Learned Uncertainty-Aware (LUNA) Bases for Bayesian Regression using Multi-Headed Auxiliary Networks

Sujay Thakur [1] Cooper Lorsung [1] Yaniv Yacoby [1] Finale Doshi-Velez [1] Weiwei Pan [1]

Abstract

Neural Linear Models (NLM) are deep models that produce predictive uncertainty by learning features from the data and then performing Bayesian linear regression over these features. Despite their popularity, few works have focused on formally evaluating the predictive uncertainties of these models. In this work, we show that traditional training procedures for NLMs can drastically underestimate uncertainty in data-scarce regions. We identify the underlying reasons for this behavior and propose a novel training procedure for capturing useful predictive uncertainties.

1. Introduction

In high-stakes, safety critical applications of machine learning, reliable measurements of model predictive uncertainty matter just as much as predictive accuracy. Traditionally, applications that require predictive uncertainty have relied on Gaussian Processes (GPs) (Rasmussen & Williams, 2006), a non-parametric model that produces high predictive uncertainty in data-scarce regions and low uncertainty in data rich ones. Recent works, however, have advocated for the use of deep Bayesian models as fast and scalable GP alternatives (Springenberg et al., 2016; Snoek et al., 2015).

Bayesian Neural Networks (BNNs) (Neal, 1995) provide a way of explicitly capturing model uncertainty - uncertainty arising from having insufficient observations to determine the “true” predictor - by placing a prior distribution over network weights. Just like GPs, rather than point estimate predictions, Bayesian inference for BNNs produces distributions over possible predictions, whose variance can be used as an indicator of model confidence during test time.

However, inference for large BNNs remains extremely challenging. For this reason, Neural Linear Models (NLM), a model with BNN-like properties but highly tractable inference, is gaining popularity (Pinsler et al., 2019; Snoek et al., 2015; Riquelme et al., 2018; Zhou & Precioso, 2019). NLM places priors only on the last layer of network weights and learns point estimates for the remaining weights; inference for the last weight layer can then be performed analytically. One interprets the deterministic network layers as a finite dimensional feature space embedding of the data, and the last layer of NLMs as performing Bayesian linear regression on the basis defined by these features. So, the NLM is also an approximation of GPs (the latter performs Bayesian linear regression on infinite dimensional bases).

Despite their increasing popularity, little work has been done to formally evaluate the quality of uncertainty estimates produced by NLMs. In the first paper to do so (Ober & Rasmussen, 2019), the authors show that NLMs can achieve high log-likelihood on test data sampled from training data-scarce regions; they treat this as evidence that NLM uncertainties can distinguish data-scarce and data rich regions. However, as noted in (Yao et al., 2019), log-likelihood measures only how well predictive uncertainty aligns with the variation in the actual data and not how well these uncertainties predict data-scarcity.

Our contributions are both theoretical and methodological: (1) we show that NLM’s predictive uncertainty may not be able to distinguish data-scarce regions from data-rich ones; furthermore, we identify the precise cause of the problem: the NLM training procedure learns feature bases incapable of expressing uncertainty in data-scarce regions, or “in-between" uncertainties (2) we propose a new basis training procedure, LUNA, for NLMs that explicitly optimizes features to produce expressive in-between uncertainties.

We show, on a number of synthetic and real datasets, that we are reliably able to identify data-scarce regions in training where NLM (and other baseline methods) struggle. Furthermore, we show that LUNA is able to learn bases that outperform NLM bases in terms of transfer learning and minimizing generalization error. Lastly, we demonstrate the utility of LUNA’s uncertainty estimates by showing that LUNA bases outperform baselines on a down-stream task for uncertainty estimation: Bayesian Optimization.

[1] John A. Paulson School of Engineering and Applied Sciences, Harvard University, Cambridge, MA, USA. Correspondence to: Weiwei Pan <weiweipan@g.harvard.edu>.

Presented at the ICML 2020 Workshop on Uncertainty and Robustness in Deep Learning. Copyright 2020 by the author(s).

2. Background

Let the input space be D dimensional, and suppose we have a dataset $\{(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_N, y_N)\}$, where $\mathbf{x}_n \in \mathbb{R}^D$ and $y_n \in \mathbb{R}$. A Neural Linear Model (NLM) consists of: (1) a feature map $\phi_\theta : \mathbb{R}^D \to \mathbb{R}^L$, parameterized by a neural network with weights θ, and (2) a Bayesian linear regression model fitted on the data embedded in the feature space:

$$\mathbf{y} \sim \mathcal{N}(\boldsymbol{\Phi}_\theta \mathbf{w}, \sigma^2 \mathbf{I}), \quad \mathbf{w} \sim \mathcal{N}(\mathbf{0}, \alpha \mathbf{I})$$

where the design matrix $\boldsymbol{\Phi}_\theta = [\widetilde{\phi_\theta(\mathbf{x}_1)}, \dots, \widetilde{\phi_\theta(\mathbf{x}_N)}]^T$ and $\widetilde{\phi_\theta(\mathbf{x}_n)}$ is the feature vector $\phi_\theta(\mathbf{x}_n)$ augmented with a 1. Thus, for NLMs, the posterior, marginal and posterior predictive distributions are all computed analytically.

Intuitively, NLM represents a neural network with L number of nodes in the last hidden layer and Gaussian priors placed on the last layer of weights, $\mathbf{w}$.

Typically, the θ is learned by training the entire network with an maximum a posteriori objective (Snoek et al., 2015), since marginalizing out $\mathbf{w}$ involves backpropagating through large matrix inversions (Appendix Section A.2):

$$\mathcal{L}_{\text{MAP}}(\theta_{\text{Full}}) = \log \mathcal{N}\left(\mathbf{y}; \boldsymbol{\Phi}_\theta \mathbf{w}, \sigma^2 \mathbf{I}\right) - \gamma \left\|\theta_{\text{Full}}\right\|_2^2 \quad (1)$$

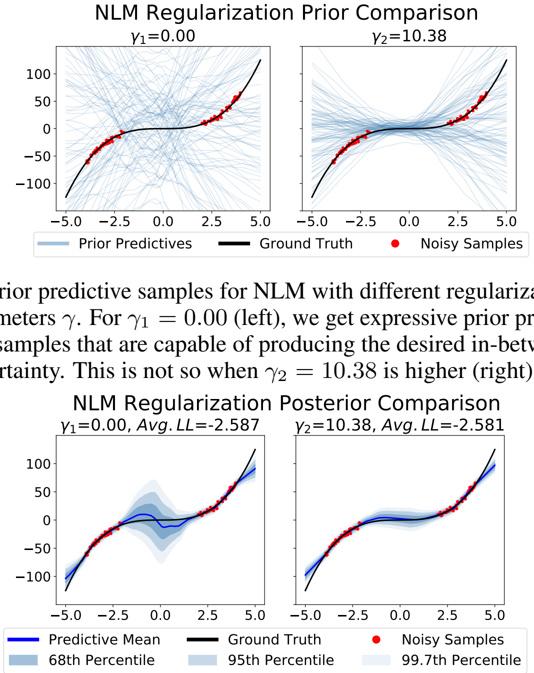
where $\theta_{\text{Full}} = (\theta, \mathbf{w})$ are weights of the full network. Inference on $\mathbf{w}$ is then performed with θ_{MAP} fixed.

3. Analysis of the Expressiveness of Neural Linear Model Uncertainties

In this section, we demonstrate that conventional training for NLMs consistently learns feature bases that span a limited class of functions in the prior predictive. As a result, the posterior predictive will be distributed over very limited function classes. Specifically, ***the lack of diversity in the posterior predictive functions leads to limited functional variations across the input domain, causing NLMs to underestimate in-between uncertainty***.

We also identify the precise cause of this problem: the training objective optimizes the feature bases to maximize the likelihood of the observed data and doing so does nothing to encourage the bases to support functions that extrapolate differently away from the observed data. In fact, we show that the regularization terms in the NLM training objective actively discourages functional diversity in the posterior. In Appendix A.2, we show that our analysis generalizes to the case where one optimizes the feature map ϕ_θ to maximize the marginal data likelihood (by integrating out $\mathbf{w}$).

Feature map regularization leads to inexpressive feature bases. When the regularization term in the NLM objective (Eq 1) is non-zero, the feature map ϕ_θ is explicitly discouraged from producing bases spanning functions that extrapolate differently away from the observed data. This is because such diversity comes at the cost of larger values in θ and does not impact the log-likelihood of the observed data. In Figure 1a, we show samples from the prior predictive for two NLMs with regularized and unregularized feature maps ϕ_θ. With the regularization parameter γ set at 0, the basis for ϕ_θ spans a diverse class of functions under the prior distribution $p(\mathbf{w})$ over their linear combinations - the prior predictive samples show variation in both the data-rich and data-scarce regions. With regularization, the feature basis spans a limited set of functions - the prior predictive samples show no variation in the data-scarce region. In Figure 6 of Appendix D.1, we reproduce the effect of regularization on NLM prior predictives for different γ over random restarts. In the following, we show that NLMs are biased towards learning inexpressive bases even when we set γ to be zero!



(a) Prior predictive samples for NLM with different regularization parameters γ. For $\gamma_1 = 0.00$ (left), we get expressive prior predictive samples that are capable of producing the desired in-between uncertainty. This is not so when $\gamma_2 = 10.38$ is higher (right).

(b) Posterior predictive distributions for NLM with different regularization parameters γ. With $\gamma_1 = 0.00$ (left), we express desired in-between uncertainty, but this is not so with $\gamma_2 = 10.38$ (right). Both models give similar test log-likelihoods, thus test log-likelihood cannot measure the quality of in-between uncertainty.

Test log-likelihood does not measure the quality of posterior in-between uncertainty. Typically, the hyperparameter γ is chosen by grid-search or BayesOpt maximizing the test log-likelihood (where the test data is sampled from the same distribution as the train data). In Figure 1b, the test log-likelihood for the NLM with a regularized feature map is a hair higher. Thus, by maximizing test log-likelihood, we choose a model with a posterior predictive that is inexpressive over the data-scarce region and hence unable to capture in-between uncertainty.

Likewise, the architecture of the feature map ϕ_θ, e.g. the number of features L, is also chosen by maximizing the test log-likelihood. In Figures 7 and 8 of Appendix D.1, we demonstrate that the number of features, more so than the depth of the network, determines the expressiveness of priors predictives (and hence posteriors predictives) of NLMs. In particular, we show that for a small number of features, even when ϕ_θ is unregularized, while some optimal (with respect to train log-likelihood) settings of θ will result in expressive prior predictives, most will not. On the other hand, when the number of features L is large, then most optimal settings of θ will correspond to expressive priors predictives. However, when we choose L to be as small as possible while maximizing test log-likelihood, then we may very well choose an architecture for which no optimal setting of θ will correspond to an expressive prior predictive.

4. Learned UNcertainty-Aware (LUNA) Bases

To ensure that the NLM learns a feature basis that spans a diverse set of functions in the prior predictive, we explicitly train the feature map ϕ_θ to encode for functional diversity. Specifically, we train M auxiliary linear regressors, $f_m(\mathbf{x}) = \mathbf{\Phi}_\theta \mathbf{w}_m$, on a shared design matrix $\mathbf{\Phi}_\theta$ (see illustration in Figure 2). Our training objective, $\mathcal{L}_{\text{LUNA}}$, maximizes the mean log-likelihood of the regressors on the training data, measured by $\mathcal{L}_{\text{FIT}}$, while encouraging for functional diversity amongst them, measured by $\mathcal{L}_{\text{DIVERSE}}$ (defined further below):

$$\mathcal{L}_{\text{LUNA}}(\Psi) = \mathcal{L}_{\text{FIT}}(\Psi) - \lambda \cdot \mathcal{L}_{\text{DIVERSE}}(\Psi) \qquad (2)$$

where $\Psi = (\theta, \mathbf{w}_1, \dots \mathbf{w}_M)$, θ parametrizes the shared designed matrix and $\mathbf{w}_m$ are the weights of f_m. The constant λ controls for the degree to which we prioritize diversity. We encourage for diversity in the regressors trained on our basis since our analysis in Section 3 shows that if the feature basis spans diverse functions under the prior $p(\mathbf{w})$, the posterior predictive will be able to capture in-between uncertainty.

After optimizing our feature map via:

$$\theta_{\text{LUNA}}, \{\mathbf{w}_{m,\text{LUNA}}\} = \text{argmax}_{\Psi} \mathcal{L}_{\text{LUNA}}(\Psi),$$

we discard the auxiliary regressors $\{\mathbf{w}_{m,\text{LUNA}}\}$ and perform Bayesian linear regression on the diversified feature basis, the LUNA basis. That is, we analytically compute the posterior $p(\mathbf{w}|\mathcal{D}, \theta_{\text{LUNA}})$ over the last Bayesian layer of weights $\mathbf{w}$ in the Neural Linear Model. ***In summary, LUNA training results in a basis that supports a diverse set of predictions by varying*** $\mathbf{w}$.

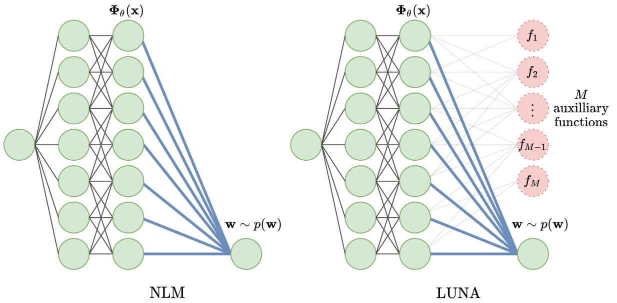

$\Phi_\theta(\mathbf{x})$
$\mathbf{w} \sim p(\mathbf{w})$
NLM
$\Phi_\theta(\mathbf{x})$
f_1
f_2
⋮
f_{M-1}
f_M
M auxilliary functions
$\mathbf{w} \sim p(\mathbf{w})$
LUNA

Figure 2. Illustration of the NLM and LUNA training. During training, in LUNA, the output layer has M neurons instead (shown in red) that learn different functions for the data. These neurons are discarded during inference. This helps the model learn more optimum features $\mathbf{\Phi}_\theta$ for well-calibrated uncertainty.

$\mathcal{L}_{\text{FIT}}$**: Fitting the Auxiliary Regressors.** We learn the regressors jointly with $\mathbf{\Phi}_\theta$, by maximizing the average train log-likelihood of the regressors on the training data, with ℓ_2 penalty on θ as well as on the weights of each regressor:

$$\mathcal{L}_{\text{FIT}}(\Psi) = \frac{1}{M} \sum_{m=1}^{M} \log \mathcal{N}(\mathbf{y}; f_m(\mathbf{x}), \sigma^2 \mathbf{I}) - \gamma \, \|\Psi\|_2^2 \, .$$

$\mathcal{L}_{\text{DIVERSE}}$**: Enforcing diversity.** We enforce diversity in the auxiliary regressors as a proxy for the diversity of the functions spanned by the feature basis. We adapt the Local Independence Training objective in (Ross et al., 2018) to encourage our regressors to extrapolate differently away from the training data, where cos sim is cosine similarity:

$$\mathcal{L}_{\text{DIVERSE}}(\Psi) = \sum_{i=1}^{M} \sum_{j=i+1}^{M} \text{CosSim}^2 \left(\nabla_{\mathbf{x}} f_i(\mathbf{x}), \nabla_{\mathbf{x}} f_j(\mathbf{x}) \right) .$$

Here we encourage extrapolation difference in every pair of regressors f_i and f_j by penalizing non-orthogonal gradients. Furthermore, we avoid expensive gradient computations using a finite difference approximation. A detailed explanation of the training objective is found in Appendix B.1.

5. Experiments

We compare LUNA to traditional NLM inference as well as a variety of other models on 3 toy problems and 5 UCI "gap" datasets (Foong et al., 2019). We show that NLMs with LUNA bases are able to distinguish data-scarce regions from data rich ones. We also show that with fewer features, NLM with LUNA bases achieves lower generalization error and LUNA bases retain their utility in transfer learning. Finally, we show that NLM with LUNA bases outperform baselines on a Bayesian Optimization task.

Our baselines are: NLM, MC Dropout (MCD) (Gal & Ghahramani, 2016), MAP (see Appendix C.3) and ensemble. Experimental setup details in Appendix C.

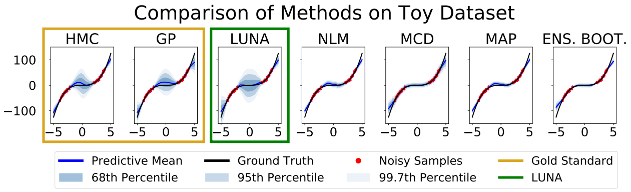


Figure 3. Comparison of posterior predictive distributions on "Cubic Gap" (see Appendix C.1). "Gold standards": BNN with Hamiltonian Monte Carlo sampling, GP. LUNA most resembles gold standards, while other methods underestimate uncertainty in gap regions.

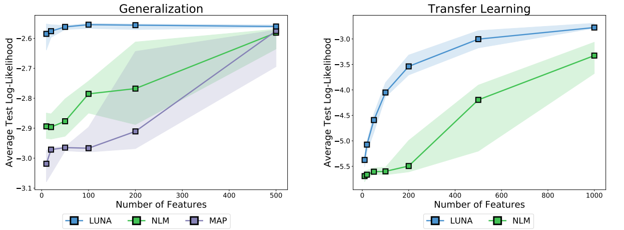
LUNA obtains better uncertainty estimation on Toy Data Following the set-up in (Ober & Rasmussen, 2019), we construct a synthetic 1-D cubic function with a gap (see "Cubic Gap" in Appendix C.1). Figure 3 shows that NLM with LUNA bases matches the performance of the "gold-standards": BNN with HMC and GP, while other baselines drastically underestimate in-between uncertainty.

LUNA obtains better uncertainty estimation for UCI Gap Datasets We use 5 UCI "gap" datasets (Foong et al., 2019) with artificially created gaps in the training set, and demonstrate that, as desired, LUNA's model uncertainty is able to distinguish between the gap region and data-rich regions of the input space. We evaluate test log-likelihoods and epistemic uncertainties for data sampled inside the gap and outside the gap (see Appendix C.2). In the gap, a good model should have higher epistemic uncertainty without a large decrease in test log-likelihood (i.e. the generalization error should be low). In Appendix D.4, we see that, across all data sets, NLM with LUNA bases has higher epistemic uncertainty for test data in gaps without a large decrease in log-likelihood. NLM with traditional training results in significant decrease of log-likelihood on gap data (the model overfits) and does not provide higher epistemic uncertainty in gaps. Finally, MCD generalizes well to gap data, but fails to provide higher epistemic uncertainty in gaps.

LUNA bases are better for generalization and transfer learning even with few features. We show that for a range of feature numbers, NLM with LUNA bases outperforms NLM when generalizing to test data sampled from the same distribution as train. We construct a synthetic 1-D dataset ("Squiggle Gap" in Appendix C.1), which unlike the cubic example, has unexpected variations in the held-out gap region. In Figure 4a, we see that with a very large number of features, all models generalize well. However, LUNA bases generalizes well for even very small number of features.

We now consider transfer learning, that is, we fix the feature map ϕ_θ and re-train the posterior $p(\mathbf{w}|\mathcal{D}, \theta)$ over the last layer using data from the gap. Figure 4b shows that LUNA bases can easily be adapted to modeling data from the gap, while NLM feature bases struggle to adapt as well, even as we increase the number of features.

LUNA performs better on BayesOpt We compare NLM with LUNA bases, NLM and GP for Bayesian optimization on a 1D function (see BayesOpt in Appendix C.1). In Figure 12 of Appendix D.3, we see LUNA and NLM both converge faster than the GP. Over 10 runs, LUNA outperforms NLM on average and has much lower variance across runs.



(a) Generalization task

(b) Transfer learning task

Figure 4. Comparison of LUNA and baselines on "Squiggle Gap" (see Appendix C.1) as the number of features are varied. **Generalization:** With any number of features, LUNA outperforms baselines on test and train data sampled from the same distribution. **Transfer learning:** With any number of features, LUNA outperforms NLM under covariate shift when the features are fixed but the Bayesian linear model is retrained on the new data.

6. Conclusion

In this paper, we show that traditional NLM training (maximizing MAP) learns feature bases that span a limited class of functions in the prior predictive distribution, and hence NLM posterior uncertainty cannot distinguish data-poor regions from data rich-ones. We identified the cause of the problem: optimizing the bases to maximize likelihood on train data does not encourage them to span diverse functions away from training data. Furthermore, we showed that this problem is not solved when the feature map ϕ_θ is learned by maximizing the marginal log-likelihood.

Based on this observation, we propose a new feature training procedure LUNA that explicitly encourages diversity in functions spanned by the learned feature basis. On toy and real data, we show that NLMs with LUNA bases outperform baselines on uncertainty estimation, generalization and transfer learning. On a toy BayesOpt task, we show that LUNA bases out-perform baselines as well as a GP.

Feature map learning in NLMs is a hyperparameter selection procedure for Bayesian linear regression. Thus, our work shows that **we need general *uncertainty-aware* frameworks for hyperparameter selection that are alternatives to likelihood-based selection**. In future work, we intend to explore theoretical connections between LUNA and likelihood-based training in this broader setting.

Acknowledgments

ST, CL, and WP are supported by the Harvard Institute of Applied Computational Sciences. YY acknowledges support from NIH 5T32LM012411-04 and from IBM Research.

References

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

Foong, A. Y. K., Li, Y., Hernández-Lobato, J. M., and Turner, R. E. 'in-between' uncertainty in bayesian neural networks. In *International Conference on Machine Learning (ICML): Workshop on Uncertainty & Robustness in Deep Learning*, 2019.

Gal, Y. and Ghahramani, Z. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *Proceedings of the 33rd International Conference on Machine Learning (ICML)*, pp. 1050–1059, 2016.

Iglewicz, B. and Hoaglin, D. *Volume 16: How to Detect and Handle Outliers*. 1993.

Kingma, D. P. and Ba, J. Adam: A Method for Stochastic Optimization. *arXiv e-prints*, art. arXiv:1412.6980, December 2014.

Neal, R. M. Bayesian learning for neural networks, 1995.

Ober, S. W. and Rasmussen, C. E. Benchmarking the neural linear model for regression. In *Advances in Approximate Bayesian Inference (AABI)*, 2019.

Pinsler, R., Gordon, J., Nalisnick, E., and Hernández-Lobato, J. M. Bayesian batch active learning as sparse subset approximation. In *Advances in Neural Information Processing Systems 32*, pp. 6359–6370, 2019.

Rasmussen, C. and Williams, C. *Gaussian Processes for Machine Learning*. Adaptive Computation and Machine Learning. MIT Press, Cambridge, MA, USA, January 2006.

Riquelme, C., Tucker, G., and Snoek, J. Deep bayesian bandits showdown: An empirical comparison of bayesian deep networks for thompson sampling. In *International Conference on Learning Representations (ICLR)*, 2018.

Ross, A. S., Pan, W., and Doshi-Velez, F. Learning qualitatively diverse and interpretable rules for classification. In *International Conference on Machine Learning (ICML): Workshop on Human Interpretability in Machine Learning*, 2018.

Snoek, J., Rippel, O., Swersky, K., Kiros, R., Satish, N., Sundaram, N., Patwary, M. M. A., Prabhat, and Adams, R. P. Scalable bayesian optimization using deep neural networks. In *Proceedings of the 32nd International Conference on Machine Learning (ICML)*, pp. 2171–2180, 2015.

Springenberg, J. T., Klein, A., Falkner, S., and Hutter, F. Bayesian optimization with robust bayesian neural networks. In *Advances in Neural Information Processing Systems 29*, pp. 4134–4142, 2016.

Yao, J., Pan, W., Ghosh, S., and Doshi-Velez, F. Quality of uncertainty quantification for bayesian neural network inference. In *International Conference on Machine Learning (ICML): Workshop on Uncertainty & Robustness in Deep Learning*, 2019.

Zhou, W. and Precioso, F. Adaptive bayesian linear regression for automated machine learning. Technical Report arXiv:1904.00577 [cs.LG], ArXiV, 2019.

A. Neural Linear Model Details

A.1. MAP Training

Here, we largely follow the specification in (Ober & Rasmussen, 2019). NLM uses a neural network to parameterize basis functions for a Bayesian linear regression model by treating the output weights of the network probabilistically, while treating the rest of the network's parameters θ as hyperparameters.

Using notation from Section 2 and following standard Bayesian linear regression analysis, we can derive the posterior predictive as

$$p(y_\star|\mathbf{x}_\star,\mathcal{D}) = \mathcal{N}(y_\star; \mathbf{w}_N^T\phi_\theta(\mathbf{x}_\star), \sigma^2+\phi_\theta(\mathbf{x}_\star)^T\mathbf{V}_N\phi_\theta(\mathbf{x}_\star))$$

where

$$\begin{aligned}\mathbf{w}_N &= \frac{1}{\sigma^2}\mathbf{V}_N{\mathbf{\Phi}_\theta}^T\mathbf{y}\\ \mathbf{V}_N^{-1} &= \frac{1}{\alpha}\mathbf{I}_{M\times M} + \frac{1}{\sigma^2}{\mathbf{\Phi}_\theta}^T\mathbf{\Phi}_\theta.\end{aligned} \tag{3}$$

For the MAP-trained NLM, we maximize the objective

$$\begin{aligned}\mathcal{L}_{\text{MAP}}(\theta_{\text{Full}}) &= \log\mathcal{N}\left(\mathbf{y};\mathbf{\Phi}_\theta\mathbf{w},\sigma^2\mathbf{I}\right) - \gamma\left\|\theta_{\text{Full}}\right\|_2^2\\ &= -\frac{N}{2}\log 2\pi\sigma^2 - \frac{1}{2\sigma^2}\left\|\mathbf{y}-\mathbf{\Phi}_\theta\mathbf{w}\right\|_2^2\\ &\quad - \gamma\left\|\theta_{\text{Full}}\right\|_2^2\end{aligned}$$

where θ_{Full} represents the parameters of the full network (including the output weights). We would then extract θ from θ_{Full} and perform the Bayesian linear regression as above.

A.2. Marginal Likelihood Training

The NLM is defined the same as above, but we optimize θ to maximize the evidence or log marginal likelihood of the data instead (by integrating out $\mathbf{w}$). For training stability and identifiability, we further regularize θ as in (Ober & Rasmussen, 2019). The full objective is hence:

$$\begin{aligned}\mathcal{L}_{\text{Marginal}}(\theta_{\text{Full}}) &= \log\int p(\mathbf{y}|\mathbf{X},\mathbf{w})p(\mathbf{w})d\mathbf{w}\\ &= -\frac{N}{2}\log 2\pi\sigma^2 - \frac{1}{2\sigma^2}\left\|\mathbf{y}-\mathbf{\Phi}_\theta\mathbf{w}\right\|_2^2\\ &\quad - \frac{M}{2}\log\alpha - \frac{1}{2\alpha}\left\|\mathbf{w}_N\right\|_2^2\\ &\quad - \frac{1}{2}\log\left|\mathbf{V}_N\right|\end{aligned}$$

The authors of (Ober & Rasmussen, 2019) note that the addition of a regularization term $\gamma\left\|\theta\right\|_2^2$ to $\mathcal{L}_{\text{Marginal}}$ is necessary for the estimates of the output noise since this tends to zero when the objective is unregularized. In Theorem A.1, we also show that without the regularization term $\gamma\left\|\theta\right\|_2^2$, the features $\mathbf{\Phi}_\theta$ experience pathological blow-up for ReLu networks, since large $\mathbf{\Phi}_\theta$ reduce the magnitude of the posterior mean $\mathbf{w}_N$, and hence increase $\mathcal{L}_{\text{Marginal}}$.

Theorem A.1. *Suppose that the activations are ReLu functions. For fixed θ, $\mathbf{w}$ and any $c > 0$, we define θ^c as θ but with the last layer of weights scaled by c, we also define $\mathbf{w}^c$ as $\frac{1}{c}\mathbf{w}$. For a sufficiently large $C > 0$ and any $c > C$, we have that $\mathcal{L}_{\text{Marginal}}(\theta_{\text{Full}}) < \mathcal{L}_{\text{Marginal}}(\theta_{\text{Full}}^c)$, where $\theta_{\text{Full}} = (\theta,\mathbf{w})$ and $\theta_{\text{Full}}^c = (\theta^c,\mathbf{w}^c)$.*

Proof. Let us first establish the relationship between $\mathbf{w}_N$ and $\mathbf{\Phi}_\theta$ in the asymptotic case $\|\mathbf{\Phi}_\theta\| \to \infty$. From Eq 3,

$$\begin{aligned}\frac{1}{\sigma^2}{\mathbf{\Phi}_\theta}^T\mathbf{\Phi}_\theta \gg \frac{1}{\alpha}\mathbf{I}_{M\times M} &\implies \mathbf{V}_N^{-1} \to \frac{1}{\sigma^2}{\mathbf{\Phi}_\theta}^T\mathbf{\Phi}_\theta\\ &\implies \mathbf{V}_N \to \sigma^2\left({\mathbf{\Phi}_\theta}^T\mathbf{\Phi}_\theta\right)^{-1}\end{aligned}$$

Hence,

$$\mathbf{w}_N \to \left({\mathbf{\Phi}_\theta}^T\mathbf{\Phi}_\theta\right)^{-1}{\mathbf{\Phi}_\theta}^T\mathbf{y} \implies \|\mathbf{w}_N\| \sim \frac{1}{\|\mathbf{\Phi}_\theta\|}.$$

Note that the loss $\|\mathbf{y}-\mathbf{\Phi}_\theta\mathbf{w}\|$ is equal to $\|\mathbf{y}-\mathbf{\Phi}_{\theta^c}\mathbf{w}^c\|$, since $\mathbf{\Phi}_{\theta^c}$ is $\mathbf{\Phi}_\theta$ scaled by c and this scaling is canceled by $\mathbf{w}^c = \frac{1}{c}\mathbf{w}$. Thus, since $\|\mathbf{w}_N^c\| < \|\mathbf{w}_N\|$, we have that $\mathcal{L}_{\text{Marginal}}(\theta_{\text{Full}}) < \mathcal{L}_{\text{Marginal}}(\theta_{\text{Full}}^c)$. □

The above theorem tells us that that we can continue to increase $\mathcal{L}_{\text{Marginal}}$ by reducing $\|\mathbf{w}_N\|$. Hence, if we do not regularize θ, the training will continually increase $\|\mathbf{\Phi}_\theta\|$ to affect a decrease in $\|\mathbf{w}_N\|$.

However, the addition of the regularization term $\gamma\left\|\theta\right\|_2^2$ to $\mathcal{L}_{\text{Marginal}}$ biases training towards inexpressive feature bases for the same reason we identified in Section 3. In Figure 10, we show that with regularization, the feature bases learned by optimizing $\mathcal{L}_{\text{Marginal}}$ are inexpressive. In Figure 9 we see that even with γ set close to zero, the learned feature bases are not consistently expressive across random restarts.

B. LUNA Details

B.1. Training Objective

As stated in section 4, we have adapted the training objective from (Ross et al., 2018). We use the cosine similarity function on the gradients of the auxiliary regressors:

$$\begin{aligned}&\text{CosSim}^2\left(\nabla_{\mathbf{x}}f_i(\mathbf{x}),\nabla_{\mathbf{x}}f_j(\mathbf{x})\right) =\\ &\qquad\frac{\left(\nabla_{\mathbf{x}}f_i(\mathbf{x})^T\nabla_{\mathbf{x}}f_j(\mathbf{x})\right)^2}{\left(\nabla_{\mathbf{x}}f_i(\mathbf{x})^T\nabla_{\mathbf{x}}f_i(\mathbf{x})\right)\left(\nabla_{\mathbf{x}}f_j(\mathbf{x})^T\nabla_{\mathbf{x}}f_j(\mathbf{x})\right)}\end{aligned} \tag{4}$$

This acts as a measure of orthogonality, equal to one when the two inputs are parallel, and 0 when they are orthogonal. A higher penalty, λ, in the training objective penalizes parallel components, hence enforcing diversity.

In practice, we approximate these gradients using a finite differences approximation. This was done to save on computational cost. That is, we approximate gradients as:

$$\nabla_{\mathbf{x}} f_i(\mathbf{x}) \approx \frac{f_i(\mathbf{x}+\delta\mathbf{x}) - f_i(\mathbf{x})}{\delta\mathbf{x}} \tag{5}$$

for some small perturbation $\delta\mathbf{x}$. We sample perturbations according to $\delta\mathbf{x} \sim \mathcal{N}(0, \sigma^2)$, where we generally use $\sigma = 0.1$.

C. Experimental Setup

C.1. Synthetic Data

Cubic Gap Example Following the set-up in (Ober & Rasmussen, 2019), we construct a synthetic 1-D dataset comprising 100 train and 100 test pairs (x, y), where x is sampled uniformly in the range $[-4, -2] \cup [2, 4]$ and y is generated as $y = x^3 + \epsilon, \epsilon \sim \mathcal{N}(0, 3^2)$.

Squiggle Gap Example We construct a synthetic 1-D dataset where train x is uniformly sampled from the range $[-4, -2] \cup [2, 4]$ and $y = x^3 + 20\exp(-x^2) \cdot \sin(10x) + \epsilon, \epsilon \sim \mathcal{N}(0, 3^2)$. As seen in Figure 5, this function is like the Cubic Gap Example, but with unexpected variations in the gap. For the generalization experiment, test x is sampled from the same range. For the transfer learning experiment, test x is sampled from the range $[-2, 2]$, inside the gap.

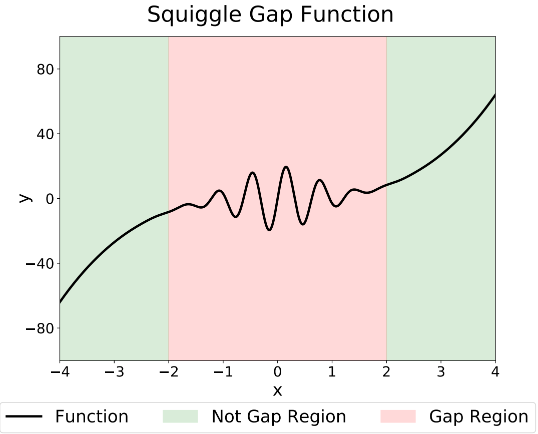


Figure 5. The squiggle gap function $y = x^3 + 20\exp(-x^2) \cdot \sin(10x)$.

BayesOpt Example We maximize the function $f(x) = \sin(0.1x)/0.1x$ in the range $[-100, 100]$, where the true maxima lies at $x = 0$. The samples from the function have added noise $\epsilon \sim \mathcal{N}(0, 10^{-8})$. Each model was run 10 times to get variance across runs.

C.2. Real Data

We used 3 standard UCI (Dua & Graff, 2017) regression data and modify them to create 5 "gap data-sets", wherein we purposefully created a gap in the data where we can test our model's in-between uncertainty (i.e. we train our model on the non-gap data and test the model's epistemic uncertainty on the gap data). We adapt the procedure from (Foong et al., 2019) to convert these UCI data sets into UCI gap data sets. For a selected input dimension, we (1) sort the data in increasing order in that dimension, and (2) remove middle $1/3$ to create a gap. We specifically selected input dimensions that have high correlation with the output in order to ensure that the learned model should have epistemic uncertainty in the gap; that is, if we select a dimension that is not useful for prediction, any model need not have increased uncertainty in the gap. The features we selected are:

- Boston Housing: "Rooms per Dwelling" (RM) and "Percentage Lower Status of the Population" (LSTAT)
- Concrete Compressive Strength: "Cement" and "Superplasticizer"
- Yacht Hydrodynamics: "Froude Number"

C.3. Training

Optimization We used the Adam optimizer (Kingma & Ba, 2014) with a learning rate of 10^{-3} in all experiments. We use 10000 iterations for the cubic and squiggle gap data experiments, 2000 iterations for the BayesOpt experiment and 20000 iterations for the real data experiments.

Architecture For the cubic gap example, all neural networks are 2-layer 50 hidden units ReLU networks (except for the full BNN, in which a 1-layer network was used to reduce computational cost). The LUNA model uses $M = 50$ auxiliary functions. The ensemble uses 50 independent networks and we use the mean and variance of the outputs as the predictive mean and uncertainty. For each network in the ensemble, we train with bootstraps of the training data. The GP model uses the sum of a Matern-$5/2$ kernel with length scale 1.0 and a white kernel with ground truth noise variance $\sigma^2 = 3^2$.

For the squiggle gap example, we use 2-layer ReLU networks with 50 units in the first hidden layer and variable units in the second hidden layer based on the experiment. The LUNA model uses $M = 50$ auxiliary functions.

In both these examples, the MAP model is simply the maximum a posteriori trained neural network with ground truth noise variance σ^2 added as the model variance.

For the BayesOpt example, the neural networks have 2 hidden layers with 10 units in each layer. They use the RBF activation with parameter $\gamma = 0.01$. The LUNA model uses $M = 20$ auxiliary functions. The GP model uses the sum of an RBF kernel with length scale 10.0 and a white kernel with ground truth noise variance $\sigma^2 = 10^{-8}$. The random guessing baseline just random uniformly samples the input domain every iteration.

C.4. Hyperparameters

We tuned hyperparameters for all models using 20% of the training data as a held-out validation set.

Synthetic Data The ground truth noise variance σ^2 was used in all models.

For the Neural Linear model (NLM), the regularization hyperparameter γ was selected by maximizing validation log-likelihood using 50 iterations of Bayesian optimization over the range $\gamma \in [10^{-3}, 10^3]$, initialized with 10 iterations of random search. The chosen value was $\gamma = 8.37$.

For the LUNA model, the regularization hyperparameter γ and the diversity hyperparameter λ were selected using grid search over $\gamma \in \{10^{-3}, 10^{-2}, 10^{-1}, 10^0, 10^1, 10^2, 10^3\}$ and $\lambda \in \{10^{-3}, 10^{-2}, 10^{-1}, 10^0, 10^1, 10^2, 10^3\}$. We first set a minimum threshold and reject all models with validation log-likelihoods below this threshold. In this case, our threshold was simply the average of all validation log-likelihoods of the grid-search models. We then select the model that has the smallest diversity error $\mathcal{L}_{\text{DIVERSE}}(\Psi)$ (see Section 4). This selection procedure ensures we select sufficiently expressive basis features while maintaining reasonable performance on the training data. The chosen values were $\gamma = 10^{-1}$ and $\lambda = 10^0$.

Since the NLM and LUNA have the same inference architecture, a reasonable value for the prior variance $\alpha = 1.0$ was used for consistency.

For the Monte Carlo Dropout (MCD) model, the dropout rate p was selected by maximizing validation log-likelihood using 50 iterations of Bayesian optimization over the range $p \in [10^{-3}, 0.5]$, initialized with 10 iterations of random search. The chosen value was $p = 0.00225$. The model precision τ was set to the inverse of the ground truth noise variance σ^2. A reasonable 1000 forward passes were used to obtain model uncertainty.

Real Data We first fit a maximum a posteriori (MAP) model to the data sets and use the variance of the output errors as the noise variance σ^2 in all models.

For the NLM, the regularization hyperparameter γ was selected by maximizing validation log-likelihood using grid search over $\gamma \in \{10^{-2}, 10^{-1}, 10^0, 10^1, 10^2\}$.

For the LUNA model, the regularization hyperparameter γ and the diversity hyperparameter λ were selected using grid search over $\gamma \in \{10^{-2}, 10^{-1}, 10^0, 10^1, 10^2\}$ and $\lambda \in \{10^{-3}, 10^{-2}, 10^{-1}, 10^0, 10^1, 10^2, 10^3\}$. We first set a minimum threshold and reject all models with validation log-likelihoods below this threshold. In this case, our threshold was simply the average of all validation log-likelihoods of the grid-search models. We then select the model that has the smallest diversity error $\mathcal{L}_{\text{DIVERSE}}(\Psi)$. This selection procedure ensures we select sufficiently expressive basis features while maintaining reasonable performance on the training data.

Since the NLM and LUNA have the same inference architecture, a reasonable value for the prior variance $\alpha = 10.0$ was used for consistency.

For the Monte Carlo Dropout (MCD) model, we follow (Gal & Ghahramani, 2016) and set the dropout rate $p = 0.05$. Since the other models used the same learned noise variance σ^2, we set the model precision τ to the inverse of this σ^2. A reasonable 1000 forward passes were used to obtain model uncertainty.

The selected hyperparameters are shown in Table 1.

D. Experimental Results

D.1. Toy Problem

We show NLM prior predictive samples and posterior predictive distributions, with and without regularization, across random restarts in Figure 6. With regularization, NLM is unable to get expressive prior predictive samples and hence increased in-between posterior predictive uncertainty. While this can be achieved with no regularization, it is inconsistent in doing so. A good initialization is needed, which is currently impossible to select a priori. For high dimensional cases, since we cannot plot the distributions, it is impossible to select these few good initialization out of random restarts. In a real case, this issue is further exacerbated because we do not know where the gaps are. We have shown in Section 3 that we cannot use log-likelihood to judge this either.

We also show the posterior predictive distributions for NLMs with different numbers of basis features across random restarts in Figure 7, and with different numbers of hidden layers across random restarts in Figure 8. We see that increasing number of features is more important than increasing depth for improving posterior predictives.

Finally, we show NLM prior predictive samples and posterior predictive distributions for marginal data likelihood training, described in Appendix A.2. This is shown for different regularization and prior variances across random restarts in Figures 9 and 10. We see the same trends here as what was observed above with traditional MAP training. With high regularization (high γ or low α), the feature bases for the learned θ do not span diverse functions in the prior predictive and hence cannot capture in-between uncertainty. With low regularization (low γ or high α), the model has the potential to capture in-between uncertainty, but it does so inconsistently across random restarts and needs to rely on good initialization.

D.2. Generalization

We show the full result from the generalization experiment (see Section 5) in Figure 11. All models, except MCD, converge to good generalization ability when a enough features are used. While MCD has relatively good performance with low feature numbers, there is not much of an improvement as more features are used. We believe this is worthy of further investigation.

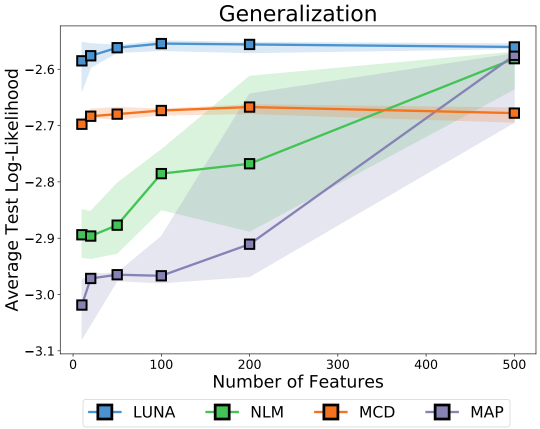


Figure 11. With any number of features, LUNA outperforms other models on test data sampled from the same distribution as train data. While the other models converge to LUNA's performance with large number of features, this does not happen with MCD.

D.3. BayesOpt Downstream Task

We show the results from the BayesOpt experiment (see Appendix B.1) in Figure 12. We see that LUNA shows the fastest convergence and has smaller variance over 10 runs than NLM.

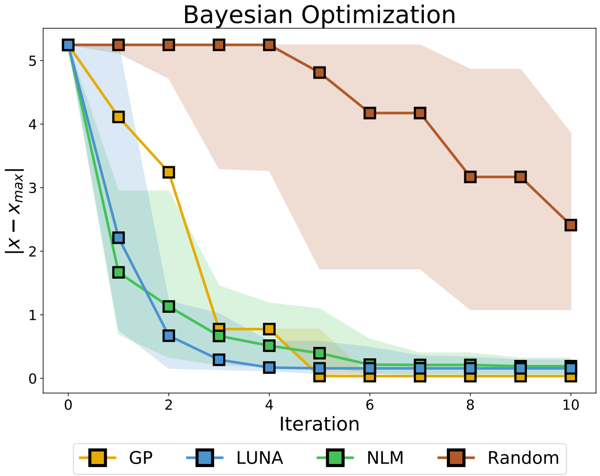


Figure 12. Bayesian Optimization using LUNA and baselines. LUNA converges the fastest and has much lower variance than NLM across runs.

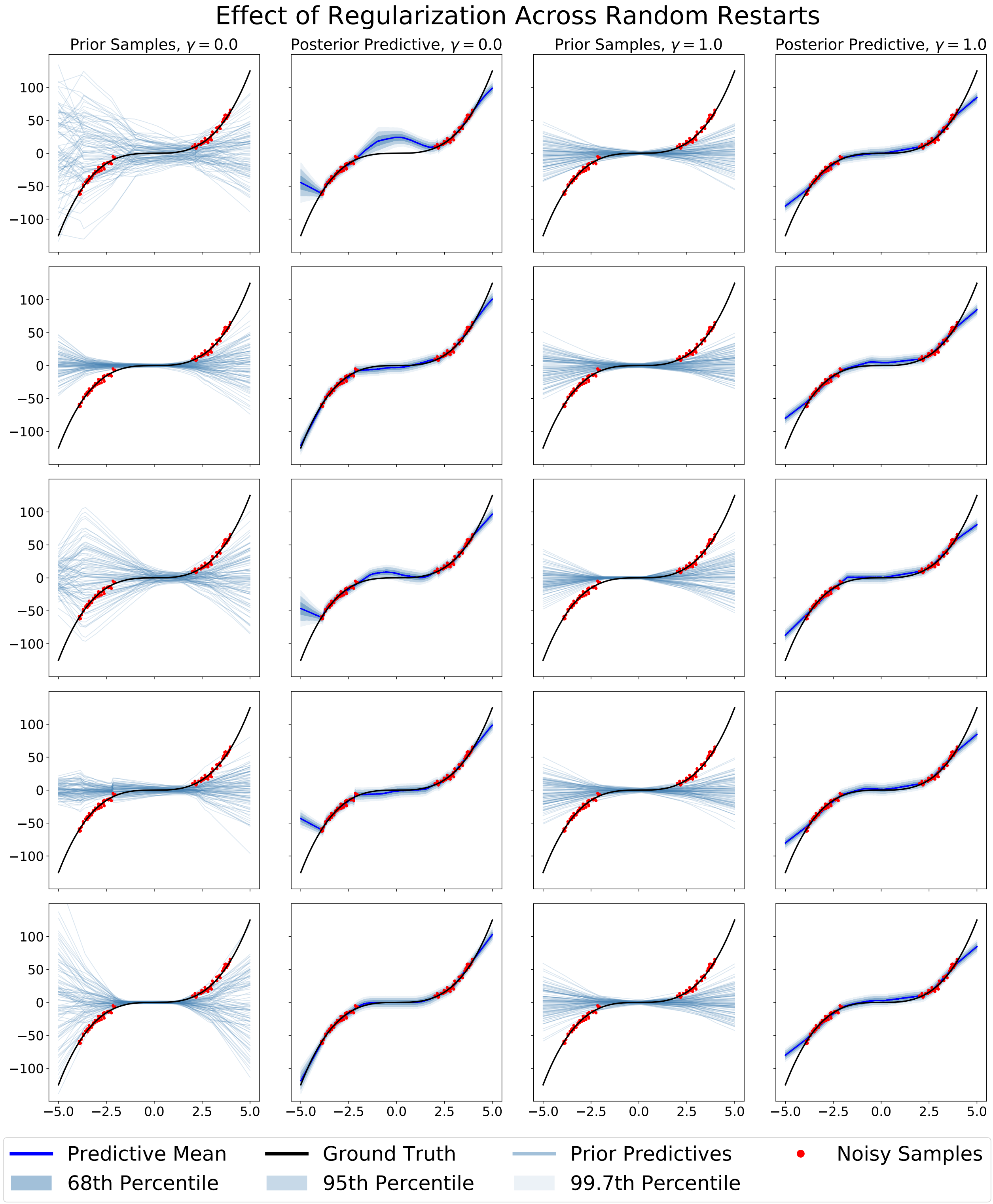


Figure 6. This experiment was run with a 2-layer ReLU network with 50 and 20 neurons in the first and second layers respectively (20 features). We used MAP training. With no regularization and very noisy priors, the NLM is able to model in-between uncertainty, albeit inconsistently. With regularization, we see the priors are not expressive enough and the NLM fails to ever capture in-between uncertainty.

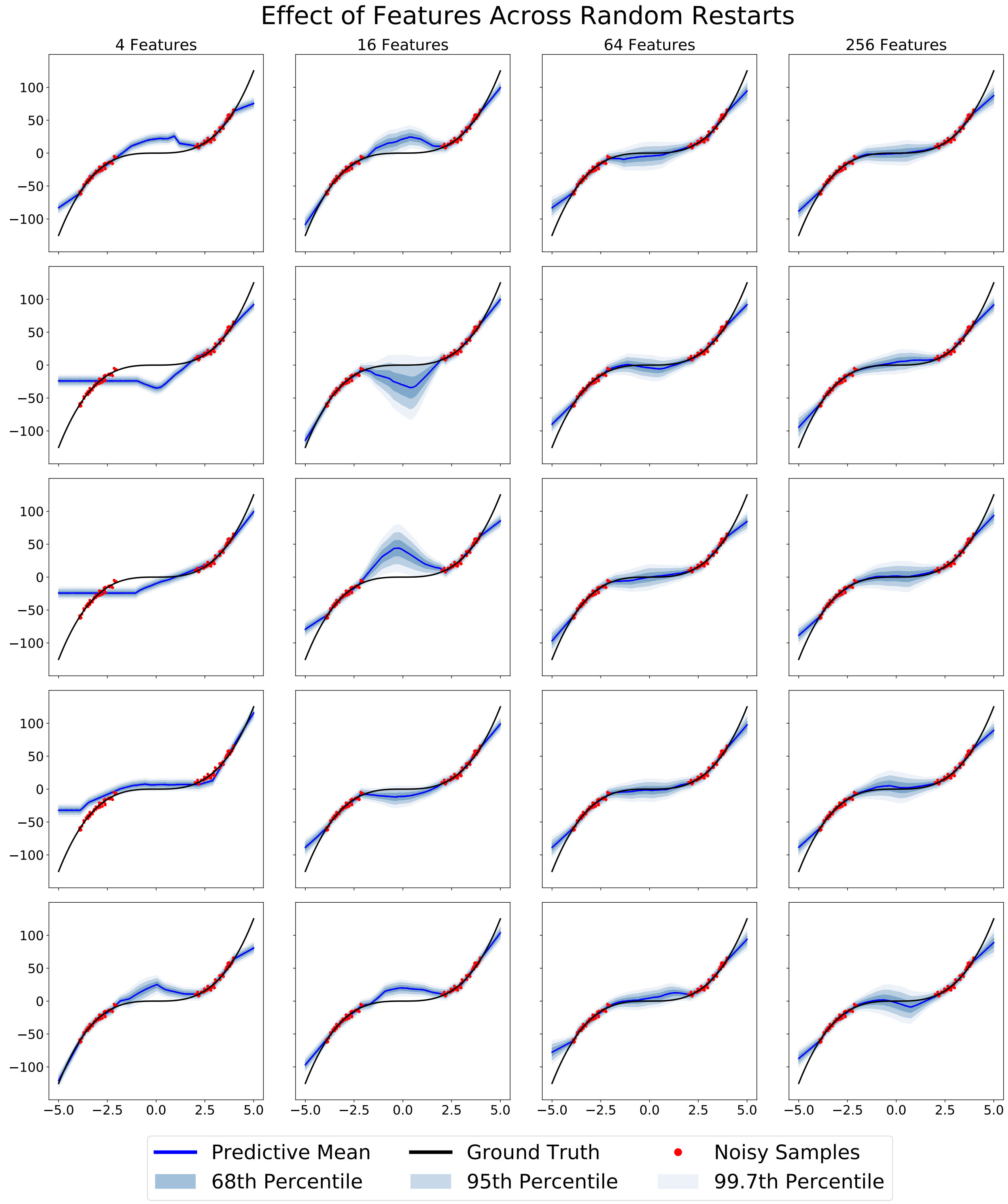


Figure 7. This experiment was run with a 2-layer ReLU network with 50 neurons in the first layer and without regularization. We used MAP training. The number of neurons in the second layer correspond to the number of features. We see clearly as model capacity increases NLM better fits the data. However, this increased capacity still fails to consistently model in-between uncertainty.

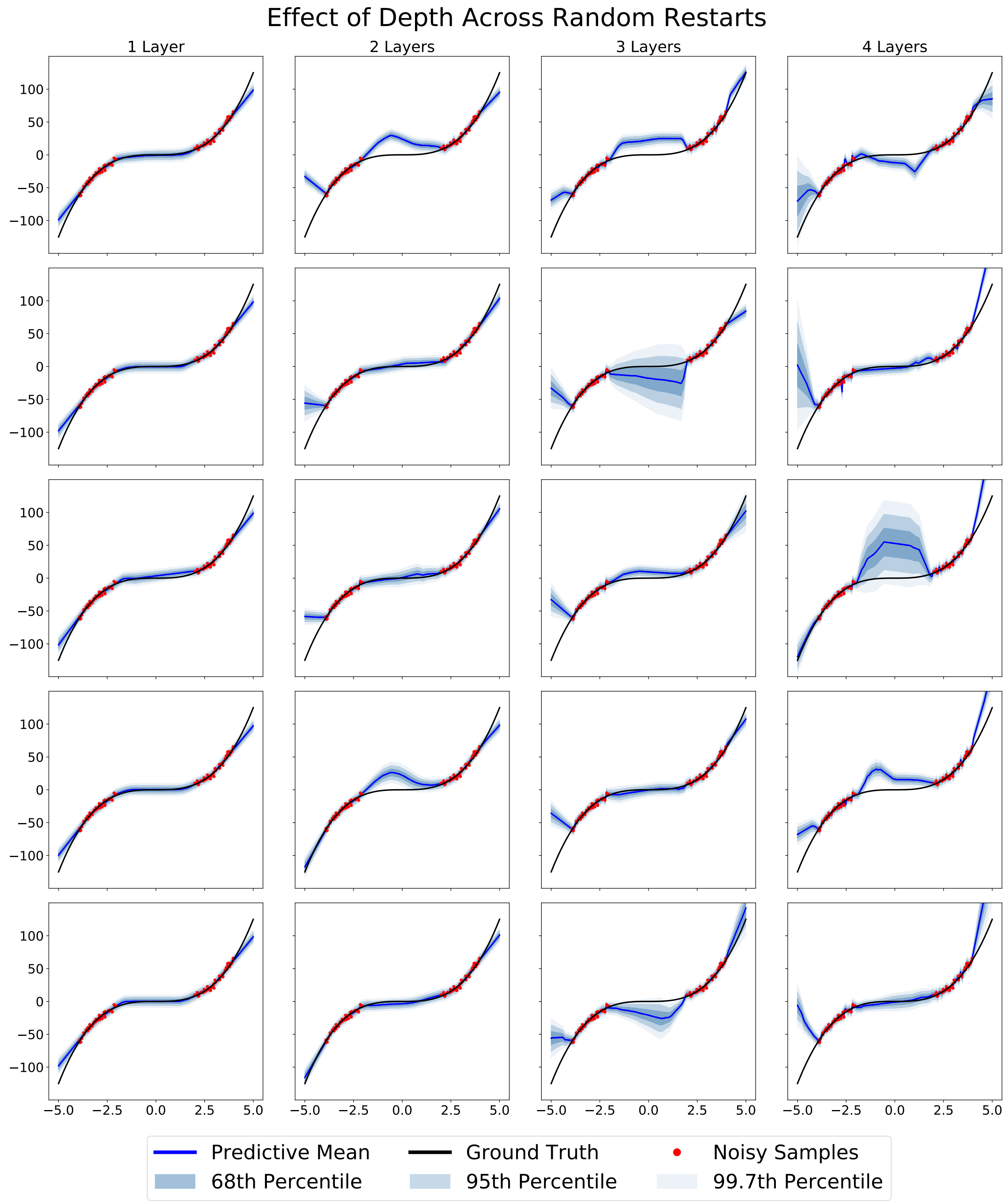


Figure 8. This experiment was run with a 2-layer ReLU network with 50 and 20 neurons in the first and second layers respectively (20 features). We used MAP training. We see that NLM is able to capture more complex relationships as capacity increases, but this increased capacity does not lead to consistent in-between uncertainty. Additionally, these added layers make NLM susceptible to overfitting.

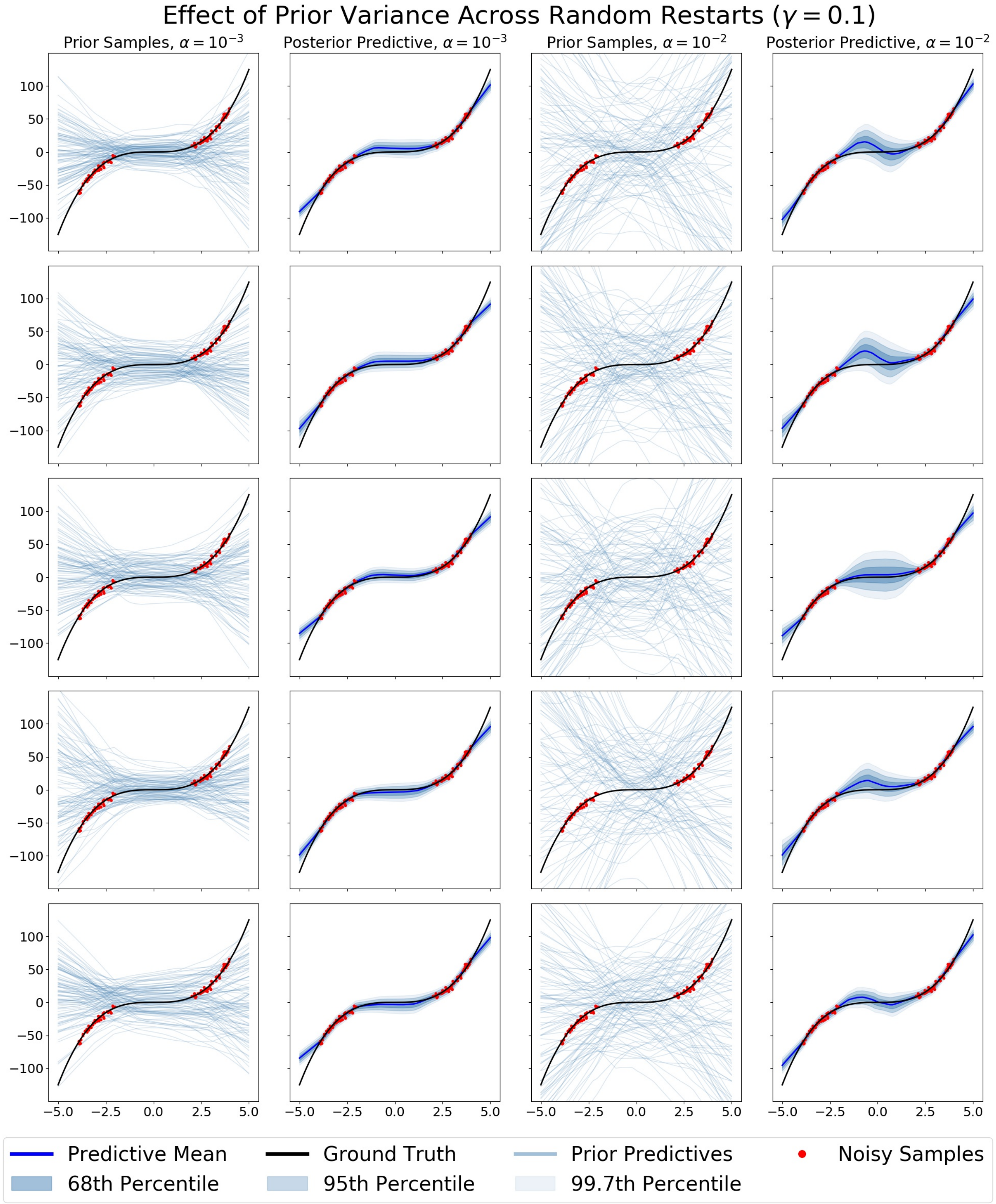


Figure 9. This experiment was run with a 2-layer ReLU network with 50 and 20 neurons in the first and second layers respectively (20 features). We used marginal likelihood training and a smaller $\gamma = 0.1$. We see that this NLM is able to capture higher in-between uncertainty when α is high enough, but is inconsistent in doing so.

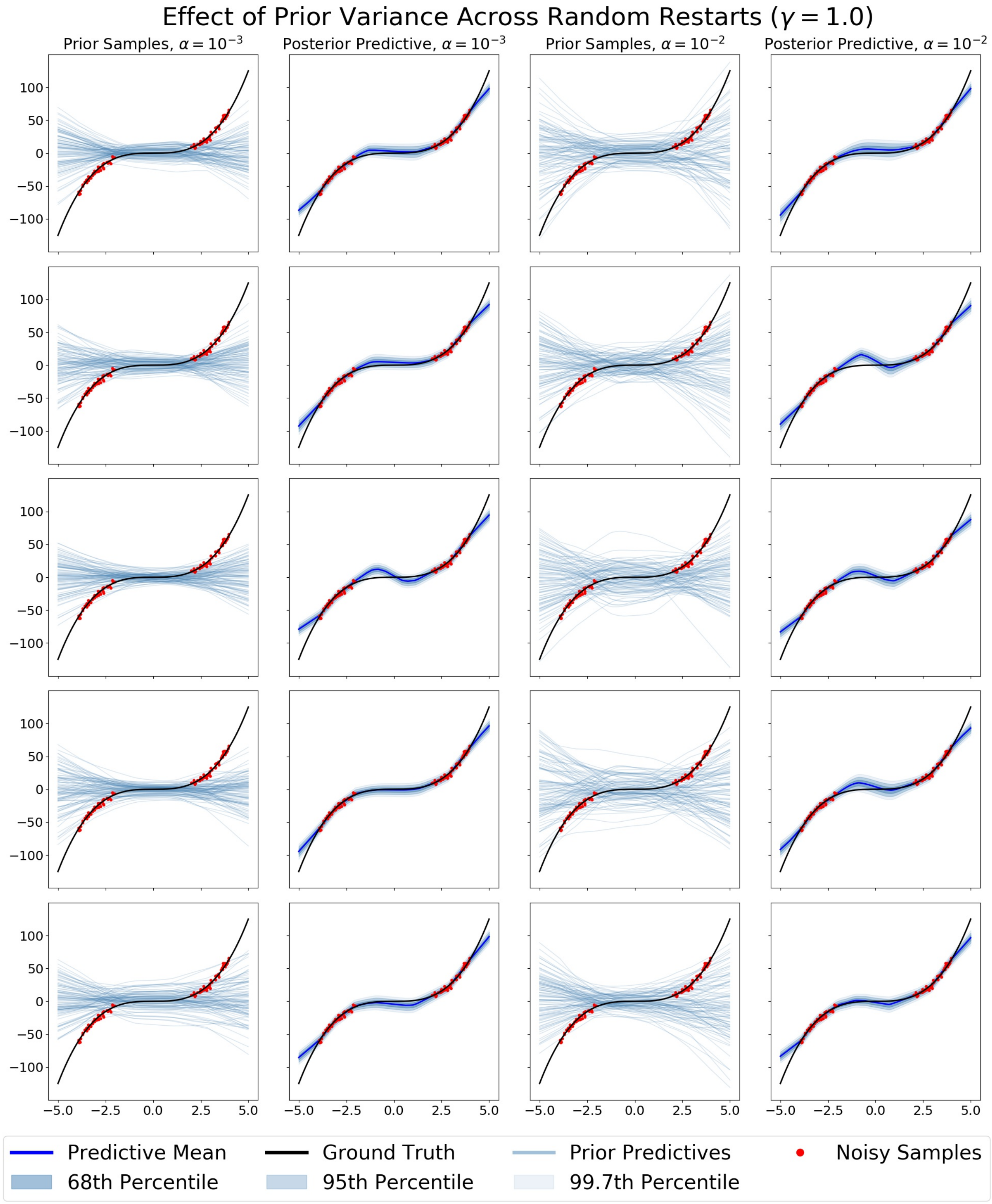


Figure 10. This experiment was run with a 2-layer ReLU network with 50 and 20 neurons in the first and second layers respectively (20 features). We used marginal likelihood training and a larger $\gamma = 1.0$. We see that this NLM is unable to capture higher in-between uncertainty even when α is high.

D.4. UCI Gap

We visualize test log-likelihoods and epistemic uncertainties inside and outside the gap for the Yacht "Froude" data set in Figure 13. Outlier removal was done, where an outlier is defined as having modified z-score greater than 3.5 (Iglewicz & Hoaglin, 1993). Outliers were calculated independently for each metric. In total, 1.33% of data points were treated as outliers and removed. Exact outlier details are shown in Table 4. Note that NLM shows catastrophic gap likelihood failure without significant gap epistemic uncertainty increase, while MCD fails to capture higher gap epistemic uncertainty. LUNA is able to capture this higher gap epistemic uncertainty while maintaining good gap log-likelihood.

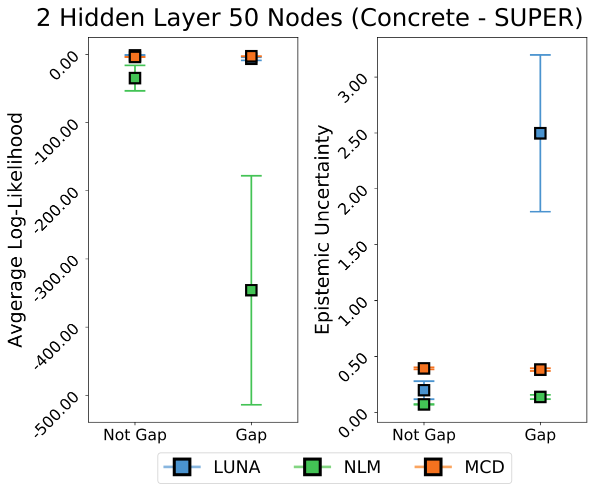


Figure 13. Average test log-likelihoods (left) and epistemic uncertainties (right) outside and inside the gap for Monte Carlo Dropout (MCD), Neural Linear model (NLM) and LUNA model, with a gap created in the "Froude Number" feature for the Yacht UCI dataset. We see that LUNA expresses higher epistemic uncertainty in the gap while maintaining good log-likelihood.

We also present tabulated values for experiments run using multiple architectures across many more features of the UCI gap data sets. The metrics reported are root mean squared error (RMSE), average log-likelihood (LL), and epistemic uncertainty (standard deviation). These are seen below in Table 2 for two layer models and Table 3 for one layer models.

	Yacht - Froude			Boston - RM			Boston - LSTAT			Concrete - CEMENT			Concrete - SUPER		
	LUNA		NLM	LUNA		NLM	LUNA		NLM	LUNA		NLM	LUNA		NLM
	γ	λ	γ	γ	λ	γ	γ	λ	γ	γ	λ	γ	γ	λ	γ
1 Layer	0.1	10.0	100.0	0.1	0.001	100.0	10.0	10.0	100.0	1.0	1.0	0.01	1.0	0.01	100.0
2 Layer	0.01	0.001	100.0	0.01	10.0	1.0	0.01	10.0	100.0	100.0	10.0	1.0	0.01	0.1	100.0

Table 1. Selected hyperparameters for all models across the different data sets.

Root Mean Square Error

	Yacht - Froude			Boston - RM			Boston - LSTAT			Concrete - CEMENT			Concrete - SUPER		
	LUNA	NLM	MCD	LUNA	NLM	MCD	LUNA	NLM	MCD	LUNA	NLM	MCD	LUNA	NLM	MCD
Not Gap	0.25 ± 0.10	2.01 ± 0.45	1.38 ± 0.04	2.39 ± 2.06	7.09 ± 1.57	1.17 ± 0.03	1.76 ± 0.91	5.17 ± 1.38	1.35 ± 0.03	1.11 ± 0.54	2.57 ± 0.79	1.28 ± 0.03	0.49 ± 0.09	1.88 ± 0.49	1.21 ± 0.02
Gap	**0.38** ± **0.17**	2.98 ± 0.43	1.08 ± 0.04	2.32 ± 1.28	5.61 ± 1.12	**0.90** ± **0.03**	2.19 ± 1.16	5.30 ± 1.41	**0.78** ± **0.03**	1.54 ± 0.43	5.38 ± 0.86	**1.07** ± **0.02**	2.77 ± 0.79	5.92 ± 1.59	**1.01** ± **0.02**

Avg. Log-Likelihood

	Yacht - Froude			Boston - RM			Boston - LSTAT			Concrete - CEMENT			Concrete - SUPER		
	LUNA	NLM	MCD	LUNA	NLM	MCD	LUNA	NLM	MCD	LUNA	NLM	MCD	LUNA	NLM	MCD
Not Gap	-0.09 ± 0.41	-212 ± 104	-12.93 ± 1.71	-4.10 ± 2.06	-524 ± 215	-4.99 ± 0.53	-2.72 ± 1.29	-145 ± 73	-5.40 ± 0.37	-2.70 ± 2.15	-66.01 ± 36.17	-3.78 ± 0.13	-1.25 ± 0.70	-34.54 ± 18.71	-3.53 ± 0.22
Gap	**-1.39** ± **2.04**	-458 ± 134	-2.67 ± 0.36	-4.58 ± 2.80	-325 ± 117	**-2.33** ± **0.14**	-3.97 ± 1.68	-155 ± 73	**-1.50** ± **0.12**	-4.59 ± 0.96	-275 ± 94	**-2.88** ± **0.05**	-7.01 ± 3.18	-346 ± 168	**-2.49** ± **0.13**

Epistemic Uncertainty (STD)

	Yacht - Froude			Boston - RM			Boston - LSTAT			Concrete - CEMENT			Concrete - SUPER		
	LUNA	NLM	MCD	LUNA	NLM	MCD	LUNA	NLM	MCD	LUNA	NLM	MCD	LUNA	NLM	MCD
Not Gap	0.57 ± 0.40	0.07 ± 0.00	0.39 ± 0.02	1.44 ± 0.41	0.19 ± 0.02	0.31 ± 0.01	2.53 ± 0.65	0.22 ± 0.03	0.30 ± 0.01	0.42 ± 0.10	0.08 ± 0.00	0.39 ± 0.01	0.20 ± 0.08	0.07 ± 0.00	0.39 ± 0.01
Gap	**0.72** ± **0.60**	0.09 ± 0.01	0.38 ± 0.01	**2.21** ± **0.86**	0.16 ± 0.02	0.33 ± 0.01	**2.87** ± **0.76**	0.23 ± 0.04	0.31 ± 0.01	**1.81** ± **0.77**	0.12 ± 0.01	0.37 ± 0.01	**2.50** ± **0.70**	0.14 ± 0.02	0.38 ± 0.01

Table 2. RMSE, average log-likelihood and epistemic uncertainty for 2-layer 50 hidden units ReLU networks. The metrics are computed both inside and outside the gap. We see that MCD and NLM consistently fail to capture in-between uncertainty in the gap across all UCI gap data sets explored. LUNA captures the expected increased uncertainty without showing catastrophic RMSE and log-likelihood failure in the gaps.

Root Mean Square Error

	Yacht - Froude			Boston - RM			Boston - LSTAT		
	LUNA	NLM	MCD	LUNA	NLM	MCD	LUNA	NLM	MCD
Not Gap	0.06 ± 0.01	0.04 ± 0.01	1.38 ± 0.05	0.53 ± 0.06	0.35 ± 0.01	1.24 ± 0.10	0.59 ± 0.07	0.44 ± 0.02	1.48 ± 0.04
Gap	0.11 ± 0.01	**0.08 ± 0.01**	1.08 ± 0.04	0.54 ± 0.07	**0.37 ± 0.01**	0.99 ± 0.03	0.63 ± 0.09	**0.39 ± 0.01**	0.82 ± 0.03

	Concrete - CEMENT			Concrete - SUPER		
	LUNA	NLM	MCD	LUNA	NLM	MCD
Not Gap	0.41 ± 0.05	0.32 ± 0.01	1.20 ± 0.02	0.49 ± 0.09	1.88 ± 0.49	1.21 ± 0.02
Gap	0.63 ± 0.13	**0.40 ± 0.02**	1.02 ± 0.02	2.77 ± 0.79	5.92 ± 1.59	**1.01 ± 0.02**

Avg. Log-Likelihood

	Yacht - Froude			Boston - RM			Boston - LSTAT		
	LUNA	NLM	MCD	LUNA	NLM	MCD	LUNA	NLM	MCD
Not Gap	1.21 ± 0.05	1.34 ± 0.02	-10.15 ± 1.65	-1.69 ± 0.47	-0.65 ± 0.05	-4.53 ± 0.36	-0.98 ± 0.13	-0.79 ± 0.07	-4.41 ± 0.35
Gap	0.78 ± 0.17	**1.09 ± 0.07**	-3.40 ± 0.40	-1.70 ± 0.57	**-0.85 ± 0.07**	-3.21 ± 0.21	-1.52 ± 0.44	**-0.60 ± 0.04**	-1.99 ± 0.09

	Concrete - CEMENT			Concrete - SUPER		
	LUNA	NLM	MCD	LUNA	NLM	MCD
Not Gap	-0.88 ± 0.29	-0.44 ± 0.08	-4.49 ± 0.20	-1.25 ± 0.70	-34.54 ± 18.71	-3.53 ± 0.22
Gap	-3.13 ± 1.55	**-0.94 ± 0.15**	-4.39 ± 0.16	-7.01 ± 3.18	-346 ± 168	**-2.49 ± 0.13**

Epistemic Uncertainty (STD)

	Yacht - Froude			Boston - RM			Boston - LSTAT		
	LUNA	NLM	MCD	LUNA	NLM	MCD	LUNA	NLM	MCD
Not Gap	0.05 ± 0.01	0.03 ± 0.00	0.43 ± 0.01	0.10 ± 0.02	0.08 ± 0.00	0.42 ± 0.01	0.19 ± 0.03	0.09 ± 0.01	0.47 ± 0.01
Gap	0.05 ± 0.00	0.04 ± 0.00	**0.37 ± 0.01**	0.12 ± 0.02	0.07 ± 0.00	**0.32 ± 0.01**	0.17 ± 0.03	0.08 ± 0.00	**0.29 ± 0.01**

	Concrete - CEMENT			Concrete - SUPER		
	LUNA	NLM	MCD	LUNA	NLM	MCD
Not Gap	0.06 ± 0.00	0.07 ± 0.00	0.37 ± 0.01	0.20 ± 0.08	0.07 ± 0.00	0.39 ± 0.01
Gap	0.07 ± 0.01	0.07 ± 0.00	**0.27 ± 0.01**	**2.50 ± 0.70**	0.14 ± 0.02	0.38 ± 0.01

Table 3. RMSE, average log-likelihood and epistemic uncertainty for 1-layer 50 hidden units ReLU networks. The metrics are computed both inside and outside the gap. We see that without the added capacity of a second layer, all models fail to capture any significant in-between uncertainty on the scale of what was observed in 2-layer networks in Table 2.

Outliers

Model	Layers	Data Feature	Metric	Gap Outliers	Not Gap Outliers
LUNA	1	Yacht - FROUDE	Avg. Log-Likelihood	1	0
LUNA	1	Boston - RM	Avg. Log-Likelihood	0	1
LUNA	1	Concrete - CEMENT	RMSE	2	0
LUNA	1	Concrete - CEMENT	Avg. Log-Likelihood	2	0
LUNA	1	Concrete - SUPER	Avg. Log-Likelihood	1	0
LUNA	2	Yacht - FROUDE	RMSE	1	1
LUNA	2	Yacht - FROUDE	Avg. Log-Likelihood	1	0
LUNA	2	Yacht - FROUDE	Epistemic Uncertainty	2	0
LUNA	2	Boston - RM	RMSE	0	1
LUNA	2	Boston - RM	Avg. Log-Likelihood	0	1
LUNA	2	Boston - LSTAT	Avg. Log-Likelihood	1	0
LUNA	2	Boston - LSTAT	Epistemic Uncertainty	2	0
LUNA	2	Concrete - CEMENT	RMSE	0	1
LUNA	2	Concrete - CEMENT	Avg. Log-Likelihood	0	2
LUNA	2	Concrete - SUPER	Avg. Log-Likelihood	1	0
NLM	1	Yacht - Froude	Epistemic Uncertainty	0	1
NLM	1	Boston - LSTAT	RMSE	1	0
NLM	1	Boston - LSTAT	Avg. Log-Likelihood	1	0
NLM	1	Boston - RM	Epistemic Uncertainty	0	1
NLM	2	Yacht - FROUDE	Avg. Log-Likelihood	0	2
MCD	1	Boston - RM	RMSE	0	1
MCD	1	Boston - LSTAT	Epistemic Uncertainty	0	1
MCD	2	Boston - LSTAT	Epistemic Uncertainty	1	1
MCD	2	Concrete - CEMENT	Epistemic Uncertainty	0	1

Table 4. The tabulated number of outlier points removed for each metric calculation.