for a given set of observed data. We show that traditional inference methods tend to yield parameters that reconstruct observed data well, but generalize in undesirable ways. In this paper, we describe the non-identifiability in BNN+LV models and propose a novel inference procedure that yield high quality predictions as well as uncertainty estimates. We demonstrate that our inference method improves upon benchmark methods across a range of synthetic and real datasets.

1. Introduction

While deep learning has been recently applied to many real-world tasks with significant success (LeCun et al., 2015), the current focus of deep learning on learning point estimates of model parameters can lead to overfitting and provides no uncertainty quantification on predictions. When machine learning models are applied to critical domains such as autonomous driving, precision health care, or criminal justice, reliable measurements of a model's predictive uncertainty may be as crucial as correctness of its predictions.

In general, prediction uncertainty comes from two sources. *Epistemic* uncertainty, or model uncertainty, comes from having insufficient knowledge about the "true" predictor. In contrast, *aleatoric* uncertainty comes from the stochasticity inherent the environment (Der Kiureghian & Ditlevsen, 2009; Kendall & Gal, 2017). Bayesian Neural Networks with latent variables (BNN+LV) (Wright, 1999; Depeweg et al., 2018) provide a way of explicitly modeling these two types of uncertainties in deep models. In particular, a BNN+LV model assumes a predictor of the following form:

$$y = f(x, z; W) + \epsilon$$

where ϵ is the output noise, W are the parameters of a neural network, z is an unobserved (latent) random variable associated with *each* (x, y) pair. In BNN+LV, distributions over W captures model uncertainty (epistemic uncertainty), while the stochastic input z captures the stochasticity of the data generating process. Together with the output noise ϵ, the stochastic input z model the sources of aleatoric uncertainty; and while z can have a fixed variance, it can capture heteroscedastic noise patterns after being transformed by the non-linear function f (Depeweg et al., 2018).

In this paper, we first show that BNN+LV models are unidentifiable in many cases. Specifically, there are multiple sets of values for network parameters and latent variables that are equally highly plausible for the observed data, but most of these parameters will parametrize networks that generalize poorly. We show that traditional training methods tend to find suboptimal solutions. Secondly, we introduce an approximate inference scheme, Noise Constrained Approximate Inference (NCAI), that explicitly mitigates the effects of model non-identifiability during training. We demonstrate that our approach consistently recovers approximate posteriors that are closer to the true posteriors on synthetic examples, and that we achieve better generalization on an array of synthetic and real-world data sets.

Related Works In Bayesian regression, one generally assumes that the irreducible noise (aleatoric uncertainty) in the data is identically, independent distributed and that structure in the predictive uncertainty arises from complex forms of epistemic uncertainty (due to insufficient observation). Bayesian Neural Networks (BNN's) capture epistemic uncertainty with a prior distribution over the parameters of a neural network predictor (MacKay, 1992; Neal, 2012). However, for many real-world tasks one needs to model complex forms of aleatoric uncertainty via heteroscedastic noise (Kendall & Gal, 2017; Depeweg et al., 2018).

Heteroscedastic noise in regression has been addressed by incorporating an input-dependent output noise variable or by incorporating a simple input noise variable that is able

[*]Equal contribution [1]Harvard University, Cambridge, MA. Correspondence to: Yaniv Yacoby <yanivyacoby@g.harvard.edu>, Weiwei Pan <weiweipan@g.harvard.edu>.

Presented at the ICML 2019 Workshop on Uncertainty and Robustness in Deep Learning.

to capture heteroscedasticity after being transformed by a non-linear predictor. Input-dependent output noise models have been formulated for Gaussian Processes (GP's) (Le et al., 2005; Wang & Neal, 2012; Kersting et al., 2007) and Bayesian Neural Networks (Kendall & Gal, 2017; Gal, 2016). On the other hand, while there are a number of works that model heteroscedastic noise for GP's through an input noise variable (Lawrence & Moore, 2007; McHutchon & Rasmussen, 2011; Damianou et al., 2014), there are only a few works that do the same for Bayesian Neural Networks (Wright, 1999; Depeweg et al., 2018).

To our knowledge, we provide the first description of model non-identifiability in Bayesian Neural Networks with Latent Variables and of how non-identifiability impacts inference. Based on our analysis of the sources of non-identifiability in BNN+LV models, we propose a novel framework for performing high quality approximate inference.

2. Background

Let $\mathcal{D} = \{(x_1, y_1), \dots (x_N, y_N)\}$ be a dataset of N observations. Where each input $x_n \in \mathbb{R}^D$ is a D-dimensional vector and each output $y_n \in \mathbb{R}^L$ is L-dimensional.

A *Bayesian Neural Network (BNN)* assumes a predictor of the form $y = f(x; W) + \epsilon$, where f is a neural network parametrized by W and ϵ is a normally distributed noise variable. Predictive uncertainty in a BNN is modeled by a posterior predictive distribution $p(y|x, \mathcal{D})$, obtained by placing a prior $p(W)$ on the network parameters and inferring a posterior distribution $p(W|\mathcal{D})$ over W.

Bayesian Neural Network with Latent Variables (BNN+LV), extends BNN's by introducing a latent variable $z_n \sim \mathcal{N}(0, \sigma_z^2 \cdot I)$ per observation (x_n, y_n) explicitly modeling white noise in the data generation process (Depeweg et al., 2018). We assume the following data generation process:

$$\begin{aligned} &W \sim p(W), \quad \epsilon_n \sim \mathcal{N}(0, \sigma_\epsilon^2), \quad z_n \sim p(z), \\ &y_n = f(x_n, z_n; W) + \epsilon_n, \quad n = 1, \dots, N. \end{aligned} \tag{1}$$

For this paper, we will set $W \sim \mathcal{N}(0, \sigma_w \cdot I)$. We note that when f is non-linear, BNN+LV is able to model heteroscedastic noise through z.

Our goal is to learn the posterior distribution, $p(W, \{z_1, \dots, z_N\}|\mathcal{D})$, over both network weights and the latent input noise. During test time, predictions are then made using the mean value of the posterior predictive distribution, $p(y|x, \mathcal{D})$: $p(y|x, \mathcal{D}) = \iint p(y|x, z, W)p(W|\mathcal{D})p(z)dzdW$.

3. Non-identifiability in BNN+LV Models

Unfortunately, BNN+LV suffers from model non-identifiability. Consider output generated by a single node neural network with LeakyRelu activation (assuming zero network biases, unit input weights and additive input noise):

$$f(x, z; W) = \max\{W(x + z), \alpha W(x + z)\}, \tag{2}$$

where $\alpha \in (0, 1)$. For any non-zero constant C, the pair $\widehat{W}^{(C)} = W/C$, $\widehat{z}^{(C)} = (C-1)x + Cz$ reconstructs the observed data equally well: $f(x, z; W) = f(x, \widehat{z}, \widehat{W})$. Now, suppose that the output is observed with Gaussian noise. Then the true values of the parameter W and the latent input noise z are equally likely as $\widehat{W}^{(C)}$ and $\widehat{z}^{(C)}$ under the likelihood. In theses cases, one typically place a prior on W in order to bias the posterior towards the ground truth parameter. However, we show in Theorem 1 (Appendix 7.1) that *the posterior over the model parameter W is biased away from the ground truth, regardless of the choice of W prior and even as the sample size grows*. Furthermore, in Appendix 7, we show that sources of non-identifiability increases when f is a neural network.

Model non-identifiability decreases the quality of the predictive distribution, even when the latter can be exactly inferred. This is because predictions are averaged during test time across weights drawn from multiple posterior high density regions, many of which, we show, will parameterize functions that generalize differently than the function generating the data. We argue that the effect of non-identifiability on approximate inference can be especially significant. Although a number of recently introduced approximate inference methods demonstrate increasing ability to approximate complex posterior distributions (Hernández-Lobato et al., 2016; Hernández-Lobato & Adams, 2015; Liu & Wang, 2016; Louizos & Welling, 2016; 2017), when approximate inference happens to capture posterior regions corresponding to functions different from the ground truth, the learned models will generalize poorly on new data (see Section 5).

4. Noise Constrained Approximate Inference

From the forms of non-identifiability we derive in Section 3 and Appendix 7.3, we see that when we scale the parameters W, the learned latent variable z becomes directly dependent on the input x or indirectly dependent on x through y, thus violating our assumption that z represents i.i.d. noise. Also, the distribution of z may no longer have the original isotropic Gaussian form assumed in the generative process. Based on these observation, we propose a framework, *Noise Constrained Approximate Inference (NCAI)*, for performing variational inference on BNN+LV models that consists of: (1) an intelligent initialization – we initialize W with parameters from a model that explains the data without using input noise and we initialize z's that are sampled i.i.d. from the prior; (2) a constrained learning procedure – we *explicitly* penalize violation of modeling assumptions during training.

Model-satisfying Initialization. Since local optima are a

major concern in BNN+LV inference, we start with settings of the variational parameters ϕ that satisfy the properties implied by our generative model (Equation 1). In particular, we initialize the variational means μ_{z_n} of the latent noise variable be draws from the prior (and thus independent of x). We initialize the variational means μ_{w_i} of the weights (except for weights associated with the input noise) with those of a neural network trained on the same data. We do so based on the observation that a neural network is often able to capture the mean of the data (but not the uncertainty). Lastly, we initialize all variational variances randomly.

Model Constrained Inference. We further ensure that the two key modeling assumptions—that the noise variables z are drawn *independently* and *identically* from $p(z)$—remain satisfied during training by adding constraints to our variational objective:

$$\begin{aligned} &\min_{\phi} -\text{ELBO}(\phi) \quad \text{s.t} \\ &\text{Dep}(x, z) = 0 \;\text{ and }\; \text{Div}(q(z), p(z)) = 0. \end{aligned} \tag{3}$$

where $\text{Dep}(x, z)$ is any metric measuring the statistical dependence between z and x (enforcing the independent sampling assumption), and $\text{Div}(q(z), p(z))$ is any metric quantifying the distance between $q(z)$ and $p(z)$ (enforcing identical sampling from $p(z)$ assumption). We solve the problem in (3) by gradient descent on the Lagrangian:

$$\begin{aligned} \mathcal{L}_{\text{NCAI}}(\phi) = &- \text{ELBO}(\phi) \\ &+ \lambda_1 \text{Dep}(x, z) + \lambda_2 \text{Div}(q(z), p(z)). \end{aligned} \tag{4}$$

where in (3) and (4), $q(z)$ is the aggregated posterior (Makhzani et al., 2015):

$$q(z) = \mathbb{E}_{p(x,y)}\left[q(z|x, y)\right] \approx \frac{1}{N}\sum_{n=1}^{N} p(z|x_n, y_n). \tag{5}$$

Choosing differentiable non-parametric forms of Dep and D is the key challenge. In Appendix 10, we describe our choices for tractable instantiations of the NCAI objective.

5. Experiments

We consider 5 synthetic datasets that are frequently used in heteroscedastic regression literature and 6 real datasets. Appendix 11 describes all datasets. Each dataset is split into 5 random train/validation/test sets. For every split of each data set, each method is evaluated on the best learned model out of 10 random restarts (details in Appendix 12).

We compare NCAI on BNN+LV with unconstrained Mean Field BBB (Blundell et al., 2015) (the latter denoted BNN+LV). We also compare selecting constraint strength parameters λ_i of NCAI through cross validation (denoted NCAI_λ) against fixing them at zero (denoted $\text{NCAI}_{\lambda=0}$). Furthermore, we compare the performance of BNN+LV's (for all inference procedures) with that of BNN's.

We evaluate the learned models for quality of fit (measured by test average log-likelihood, Root Mean Square Error, calibration of posterior predictives) and the learned latent variables for satisfaction of the white noise assumption (measured by the Henze-Zirkler test-statistic for normality, mutual information, Jensen-Shannon and KL divergence between the recovered and true noise priors). Computational details for evaluation metrics are in Appendix 12.

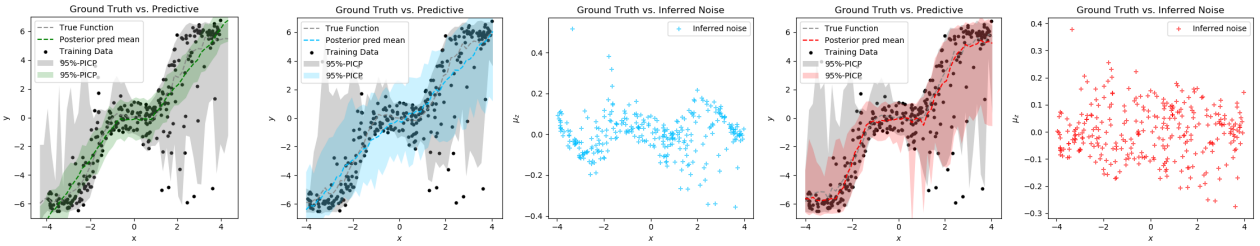
Experimental results are summarized in Table 1, 8 and 3 (additional evaluations of model generalization and model assumption satisfaction are summarized in Appendix 13). Overall, we see that training with NCAI_λ recovers latent variables that better satisfy model assumptions – have low mutual information with x and are distributed like an isotropic Gaussian. NCAI_λ also learns models with improved generalization – across average marginal log-likelihood and predictive RMSE, our method is comparable or better on all datasets except for Energy Efficiency, where the BNN model performs best in terms of test log-likelihood but drastically underestimates the uncertainty in the data (see posterior predictive metrics in Table 14). Furthermore, we see that intelligent initialization without constraints ($\text{NCAI}_{\lambda=0}$), while always outperforming the baselines, *does not* always learn the best models – that is, the constraints imposed in NCAI are indeed necessary.

Figure 1 shows a qualitative comparison of the posterior predictive distributions of BNN+LV trained with NCAI_λ compared with benchmarks (posterior predictives visualization for all univariate data sets are in the Appendix 13). We see that, as expected, BNNs underestimate the posterior predictive uncertainty, whereas BNN+LV with unconstrained inference improves upon the BNN in terms of log-likelihood by expanding posterior predictive uncertainty nearly symmetrically about the predictive mean. The predictive distribution obtained by BNN+LV trained with NCAI, however, is able to capture the asymmetry of the observed heteroscedasticity. Furthermore, while unconstrained inference on BNN+LV recovers latent noise that is highly correlated with y, NCAI_λ recovers latent noise that better aligns with the data generating model.

6. Discussion & Conclusion

In this paper we identify a key issue – model non-identifiability – with a class of flexible latent variable models for Bayesian regression, BNN+LV. By analyzing the sources of non-identifiability in BNN+LV models, we propose a novel approximate inference framework, NCAI, that explicitly enforces model assumptions during training.

Non-identifiability negatively impacts inference in the-



(a) BNN (b) BNN+LV with Mean Field BBB (c) BNN+LV with NCAI

Figure 1. Comparison of posterior predictives. BNN captures trend but underestimates variance; BNN+LV with Mean Field BBB captures more variance but learns z's that dependend on the input. BNN+LV with NCAI$_\lambda$ best captures heteroscedasticity and learns z's that best resemble white noise.

	Heavy Tail	*Goldberg*	*Williams*	*Yuan*	*Depeweg*
BNN	-2.47 ± 0.083	-1.055 ± 0.08	-1.591 ± 0.417	-2.846 ± 0.346	-2.306 ± 0.059
BNN+LV	-1.867 ± 0.078	-1.026 ± 0.056	-1.033 ± 0.156	-1.278 ± 0.164	-2.342 ± 0.048
NCAI$_{\lambda=0}$	-1.481 ± 0.018	$\mathbf{-0.962 \pm 0.040}$	$\mathbf{-0.414 \pm 0.184}$	$\mathbf{-1.211 \pm 0.083}$	$\mathbf{-1.973 \pm 0.049}$
NCAI$_\lambda$	$\mathbf{-1.426 \pm 0.042}$	-0.963 ± 0.041	$\mathbf{-0.414 \pm 0.184}$	$\mathbf{-1.211 \pm 0.083}$	$\mathbf{-1.973 \pm 0.049}$

Table 1. Comparison of *test log-likelihood* on synthetic datasets ($\pm$ std). For all datasets NCAI$_\lambda$ training yields comparable if not better generalization. NCAI training outperforms BNN+LV with Mean Field BBB as well as BNN. Results for RMSE are in Appendix 13.

	Heavy Tail	Goldberg	Williams	Yuan	Depeweg
BNN+LV	0.243 ± 0.079	0.229 ± 0.113	0.982 ± 0.121	$\mathbf{0.24 \pm 0.129}$	0.428 ± 0.04
NCAI$_{\lambda=0}$	0.051 ± 0.049	$\mathbf{0.02 \pm 0.024}$	$\mathbf{0.519 \pm 0.091}$	0.283 ± 0.112	$\mathbf{0.032 \pm 0.017}$
NCAI$_\lambda$	$\mathbf{0.036 \pm 0.04}$	0.046 ± 0.067	$\mathbf{0.519 \pm 0.091}$	0.283 ± 0.112	$\mathbf{0.032 \pm 0.017}$

Table 2. Comparison of *mutual information* between z and x on synthetic datasets ($\pm$ std). Across all but one of the datasets, NCAI$_\lambda$ training learns z's that has the least mutual information. Additional evaluations of model assumption satisfaction are in Appendix 13.

	Abalone	Airfoil	Energy	Lidar	Wine	Yacht
BNN	-1.248 ± 0.153	-0.995 ± 0.143	$\mathbf{1.281 \pm 0.171}$	-0.31 ± 0.069	-1.143 ± 0.027	0.818 ± 0.187
BNN+LV	-0.843 ± 0.071	-0.512 ± 0.083	0.573 ± 0.288	0.129 ± 0.131	-1.709 ± 0.22	0.638 ± 0.121
NCAI$_{\lambda=0}$	$\mathbf{-0.831 \pm 0.086}$	$\mathbf{-0.462 \pm 0.056}$	0.862 ± 0.138	$\mathbf{0.269 \pm 0.107}$	-1.147 ± 0.025	$\mathbf{0.832 \pm 0.077}$
NCAI$_\lambda$	$\mathbf{-0.831 \pm 0.086}$	$\mathbf{-0.462 \pm 0.056}$	0.898 ± 0.452	0.263 ± 0.11	$\mathbf{-0.849 \pm 0.038}$	$\mathbf{0.832 \pm 0.077}$

Table 3. Comparison of *test log-likelihood* on real datasets ($\pm$ std). Across all but one dataset BNN+LV with NCAI$_\lambda$ training yields better or comparable generalization. In particular, NCAI training outperforms BNN+LV with Mean Field BBB. Results for RMSE are in Appendix 13. Evaluations of model assumption satisfaction are in Appendix 13

ory and practice. In Section 3 we show that BNN+LV models are generally non-identifiable, the data generating model is difficult to learn regardless of the choices of priors and the quantity of observed data. Specifically, in the posterior distribution, ground truth model parameters (and true latent noise variables) can be as less likely as parameters that generalize poorly. At test time, averaging over models or sampling a single model from this posterior will decrease predictive quality, even when inference can be performed exactly. Empirically, we show that non-identifiability poses problems for inference on *most* datasets (BNN+LV with unconstrained training results in inferior models).

The ELBO cannot distinguish optimal and suboptimal models. We empirically verified that the ELBO cannot distinguish qualitatively different solutions. Across all synthetic data sets, we've observed cases where the ELBO evaluates a superior model as equal to an inferior one. Thus, by optimizing ELBO alone one cannot hope to consistently recover models that match the data generation process.

The NCAI constraints are necessary and effective. In Section 3 and Appendix 7, we show that when learned models reconstruct the observed data well but generalize poorly, the discrepancy is often attributable to the latent variable encoding the data. This encoding yields latent variables that violate our white-noise assumption and justifies the constraints that we impose in NCAI. Experiments show that NCAI$_\lambda$ training recovers z's that satisfy model assumptions and W's that generalize well, providing empirical evidence that our constraints are necessary and effective.

Overall On synthetic and real datasets we demonstrate the ability of NCAI to recover latent variables that better satisfy the white-noise assumption as well as to learn models that have improved generalization.

Acknowledgements

Yaniv Yacoby is partially supported by an IBM Faculty Research Award. Weiwei Pan is supported by the Institute of Applied Computational Science at Harvard University.

References

Kolmogorov–Smirnov Test, pp. 283–287. Springer New York, New York, NY, 2008. ISBN 978-0-387-32833-1. doi: 10.1007/978-0-387-32833-1_214. URL https://doi.org/10.1007/978-0-387-32833-1_214.

Blundell, C., Cornebise, J., Kavukcuoglu, K., and Wierstra, D. Weight uncertainty in neural networks. *arXiv preprint arXiv:1505.05424*, 2015.

Damianou, A. C., Titsias, M. K., and Lawrence, N. D. Variational inference for uncertainty on the inputs of gaussian process models. *arXiv preprint arXiv:1409.2287*, 2014.

Depeweg, S., Hernandez-Lobato, J.-M., Doshi-Velez, F., and Udluft, S. Decomposition of uncertainty in bayesian deep learning for efficient and risk-sensitive learning. In *International Conference on Machine Learning*, pp. 1192–1201, 2018.

Der Kiureghian, A. and Ditlevsen, O. Aleatory or epistemic? does it matter? *Structural Safety*, 31(2):105–112, 2009.

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

Gal, Y. *Uncertainty in Deep Learning*. PhD thesis, University of Cambridge, 2016.

Goldberg, P. W., Williams, C. K., and Bishop, C. M. Regression with input-dependent noise: A gaussian process treatment. In *Advances in neural information processing systems*, pp. 493–499, 1998.

Henze, N. and Zirkler, B. A class of invariant consistent tests for multivariate normality. *Communications in Statistics - Theory and Methods*, 19(10):3595–3617, 1990. doi: 10.1080/03610929008830400. URL https://doi.org/10.1080/03610929008830400.

Hernández-Lobato, J. M. and Adams, R. Probabilistic backpropagation for scalable learning of bayesian neural networks. In *International Conference on Machine Learning*, pp. 1861–1869, 2015.

Hernández-Lobato, J. M., Li, Y., Rowland, M., Hernández-Lobato, D., Bui, T., and Turner, R. E. Black-box α-divergence minimization. 2016.

Kendall, A. and Gal, Y. What uncertainties do we need in bayesian deep learning for computer vision? In *Advances in neural information processing systems*, pp. 5574–5584, 2017.

Kersting, K., Plagemann, C., Pfaff, P., and Burgard, W. Most likely heteroscedastic gaussian process regression. In *Proceedings of the 24th international conference on Machine learning*, pp. 393–400. ACM, 2007.

Kingma, D. P. and Welling, M. Auto-Encoding Variational Bayes. December 2013. arXiv: 1312.6114.

Kraskov, A., Stögbauer, H., and Grassberger, P. Estimating mutual information. 69:066138, June 2004. doi: 10.1103/PhysRevE.69.066138.

Lawrence, N. D. and Moore, A. J. Hierarchical gaussian process latent variable models. In *Proceedings of the 24th international conference on Machine learning*, pp. 481–488. ACM, 2007.

Le, Q. V., Smola, A. J., and Canu, S. Heteroscedastic gaussian process regression. In *Proceedings of the 22nd international conference on Machine learning*, pp. 489–496. ACM, 2005.

LeCun, Y., Bengio, Y., and Hinton, G. Deep learning. *nature*, 521(7553):436, 2015.

Liu, Q. and Wang, D. Stein variational gradient descent: A general purpose bayesian inference algorithm. In *Advances In Neural Information Processing Systems*, pp. 2378–2386, 2016.

Louizos, C. and Welling, M. Structured and efficient variational deep learning with matrix gaussian posteriors. In *International Conference on Machine Learning*, pp. 1708–1716, 2016.

Louizos, C. and Welling, M. Multiplicative normalizing flows for variational bayesian neural networks. *arXiv preprint arXiv:1703.01961*, 2017.

MacKay, D. J. A practical bayesian framework for backpropagation networks. *Neural computation*, 4(3):448–472, 1992.

Makhzani, A., Shlens, J., Jaitly, N., Goodfellow, I., and Frey, B. Adversarial Autoencoders. *arXiv e-prints*, art. arXiv:1511.05644, Nov 2015.

McHutchon, A. and Rasmussen, C. E. Gaussian process training with input noise. In *Advances in Neural Information Processing Systems*, pp. 1341–1349, 2011.

Neal, R. M. *Bayesian learning for neural networks*, volume 118. Springer Science & Business Media, 2012.

Sigrist, M., W, S., Winefordner, J., and Kolthoff, I. *Air Monitoring by Spectroscopic Techniques*. Chemical Analysis: A Series of Monographs on Analytical Chemistry and Its Applications. Wiley, 1994. ISBN 9780471558750. URL https://books.google.com/books?id=_vkyfs5_OAoC.

Wang, C. and Neal, R. M. Gaussian process regression with heteroscedastic or non-gaussian residuals. *arXiv preprint arXiv:1212.6246*, 2012.

Williams, P. M. Using neural networks to model conditional multivariate densities. *Neural Computation*, 8(4):843–854, 1996.

Wright, W. Bayesian approach to neural-network modeling with input uncertainty. *IEEE Transactions on Neural Networks*, 10(6):1261–1270, 1999.

Yuan, M. and Wahba, G. Doubly penalized likelihood estimator in heteroscedastic regression. *Statistics & probability letters*, 69(1):11–20, 2004.

7. Theoretical Analysis of the Impact of Model Non-identifiability on Inference

7.1. Theorems for Single-Node BNN+LV Models

Assume that our generative process is the following:

$$\begin{aligned} W &\sim \mathcal{N}(0,1) \\ z_n &\sim \mathcal{N}(0,\sigma_z^2) \\ x_n &\sim \mathcal{N}(0,\sigma_x^2) \\ \epsilon &\sim \mathcal{N}(0,0.001) \\ y_n &= \max\{W(x+z), \alpha W(x+z)\} + \epsilon \end{aligned} \tag{6}$$

where α is a fixed constant in (0, 1). For any non-zero constant C, define $\widehat{W}^{(C)} = W/C$ and $\widehat{z}_n^{(C)} = (C-1)x_n + Cz_n$.

Theorem 1 (**Bias in the Posterior**). *For every pair of ground truth parameter W and choice of prior $\mathcal{N}(\mu_W, \sigma_W^2)$ for W, there exist a non-zero $\mathcal{C}$ such that, for every $c \in (0, \mathcal{C})$, the scaled values $(\widehat{W}^{(c)}, \{\widehat{z}_n^{(c)}\})$ become more likely than $(W, \{z_n\})$ under the expected posterior as the sample size N grows.*

Proof. Given W and $\{z_n\}$, since

$$\max\{W(x+z_n), \alpha W(x+z_n)\} \tag{7}$$

$$= \max\left\{\widehat{W}^{(C)}(x+\widehat{z}_n^{(C)}), \alpha\widehat{W}^{(C)}(x+\widehat{z}_n^{(C)})\right\} \tag{8}$$

we have that the ground truth values $(W, \{z_n\})$ are as likely as $(\widehat{W}^{(C)}, \widehat{z}_n^{(C)})$ under the likelihood, that is,

$$\prod_n p(y_n|x_n, z_n, W) = \prod_n p\left(y_n|x_n, \widehat{z}_n^{(C)}, \widehat{W}^{(C)}\right). \tag{9}$$

Thus, to compare the likelihood of $(W, \{z_n\})$ and $(\widehat{W}^{(C)}, \{\widehat{z}_n^{(C)}\})$ under the log posterior, we need only to compare their log prior values. Now, define K to be the difference between the log prior of W evaluated at the ground truth and $\widehat{W}^{(C)}$ respectively,

$$K \stackrel{\text{def}}{=} \log\mathcal{N}(W;\mu_W,\sigma_W^2) - \log\mathcal{N}\left(\widehat{W}^{(C)};\mu_W,\sigma_W^2\right), \tag{10}$$

and define M to be the difference between the log prior of z evaluated at the ground truth $\{z_n\}$ and $\{\widehat{z}_n^{(C)}\}$ respectively,

$$M \stackrel{\text{def}}{=} \sum_n \left(\log\mathcal{N}(z_n;0,\sigma_z^2) - \log\mathcal{N}\left(\widehat{z}_n^{(C)};0,\sigma_z^2\right)\right). \tag{11}$$

Now, the expected value of M over different samples of training data can be written as

$$\begin{aligned} \mathbb{E}_{z_n,x_n}[M] =& K_z - N * \mathrm{Var}_{z_n}[z_n] && (12)\\ & - (K_z - N * \mathrm{Var}_{z_n,x_n}[\widehat{z}^{(C)}]) && (13)\\ =& N(\mathrm{Var}_{z_n,x_n}[\widehat{z}^{(C)}] - \mathrm{Var}_{z_n}[z_n]) && (14)\\ =& N((C-1)^2\sigma_x^2 + (C^2-1)\sigma_z^2) && (15) \end{aligned}$$

where K_z is the normalizing constant for the prior distribution of z. In the above, Equation 13 follows straightforwardly from Equation 11 by taking the log of the normal pdf's and then applying the expectation; Equation 15 follows from 14 by noting our definition:

$$\widehat{z}_n^{(C)} = (C-1)x_n + Cz_n. \tag{16}$$

We observe that for $0 < C < 1$, we have $\mathrm{Var}_{z_n,x_n}[\widehat{z}^{(C)}] < \mathrm{Var}_{z_n}[z_n]$, hence $\mathbb{E}_{z_n,x_n}[M]$ is less than zero. Thus, as $N \to \infty$, we have that $\mathbb{E}_{z_n,x_n}[M] + K \to -\infty$. In other words, as the training set increases in size, the ground truth values $(W, \{z_n\})$ become less and less likely than $(\widehat{W}^{(C)}, \{\widehat{z}_n^{(C)}\})$ under the expected log posterior, thus completing the proof. □

Example 1 (**A Case wherein Non-identifiability Biases the Posterior Predictive**). For the model in Equation 6, suppose that the ground truth parameter W is equal to 1 and set $\sigma_x^2 = 1$ and $\sigma_z^2 = 0.5$. We show empirically that the posterior mean, $\mathbb{E}_{w^*\sim p(W|\text{Data})}[w^*]$, of W is not equal to 1, and that the posterior predictive mean,

$$\mathbb{E}_{\substack{z^*\sim p(z)\\ \epsilon^*\sim\mathcal{N}(0,\sigma_\epsilon^2)}} \mathbb{E}_{w^*\sim p(W|\text{Data})}[f(x^*,z^*;w^*) + \epsilon^*], \tag{17}$$

of this model is biased away from the ground truth predictive mean,

$$\mathbb{E}_{\substack{z^*\sim p(z)\\ \epsilon^*\sim\mathcal{N}(0,\sigma_\epsilon^2)}} [f(x^*,z^*;w^*) + \epsilon^*]. \tag{18}$$

We compute these expectation numerically using Monte Carlo estimation with 250000 samples and we apply a number of computational techniques for encouraging numerical stability, the results are summarized in Figure 2. We see that the ground truth value is not recovered by the posterior mean of W even as sample size increases. We also see that the posterior predictive mean is biased away from the ground truth even with large numbers of observations.

Theorems 1 and Example 1 indicate that non-identifiability in the functional form of $f(x, z; W)$ can negatively impact inference – the learned model does not generalize well. In the following, we show an example where this is not the case.

Example 2 (**A Case wherein Non-identifiability Does Not Impact Generalization**). Assume that our generative process is the following:

$$\begin{aligned} W &\sim N(0,1) \\ z_n &\sim N(0, 0.5) \\ x_n &\sim N(0,1) \\ \epsilon &\sim N(0, 0.001) \\ y_n &= W \cdot (x_n + z_n)^3 + \epsilon \end{aligned} \tag{19}$$

Suppose that the ground truth parameter W is equal to 1. Following the proof for theorem 1, we can show that the alternate model $\widehat{W} = 8$ and $\widehat{z}_n = -0.5x + 0.5z_n$ is valued as more likely than the ground truth in the expected log posterior distribution $p(W, z_1, \ldots, z_n|\text{Data})$ as the training data increases. That is, the posterior is biased away from the ground truth. On the other hand, the posterior predictive mean,

$$\mathop{\mathbb{E}}_{\substack{z^*\sim p(z)\\ \epsilon^*\sim\mathcal{N}(0,\sigma_\epsilon^2)}} \mathop{\mathbb{E}}_{w^*\sim p(W|\text{Data})} [w^*(x^* + z^*)^3 + \epsilon^*], \tag{20}$$

of this model is unbiased, i.e. it is equal to the ground truth predictive mean

$$\mathop{\mathbb{E}}_{\substack{z^*\sim p(z)\\ \epsilon^*\sim\mathcal{N}(0,\sigma_\epsilon^2)}} [(x^* + z^*)^3 + \epsilon^*]. \tag{21}$$

Note that we can write:

$$\mathop{\mathbb{E}}_{\substack{z^*\sim p(z)\\ \epsilon^*\sim\mathcal{N}(0,\sigma_\epsilon^2)}} \left(\mathop{\mathbb{E}}_{w^*\sim p(W|\text{Data})} [w^*](x^* + z^*)^3 + \epsilon^* \right). \tag{22}$$

Thus, showing the equality of the posterior predictive mean and the true predictive mean is equivalent to showing that $\mathop{\mathbb{E}}_{w^*\sim p(W|\text{Data})}[w^*] = 1$. We compute this expectation numerically using Monte Carlo estimation with 250000 samples, the result is summarized in Figure 3. We see that as the training data set grows in size, $\mathop{\mathbb{E}}_{w^*\sim p(W|\text{Data})}[w^*]$ converges to 1.

7.2. Theorems for 1-Layer BNN+LV Models

Assume that our generative process is the following:

$$\begin{aligned} W^x, W^z, W^{\text{out}} &\sim \mathcal{N}(0, \mathbb{I}) \\ b^{\text{hidden}}, b^{\text{out}} &\sim \mathcal{N}(0, 1) \\ z_n &\sim \mathcal{N}(0, \Sigma_z) \\ x_n &\sim \mathcal{N}(0, \Sigma_x) \\ \epsilon &\sim N(0, 0.001) \\ a^{\text{hidden}} &= g\left(W^x x + W^z z + b^{\text{hidden}}\right), \\ y_n &= (a^{\text{hidden}})^\top W^{\text{out}} + b^{\text{out}} + \epsilon \end{aligned} \tag{23}$$

Let W denote the set

$$\{W^x, W^z, W^{\text{out}}, b^{\text{hidden}}, b^{\text{out}}\}. \tag{24}$$

For a given W, let $\widehat{W}$ denote the set

$$\{\widehat{W}^x, \widehat{W}^z, W^{\text{out}}, \widehat{b}^{\text{hidden}}, b^{\text{out}}\}. \tag{25}$$

where we define

$$\widehat{W}^x = W^x + W^z S, \tag{26}$$

$$\widehat{W}^z = R, \tag{27}$$

$$\widehat{z} = Tz - TSx - U, \tag{28}$$

$$\widehat{b}^{\text{hidden}} = b + RU. \tag{29}$$

Theorem 2 (**Bias in the Posterior**). *For every pair of ground truth parameters W and any choice of prior $\mathcal{N}(\mu_W, \Sigma_W)$ on W, there exist scaled values $\left(\widehat{W}, \{\widehat{z}_n\}\right)$ that become more likely than $(W, \{z_n\})$ under the expected posterior as the sample size N grows.*

Proof. The proof follows in the same fashion as the one for Theorem 1 For any $0 < \epsilon < 1$. Let S be the identity matrix $\mathbb{I}$; let $T = \epsilon\mathbb{I}$, $R = \frac{1}{\epsilon}W^z$ and $U = 0$. Then we have

$$\widehat{W}^x = W^x + W^z, \tag{30}$$

$$\widehat{W}^z = \frac{1}{\epsilon}W^z, \tag{31}$$

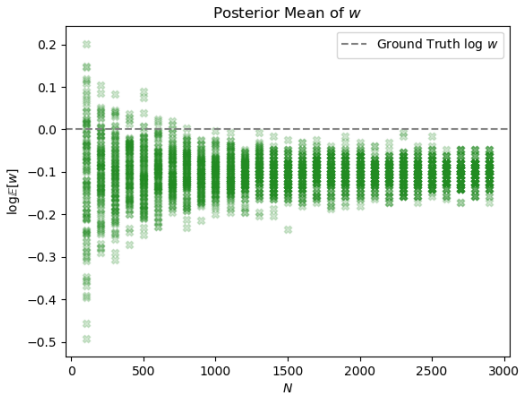
$$\widehat{b}^{\text{hidden}} = b^{\text{hidden}}, \tag{32}$$

$$\widehat{z}_n = \epsilon(z_n - x_n). \tag{33}$$

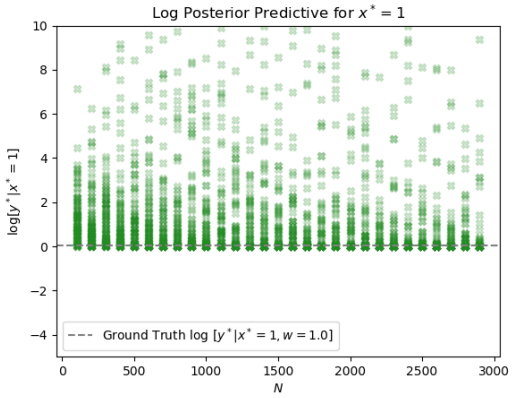
Clearly, the alternate values $\widehat{W}$ for the model parameters and the alternate values for the latent noise $\widehat{z}_n$ reconstruct the observed data as well as the ground truth: $p(y_n|x_n, z_n; W) = p(y_n|x_n, \widehat{z}_n; \widehat{W})$. We now compare the two sets of values under the log priors. In particular, define

$$K \stackrel{\text{def}}{=} (\log\mathcal{N}(W^x; \mu_W, \Sigma_W) + \log\mathcal{N}(W^z; 0, \mathbb{I})) \tag{34}$$

$$- \left(\log\mathcal{N}(\widehat{W}^x; 0, \mathbb{I}) + \log\mathcal{N}(\widehat{W}^z; 0, \mathbb{I})\right) \tag{35}$$



(a) Log posterior mean of W as number of observations increases



(b) Log posterior predictive mean of W at $x^* = 1$ as number of observations increases

Figure 2. Visualization of the posterior mean of W and the log posterior predictive for $x^* = 1$ in the model described by Equation 6. Each point in the scatter plot is the corresponding mean computed for a particular sample of training data of size N. The ground truth value of W is 1 ($\log 0$), which is not recovered by the posterior mean of W even as the number of training points N increases. The true predictive mean for $x^* = 1$ is approximately 0.445, which is not recovered by the posterior predictive mean even a sample size increases.

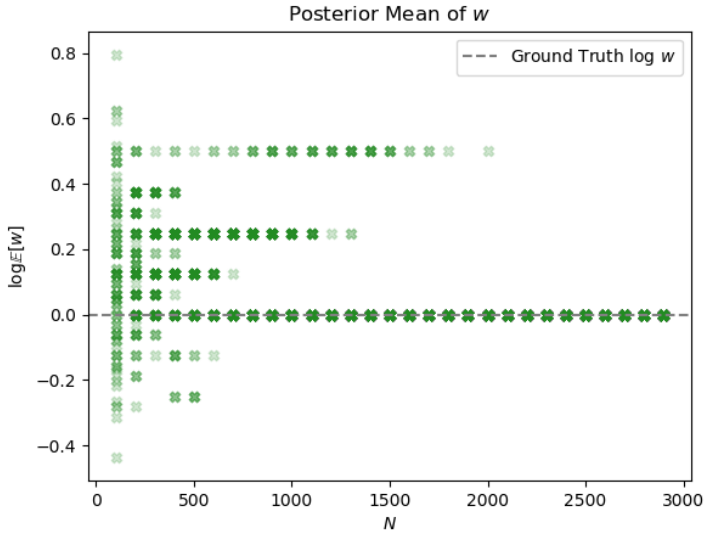


Figure 3. Visualization of the log posterior mean of W in the model described by Equation 19. Each point in the scatter plot is the posterior mean computed for a particular sample of training data of size N. The ground truth value of W is 1 ($\log 0$), which is recovered by the posterior mean of W as the number of training points N increases.

and

$$M \stackrel{\text{def}}{=} \sum_{n=1}^{N} \left(\log \mathcal{N}(z_n; 0, \Sigma_z) - \log \mathcal{N}(\widehat{z}_n; 0, \Sigma_z)\right) \quad (36)$$

Then we have that

$$\mathbb{E}_{z_n, x_n}[M] = \sum_d \left((K_{z^d} - N * \mathrm{Var}_{z_n}[z_n^d]) \right. \quad (37)$$

$$\left. -(K_{z^d} - N * \mathrm{Var}_{z_n, x_n}[\widehat{z}_n^d]) \right), \quad (38)$$

$$= N \sum_d \left(\mathrm{Var}_{z_n, x_n}[\widehat{z}_n^d]) - \mathrm{Var}_{z_n}[z_n^d] \right), \quad (39)$$

where K_{z^d} is the normalizing constant for the distribution of

z^d. Choose ϵ such that $\mathrm{Var}_{z_n,x_n}[\widehat{z}_n^d] < \mathrm{Var}_{z_n}[z_n^d]$, for each dimension $0 < d < D$. Then, as N becomes sufficiently large, $\mathbb{E}_{z_n,x_n}[M] + K$ becomes negative and large. In other words, the alternate values $(\widehat{W}, \{\widehat{z_n}\})$ is more likely under the expected log posterior than the ground truth values (W, $\{z_n\}$). □

7.3. Additional types of non-identifiability for 1-Layer BNN+LV Models

Assume that the activation function g is invertible. Let $\{(x_n, y_n)\}_{n=1}^N$ be a set of observed data generated by the model parameters W. Define $\widehat{b}^{\text{hidden}}, b^{\text{out}}$ to be zero, $\widehat{W}^x$ to be the $H \times D$ zero matrix and $\widehat{W}^{\text{out}}$ to be the $1 \times H$ matrix of consisting of $\frac{1}{DH}$ in all entries. Finally, let $\widehat{W}^z$ be a $H \times D$ matrix of 1's and let $\widehat{z}_n = g^{-1}(y_n)$.

Then, we have that $y_n = f(x_n, \widehat{z}_n; \widehat{W})$. That is, the alternate set of model parameters $\widehat{W}^x$ reconstructs the observed data perfectly. We note that in this case, the latent noise variable z is a function of the observed output y and is hence dependent on the input x.

8. Mutual Information Computation

For our model, the mutual information between x and z is intractable to compute as is:

$$I(x;z) = D_{\mathrm{KL}}[q(z|x)p(x) \,\|\, q(z)p(x)] \tag{40}$$

$$= \mathbb{E}_{q(z|x)p(x)}\left[\log \frac{q(z|x)p(x)}{q(z)p(x)}\right] \tag{41}$$

$$= \frac{1}{N}\sum_{n=1}^{N} \mathbb{E}_{q(z_n|x_n)}\left[\log \frac{q(z_n|x_n)}{\frac{1}{N}\sum_{n=1}^{N} q(z_n|x_n, y_n)}\right] \tag{42}$$

where

$$q(z_n|x_n) = \mathbb{E}_{q(y_n|x_n)}\left[q(z_n|x_n, y_n)\right] \tag{43}$$

$$q(y_n|x_n) = \mathbb{E}_{q(Z,W|\mathcal{D})}\left[p(y_n|x_n, z_n, W)\right] \tag{44}$$

The nested expectations in the above formulation require too many samples in order to evaluate $I(x;z)$ with low variance. For this reason we choose to compute the above correlation based proxies.

9. Optimization Techniques

We perform a number of optimization 'tricks' to encourage convergence to a desirable local optimum.

Choosing Hyper-parameters We choose hyper-parameters of the priors as well as the likelihood using empirical Bayes (MAP Type II). That is, we place Inverse Gamma priors ($\alpha = 3.0, \beta = 0.5$) on the variances of the network weights and the latent variables; then we approximate the negative ELBO with the MAP estimates of the variances, making the assumption that these term dominate the respective integrals in which they appear:

$$\begin{aligned} -\text{ELBO}(\phi) \approx &- \mathbb{E}_{q(Z,W|\phi)}[p(Y|X,W,Z)] \\ &+ D_{\mathrm{KL}}[q(W|\phi) \,\|\, p(W|s_w^*)p(s_w^*)] \\ &+ D_{\mathrm{KL}}[q(Z|\phi) \,\|\, p(Z|s_z^*)p(s_z^*)] \end{aligned}$$

where we define:

$$\begin{aligned} s_z^* &= \operatorname*{argmin}_{s_z} D_{\mathrm{KL}}[q(Z|\phi) \,\|\, p(Z|s_z)p(s_z)] \\ &= \frac{2\beta + \frac{1}{N}\sum_n \left[\mathrm{tr}\left(\Sigma_{q_n}\right) + \mu_{q_n}^T \mu_{q_n}\right]}{K + 2\alpha - 2}, \\ s_w^* &= \operatorname*{argmin}_{s_w} D_{\mathrm{KL}}[q(W|\phi) \,\|\, p(W|s_w)p(s_w)] \\ &= \frac{2\beta + \mathrm{tr}\left(\Sigma_q\right) + \mu_q^T \mu_q}{H + 2\alpha - 2}. \end{aligned}$$

with K and H as the dimensionality of z_n and W, respectively. The optimal variances, s_z^*, s_w^*, have analytic solutions. In training, we update the objective as well as the optimal hyper-parameters via coordinate descent. That is, we iteratively compute s_z^*, s_w^* in closed-form given the current ϕ, and then optimize ϕ while holding s_z^*, s_w^* fixed.

10. Choosing Differentiable Forms of the NCAI Objective

Tractable training with NCAI depends on instantiating a differentiable form of the training Equation 4. In the following, we choose computationally efficient proxies for the two constraints in the NCAI objective.

Defining $\mathbf{Div(q(z)||p(z))}$. As a proxy for $\mathrm{Div}(q(z), p(z))$, we penalize the Henze-Zirkler non-parametric test-statistic for normality (Henze & Zirkler, 1990) applied to the set of latent noise means, $\{\mu_{z_n}\}$. This encourages the aggregated posterior $q(z)$ to be Gaussian, and hence the learned z_n's will appear as if sampled from this Gaussian. In addition, we penalize the ℓ_2 penalty of the off diagonal terms of the empirical covariance of the latent noise means:

$$\lambda_1 \mathrm{HZ}(\{\mu_{z_n}\}_{n=1}^N) + \lambda_2 \left\|\text{offdiag}\ \widehat{\Sigma}\left(\{\mu_{z_n}\}_{n=1}^N\right)\right\|_2 \tag{45}$$

This ensures that the learned z_n's are independent of each other. We find that, in practice, unlike more traditional divergences (e.g. reverse/forward-KL, Jensen-Shannon, MMD (**?**)), our proxy cannot be trivially minimized by inflating the variational variances, $\sigma_{z_n}^2$.

Defining $\mathbf{Dep(x;z)}$. Ensuring that z_n's are independent of each other is not sufficient to satisfy the properties implied in the generative model. From our analysis in Section **??**,

we see that the latent variable can compensate for incorrectly learned network weights by absorbing a copy of the input, x, or by becoming dependent on x through encoding for y. We therefore penalize the dependence between x and z by penalizing correlation between x and z and the correlation between y and z:

$$\begin{aligned}&\lambda_3 \,\text{PairwiseCorr}(\{x_n\}, \{\mu_{z_n}\}) \\ &+ \lambda_4 \,\text{PairwiseCorr}(\{y_n\}, \{\mu_{z_n}\})\end{aligned} \tag{46}$$

where $\text{PairwiseCorr}(\cdot, \{\mu_{z_n}\})$ is a measure of the average correlation between pairs of dimensions in x or y and the latent noise means $\{\mu_{z_n}\}$. We find that, in practice, unlike mutual information lower bounds and estimators (e.g. MINE (**?**)) and upper bounds, our proxy cannot be trivially minimized by inflating the variational variances, $\sigma^2_{z_n}$. See Appendix 8 for details about the difficulty in directly minimizing mutual information for this model.

Relationship between $\mathbf{Div(q(z)||p(z))}$ and $\mathbf{Dep(x;z)}$. The two constraints are theoretically orthogonal to one another: a small $\text{Dep}(x, z)$ does not imply a small $\text{Div}(q(z), p(z))$, and vice versa. For example, one can adversarially construct z's and x's such that $\text{Div}(q(z), p(z))$ is small and $\text{Dep}(x, z)$ is high by initializing the variational parameters ϕ such that $q(z) = p(z)$, and then, for the given x's pairing small x's with small z's. As such, both constraints are theoretically necessary.

Defining the NCAI Objective. Finally, we incorporate the ELBO and the differentiable forms of the constraints (as exponentially smoothed penalties) into Equation 4:

$$\begin{aligned}\mathcal{L}_{\text{NCAI}}(\phi) = &- \text{ELBO}(\phi) \\ &+ \lambda_1 N \exp\left(\frac{\text{HZ}(\{\mu_{z_1}, \ldots, \mu_{z_N}\}}{\epsilon_T}\right) \\ &+ \lambda_2 N \|\text{offdiag}\ \widehat{\Sigma}(\{\mu_{z_1}, \ldots, \mu_{z_N}\})\|_2 \\ &+ \lambda_3 N \exp\left(\frac{\text{PairwiseCorr}(\{x_n\}, \{\mu_{z_n}\})}{\epsilon_x}\right. \\ &\left. + \frac{\text{PairwiseCorr}(\{y_n\}, \{\mu_{z_n}\})}{\epsilon_y}\right)\end{aligned} \tag{47}$$

where $\epsilon_T, \epsilon_x, \epsilon_y$ control the growth rate of the exponential penalties. We minimize the negative ELBO following Bayes by Backprop (BBB) (Blundell et al., 2015): back-propagating through $\mathbb{E}_{q(Z,W|\phi)}[\cdot]$ in the ELBO using the reparametarization trick (Kingma & Welling, 2013), computing the KL-divergence terms and constraints using closed-form expressions.

11. Datasets

Synthetic Data: We consider 4 synthetic datasets, most of which have been widely used to evaluate heteroscedastic regression models (Wright, 1999; Kersting et al., 2007; Wang & Neal, 2012; Goldberg et al., 1998):

1. **Goldberg** (Goldberg et al., 1998): targets are given by $y = 2\sin(2\pi x) + \epsilon(x)$, where $\epsilon(x) \sim \mathcal{N}(0, x + 0.5)$. Evaluated on 200 training input, 200 validation and 200 test inputs uniformly sampled from $[0, 1]$.
2. **Yuan** (Yuan & Wahba, 2004): targets are given by $y = 2[\exp\{-30(x - 0.25)^2 + \sin(\pi x^2)\}] - 2 + \epsilon(x)$, where $\epsilon(x) \sim \mathcal{N}(0, \exp\{\sin(2\pi x)\})$. Evaluated on 200 training input, 200 validation and 200 test inputs uniformly sampled from $[0, 1]$.
3. **Williams** (Williams, 1996): the targets are given by $y = \sin(2.5x) \cdot \sin(1.5x) + \epsilon(x)$, where $\epsilon(x) \sim \mathcal{N}(0, 0.01 + 0.25(1 - \sin(2.5x))^2)$. Evaluated on 200 training input, 200 validation and 200 test inputs uniformly sampled from $[0, 1]$.

Synthetic Data Generated with Ground Truth: We also generate two synthetic data-sets with corresponding ground truth in order to guarantee that our generative process matches our data. We generate these data-sets by training a neural network to map the x_n's and ground truth z_n's to y_n's, specified by some function. We then re-generate the y_n's from the learned neural network and treat that network as the ground truth function.

The two data-sets we have generated in this way are the following:

1. **Heavy-Tail**: targets are given by a neural network approximation of $y = 6\tanh(0.1x^3(z + 1)^6 - 10xz^2 + z) + \epsilon$, where $\epsilon \sim \mathcal{N}(0, 0.1)$ and $z \sim \mathcal{N}(0, 0.01)$. Evaluated on 300 training input, 300 validation and 300 test inputs uniformly sampled from $[-4, 4]$.
2. **Depeweg** (Depeweg et al., 2018): targets are given by a neural network approximation of $y = 7\sin(x) + 3|\cos(x/2)|z + \epsilon$, where $\epsilon \sim \mathcal{N}(0, 0.1)$ and $z \sim \mathcal{N}(0, 1.0)$. Evaluated on 750 training input, 250 validation and 250 test inputs uniform mixture of the following gaussians: $\mathcal{N}(0, -4.0, 0.16), \mathcal{N}(0, 0, 0.81), \mathcal{N}(0, 4.0, 0.16)$. (Note the original data-set from (Depeweg et al., 2018) sampled $y = 7\sin(x) + 3|\cos(x/2)|\epsilon$, where $\epsilon \sim \mathcal{N}(0, 1.0)$).

Real Data: We use 6 UCI datasets (Dua & Graff, 2017) and a dataset commonly used in the heteroscedastic literature, Lidar (Sigrist et al., 1994) (see Table 4 for details).

12. Experimental Details and Evaluation

Architecture The network architectures we use for all experiments are summarized in Tables 4 and 5. We note

that we have purposefully selected lower capacity architectures to encourage for the non-identifiability described in the paper to occur in practice. We also note that the non-identifiability occurs even when the ground-truth network capacity is known, as in the case of the HeavyTail and Depeweg data-sets, which have been generated using a neural network. As such, even when the data was generated by the same generative process as the one assumed by the model, the problem of non-identifiability still occurs.

Train/Validation/Test Data-splits We each data-set into train/validation/test set 6 times. We use the first data-split to select hyper-parameters by selecting the hyper-parameters that yield the best average log-likelihood performance on the validation set across 10 random restarts. After having selected the hyper-parameter for each method, we select between NCAI$_{\lambda=0}$ and NCAI$_{\lambda}$ by picking the approach that yielded the best log-likelihood performance on the validation set across the 10 random restarts. Now, using the selected hyper-parameters and form of NCAI, we train our models on the remaining 5 data-splits, averaging the best-of-10 random restarts across the data-splits (using the validation log-likelihood).

For Abalone, Airfoil, Boston Housing, Energy Efficiency, Lidar, Wine Quality Red, Yacht, Goldberg, Williams, Yuan, we splits the data into a $\%70$ training set, $\%20$ validation set and $\%10$.

Hyperparameter Selection: For the data-sets Abalone, Airfoil, Boston Housing, Energy Efficiency, Lidar, Wine Quality Red, Yacht, Goldberg, Williams, Yuan, we used grid-searched over the following parameters:

- BNN: $\sigma_\epsilon^2 = \{1.0, 0.1, 0.01\}$
- BNN+LV: $\sigma_\epsilon^2 = \{0.1, 0.01\}$
- NCAI:
  $\sigma_\epsilon^2 = \{0.1, 0.01\}$,
  $\lambda_2 = \{10.0\}$,
  $\epsilon_T = \{0.01, 0.0003\}$,
  $\epsilon_y = \{0.1, 0.5\}$,
  $\epsilon_x = \{0.5, 1.0\}$

For HeavyTails we grid-searched over the following parameters:

- BNN: $\sigma_\epsilon^2 = \{1.0, 0.1, 0.5\}$
- BNN+LV: $\sigma_\epsilon^2 = \{0.1\}, \sigma_z^2 = \{0.01\}$
- NCAI:
  $\sigma_\epsilon^2 = \{0.1\}$,
  $\lambda_2 = \{10.0\}$,
  $\epsilon_T = \{0.01, 0.0003\}$,
  $\epsilon_y = \{0.1, 0.5\}$,
  $\epsilon_x = \{0.5, 1.0\}$

For Depeweg we grid-searched over the following parameters:

- BNN: $\sigma_\epsilon^2 = \{1.0, 0.1, 0.5\}$
- BNN+LV: $\sigma_\epsilon^2 = \{0.1\}, \sigma_z^2 = \{1.0\}$
- NCAI:
  $\sigma_\epsilon^2 = \{0.1\}$,
  $\lambda_2 = \{10.0\}$,
  $\epsilon_T = \{0.01, 0.0003\}$,
  $\epsilon_y = \{0.1, 0.5\}$,
  $\epsilon_x = \{0.5, 1.0\}$

12.1. Evaluation Metrics

Quality of Fit We measure the *training reconstruction MSE*, the ability of the model to reconstruct the training targets with the learned weights and latent variables:

$$\frac{1}{N}\sum_{n=1}^{N} \mathbb{E}_{q(z_n)q(W)}\left[\|y_n - f(z_n, W)\|_2^2\right]. \tag{48}$$

At test time, we measure the quality of the posterior predictive distribution of the model by computing the *average marginal log-likelihood*

$$\frac{1}{N}\sum_{n=1}^{N} \mathbb{E}_{p(z)q(W)}\left[\log p(y_n|x_n, W, z_n)\right]. \tag{49}$$

We also compute the predictive quality of the model by computing the *predictive MSE*:

$$\frac{1}{N}\sum_{n=1}^{N} \left[\|y_n - \mathbb{E}_{p(z_n)q(W)} f(z_n, W)\|_2^2\right]. \tag{50}$$

Note that the difference between the reconstruction MSE (48) and the predictive MSE (50) is that in the latter we sample the latent variables from the prior distributions rather than the learned posterior distributions.

Posterior Predictive Calibration We measure the quality of the model's predictive uncertainty by computing the percentage of observations for which the ground truth y lies within a 95% predictive-interval (PI) of the learned model – this quantity is called the Prediction Interval Coverage Probability (PICP). We measure the tightness of the model's predictive uncertainty by computing the 95% Mean Prediction Interval Width (MPIW).

Data-set	Size	Dimensionality	Hidden Nodes
Lidar	221	1	10
Yacht	309	6	5
Energy Efficiency	768	8	10
Airfoil	Subsampled to 1000	5	30
Abalone	Subsampled to 1000	10 (1-hot for categorical)	10
Wine Quality Red	1600	1	20

Table 4. Experimental Details for the Real Data-sets

Data-set	Number of Hidden Nodes	Number of Layers
Williams	20	2
Yuan	20	1
Goldberg	20	1
HeavyTail	50	1
Depeweg	50	1

Table 5. Experimental Details for the Synthetic Data-sets

Satisfaction of Model Assumptions We estimate the mutual information between x and z by computing the Kraskov nearest-neighbor based estimator (Kraskov et al., 2004) (with 5 nearest neighbors) on the x's and the means of the z's: $\hat{I}(x; \mu_z)$. We use μ_{z_n}'s instead of $z \sim q(z)$, since if the $\sigma^2_{z_n}$'s are large the dependence between z's and x's is more difficult to detect.

For the univariate case, when $D = K = 1$, we use the Kolmogorov-Smirnov (KS) two-sample test statistic (kst, 2008) to evaluate divergence between $q(z)$ and $p(z)$. When computing the test statistic, we represent $q(z)$ using μ_{z_n}'s and $p(z)$ using its samples. This is because the $\sigma^2_{z_n}$'s are large, the distance between $q(z)$ and $p(z)$ more difficult to detect. A lower KS test-statistic indicates that $q(z)$ and $p(z)$ are more similar. We compute the Jensen-Shannon divergence between $q(z)$ and $p(z)$ in multivariate cases.

13. Experimental Results

Qualitative Evaluation For the univariate datasets (all synthetic data sets as well as Lidar), we provide visualizations of the posterior predictive distributions of NCAI and benchmarks against the ground truth, as well as the joint distribution of the input and learned latent noise (see Figures 4, 5, 6, 7, 8). We find that in all cases, NCAI training produces qualitatively superior posterior predictives and learned latent noise that is less dependent on the input: NCAI captures the trend of the data while estimating a tight uncertainty around the data. This is in contrast to the BNN, which does not capture heteroscedasticity, and to the BNN+LV which often has difficulty capturing the trend in the data well and tends to over-estimate the uncertainty. We also find that NCAI qualitatively satisfies our modeling assumptions: one can visualize discern that the z's learned by NCAI are less dependent on the x's than the z's learned by BNN+LV. Lastly, we note that even though NCAI learns z's that are less dependent on the x's, it is still uncapable to remove *all* dependence. This is because minimizing the information shared between the x's and the z's is intractable (see Appendix 8). Even by reducing some of the dependence, however, NCAI is able to model the data significantly better.

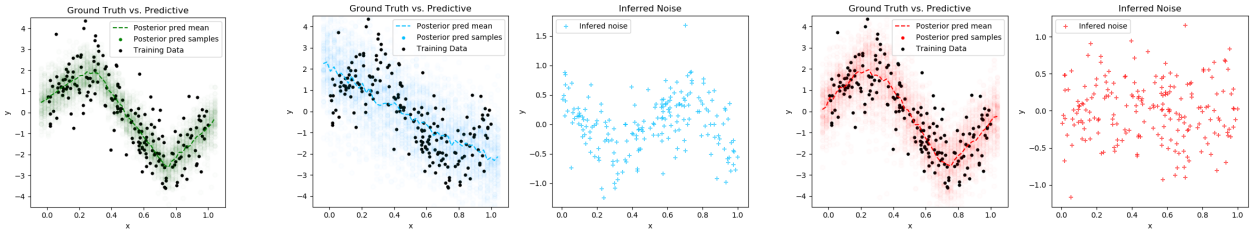
Quantitative Evaluation For all datasets, we compare NCAI training with benchmarks evaluated under all our metrics (generalization, calibration and modeling assumption satisfaction – see Section 12.1). We find that BNN+LV with NCAI training consistently outperform BNN+LV with Mean Field BBB (in terms of average test log likelihood, RMSE) and recovers latent noise variables that better satisfy modeling assumptions (mutual information, divergence metrics, normality test statistics – see Table **??**). We note that the BNN, when properly trained, is able to capture the trends in the data (measured by RMSE) but tends to underestimate the variance (log likelihood and calibration) – this tendency is especially apparent in the presence of heteroscedastic noise. This is especially apparent on the Energy Efficiency dataset, in which the BNN achieves the highest log-likelihood on average, while significantly underestimating the uncertainty; the $\%95$-PICP and MPIW show that BNN has a small predictive interval width that only covers about $\%80$ of the data, whereas NCAI overs about $\%94$ of the data (see Table 14 for more details).

Selecting between NCAI$_{\lambda=0}$ and NCAI$_{\lambda>0}$ Generally, we observe that on data-sets in which the noise is roughly symmetric around the posterior predictive mean (as in the Goldberg, Yuan, Williams, Lidar, and Depeweg data-sets) NCAI$_{\lambda=0}$ and NCAI$_{\lambda>0}$ perform comparably well on average test log-likelihood – see Tables 16, 18, 17, 18 and 22.

However, when the noise is skewed around the posterior predictive mean (like in the HeavyTail dataset), we find that NCAI$_{\lambda>0}$ out-performs NCAI$_{\lambda=0}$ – see Table 15. This is because NCAI$_{\lambda=0}$ first fits the variational parameters of the weights to best capture as best as possible, often fitting a function that represents the mean. After the warm-start, when training with respect to the variational parameters of the z's, the uncertainty is increased about the mean to best capture the data, often in a way that does not significantly alter the parameters of the weights, thereby resulting in a posterior predictive with symmetric noise.

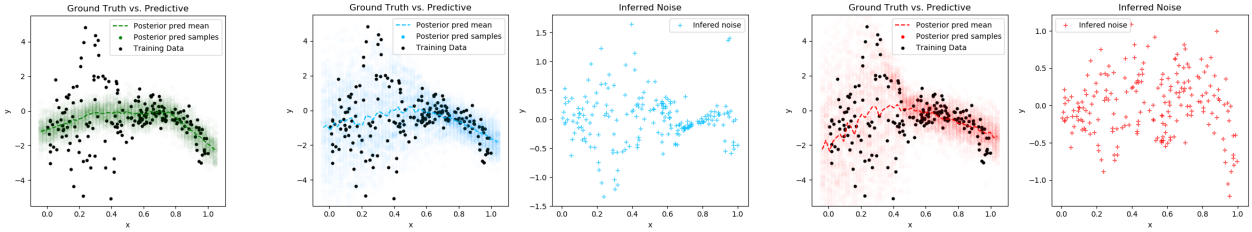
Visualization of experimental results for all univariate data sets are in Section 14, table summaries of quantitative experimental results are in Section

14. Experimental Results: Visualilzations



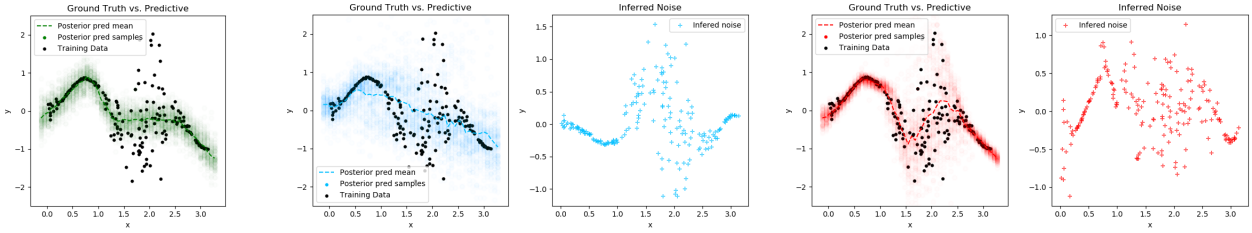
(a) BNN with Mean Field BBB (b) BNN+LV with Mean Field BBB (c) BNN+LV with NCAI

Figure 4. Comparison of the posterior predictives for Goldberg.



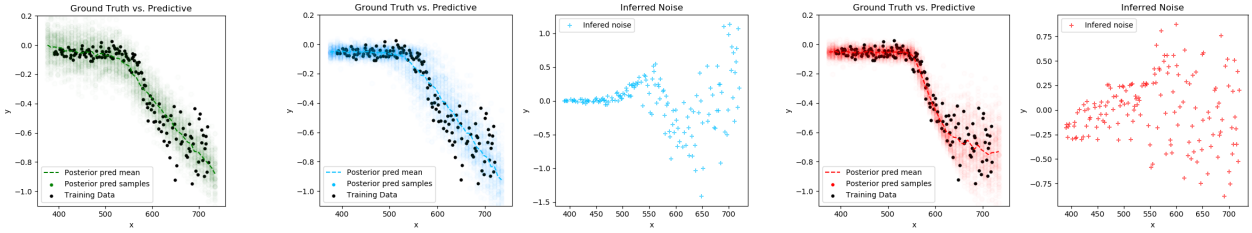
(a) BNN with Mean Field BBB (b) BNN+LV with Mean Field BBB (c) BNN+LV with NCAI

Figure 5. Comparison of the posterior predictives for Yuan.



(a) BNN with Mean Field BBB (b) BNN+LV with Mean Field BBB (c) BNN+LV with NCAI

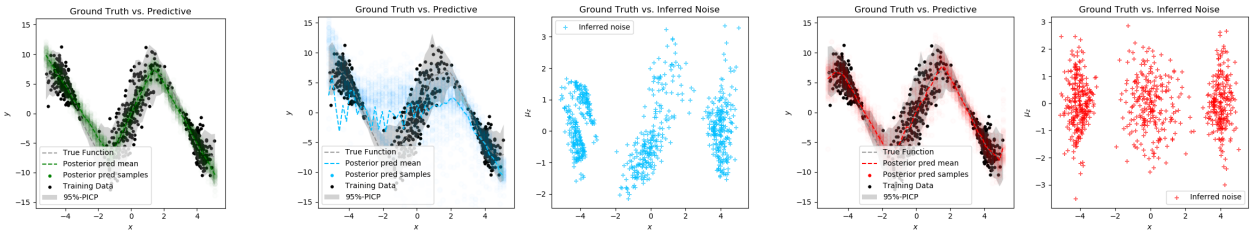
Figure 6. Comparison of the posterior predictives for Williams.



(a) BNN with Mean Field BBB (b) BNN+LV with Mean Field BBB (c) BNN+LV with NCAI

Figure 7. Comparison of the posterior predictives for Lidar.

15. Experimental Results: Tables



(a) BNN with Mean Field BBB (b) BNN+LV with Mean Field BBB (c) BNN+LV with NCAI

Figure 8. Comparison of the posterior predictives for Depeweg.

Log-Likelihood on Test Data for Synthetic Data sets

	Heavy Tail	Goldberg	Williams	Yuan	Depeweg
BNN	-2.47 ± 0.083	-1.055 ± 0.08	-1.591 ± 0.417	-2.846 ± 0.346	-2.306 ± 0.059
BNN+LV	-1.867 ± 0.078	-1.026 ± 0.056	-1.033 ± 0.156	-1.278 ± 0.164	-2.342 ± 0.048
NCAI$_{\lambda=0}$	-1.481 ± 0.018	-0.962 ± 0.040	-0.414 ± 0.184	-1.211 ± 0.083	-1.973 ± 0.049
NCAI$_{\lambda}$	-1.426 ± 0.042	-0.963 ± 0.041	-0.414 ± 0.184	-1.211 ± 0.083	-1.973 ± 0.049

Table 6. Comparison of model generalization in terms of test log-likelihood on synthetic datasets ($\pm$ std). Across all datasets BNN+LV with NCAI$_\lambda$ training yields comparable if not better generalization. NCAI training always outperforms BNN+LV with Mean Field BBB as well as BNN (the latter comparison is in terms of test log-likelihood).

RMSE on Test Data for Synthetic Data sets

	Heavy Tail	Goldberg	Williams	Yuan	Depeweg
BNN	1.831 ± 0.074	0.335 ± 0.025	1.017 ± 0.06	0.607 ± 0.035	1.953 ± 0.071
BNN+LV	1.882 ± 0.088	0.376 ± 0.032	1.118 ± 0.096	0.622 ± 0.039	3.523 ± 0.501
NCAI$_{\lambda=0}$	1.787 ± 0.094	0.339 ± 0.026	0.978 ± 0.083	0.619 ± 0.039	1.932 ± 0.059
NCAI$_{\lambda}$	1.79 ± 0.09	0.337 ± 0.025	0.978 ± 0.083	0.619 ± 0.039	1.932 ± 0.059

Table 7. Comparison of RMSE on synthetic datasets ($\pm$ std). Across all datasets BNN+LV with NCAI$_\lambda$ training yields comparable if not better generalization. NCAI training always outperforms BNN+LV with Mean Field BBB as well as BNN.

Mutual Information on Test Data for Synthetic Data sets

	Heavy Tail	Goldberg	Williams	Yuan	Depeweg
BNN+LV	0.243 ± 0.079	0.229 ± 0.113	0.982 ± 0.121	$\mathbf{0.24 \pm 0.129}$	0.428 ± 0.04
NCAI$_{\lambda=0}$	0.051 ± 0.049	$\mathbf{0.02 \pm 0.024}$	$\mathbf{0.519 \pm 0.091}$	0.283 ± 0.112	$\mathbf{0.032 \pm 0.017}$
NCAI$_{\lambda}$	$\mathbf{0.036 \pm 0.04}$	0.046 ± 0.067	$\mathbf{0.519 \pm 0.091}$	0.283 ± 0.112	$\mathbf{0.032 \pm 0.017}$

Table 8. Comparison of model assumption satisfaction (in terms of mutual information between z and x) on synthetic datasets ($\pm$ std). Across all datasets, NCAI$_\lambda$ training learns z's has the least mutual information with x's and looks the most Gaussian (lowest HZ).

HZ Metric on Test Data for Synthetic Data sets

	Heavy Tail	Goldberg	Williams	Yuan	Depeweg
BNN+LV	4.701 ± 5.439	0.918 ± 0.41	$\mathbf{6.445 \pm 2.818}$	$\mathbf{5.252 \pm 5.607}$	6.408 ± 2.439
NCAI$_{\lambda=0}$	7.137 ± 5.436	0.621 ± 0.234	7.248 ± 2.598	8.091 ± 5.185	$\mathbf{0.792 \pm 0.357}$
NCAI$_{\lambda}$	$\mathbf{0.027 \pm 0.011}$	$\mathbf{0.026 \pm 0.038}$	7.248 ± 2.598	8.091 ± 5.185	$\mathbf{0.792 \pm 0.357}$

Table 9. Comparison of model assumption satisfaction (in terms of the HZ metric) on synthetic datasets ($\pm$ std). Across all datasets, NCAI$_\lambda$ training learns z's has the least mutual information with x's and looks the most Gaussian (lowest HZ).

Mutual Information on Test Data for Real Data sets

	Abalone	Airfoil	Energy	Wine	Lidar	Yacht
BNN+LV	0.152 ± 0.015	0.485 ± 0.054	0.139 ± 0.086	0.045 ± 0.012	0.667 ± 0.061	0.077 ± 0.012
NCAI$_{\lambda=0}$	0.149 ± 0.078	0.29 ± 0.021	0.162 ± 0.063	0.047 ± 0.011	0.373 ± 0.037	0.087 ± 0.012
NCAI$_{\lambda}$	0.149 ± 0.078	0.29 ± 0.021	0.226 ± 0.041	0.029 ± 0.008	0.842 ± 0.06	0.087 ± 0.012

Table 10. Comparison of model assumption satisfaction on real datasets ($\pm$ std). Across most datasets, NCAI training learns z's has the least mutual information with x's and looks the most Gaussian (lowest HZ).

HZ Metric on Test Data for Real Data sets

	Abalone	Airfoil	Energy	Wine	Lidar	Yacht
BNN+LV	26.148 ± 4.394	28.108 ± 3.205	16.976 ± 3.519	52.566 ± 2.633	5.09 ± 0.991	27.059 ± 4.144
NCAI$_{\lambda=0}$	17.975 ± 6.725	49.122 ± 11.426	19.071 ± 5.414	53.201 ± 1.247	7.804 ± 1.727	51.283 ± 9.548
NCAI$_{\lambda}$	17.975 ± 6.725	49.122 ± 11.426	1.186 ± 0.558	1.641 ± 0.242	0.005 ± 0.001	51.283 ± 9.548

Table 11. Comparison of model assumption satisfaction, in terms of the HZ metric on real datasets ($\pm$ std). Across most datasets, NCAI training learns z's that generally looks the most Gaussian (lowest HZ).

	NCAI$_{\lambda}$	NCAI$_{\lambda=0}$	BNN	BNN+LV
$D_{\text{JS}}(q(z)\|\|p(z))$	0.007 ± 0.004	0.021 ± 0.019	N/A	0.009 ± 0.003
$D_{\text{KL}}(p(z)\|\|q(z))$	0.015 ± 0.012	0.05 ± 0.046	N/A	0.015 ± 0.006
$\hat{I}(x;\mu_z)$	0.146 ± 0.131	0.149 ± 0.078	N/A	0.152 ± 0.015
$\hat{I}(x;z)$	0.015 ± 0.006	0.012 ± 0.005	N/A	0.011 ± 0.002
HZ$(\{\mu_{z_1},\ldots,\mu_{z_N}\})$	0.839 ± 0.15	17.975 ± 6.725	N/A	26.148 ± 4.394
s_w^*	0.2 ± 0.018	1.272 ± 1.043	0.948 ± 0.68	0.202 ± 0.02
s_y^*	0.01 ± 0.0	0.01 ± 0.0	0.1 ± 0.0	0.1 ± 0.0
s_z^*	0.252 ± 0.004	0.248 ± 0.001	N/A	0.249 ± 0.0
95%-MPIW Test (Unnorm)	7.564 ± 0.459	7.027 ± 0.357	4.112 ± 0.103	7.127 ± 0.311
95%-MPIW Train (Unnorm)	7.716 ± 0.448	7.096 ± 0.446	4.111 ± 0.099	7.248 ± 0.215
95%-PICP Test	94.3 ± 2.515	92.3 ± 0.975	76.9 ± 4.292	94.4 ± 2.329
95%-PICP Train	94.771 ± 0.559	93.343 ± 1.743	77.771 ± 1.476	94.4 ± 0.509
PairwiseCorr(x,μ_z)	0.001 ± 0.0	0.006 ± 0.002	N/A	0.005 ± 0.002
PairwiseCorr(y,μ_z)	0.032 ± 0.003	0.063 ± 0.023	N/A	0.115 ± 0.016
Post-Pred Avg-LL Test	-0.837 ± 0.105	-0.831 ± 0.086	-1.248 ± 0.153	-0.843 ± 0.071
Post-Pred Avg-LL Train	-0.798 ± 0.051	-0.799 ± 0.064	-1.086 ± 0.099	-0.832 ± 0.022
RMSE Test (Unnorm)	0.208 ± 0.012	0.205 ± 0.013	0.194 ± 0.011	0.204 ± 0.012
RMSE Train (Unnorm)	0.208 ± 0.012	0.205 ± 0.013	0.194 ± 0.011	0.204 ± 0.011
Recon MSE	0.011 ± 0.0	0.012 ± 0.0	N/A	0.165 ± 0.002
Hyperparams	$\sigma_\epsilon^2 = 0.01$, $\lambda_2 = 10$, $\epsilon_T = 0.01$, $\epsilon_x = 0.54$, $\epsilon_y = 1.0$	$\sigma_\epsilon^2 = 0.01$	$\sigma_\epsilon^2 = 0.1$	$\sigma_\epsilon^2 = 0.1$

Table 12. Experiment Evaluation Summary for Abalone ($\pm$ std).

	NCAI$_\lambda$	NCAI$_{\lambda=0}$	BNN	BNN+LV
$D_{\text{JS}}(q(z)\|\|p(z))$	-0.0 ± 0.003	-0.001 ± 0.003	N/A	0.001 ± 0.004
$D_{\text{KL}}(p(z)\|\|q(z))$	0.002 ± 0.002	-0.0 ± 0.002	N/A	0.005 ± 0.003
$\hat{I}(x;\mu_z)$	0.262 ± 0.03	0.29 ± 0.021	N/A	0.485 ± 0.054
$\hat{I}(x;z)$	0.107 ± 0.005	0.105 ± 0.008	N/A	0.174 ± 0.025
HZ($\{\mu_{z_1}, \ldots, \mu_{z_N}\}$)	0.25 ± 0.11	49.122 ± 11.426	N/A	28.108 ± 3.205
s_w^*	0.548 ± 0.084	0.509 ± 0.076	0.762 ± 0.182	0.105 ± 0.032
s_y^*	0.1 ± 0.0	0.1 ± 0.0	0.1 ± 0.0	0.1 ± 0.0
s_z^*	0.25 ± 0.001	0.247 ± 0.0	N/A	0.25 ± 0.0
95%-MPIW Test (Unnorm)	10.426 ± 0.81	10.283 ± 0.423	8.999 ± 0.164	17.023 ± 1.972
95%-MPIW Train (Unnorm)	10.294 ± 0.732	10.15 ± 0.41	8.985 ± 0.148	16.974 ± 1.925
95%-PICP Test	94.7 ± 2.797	95.5 ± 1.969	91.9 ± 1.475	92.9 ± 1.245
95%-PICP Train	97.429 ± 0.598	97.286 ± 0.769	93.2 ± 2.077	94.686 ± 0.584
PairwiseCorr(x, μ_z)	0.001 ± 0.0	0.005 ± 0.002	N/A	0.004 ± 0.001
PairwiseCorr(y, μ_z)	0.002 ± 0.001	0.03 ± 0.01	N/A	0.162 ± 0.069
Post-Pred Avg-LL Test	-0.48 ± 0.076	-0.462 ± 0.056	-0.512 ± 0.083	-0.995 ± 0.143
Post-Pred Avg-LL Train	-0.407 ± 0.043	-0.401 ± 0.026	-0.422 ± 0.043	-0.972 ± 0.137
RMSE Test (Unnorm)	0.05 ± 0.005	0.05 ± 0.003	0.05 ± 0.003	0.094 ± 0.009
RMSE Train (Unnorm)	0.05 ± 0.005	0.05 ± 0.003	0.05 ± 0.003	0.094 ± 0.008
Recon MSE	0.177 ± 0.01	0.174 ± 0.006	N/A	0.139 ± 0.013
Hyperparams	$\sigma_\epsilon^2 = 0.1$, $\lambda_2 = 10$, $\epsilon_T = 0.01$, $\epsilon_x = 0.5$, $\epsilon_y = 1.0$	$\sigma_\epsilon^2 = 0.1$	$\sigma_\epsilon^2 = 0.1$	$\sigma_\epsilon^2 = 0.1$

Table 13. Experiment Evaluation Summary for Airfoil($\pm$ std).

	NCAI$_\lambda$	NCAI$_{\lambda=0}$	BNN	BNN+LV
$D_{\text{JS}}(q(z)\|\|p(z))$	0.01 ± 0.005	0.011 ± 0.009	N/A	0.032 ± 0.012
$D_{\text{KL}}(p(z)\|\|q(z))$	0.021 ± 0.01	0.025 ± 0.01	N/A	0.066 ± 0.029
$\hat{I}(x;\mu_z)$	0.226 ± 0.041	0.162 ± 0.063	N/A	0.139 ± 0.086
$\hat{I}(x;z)$	0.14 ± 0.006	0.122 ± 0.011	N/A	0.127 ± 0.02
HZ($\{\mu_{z_1},\ldots,\mu_{z_N}\}$)	1.186 ± 0.558	19.071 ± 5.414	N/A	16.976 ± 3.519
s_w^*	0.496 ± 0.403	2.91 ± 2.843	1.87 ± 1.005	0.921 ± 1.313
s_y^*	0.01 ± 0.0	0.01 ± 0.0	0.01 ± 0.0	0.01 ± 0.0
s_z^*	0.251 ± 0.003	0.245 ± 0.001	N/A	0.247 ± 0.0
95%-MPIW Test (Unnorm)	11.828 ± 2.308	10.534 ± 1.751	5.704 ± 0.16	15.159 ± 5.563
95%-MPIW Train (Unnorm)	11.925 ± 2.154	10.329 ± 1.331	5.71 ± 0.146	13.813 ± 2.45
95%-PICP Test	94.51 ± 2.384	93.464 ± 1.533	81.438 ± 2.758	94.51 ± 2.981
95%-PICP Train	95.139 ± 2.296	94.36 ± 1.556	84.527 ± 4.621	94.731 ± 2.809
PairwiseCorr(x,μ_z)	0.002 ± 0.001	0.005 ± 0.004	N/A	0.008 ± 0.004
PairwiseCorr(y,μ_z)	0.003 ± 0.001	0.014 ± 0.006	N/A	0.022 ± 0.006
Post-Pred Avg-LL Test	0.898 ± 0.452	0.862 ± 0.138	1.281 ± 0.171	0.573 ± 0.288
Post-Pred Avg-LL Train	0.953 ± 0.393	0.941 ± 0.108	1.443 ± 0.16	0.657 ± 0.205
RMSE Test (Unnorm)	0.035 ± 0.007	0.029 ± 0.005	0.016 ± 0.002	0.041 ± 0.011
RMSE Train (Unnorm)	0.035 ± 0.007	0.029 ± 0.005	0.016 ± 0.002	0.041 ± 0.011
Recon MSE	0.028 ± 0.001	0.031 ± 0.003	N/A	0.028 ± 0.002
Hyperparams	$\sigma_\epsilon^2 = 0.01$, $\lambda_2 = 10$, $\epsilon_T = 0.0003$, $\epsilon_x = 0.1$, $\epsilon_y = 1.0$	$\sigma_\epsilon^2 = 0.01$	$\sigma_\epsilon^2 = 0.01$	$\sigma_\epsilon^2 = 0.01$

Table 14. Experiment Evaluation Summary for Energy Efficiency($\pm$ std).

	NCAI$_\lambda$	NCAI$_{\lambda=0}$	BNN	BNN+LV
$D_{\text{JS}}(q(z)\|\|p(z))$	0.014 ± 0.004	0.019 ± 0.006	N/A	0.037 ± 0.022
$D_{\text{KL}}(p(z)\|\|q(z))$	0.032 ± 0.009	0.041 ± 0.015	N/A	0.082 ± 0.058
$\hat{I}(x;\mu_z)$	0.036 ± 0.04	0.051 ± 0.049	N/A	0.243 ± 0.079
$\hat{I}(x;z)$	0.018 ± 0.025	0.023 ± 0.03	N/A	0.214 ± 0.052
HZ($\{\mu_{z_1}, \ldots, \mu_{z_N}\}$)	0.027 ± 0.011	7.137 ± 5.436	N/A	4.701 ± 5.439
s_w^*	2.643 ± 0.226	2.355 ± 0.28	0.12 ± 0.007	1.246 ± 0.149
s_y^*	0.1 ± 0.0	0.1 ± 0.0	0.5 ± 0.0	0.1 ± 0.0
s_z^*	0.01 ± 0.0	0.01 ± 0.0	N/A	0.01 ± 0.0
95%-MPIW Test (Unnorm)	5.428 ± 0.403	5.418 ± 0.729	2.979 ± 0.016	6.789 ± 0.408
95%-MPIW Train (Unnorm)	5.371 ± 0.38	5.368 ± 0.7	2.98 ± 0.005	6.67 ± 0.342
95%-PICP Test	94.933 ± 0.723	93.333 ± 1.054	74.2 ± 2.834	94.733 ± 0.641
95%-PICP Train	95.4 ± 0.894	93.467 ± 2.116	73.867 ± 3.288	94.867 ± 1.095
KS Test-Stat	0.023 ± 0.004	0.051 ± 0.028	N/A	0.058 ± 0.035
PairwiseCorr(x, μ_z)	0.001 ± 0.0	0.0 ± 0.0	N/A	0.0 ± 0.0
PairwiseCorr(y, μ_z)	0.111 ± 0.017	0.068 ± 0.086	N/A	0.126 ± 0.107
Post-Pred Avg-LL Test	-1.426 ± 0.042	-1.481 ± 0.018	-2.47 ± 0.083	-1.867 ± 0.078
Post-Pred Avg-LL Train	-1.399 ± 0.058	-1.429 ± 0.066	-2.6 ± 0.143	-1.894 ± 0.078
RMSE Test (Unnorm)	1.79 ± 0.09	1.787 ± 0.094	1.831 ± 0.074	1.882 ± 0.088
RMSE Train (Unnorm)	1.789 ± 0.09	1.787 ± 0.094	1.831 ± 0.074	1.883 ± 0.087
Recon MSE	0.142 ± 0.003	0.16 ± 0.017	N/A	0.13 ± 0.007
Hyperparams	$\sigma_z^2 = 0.01$, $\sigma_\epsilon^2 = 0.1$, $\lambda_2 = 10$, $\epsilon_T = 0.01$, $\epsilon_x = 0.1$, $\epsilon_y = 1.0$	$\sigma_z^2 = 0.01$,$\sigma_\epsilon^2 = 0.1$	$\sigma_\epsilon^2 = 0.5$	$\sigma_z^2 = 0.01$,$\sigma_\epsilon^2 = 0.1$

Table 15. Experiment Evaluation Summary for Heavy-Tail ($\pm$ std).

	NCAI$_\lambda$	NCAI$_{\lambda=0}$	BNN	BNN+LV
$D_{\text{JS}}(q(z)\|\|p(z))$	0.002 ± 0.002	0.001 ± 0.002	N/A	0.003 ± 0.003
$D_{\text{KL}}(p(z)\|\|q(z))$	0.001 ± 0.001	0.002 ± 0.002	N/A	0.007 ± 0.002
$\hat{I}(x;\mu_z)$	0.046 ± 0.067	0.02 ± 0.024	N/A	0.229 ± 0.113
$\hat{I}(x;z)$	-0.006 ± 0.008	-0.011 ± 0.007	N/A	0.076 ± 0.101
HZ($\{\mu_{z_1}, \ldots, \mu_{z_N}\}$)	0.026 ± 0.038	0.621 ± 0.234	N/A	0.918 ± 0.41
s_w^*	0.456 ± 0.093	0.463 ± 0.087	0.627 ± 0.039	0.416 ± 0.152
s_y^*	0.1 ± 0.0	0.1 ± 0.0	0.1 ± 0.0	0.1 ± 0.0
s_z^*	0.249 ± 0.002	0.247 ± 0.0	N/A	0.248 ± 0.001
95%-MPIW Test (Unnorm)	3.969 ± 0.184	4.091 ± 0.21	2.265 ± 0.102	4.226 ± 0.676
95%-MPIW Train (Unnorm)	3.948 ± 0.171	4.099 ± 0.176	2.264 ± 0.096	4.298 ± 0.674
95%-PICP Test	91.8 ± 2.308	92.7 ± 1.924	73.4 ± 3.681	91.8 ± 2.49
95%-PICP Train	93.9 ± 0.894	94.1 ± 0.822	75.5 ± 2.598	93.5 ± 1.173
KS Test-Stat	0.016 ± 0.004	0.019 ± 0.005	N/A	0.025 ± 0.003
PairwiseCorr(x, μ_z)	0.001 ± 0.001	0.0 ± 0.0	N/A	0.0 ± 0.0
PairwiseCorr(y, μ_z)	0.247 ± 0.148	0.403 ± 0.04	N/A	0.193 ± 0.227
Post-Pred Avg-LL Test	-0.963 ± 0.041	-0.962 ± 0.04	-1.055 ± 0.08	-1.026 ± 0.056
Post-Pred Avg-LL Train	-0.885 ± 0.03	-0.884 ± 0.033	-0.95 ± 0.07	-0.981 ± 0.107
RMSE Test (Unnorm)	0.337 ± 0.025	0.339 ± 0.026	0.335 ± 0.025	0.376 ± 0.032
RMSE Train (Unnorm)	0.337 ± 0.026	0.339 ± 0.026	0.335 ± 0.025	0.376 ± 0.031
Recon MSE	0.16 ± 0.008	0.151 ± 0.007	N/A	0.156 ± 0.013
Hyperparams	$\sigma_\epsilon^2 = 0.1$, $\lambda_2 = 10$, $\epsilon_T = 0.0003$, $\epsilon_x = 0.1$, $\epsilon_y = 1.0$	$\sigma_\epsilon^2 = 0.1$	$\sigma_\epsilon^2 = 0.1$	$\sigma_\epsilon^2 = 0.1$

Table 16. Experiment Evaluation Summary for Goldberg($\pm$ std).

	NCAI$_\lambda$	NCAI$_{\lambda=0}$	BNN	BNN+LV
$D_{\text{JS}}(q(z)\|\|p(z))$	0.012 ± 0.005	0.015 ± 0.005	N/A	0.006 ± 0.005
$D_{\text{KL}}(p(z)\|\|q(z))$	0.025 ± 0.011	0.031 ± 0.009	N/A	0.016 ± 0.008
$\hat{I}(x;\mu_z)$	0.614 ± 0.075	0.519 ± 0.091	N/A	0.982 ± 0.121
$\hat{I}(x;z)$	0.155 ± 0.048	0.059 ± 0.02	N/A	0.235 ± 0.035
HZ($\{\mu_{z_1}, \ldots, \mu_{z_N}\}$)	0.015 ± 0.019	7.248 ± 2.598	N/A	6.445 ± 2.818
s_w^*	2.927 ± 1.612	2.368 ± 1.55	0.75 ± 0.065	0.997 ± 0.943
s_y^*	0.01 ± 0.0	0.01 ± 0.0	0.1 ± 0.0	0.1 ± 0.0
s_z^*	0.247 ± 0.002	0.246 ± 0.001	N/A	0.247 ± 0.001
95%-MPIW Test (Unnorm)	1.468 ± 0.207	1.314 ± 0.126	0.89 ± 0.036	1.881 ± 0.171
95%-MPIW Train (Unnorm)	1.479 ± 0.249	1.31 ± 0.162	0.889 ± 0.033	1.908 ± 0.165
95%-PICP Test	95.4 ± 1.517	92.9 ± 1.294	77.2 ± 3.439	92.9 ± 3.008
95%-PICP Train	96.9 ± 0.894	95.0 ± 0.5	78.8 ± 3.978	95.1 ± 0.418
KS Test-Stat	0.03 ± 0.008	0.034 ± 0.008	N/A	0.033 ± 0.011
PairwiseCorr(x, μ_z)	0.005 ± 0.002	0.001 ± 0.001	N/A	0.0 ± 0.0
PairwiseCorr(y, μ_z)	0.018 ± 0.007	0.41 ± 0.113	N/A	0.572 ± 0.068
Post-Pred Avg-LL Test	-0.489 ± 0.154	-0.414 ± 0.184	-1.591 ± 0.417	-1.033 ± 0.156
Post-Pred Avg-LL Train	-0.228 ± 0.125	-0.195 ± 0.108	-1.357 ± 0.119	-0.965 ± 0.078
RMSE Test (Unnorm)	0.987 ± 0.103	0.978 ± 0.083	1.017 ± 0.06	1.118 ± 0.096
RMSE Train (Unnorm)	0.988 ± 0.101	0.979 ± 0.083	1.017 ± 0.06	1.117 ± 0.096
Recon MSE	0.017 ± 0.001	0.017 ± 0.001	N/A	0.145 ± 0.004
Hyperparams	$\sigma_\epsilon^2 = 0.01$, $\lambda_2 = 10$, $\epsilon_T = 0.01$, $\epsilon_x = 0.5$, $\epsilon_y = 0.5$	$\sigma_\epsilon^2 = 0.01$	$\sigma_\epsilon^2 = 0.1$	$\sigma_\epsilon^2 = 0.1$

Table 17. Experiment Evaluation Summary for Williams($\pm$ std).

	NCAI$_\lambda$	NCAI$_{\lambda=0}$	BNN	BNN+LV
$D_{\text{JS}}(q(z)\|\|p(z))$	0.005 ± 0.003	0.006 ± 0.003	N/A	0.008 ± 0.003
$D_{\text{KL}}(p(z)\|\|q(z))$	0.01 ± 0.005	0.013 ± 0.006	N/A	0.012 ± 0.004
$\hat{I}(x;\mu_z)$	0.254 ± 0.057	0.283 ± 0.112	N/A	0.24 ± 0.129
$\hat{I}(x;z)$	0.128 ± 0.025	0.006 ± 0.03	N/A	0.028 ± 0.017
HZ($\{\mu_{z_1}, \ldots, \mu_{z_N}\}$)	0.004 ± 0.004	8.091 ± 5.185	N/A	5.252 ± 5.607
s_w^*	0.418 ± 0.083	0.304 ± 0.028	0.251 ± 0.137	0.311 ± 0.031
s_y^*	0.1 ± 0.0	0.1 ± 0.0	0.1 ± 0.0	0.1 ± 0.0
s_z^*	0.248 ± 0.001	0.248 ± 0.0	N/A	0.249 ± 0.0
95%-MPIW Test (Unnorm)	6.862 ± 1.262	5.243 ± 0.573	2.007 ± 0.155	5.275 ± 0.569
95%-MPIW Train (Unnorm)	6.346 ± 0.821	4.906 ± 0.354	2.006 ± 0.153	4.957 ± 0.36
95%-PICP Test	95.5 ± 2.151	94.5 ± 2.0	63.4 ± 1.981	93.6 ± 2.485
95%-PICP Train	97.4 ± 1.14	95.5 ± 0.707	69.6 ± 4.519	94.9 ± 0.822
KS Test-Stat	0.031 ± 0.012	0.027 ± 0.006	N/A	0.029 ± 0.009
PairwiseCorr(x, μ_z)	0.0 ± 0.0	0.0 ± 0.0	N/A	0.0 ± 0.0
PairwiseCorr(y, μ_z)	0.109 ± 0.036	0.879 ± 0.028	N/A	0.813 ± 0.159
Post-Pred Avg-LL Test	-1.285 ± 0.066	-1.211 ± 0.083	-2.846 ± 0.346	-1.278 ± 0.164
Post-Pred Avg-LL Train	-1.111 ± 0.065	-1.04 ± 0.057	-2.347 ± 0.154	-1.079 ± 0.065
RMSE Test (Unnorm)	0.635 ± 0.042	0.619 ± 0.039	0.607 ± 0.035	0.622 ± 0.039
RMSE Train (Unnorm)	0.635 ± 0.042	0.62 ± 0.039	0.607 ± 0.035	0.622 ± 0.039
Recon MSE	0.145 ± 0.008	0.159 ± 0.007	N/A	0.153 ± 0.006
Hyperparams	$\sigma_\epsilon^2 = 0.1$, $\lambda_2 = 10$, $\epsilon_T = 0.01$, $\epsilon_x = 0.1$, $\epsilon_y = 1.0$	$\sigma_\epsilon^2 = 0.1$	$\sigma_\epsilon^2 = 0.1$	$\sigma_\epsilon^2 = 0.1$

Table 18. Experiment Evaluation Summary for Yuan($\pm$ std).

	NCAI$_\lambda$	NCAI$_{\lambda=0}$	BNN	BNN+LV
$D_{\text{JS}}(q(z)\|\|p(z))$	0.012 ± 0.012	0.002 ± 0.002	N/A	0.001 ± 0.002
$D_{\text{KL}}(p(z)\|\|q(z))$	0.042 ± 0.012	0.003 ± 0.003	N/A	0.003 ± 0.003
$\hat{I}(x;\mu_z)$	0.029 ± 0.008	0.047 ± 0.011	N/A	0.045 ± 0.012
$\hat{I}(x;z)$	0.013 ± 0.007	0.022 ± 0.003	N/A	0.02 ± 0.004
HZ($\{\mu_{z_1}, \ldots, \mu_{z_N}\}$)	1.641 ± 0.242	53.201 ± 1.247	N/A	52.566 ± 2.633
s_w^*	0.209 ± 0.017	0.15 ± 0.002	0.197 ± 0.183	0.147 ± 0.005
s_y^*	0.01 ± 0.0	0.1 ± 0.0	0.1 ± 0.0	0.1 ± 0.0
s_z^*	0.257 ± 0.001	0.251 ± 0.0	N/A	0.251 ± 0.0
95%-MPIW Test (Unnorm)	2.4 ± 0.121	2.45 ± 0.04	1.028 ± 0.014	2.466 ± 0.026
95%-MPIW Train (Unnorm)	2.396 ± 0.106	2.439 ± 0.04	1.027 ± 0.013	2.463 ± 0.034
95%-PICP Test	94.796 ± 1.185	94.279 ± 1.479	61.191 ± 2.176	94.734 ± 1.426
95%-PICP Train	94.897 ± 1.145	94.893 ± 0.286	63.265 ± 1.179	94.969 ± 0.257
PairwiseCorr(x, μ_z)	0.001 ± 0.0	0.004 ± 0.001	N/A	0.003 ± 0.0
PairwiseCorr(y, μ_z)	0.049 ± 0.01	0.142 ± 0.041	N/A	0.154 ± 0.055
Post-Pred Avg-LL Test	-0.849 ± 0.038	-1.147 ± 0.025	-1.709 ± 0.22	-1.143 ± 0.027
Post-Pred Avg-LL Train	-0.805 ± 0.033	-1.119 ± 0.013	-1.479 ± 0.056	-1.123 ± 0.015
RMSE Test (Unnorm)	0.983 ± 0.023	0.976 ± 0.016	0.92 ± 0.022	0.981 ± 0.017
RMSE Train (Unnorm)	0.983 ± 0.023	0.976 ± 0.017	0.92 ± 0.022	0.981 ± 0.017
Recon MSE	0.011 ± 0.001	0.114 ± 0.001	N/A	0.113 ± 0.001
Hyperparams	$\sigma_\epsilon^2 = 0.01$, $\lambda_2 = 10$, $\epsilon_T = 0.01$, $\epsilon_x = 0.1$, $\epsilon_y = 0.5$	$\sigma_\epsilon^2 = 0.01$	$\sigma_\epsilon^2 = 0.1$	$\sigma_\epsilon^2 = 0.1$

Table 19. Experiment Evaluation Summary for Wine Quality Red($\pm$ std).

	NCAI$_\lambda$	NCAI$_{\lambda=0}$	BNN	BNN+LV
$D_{\text{JS}}(q(z)\|\|p(z))$	0.006 ± 0.008	-0.0 ± 0.002	N/A	0.005 ± 0.004
$D_{\text{KL}}(p(z)\|\|q(z))$	0.003 ± 0.006	0.001 ± 0.005	N/A	0.002 ± 0.004
$\hat{I}(x;\mu_z)$	0.086 ± 0.013	0.087 ± 0.012	N/A	0.077 ± 0.012
$\hat{I}(x;z)$	0.108 ± 0.002	0.108 ± 0.004	N/A	0.107 ± 0.001
HZ($\{\mu_{z_1}, \ldots, \mu_{z_N}\}$)	5.42 ± 0.747	51.283 ± 9.548	N/A	27.059 ± 4.144
s_w^*	1.094 ± 0.903	1.137 ± 0.84	1.395 ± 1.155	0.384 ± 0.026
s_y^*	0.01 ± 0.0	0.01 ± 0.0	0.01 ± 0.0	0.01 ± 0.0
s_z^*	0.251 ± 0.001	0.246 ± 0.001	N/A	0.247 ± 0.0
95%-MPIW Test (Unnorm)	6.734 ± 0.212	6.712 ± 0.235	6.163 ± 0.178	8.095 ± 0.762
95%-MPIW Train (Unnorm)	6.724 ± 0.223	6.703 ± 0.217	6.163 ± 0.193	8.118 ± 0.601
95%-PICP Test	99.016 ± 2.199	98.689 ± 2.933	97.377 ± 5.865	98.033 ± 2.137
95%-PICP Train	99.444 ± 0.387	99.444 ± 0.387	97.593 ± 2.544	98.611 ± 0.655
PairwiseCorr(x, μ_z)	0.007 ± 0.003	0.007 ± 0.002	N/A	0.007 ± 0.003
PairwiseCorr(y, μ_z)	0.005 ± 0.002	0.022 ± 0.01	N/A	0.017 ± 0.01
Post-Pred Avg-LL Test	0.836 ± 0.074	0.832 ± 0.077	0.818 ± 0.187	0.638 ± 0.121
Post-Pred Avg-LL Train	0.865 ± 0.025	0.872 ± 0.024	0.868 ± 0.074	0.678 ± 0.047
RMSE Test (Unnorm)	0.005 ± 0.001	0.005 ± 0.001	0.005 ± 0.001	0.008 ± 0.001
RMSE Train (Unnorm)	0.005 ± 0.001	0.005 ± 0.001	0.005 ± 0.001	0.008 ± 0.001
Recon MSE	0.014 ± 0.0	0.014 ± 0.0	N/A	0.014 ± 0.001
Hyperparams	$\sigma_\epsilon^2 = 0.01$, $\lambda_2 = 10$, $\epsilon_T = 0.01$, $\epsilon_x = 0.5$, $\epsilon_y = 0.5$	$\sigma_\epsilon^2 = 0.01$	$\sigma_\epsilon^2 = 0.01$	$\sigma_\epsilon^2 = 0.01$

Table 20. Experiment Evaluation Summary for Yacht($\pm$ std).

	NCAI$_\lambda$	NCAI$_{\lambda=0}$	BNN	BNN+LV
$D_{\text{JS}}(q(z)\|\|p(z))$	0.004 ± 0.002	0.004 ± 0.001	N/A	0.006 ± 0.002
$D_{\text{KL}}(p(z)\|\|q(z))$	0.008 ± 0.003	0.008 ± 0.004	N/A	0.01 ± 0.003
$\hat{I}(x;\mu_z)$	0.842 ± 0.06	0.373 ± 0.037	N/A	0.667 ± 0.061
$\hat{I}(x;z)$	0.035 ± 0.012	-0.015 ± 0.007	N/A	0.146 ± 0.026
HZ($\{\mu_{z_1}, \ldots, \mu_{z_N}\}$)	0.005 ± 0.001	7.804 ± 1.727	N/A	5.09 ± 0.991
s_w^*	0.488 ± 0.026	0.444 ± 0.019	0.28 ± 0.132	0.231 ± 0.002
s_y^*	0.01 ± 0.0	0.01 ± 0.0	0.1 ± 0.0	0.01 ± 0.0
s_z^*	0.247 ± 0.0	0.247 ± 0.0	N/A	0.248 ± 0.0
95%-MPIW Test (Unnorm)	0.258 ± 0.026	0.247 ± 0.022	0.366 ± 0.005	0.313 ± 0.03
95%-MPIW Train (Unnorm)	0.293 ± 0.011	0.277 ± 0.012	0.366 ± 0.005	0.336 ± 0.014
95%-PICP Test	96.818 ± 2.591	96.364 ± 2.033	96.364 ± 3.447	95.455 ± 2.784
95%-PICP Train	95.871 ± 0.736	94.968 ± 0.957	93.161 ± 1.08	93.29 ± 0.866
KS Test-Stat	0.028 ± 0.005	0.025 ± 0.005	N/A	0.024 ± 0.009
PairwiseCorr(x, μ_z)	0.0 ± 0.0	0.0 ± 0.0	N/A	0.0 ± 0.0
PairwiseCorr(y, μ_z)	0.007 ± 0.003	0.084 ± 0.009	N/A	0.121 ± 0.008
Post-Pred Avg-LL Test	0.263 ± 0.11	0.269 ± 0.107	-0.31 ± 0.069	0.129 ± 0.131
Post-Pred Avg-LL Train	0.155 ± 0.043	0.159 ± 0.046	-0.386 ± 0.035	-0.021 ± 0.053
RMSE Test (Unnorm)	0.995 ± 0.059	0.988 ± 0.061	1.143 ± 0.087	1.231 ± 0.057
RMSE Train (Unnorm)	0.994 ± 0.059	0.988 ± 0.062	1.143 ± 0.087	1.231 ± 0.056
Recon MSE	0.017 ± 0.0	0.017 ± 0.0	N/A	0.015 ± 0.001
Hyperparams	$\sigma_\epsilon^2 = 0.01$, $\lambda_2 = 10$, $\epsilon_T = 0.01$, $\epsilon_x = 0.5$, $\epsilon_y = 0.5$	$\sigma_\epsilon^2 = 0.01$	$\sigma_\epsilon^2 = 0.1$	$\sigma_\epsilon^2 = 0.01$

Table 21. Experiment Evaluation Summary for Lidar($\pm$ std).

	NCAI$_\lambda$	NCAI$_{\lambda=0}$	BNN	BNN+LV
$D_{\text{JS}}(q(z)\|\|p(z))$	0.003 ± 0.001	0.005 ± 0.001	N/A	0.031 ± 0.005
$D_{\text{KL}}(p(z)\|\|q(z))$	0.008 ± 0.002	0.009 ± 0.002	N/A	0.357 ± 0.088
$\hat{I}(x;\mu_z)$	0.057 ± 0.017	0.032 ± 0.017	N/A	0.428 ± 0.04
$\hat{I}(x;z)$	0.047 ± 0.015	0.024 ± 0.014	N/A	0.387 ± 0.045
HZ($\{\mu_{z_1}, \ldots, \mu_{z_N}\}$)	0.015 ± 0.004	0.792 ± 0.357	N/A	6.408 ± 2.439
s_w^*	34.575 ± 17.89	22.305 ± 7.342	1.805 ± 0.094	12.39 ± 4.903
s_y^*	0.1 ± 0.0	0.1 ± 0.0	1.0 ± 0.0	0.1 ± 0.0
s_z^*	1.0 ± 0.0	1.0 ± 0.0	N/A	1.0 ± 0.0
95%-MPIW Test (Unnorm)	7.375 ± 0.263	7.145 ± 0.16	4.011 ± 0.006	22.165 ± 10.073
95%-MPIW Train (Unnorm)	7.433 ± 0.299	7.114 ± 0.217	4.011 ± 0.001	22.267 ± 10.346
95%-PICP Test	93.84 ± 1.78	93.44 ± 1.757	73.68 ± 1.842	96.0 ± 1.095
95%-PICP Train	95.493 ± 0.256	95.227 ± 0.289	75.493 ± 1.489	96.773 ± 0.446
KS Test-Stat	0.014 ± 0.001	0.02 ± 0.002	N/A	0.044 ± 0.007
PairwiseCorr(x, μ_z)	0.003 ± 0.001	0.0 ± 0.0	N/A	0.0 ± 0.0
PairwiseCorr(y, μ_z)	0.035 ± 0.009	0.138 ± 0.014	N/A	0.161 ± 0.039
Post-Pred Avg-LL Test	-1.979 ± 0.04	-1.973 ± 0.049	-2.306 ± 0.059	-2.342 ± 0.048
Post-Pred Avg-LL Train	-1.92 ± 0.021	-1.895 ± 0.018	-2.217 ± 0.069	-2.229 ± 0.04
RMSE Test (Unnorm)	1.985 ± 0.051	1.932 ± 0.059	1.953 ± 0.071	3.523 ± 0.501
RMSE Train (Unnorm)	1.985 ± 0.051	1.933 ± 0.059	1.953 ± 0.071	3.521 ± 0.501
Recon MSE	0.124 ± 0.002	0.122 ± 0.001	N/A	0.123 ± 0.005
Hyperparams	$\sigma_z^2 = 1.0$, $\sigma_\epsilon^2 = 0.1$, $\lambda_2 = 10$, $\epsilon_T = 0.01$, $\epsilon_x = 0.5$, $\epsilon_y = 1.0$	$\sigma_z^2 = 1.0, \sigma_\epsilon^2 = 0.1$	$\sigma_\epsilon^2 = 1.0$	$\sigma_z^2 = 1.0$,$\sigma_\epsilon^2 = 0.1$

Table 22. Experiment Evaluation Summary for Depeweg($\pm$ std).