Output-Constrained Bayesian Neural Networks

Wanqian Yang [*1] **Lars Lorch** [*1] **Moritz A. Graule** [*1] **Srivatsan Srinivasan** [1] **Anirudh Suresh** [1] **Jiayu Yao** [1]
Melanie F. Pradier [1] **Finale Doshi-Velez** [1]

Abstract

Bayesian neural network (BNN) priors are defined in parameter space, making it hard to encode prior knowledge expressed in function space. We formulate a prior that incorporates functional constraints about what the output can or cannot be in regions of the input space. Output-Constrained BNNs (OC-BNN) represent an interpretable approach of enforcing a range of constraints, fully consistent with the Bayesian framework and amenable to black-box inference. We demonstrate how OC-BNNs improve model robustness and prevent the prediction of infeasible outputs in two real-world applications of healthcare and robotics.

1. Introduction

BNNs combine powerful function approximators with the ability to model uncertainty, making them useful in domains where (i) training data is expensive or limited, or (ii) inaccurate predictions are prohibitively costly and decision-making must be informed by our level of confidence (MacKay, 1995; Neal, 1995). Domain experts often have prior knowledge about the modeled function and the ability to encode such information on top of training data can thus improve performance. However, BNNs define prior distributions over parameters, whose high dimensionality and lack of interpretability make the incorporation of functional beliefs close to impossible.

We present an interpretable approach for incorporating prior functional information into BNNs in the form of constraints, while staying consistent with the Bayesian framework. We then apply our method to two domains where the ability to encode such constraints is crucial: **(i)** prediction of clinical actions in health care, where constraints prevent unsafe actions for certain physiological inputs, and **(ii)** human motion prediction, where joint positions are constrained by anatomically feasible ranges.

Our contributions are: **(a)** we introduce *constraint priors*, capable of incorporating both *negative* constraints (where the function cannot be) and *positive* constraints (where the function should be), applicable with any black-box inference algorithm normally used with BNNs, and **(b)** we demonstrate the application of constraint priors with a variety of suitable inference methods on toy problems as well as two large and high-dimensional real-world data sets.

2. Related Work

Most closely related to our work, (Lorenzi & Filippone, 2018) considered function-space equality and inequality constraints of deep probabilistic models. However, they focused on deep Gaussian processes (DGPs) rather than BNNs, and on low-dimensional data from simulated ODE systems, whereas we consider high-dimensional real-world settings. They also do not consider classification settings.

(Hafner et al., 2018) specify a Gaussian function prior with the goal of preventing overconfident BNN predictions out-of-distribution. In contrast, we use ″positive constraints″ to guide the function where it should be. Also related are functional BNNs by (Sun et al., 2019), where variational inference is performed in function-space using a stochastic process model. Their view is more general—and accordingly, more complex to optimize—while we focus on constraints in specific regions of the input-output space.

3. Background

A conventional BNN, operating in the function (or input-output) space $\mathcal{X} \times \mathcal{Y}$, typically has a prior over parameters $p(\mathcal{W})$, where $\mathcal{W}$ are the neural network weights and biases. Given data $\mathcal{D} = \{\mathbf{x}_n, \mathbf{y}_n\}_{n=1}^N$, we

[*]Equal contribution [1]Harvard University. Correspondence to: Wanqian Yang <yangw@college.harvard.edu>, Lars Lorch <lars.lorch@gmail.com>.

Presented at the ICML 2019 Workshop on Uncertainty and Robustness in Deep Learning and Workshop on Understanding and Improving Generalization in Deep Learning. Long Beach, CA, 2019.

perform inference to obtain the posterior $p(\mathcal{W} \mid \mathcal{D}) \propto p(\mathcal{W})p(\mathcal{D} \mid \mathcal{W})$. The posterior predictive for the output $\mathbf{y}'$ for some new input $\mathbf{x}'$ is obtained by integrating over the posterior distribution of $\mathcal{W}$:

$$p(\mathbf{y}' \mid \mathbf{x}', \mathcal{D}) = \int_{\mathcal{W}} p(\mathbf{y}' \mid \mathbf{x}', \mathcal{W}) p(\mathcal{W} \mid \mathcal{D}) d\,\mathcal{W} \quad (1)$$

The space of $\mathcal{W}$ is high-dimensional and the relationship between the weights and the function is non-intuitive. As such, the prior $p(\mathcal{W})$ is often trivially chosen as an isotropic Gaussian:

$$p(\mathcal{W}) = \prod_i \mathcal{N}(\mathcal{W}_i; 0, \sigma_p^2) \quad (2)$$

4. Output-Constrained BNNs

We consider two kinds of "expert knowledge": *positive* constraints define regions where a function *should* be, and *negative* constraints define regions where a function *cannot* be. This delineation is not arbitrary — the level of prior knowledge (strongly vs. weakly informative) and the task (regression or classification) may suggest the use of different prior constraints.

Defining constrained regions Formally, a *positive* constrained region $\mathcal{C}^+$ is a set of input-output tuples $(\mathbf{x}, \mathbf{y})$ defining where outputs given certain inputs should be. Conversely, a *negative* constrained region $\mathcal{C}^-$ is a set of tuples $(\mathbf{x}, \mathbf{y})$ defining where outputs given certain inputs cannot be. We will use $\mathcal{C}$ when describing properties of constrained regions of both kinds and denote $\mathcal{C}_\mathbf{x}$ for all $\mathbf{x}$ in $\mathcal{C}$ and $\mathcal{C}_\mathbf{y}$ for all $\mathbf{y}$ in $\mathcal{C}$. Given this formulation, it is our goal to enforce

$$\begin{aligned} \int_{\mathcal{W}} p(\mathbf{y}' \notin \mathcal{C}_\mathbf{y}^+ \mid \mathbf{x}' \in \mathcal{C}_\mathbf{x}^+, \mathcal{W}) p(\mathcal{W} \mid \mathcal{C}^+, \mathcal{D}) d\,\mathcal{W} \approx 0 \\ \int_{\mathcal{W}} p(\mathbf{y}' \in \mathcal{C}_\mathbf{y}^- \mid \mathbf{x}' \in \mathcal{C}_\mathbf{x}^-, \mathcal{W}) p(\mathcal{W} \mid \mathcal{C}^-, \mathcal{D}) d\,\mathcal{W} \approx 0 \end{aligned} \quad (3)$$

Note that (3) is simply the posterior predictive distribution conditioned on $\mathcal{C}$. The generality of this approach allows for the incorporation of very complicated yet interpretable constraints *a priori*, such as for example arbitrary equality, inequality and logical (if-then and either-or) constraints.

Constraint prior We connect the weight space of the BNN with constraints through the distribution:

$$g(W \mid \mathcal{C}) = g(\phi(\mathcal{C}_\mathbf{x}; \mathcal{W}); \mathcal{C}_\mathbf{y}, \theta) \quad (4)$$

where $\phi(\mathbf{x}; \mathcal{W})$ is the BNN forward pass and θ is the set of tuneable hyperparameters of g. Accordingly, a *constraint prior* $p_\mathcal{C}(\mathcal{W})$ can then be constructed as:

$$p_\mathcal{C}(\mathcal{W}) := p(\mathcal{W})\, g(\mathcal{W} \mid \mathcal{C}), \quad (5)$$

achieving the goal of expressing prior function knowledge in weight space while retaining the weight-space prior $p(\mathcal{W})$. Intuitively, $g(\mathcal{W} \mid \mathcal{C})$ measures the BNN's adherence to the constrained region.

It remains to describe how g is defined. For positive constraints $\mathcal{C}^+$, g measures how close $\phi(\mathcal{C}_\mathbf{x}^+; \mathcal{W})$ lies to $\mathcal{C}_\mathbf{y}$, for which natural choices of distributions exist for both regression and classification. For negative constraints $\mathcal{C}^-$, we define g as the expected violation of $\mathcal{C}_\mathbf{y}^-$ given $\phi(\mathcal{C}_\mathbf{x}^-; \mathcal{W})$ using a classifier function. Complete definitions of g for positive and negative priors are provided in Appendix A; details on inference procedures are provided in Appendix B.

5. Demonstrations on Synthetic Data

This section provides proof of concepts of OC-BNNs using 2-dimensional synthetic examples. Refer to Appendix C for experimental details and Appendix D for additional results. For regression, the posteriors are visualized in black/gray for baseline BNNs, and blue for OC-BNNs. Negative constrained regions are red; positive (Gaussian) constraints are green. For the classification example, the three classes are color-coded red, green and blue.

OC-BNNs model uncertainty in a manner that respects constrained regions while explaining training data. Figure 1 demonstrates this for both the regression and classification setting. Correct predictions are maintained with similar uncertainty levels as the baseline while constraints are correctly enforced with uncertainty levels changing to reflect that. These examples demonstrate how OC-BNNs enforce constraints without sacrificing predictive accuracy.

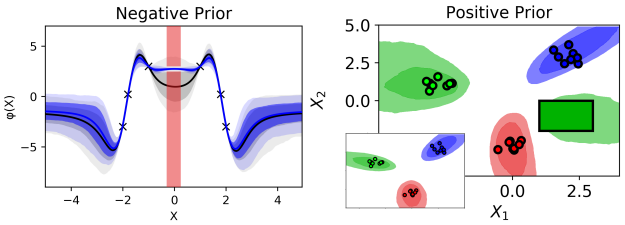

Figure 1. On both tasks, OC-BNNs reduce uncertainty in constrained regions while fitting data well. **(left)** 1D regression. The constraint is composed of two negative regions (red) separated by a small gap. Uncertainty of OC-BNNs (blue) drops sharply in the constrained region compared to the baseline (gray). **(right)** 2D classification with three classes. Constrained region enforces the prediction of green class in the green rectangle (baseline depicted in inset).

OC-BNNs encourage correct out-of-distribution behavior. Figure 2 (left) depicts sparse data, along with out-of-distribution positive constrained regions. The posterior predictive *in-distribution* closely mimics the

baseline, while the posterior *out-of-distribution* (OOD) learns to avoid the constrained region. This demonstrates that OC-BNNs function well away from the data, which is important because we typically want to enforce functional constraints when there is a lack of observed training data for the model to learn from.

OC-BNNs can capture posterior multimodality. While negative constraints $\mathcal{C}^-$ do not explicitly define multimodal posterior predictives, a bounded constrained region does imply that the posterior predictive might have probability mass on either side of the bounded region (i.e. for all d dimensions of $\mathcal{Y} = \mathbb{R}^d$). Figure 2 (right), demonstrates that we capture challenging posterior predictives.

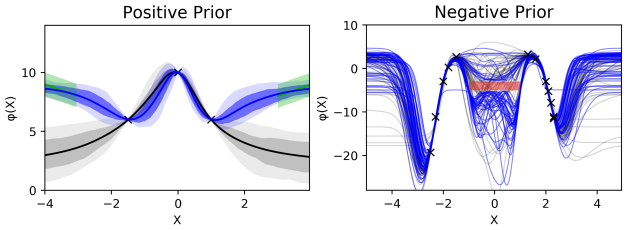

Figure 2. OC-BNNs capture important posterior qualities such as correct OOD behavior and multimodality. **(left)** OC-BNNs (blue) maintain the same in-distribution uncertainty as the baseline (gray) while adhering to OOD positive constraints (green) on either side of the plot. **(right)** OC-BNNs (blue) posterior samples go both below and above the negative constraint box (red).

6. Applications

6.1. Clinical action prediction

MIMIC-III (Johnson et al., 2016) is a benchmark database containing time series data of various physiological measurements and clinical actions prescribed belonging to $> 40,000$ intensive care patients who stayed at the Beth Israel Deaconess Medical Center between 2001 and 2012.

Problem Formulation From the raw time-series data, we construct a balanced dataset for a time-independent classification task of hypotension management. There are 9 features representing various physiological states, such as mean blood pressure and lactate levels. The goal is to predict if clinical action (either vasopressor or IV fluid) should be taken.

Constraints The constraint imposed is that for `mean blood pressure` less than 65 units, some action should be taken, which is physiologically realistic. We apply the positive (Dirichlet) constraint prior (Appendix A), as well as the weights-only prior baseline. In the given data, some training points fall within the constrained region. We train our model both with and without artificially filtering out all points within the positive constrained region.

		filtered		unfiltered	
		BNN	**OC-BNN**	**BNN**	**OC-BNN**
Train	ACC	0.745	0.741	0.881	0.878
	F1	0.805	0.801	0.882	0.880
	VIOL	0.151	0.149	N/A	N/A
Test	ACC	0.660	0.665	0.647	0.649
	F1	0.746	0.748	0.725	0.736
	VIOL	0.132	0.126	**0.117**	**0.039**

Table 1. Results for the MIMIC experiments with and without filtering out the points in the constrained region. Accuracy and F1 score remain unchanged when using OC-BNNs. For the experiment with filtration, the violation factor decreases by a factor of 3 when using OC-BNNs.

OC-BNNs maintain classification accuracy while reducing physiologically infeasible constraint violations. Table 1 displays experimental results, with statistics computed from the posterior mean. In addition to standard accuracy (ACC) and F1 score, we measure the violation fraction (VIOL), which is the fraction of predictions on held-out points that violate the constraints. The results show that OC-BNNs match standard BNNs on all predictive accuracy metrics, with significantly lower violation of the constrained region for the case where points originally in the constrained region are filtered out.

6.2. Human motion prediction

We evaluate OC-BNNs on data of humans conducting various motions available at (Kratzer, 2019) as described in (Kratzer et al., 2018). This data contains human upper body poses across many reaching tasks at a frame rate of 120Hz. The poses are provided in the form of upper body joint angles.

Problem formulation Given a subset of trajectories in (Kratzer, 2019), our goal is to predict joint angles 20 frames in the future from angles at the current time frame and the numerically computed joint velocities and accelerations. In the following, we limit ourselves to abduction and flexion (further denoted as Y- and Z-rotation to match the nomenclature in the original data (Kratzer, 2019)) of the left and right shoulder during right-handed reaching motions.

The joint angles in the test data were perturbed with normally distributed noise (μ=0, σ=2 degrees) to simulate a scenario in which a human motion prediction model is trained on data recorded in a high-end motion capture lab, and then used to predict motion from data obtained by noisy wearable sensors.

Constraints Several anatomical feasibility or functional range constraints for each of the joint angles could be applied, e.g. as described in (Namdari et al., 2012). We derived constraints on the joint limits from the reaching motions provided in (Kratzer, 2019) as the empirically observed extrema across all motions, which is modeled using the negative constraint prior.

OC-BNNs prevent infeasible predictions. We compare a BNN and OC-BNN using the negative prior and the empirical bounds on joint angles. Both models are compared in (i) RMSE using the posterior predictive mean (RMSE) [$1{\cdot}10^3$], (ii) held-out data log likelihood of $\mathcal{N}(\mu_{pp}, \sigma^2_{pp})$ with posterior predictive mean μ_{pp} and variance σ_{pp} (HO-LL), and (iii) posterior predictive violation defined as the percentage of probability mass in an infeasible constrained region (PP-VIOL) [%], each evaluated at all target points.

These metrics are summarized in Table 2. We find that OC-BNNs reduce the possibility of making an infeasible prediction to less than 0.001%, substantially improving on BNNs. Figure 3 shows exemplary motion predictions obtained with both BNN and OC-BNN for five consecutive points in a test trajectory.

		BNN	OC-BNN
Train	RMSE	0.929	1.252
	HO-LL	1718.409	1342.602
	PP-VIOL	**0.046**	**0.000**
Test	RMSE	7.320	12.127
	HO-LL	101.129	-683.697
	PP-VIOL	**18.447**	**0.000**

Table 2. Results for human motion prediction. While predictive performance and held-out log likelihood are similar, OC-BNNs (negative prior) reduce the chance of predicting an infeasible position to 0.0 % while BNNs make infeasible predictions in 18.4% of cases.

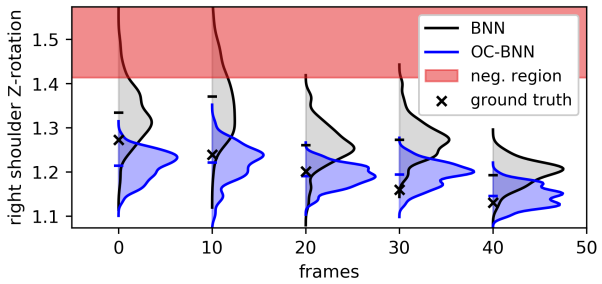

Figure 3. Consecutive predictions of right Z-rotation during an exemplary test trajectory with 10 frame gaps. The posterior predictive of BNN (black) and OC-BNN (blue) given the input state 20 frames earlier are plotted along the y-axis. While both BNN and OC-BNN do not perfectly generalize to the test set, the expert knowledge of valid joint positions enforces feasible and thus more robust predictions.

7. Discussion

OC-BNNs prevent constraint violation while fitting low- and high-dimensional data. Our results highlight that incorporating expert knowledge into OC-BNNs helps enforcing feasible and thus more robust predictions. Results for both datasets in Section 6 demonstrate that constraint violation metrics are reduced significantly, whereas accuracy metrics are nearly unchanged. This affirms the behavior observed in the synthetic examples in Section 5.

Training data in constrained region can outweigh prior effect. The clinical dataset results show that the presence of data in $\mathcal{C}$ reduces the effect of constraint priors. This is expected and in accordance with the Bayesian framework, where the likelihood effect will crowd out the prior given enough training data, and also suggests that the practitioner can use OC-BNNs even for situations where the constraints themselves may not be fully satisfied.

OC-BNNs can facilitate data imputation. The fact that OC-BNNs model uncertainty correctly in constrained regions without losing predictive accuracy, even for high-dimensional datasets, show that OC-BNNs can encode imputation in input regions without training data. Rather than directly modifying the training set through imputation, prior beliefs about missing data can instead be formulated as constraints.

When to use which prior? In the regression setting, negative priors are weakly informative whereas positive priors tend to be strongly informative – one or both of the prior types can be used depending on domain knowledge. While the negative prior formulation does not apply to classification cases, this does not pose a problem as negative and positive constraints are complements in discrete space.

8. Conclusion and Outlook

We describe OC-BNNs, a formulation to incorporate expert knowledge into BNNs by prescribing positive and negative (i.e., desired and forbidden) regions, and demonstrate their application to synthetic and real-world data. We show that OC-BNNs generally maintain the desirable properties of regular BNNs while their predictions follow the prescribed constraints. This makes them a promising tool for settings like healthcare, where models trained on sparse data may be augmented with expert knowledge. In addition, OC-BNNs may find applications in safe reinforcement learning, e.g. in tasks where certain actions are known to have catastrophic consequences.

Acknowledgements

MG and FDV acknowledge support from AFOSR FA 9550-17-1-0155. LL and WY acknowledge support from the John A. Paulson School of Engineering and Applied Sciences at Harvard University.

References

Hafner, D., Tran, D., Lillicrap, T., Irpan, A., and Davidson, J. Reliable uncertainty estimates in deep neural networks using noise contrastive priors. In *eprint arXiv:1807.09289*, 2018.

Johnson, A. E., Pollard, T. J., Shen, L., Li-wei, H. L., Feng, M., Ghassemi, M., Moody, B., Szolovits, P., Celi, L. A., and Mark, R. G. Mimic-iii, a freely accessible critical care database. *Scientific data*, 3:160035, 2016.

Kratzer, P. mocap-mlr-datasets. https://github.com/charlespwd/project-title, 2019.

Kratzer, P., Toussaint, M., and Mainprice, J. Towards combining motion optimization and data driven dynamical models for human motion prediction. In *2018 IEEE-RAS 18th International Conference on Humanoid Robots (Humanoids)*, pp. 202–208. IEEE, 2018.

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.

Lorenzi, M. and Filippone, M. Constraining the dynamics of deep probabilistic models. *arXiv preprint arXiv:1802.05680*, 2018.

MacKay, D. J. C. Probable networks and plausible predictions – a review of practical bayesian methods for supervised neural networks. In *Network: Computation in Neural Systems, 6:3, 469-505*, 1995.

Namdari, S., Yagnik, G., Ebaugh, D. D., Nagda, S., Ramsey, M. L., Williams Jr, G. R., and Mehta, S. Defining functional shoulder range of motion for activities of daily living. *Journal of shoulder and elbow surgery*, 21(9):1177–1183, 2012.

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

Neal, R. M. Mcmc using hamiltonian dynamics. In *Handbook of Markov Chain Monte Carlo*, 2012.

Sun, S., Zhang, G., Shi, J., and Grosse, R. Functional variational bayesian neural networks. *arXiv preprint arXiv:1903.05779*, 2019.

A. Constraint Priors

In this section, we describe the detailed functional forms of our positive and negative constraints and priors for both classification and regression settings, noting aspects important for inference.

A.1. Positive constraint prior

Since $\mathcal{C}^+$ describes the set of points that the learned function should model, $g(\phi(\mathcal{C}_\mathbf{x}; \mathcal{W}); \mathcal{C}_\mathbf{y}, \theta)$ has the straightforward interpretation of measuring how closely $\phi(\mathcal{C}_\mathbf{x}; \mathcal{W})$ lies to $\mathcal{C}_\mathbf{y}$. Most common probability distributions as well as (possibly improper) user-defined distributions are amenable, though differentiability may be a condition for certain inference methods. In particular, natural choices of distributions exist for both regression and classification.

Regression In the simplest setting, for which there is a known ground-truth function described perfectly by $\mathcal{C}^+$, the Gaussian distribution is a natural choice:

$$g(\mathcal{W} \,|\, \mathcal{C}^+) = \prod_{\mathbf{x},\mathbf{y} \sim \pi(\mathcal{C}^+)} \mathcal{N}(\phi(\mathbf{x}; \mathcal{W}); \mathbf{y}, \sigma_+^2) \quad (6)$$

where $\pi(\mathcal{C}^+)$ is a sampling distribution for $\mathcal{C}^+$, which is necessary for tractability if $\mathcal{C}^+$ is large or infinite. $\pi(\mathcal{C}^+)$ itself can be user-defined as the domain allows, allowing for flexibility in sampling. σ_+ is the tuneable standard deviation of the Gaussian, controlling strictness of deviation from $\mathcal{C}^+$. More generally, it is possible that there exists multiple $\mathbf{y} \in \mathcal{C}_y^+$ for some $\mathbf{x} \in \mathcal{C}_x^+$. This can be expressed using multimodal distributions, for example:

$$g(\mathcal{W} \,|\, \mathcal{C}^+) = \prod_{\mathbf{x}, \{\mathbf{y}\}_k^K \sim \pi(\mathcal{C}^+)} \sum_{k=1}^{K} \omega_k \mathcal{N}(\phi(\mathbf{x}; \mathcal{W}); \mathbf{y}_k, \sigma_+^2) \quad (7)$$

where ω_k are the user-defined mixture weights.

Classification $\mathcal{C}_\mathbf{y}^+$ describes the classes that the BNN is constrained to for the corresponding $\mathcal{C}_\mathbf{x}^+$. In the discrete setting, the natural distribution is the Dirichlet. For K classes,

$$g(\mathcal{W} \,|\, \mathcal{C}^+) = \prod_{\mathbf{x},\mathbf{y} \sim \pi(\mathcal{C}^+)} \mathcal{D}ir(\phi(\mathbf{x}; \mathcal{W}); \alpha) \quad (8)$$

where $\alpha_k = \begin{cases} 1 & \text{if } y_k = 1 \\ 1 - \alpha_\sigma & \text{otherwise} \end{cases}$ for some controllable penalty α_σ.

A.2. Negative constraint prior

The negative constraint prior enforces the infeasibility of regions in function space and is constructed by placing little prior probability on high expected vio-

lation of $\mathcal{C}^-$:

$$g(\mathcal{W}\,|\mathcal{C}^-) = \exp\left(\mathop{\mathbb{E}}_{\mathbf{x}\sim\pi(C_{\mathbf{x}}^-)}\left[-\gamma\; c\big(\mathbf{x}, \phi(\mathbf{x},\mathcal{W}); \mathcal{C}^-\big)\right]\right) \quad (9)$$

In (9), $c(\mathbf{x},\mathbf{y};\mathcal{C}^-)$ is a classifier function that encodes softly whether or not $(\mathbf{x},\mathbf{y})$ is in $\mathcal{C}^-$, which allows black-box use with any inference technique:

$$c(\mathbf{x},\mathbf{y};\mathcal{C}^-) = \sum_{j=1}^{J}\prod_{k=1}^{K_j} \sigma_{\tau_0,\tau_1}\Big(f_j\big(\mathbf{x},\mathbf{y}\big)_k\Big) \quad (10)$$

The definition of $c(\mathbf{x},\mathbf{y};\mathcal{C}^-)$ assumes that the negative region $\mathcal{C}^-$ is defined by J sets of K_j inequality constraints $f_j(\mathbf{x},\mathbf{y}) \leq 0$, i.e. $\mathcal{C}^- = \bigcup_{j=1}^{J}\mathcal{C}_j^-$ with $\mathcal{C}_j^- = \{(\mathbf{x},\mathbf{y}) \mid f_j(\mathbf{x},\mathbf{y}) \leq 0\}$, which can define arbitrary linear and nonlinear shapes in the input-output space. $\sigma_{\tau_0,\tau_1}(z)$ is a soft indicator of whether a constraint of the form $z \leq 0$ is satisfied, a more generally-parameterizable sigmoidal activation defined as

$$\sigma_{\tau_0,\tau_1}(z) = \big(\tanh(-\tau_0 z)+1\big)\big(\tanh(-\tau_1 z)+1\big) \quad (11)$$

If *all* constraints for at least one infeasible region $\mathcal{C}_j^-$ are satisfied, our prior knowledge is violated and $c(\mathbf{x},\mathbf{y};\mathcal{C}^-)$ is far from 0. Otherwise, at least one constraint of all infeasible regions is violated and our prior beliefs satisfied; $c(\mathbf{x},\mathbf{y};\mathcal{C}^-)$ is close to 0. Contrary to other classification functions, the product of two tanh functions with different scales τ_0, τ_1 enables a sharp and steep overall classification of violating values in $z > 0$ and a smoother and flatter classification for satisfying values in $z \leq 0$, making gradients less vanishing for constraint-satisfying, i.e. region-violating inputs. We use $\tau_0 = 15, \tau_1 = 2$.

B. Inference

Constraint priors can be substituted for the traditional prior term $p(\mathcal{W})$ with any black-box sampling or variational inference (VI) algorithm. Here, we provide a summary of the algorithms we use and describe the trivial modifications used to incorporate constraint priors $p_{\mathcal{C}}(\mathcal{W})$. Note that the general form of $p_{\mathcal{C}}(\mathcal{W})$ is not normalized, which does not pose a problem for inference in practice.

Hamiltonian Monte Carlo (HMC) HMC (Neal, 2012) is a MCMC method considered to be the "gold standard" in posterior sampling even though not being scalable. We substitute $p(\mathcal{W})$ by $p_{\mathcal{C}}(\mathcal{W})$ in the potential energy term $U(\mathcal{W})$ computed at each sampling iteration:

$$U(\mathcal{W}) = -\log p_{\mathcal{C}}(\mathcal{W}) - \log p(\mathcal{D}\,|\,\mathcal{W}) \quad (12)$$

As the presence of $g(\mathcal{W}\,|\,\mathcal{C})$ increases the magnitude of the prior $p_{\mathcal{C}}(\mathcal{W})$, empirical performance typically improves by using a smaller step-size than with $p(\mathcal{W})$ for the same dataset.

Stein Variational Gradient Descent (SVGD) SVGD (Liu & Wang, 2016) is a VI method where a set of S particles (in our case, $\{\mathcal{W}_s\}_{s=1}^S$) are optimized via functional gradient descent to mimic the true posterior. SVGD combines the efficiency of VI methods with the ability of MCMC methods to capture more expressive posterior approximations. $p(W)$ is substituted by $p_{\mathcal{C}}(\mathcal{W})$ in the computation of the functional gradient:

$$\hat{\phi}^*(\mathcal{W}) = \frac{1}{S}\sum_{s=1}^{S}\Big[k(\mathcal{W}_s,\mathcal{W})\nabla_{\mathcal{W}_s}[\log p_{\mathcal{C}}(\mathcal{W}_s) + \log p(\mathcal{D}\,|\,\mathcal{W}_s)] + \nabla_{\mathcal{W}_s}k(\mathcal{W}_s,\mathcal{W})\Big] \quad (13)$$

Our implementation of SVGD uses the weighted RBF kernel $k(x,x') = \exp(-\frac{1}{h}||x-x'||_2^2)$ and adapting bandwith h as suggested in (Liu & Wang, 2016) as well as mini-batched data $\mathcal{D}$ for tractability.

C. Experimental Details

C.1. Synthetic Examples

For all experiments, the BNN used comprises a single hidden layer with 10 nodes, and Radial Basis Function (RBF) activations $\sigma(x) = \exp\{-x^2\}$.

All regression plots show the posterior mean function (bold line) as well as the confidence intervals for $\sigma = 1$ (dark shading) and $\sigma = 2$ (light shading).

Figure 1: (left) The constrained regions are $y < 2.5$ and $y > 3$ for $x \in [-0.3, 0.3]$. The function generating the training points is $y = -x^4 + 3x^2 + 1$. The negative prior formulation is used. **(right)** The input space is 2-dimensional and there are 3 classes (color-coded) with 8 training points in each class, generated from the Gaussian means $(-3,1), (0,-3)$ and $(2,3)$. The constrained region is $[1,3]\times[-2,0]$ and defined such that points within the box *should* be classified as green. The positive prior is used. HMC (10000 burn-in, 1000 samples collected at intervals of 10) is used for both examples.

Figure 2: (left) The positive constraints are $y = -x + 5$ for $x \in [-5.0, -3.0]$ and $y = x + 5$ for $x \in [3.0, 5.0]$. Both constraints are Gaussian with the $\sigma = 0.5$. The 3 training points are arbitrarily defined. HMC (10000 burn-in, 1000 samples collected at intervals of 10) is used. **(Right)** The constrained boxed region is $x \in [-1.0, 1.0]$ and $y \in [-5.0, 3.0]$. The function generat-

ing the training points is $y = -x^4 + 3x^2 + 1$. SVGD with 75 particles is used with Adagrad.

C.2. Clinical action prediction

For all experiments, the BNN used comprises a 2 hidden layers of 200 nodes each and RBF activations. SVGD is used for inference with 50 particles, 1500 iterations, Adagrad optimization, and a suitable batch size. The size of the full dataset is 298K; this reduces to 125K when points in the constrained region are filtered out. Details on the prior formulation for can be found in A. The Dirichlet parameter is set to 10 for allowed classes and 0.01 for forbidden classes.

C.3. Human motion prediction

For these experiments, the BNN used comprises a 2 hidden layers of 100 nodes each and RBF activations. For inference, we again used SVGD and Adagrad with 50 particles and 1000 iterations. The negative prior used 50 samples from $\pi(\mathcal{C}_{\mathbf{x}}^{-})$ and $\gamma = 10,000$, see Eq. 9.

We randomly chose a subset of 10 right-handed reaching trajectories from (Kratzer, 2019). This data was randomly split into 5 training and 5 test trajectories, which amounts to 243 train Markov states of sensors for training and 142 states for evaluation. Given this problem setting, the regression task had 12-dimensional inputs and 4-dimensional targets. The number of training trajectories was kept low to increase sparsity and the difficulty of successful robust generalization.

D. Additional Results

D.1. Additional Synthetic Examples

Figure 4 shows additional examples for out-of-distribution and multimodal behavior. **(left)** Out-of-distribution negative constraints. The negative constraints are $y > -x + 7$ and $y < -x + 2$ for $x \in [-5.0, -3.0]$ and $y > x + 7$ and $y < x + 2$ for $x \in [3.0, 5.0]$. The training points are identical to those in the left plot of Figure 2. HMC (10000 burn-in, 1000 samples collected at intervals of 10) is used. **(right)** Multimodal positive constraints. The two positive functions are $y = -0.2x^3 + 0.5x^2 + 0.7x - 0.5$ and $y = 0.2x^3 - 0.15x^2 + 3.5$, both for the domain $x \in [-1.0, 1.0]$. The training points were arbitrarily defined. An equally-weighted mixture of two Gaussians with $\sigma = 0.5$ is used as the positive constraint prior. SVGD with 75 particles and Adagrad are used.

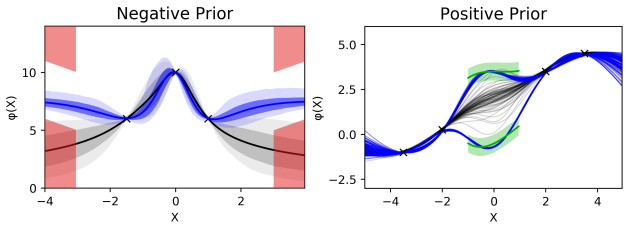

Figure 4. **(left)** The same training set as Figure 2 (left), but with negative constraints defined out-of-distribution. OC-BNNs fit the sparse data while avoiding the constraints. **(right)** Positive prior with mixture of two Gaussians. Using SVGD, individual OC-BNN samples (blue) capture both modes.