High Dimensional Level Set Estimation with Bayesian Neural Network

Huong Ha*, Sunil Gupta, Santu Rana, Svetha Venkatesh
Applied Artificial Intelligence Institute (A^2I^2), Deakin University, Geelong, Australia

Abstract

Level Set Estimation (LSE) is an important problem with applications in various fields such as material design, biotechnology, machine operational testing, etc. Existing techniques suffer from the scalability issue, that is, these methods do not work well with high dimensional inputs. This paper proposes novel methods to solve the high dimensional LSE problems using Bayesian Neural Networks. In particular, we consider two types of LSE problems: (1) *explicit* LSE problem where the threshold level is a fixed user-specified value, and, (2) *implicit* LSE problem where the threshold level is defined as a percentage of the (unknown) maximum of the objective function. For each problem, we derive the corresponding theoretic information based acquisition function to sample the data points so as to maximally increase the level set accuracy. Furthermore, we also analyse the theoretical time complexity of our proposed acquisition functions, and suggest a practical methodology to efficiently tune the network hyper-parameters to achieve high model accuracy. Numerical experiments on both synthetic and real-world datasets show that our proposed method can achieve better results compared to existing state-of-the-art approaches.

Introduction

In engineering practice, there are numerous problems which require accurately identifying the regions where the value of a black-box expensive function is higher or lower than a given threshold level. A specific example is alloy design where the task is to determine the regions of alloy compositions that have a desired property (e.g. tensile strength) above a certain level. To solve this, the desired property of a number of alloy compositions are measured and the compositions of interest can be estimated from these measurements. As the measurement process is expensive, thus it is required to minimize the total number of measurements while still maintaining accurate identification for the region of interest. Another example is the algorithmic assurance problem, with the goal being to identify the range of inputs where a machine learning model performs as expected. Specifically, before deploying a machine learning model, it is of interest to identify the range of inputs where the deviation between the machine learning model and the ground truth is lower than a user-specified threshold. A traditional approach is to label the data in the applied domain and estimate the region of interest. As the data labelling process is costly, it is desired to accurately identify this region with a minimal number of sampled data. Other applications of the level set estimation problem can be found in the field of environmental monitoring, manufacturing quality control process, biotechnology, etc (Bryan et al. 2005; Gotovos et al. 2013; Bogunovic et al. 2016; Zanette, Zhang, and Kochenderfer 2019; Iwazaki, Inatsu, and Takeuchi 2019).

In the statistics and machine learning literature, this problem is called the Level Set Estimation (LSE) problem. To solve an LSE problem, the general idea is to formulate it as an active learning problem. First, a small number of measurements of the black-box function are taken to construct a training dataset, and a surrogate model is learned from this training dataset to predict the measurements of all the unmeasured points in the domain. Second, an acquisition function is constructed based on the surrogate model to decide which data point in the domain should be next measured. The black-box function is then evaluated at the chosen data point, and the training dataset is updated with the new measurement. This iterative process continues until the measuring budget is depleted. The regions of interest are then estimated using the learned surrogate model.

There have been a number of research works proposing new methods to tackle the LSE problem, however, all of these approaches are limited due to the use of Gaussian process (Rasmussen and Williams 2006) as the surrogate model. In particular, it is widely known that standard Gaussian process (GP) (i) does not scale well with high dimensional inputs due to its cubic time complexity, and, (ii) has low representational power because of the limited choices of kernels (Krauth and E. Bonilla 2017). In this work, we address this issue by suggesting the use of Bayesian Neural Network (BNN) as the surrogate model. Bayesian neural network is a special type of neural network that can produce a probability distribution over the weight parameters (MacKay 1992; Neal 1995), thus, it can provide both the prediction and prediction uncertainty for any new data point.

We first consider the *explicit* LSE problem where the threshold level is a fixed user-specified value. Using the uncertainty estimates from the BNN, we derive a theoretic information based acquisition function which can sample the

*Correspondence to: huong.ha@rmit.edu.au

data points that help to most accurately classify the data points in the domain into two sets with outputs below and above the threshold level. We next address the *implicit* LSE problem where the threshold level is defined as a percentage of the (unknown) maximum of the objective function. As the threshold level is unknown, the challenge with the *implicit* LSE problem is that we need to sample the data points that help to most accurately (1) learn the implicit threshold level (or the objective function maximum), and, (2) classify the data points with respect to this estimated threshold level. To solve this problem, we unify the two goals into one goal which is to sample the data point for which if its function value is known, the estimated super- and sub-level set are most accurate. We then construct a novel theoretic information based acquisition function that aims to reduce the uncertainty of the estimated super(sub)-level set, thus, helps to maximally increase the classification accuracy. Finally, we analyse the theoretical time complexity of our proposed acquisition functions, and suggest a practical methodology to efficiently tune the BNN hyper-parameters.

Contributions The main contributions of our paper can be summarized as follows:

- Proposing for the first time the use of BNN to tackle the LSE problems for high dimensional inputs,
- Deriving two novel sampling methodologies to address the *explicit* and *implicit* LSE problems,
- Demonstrating the effectiveness of our proposed methods on both synthetic and real-world problems and comparing with state-of-the-art approaches.

Related Work There are various research works aiming to tackle the LSE problem. The work in (Bryan et al. 2005) addresses the *explicit* LSE problem by suggesting the *Straddle* heuristic that aims to select the data point that is both predicted to be near the threshold level and also uncertain in terms of its predicted function value. The work was then extended in (Bryan and Schneider 2008) to handle the case when the black-box objective function is a composite function. Later, the work in (Gotovos et al. 2013) proposed an algorithm that can solve both the *explicit* and *implicit* LSE problems. Recently, the work in (Shekhar and Javidi 2019) proposed a new method for the *explicit* LSE problem that achieves better theoretical guarantee compared to state-of-the-art methods whilst maintaining a reasonable computational complexity. There are also various works aiming to address some compplex settings of the *explicit* LSE problem such as finding the regions when the probability of the function values larger than a threshold (Zanette, Zhang, and Kochenderfer 2019), or when the inputs are uncertain (Iwazaki, Inatsu, and Takeuchi 2019) or cost dependent (Inatsu et al. 2019). All of these works rely on Gaussian Process as the surrogate model, and therefore, suffer from the problem of scalability. In the experimental results, we will demonstrate that our proposed approaches outperform these works especially on high dimensional problems.

In a closely related context, there are research works in the field of Bayesian Optimization (BO) with the goal of obtaining the argmax of the black-box objective function using a minimal numbers of sampled data (Snoek, Larochelle, and Adams 2012; Shahriari et al. 2016). The algorithm *TRUVAR* (Bogunovic et al. 2016) was developed to unify the *explicit* LSE and BO problem. There are also works that use the ideas behind the LSE problem to develop safe BO methodologies (Sui et al. 2015, 2018). Finally, there is the work in (Garnett et al. 2012) considering the problem of *active search*, with the goal to sample as many points as possible from one of the level set domains.

Problem Statement and Background

Let us consider an expensive black-box function $f : \mathcal{X} \to \mathbb{R}$, where $\mathcal{X}$ is a discrete compact subset of $\mathbb{R}^d$. We assume that we only have noisy measurements of $f(x)$, i.e. querying f at the data point $x \in \mathcal{X}$ results in the noisy function value $f(x) + \epsilon$ where the noise ϵ follows the normal distribution $\mathcal{N}(0, \sigma_\epsilon^2)$ and is independent from x or $f(x)$. We consider two types of LSE problems: *explicit LSE* and *implicit LSE*:

- The *explicit LSE* problem is to classify all the data points in $\mathcal{X}$ into two subsets based on a user-specified threshold level h ($h \in \mathbb{R}$), i.e. to classify $\forall x \in \mathcal{X}$ into a super-level set $H = \{x \in \mathcal{X} \mid f(x) > h\}$ and a sub-level set $L = \{x \in \mathcal{X} \mid f(x) \leq h\}$.
- The *implicit LSE* problem is to classify all the data points in $\mathcal{X}$ into two subsets based on the relation with the function maximum, i.e. to classify $\forall x \in \mathcal{X}$ into a super-level set $H = \{x \in \mathcal{X} \mid f(x) > l \max_{x \in \mathcal{X}} f(x)\}$ and a sub-level set $L = \{x \in \mathcal{X} \mid f(x) \leq l \max_{x \in \mathcal{X}} f(x)\}$ where l ($0 \leq l \leq 1$) is a user-specified threshold.

Given a budget of T function evaluations, our goal is to design sequential query point selection strategies to achieve the highest classification accuracies for the two LSE problems.

Bayesian Neural Networks Bayesian neural networks (BNN) provide a probabilistic interpretation of neural networks by inferring probability distribution of the networks' weights (MacKay 1992; Neal 1995). Given a training data $\mathcal{D}_{tr} = \{x_i, y_i\}_{i=1}^N$, a BNN can provide the posterior distribution $p(\omega|\mathcal{D}_{tr})$ with ω being the neural network weights. This special property of BNNs offers a way to compute the uncertainty of the network's prediction. In particular, for each data point x, the predicted distribution $p(y|x, \mathcal{D}_{tr})$ can be inferred from the posterior distribution $p(\omega|\mathcal{D}_{tr})$ using the formula $p(y|x, \mathcal{D}_{tr}) = \int p(y|x, \omega)p(\omega|\mathcal{D}_{tr})\,\mathrm{d}\omega$.

In practice, performing exact inference to obtain $p(\omega|\mathcal{D}_{tr})$ is generally intractable, hence we use a variational approximation technique to approximate this posterior. Specifically, we employ the MC-dropout method (Gal and Ghahramani 2016). There are two main reasons for this choice. First, it is both scalable and theoretically guaranteed, i.e. it is shown to be equivalent to performing approximate variational inference to find a distribution $q_\theta^*(\omega)$ in a tractable family that minimizes the Kullback-Leibler divergence to the true model posterior $p(\omega|\mathcal{D}_{tr})$ (Gal and Ghahramani 2016). Second, the method can directly provide a reasonable amount of samples $\hat{\omega}_j$ from the approximate posterior $q_\theta^*(\omega)$, which is a key ingredient of our proposed sampling strategy.

Level Set Estimation with Bayesian Neural Network

The Explicit LSE Problem

One simple solution is to formulate the *explicit* LSE problem as a classification active learning problem that aims to efficiently train a binary classifier to predict whether a data point belongs to the super- or sub-level set. However, this approach is infeasible in the important cases when the threshold h is close to $\min_{x\in\mathcal{X}} f(x)$ or $\max_{x\in\mathcal{X}} f(x)$. In these cases, the data points in the domain $\mathcal{X}$ mostly belong to one single class, thus, any standard classification active learning method results in incorrect classification for the minor class.

To overcome this issue, we formulate the *explicit* LSE problem as a regression active learning problem, i.e. actively train a network to predict the function value $f(x), \forall x \in \mathcal{X}$. Using the network prediction $\mathcal{A}(x)$, we can approximate the super-level set H and sub-level set L as $\hat{H} = \{x \in \mathcal{X} \mid \mathcal{A}(x) > h\}$ and $\hat{L} = \{x \in \mathcal{X} \mid \mathcal{A}(x) \leq h\}$. However, unlike the standard regression active learning problem that aims to train a network to most accurately predict the function values for all data points, our goal is to train a network to most accurately predict the function values of only those data points that affect the level set classification accuracy. At iteration $t+1$, given the current observed data $\mathcal{D}_t$, using the mutual information (Houlsby et al. 2011; Gal, Islam, and Ghahramani 2017) as a way to represent uncertainty, we propose the acquisition function as,

$$\begin{aligned}\alpha_{\text{exp}}(x;\mathcal{D}_t) &= \mathbb{I}(I_y;\omega|x,\mathcal{D}_t)\\ &= \mathbb{H}(I_y|x,\mathcal{D}_t) - \mathbb{E}_{p(\omega|\mathcal{D}_t)}[\mathbb{H}(I_y|x,\omega)], \quad (1)\end{aligned}$$

where y denotes the predicted function value corresponding to the data point x, I_y is equal 1 if $y > h$ and equal 0 otherwise, $\mathbb{H}(I_y|x,\mathcal{D}_t)$ denotes the entropy of I_y given the data point x and the current observed data $\mathcal{D}_t$, and $\mathbb{H}(I_y|x,\omega)$ denotes the entropy of I_y given x and the model weights ω. The value of $\alpha_{\text{exp}}(x;\mathcal{D}_t)$ is highest when the BNN is most uncertain whether x belongs to the super- or sub-level set.

Using the fact that $p(I_y = c|x,\mathcal{D}_t) = \int p(I_y = c|x,\omega)p(\omega|\mathcal{D}_t)\,\mathrm{d}\omega$, we can rewrite $\alpha_{\text{exp}}(x;\mathcal{D}_t)$ as,

$$\begin{aligned}\alpha_{\text{exp}}(x;\mathcal{D}_t) = &-\sum_{c\in\{0,1\}}\Big(\int p(I_y=c|x,\omega,\mathcal{D}_t)p(\omega|\mathcal{D}_t)\,\mathrm{d}\omega\\ &\times\log\int p(I_y=c|x,\omega,\mathcal{D}_t)p(\omega|\mathcal{D}_t)\,\mathrm{d}\omega\Big)\\ &+\mathbb{E}_{p(\omega|\mathcal{D}_t)}\sum_{c\in\{0,1\}} p(I_y=c|x,\omega)\log p(I_y=c|x,\omega),\end{aligned}$$

where $c \in \{0,1\}$ and $p(I_y = c|x,\omega,\mathcal{D}_t)$ is set to be c if $y|x,\omega$ is larger than h, and $1-c$ vice versa. Finally, approximating the true posterior $p(\omega|\mathcal{D}_t)$ as the MC-dropout posterior $q^*_\theta(\omega)$, and using the stochastic forward passes $\{\hat{\omega}_j\}_{j=1}^M$ provided by MC-dropout, we can compute $\alpha_{\text{exp}}(x;\mathcal{D}_t)$ as,

$$\begin{aligned}\alpha_{\text{exp}}(x;\mathcal{D}_t) \approx &-\sum_{c\in\{0,1\}}\Big(\frac{1}{M}\sum_M \hat{p}_c^j\Big)\log\Big(\frac{1}{M}\sum_M \hat{p}_c^j\Big)\\ &+\frac{1}{M}\sum_{c,j}\hat{p}_c^j\log\hat{p}_c^j, \quad (2)\end{aligned}$$

Algorithm 1 ExpHLSE: Explicit High Dimensional LSE via Bayesian Neural Network

Input: Function f, domain $\mathcal{X}$, threshold h, Bayesian neural network $\mathcal{M}$, acquisition function $\alpha_{\text{exp}}(x)$, initial observations $\mathcal{D}_{init}$, evaluation budget T.
Output: Estimated super- and sub-level sets $\hat{H}$, $\hat{L}$.
Initialize $\mathcal{D}_0 = \mathcal{D}_{init}$,
for $t = 1, 2, \ldots, T$ **do**
  Train the MC-Dropout BNN using $\mathcal{D}_{t-1}$
  Generate M forward passes $\{\hat{\omega}_j\}_{j=1}^M$ from the BNN
  Find $x^* = \text{argmax}\ \alpha_{\text{exp}}(x)(x;\mathcal{D}_{t-1})$ using Eq. (2)
  Update observations $\mathcal{D}_t = \mathcal{D}_{t-1} \cup \{x^*, f(x^*)\}$
end for
Train the BNN using $\mathcal{D}_T$
Compute the estimated super- and sub-level sets $\hat{H} = \{x \in \mathcal{X} \mid \mathcal{A}(x) > h\}$ and $\hat{L} = \{x \in \mathcal{X} \mid \mathcal{A}(x) \leq h\}$ where $\mathcal{A}(x)$ is the BNN prediction for the data point x.

with $\hat{p}_c^j$ being the probability of input x with model parameters $\hat{\omega}_j \sim q^*_\theta(\omega)$ to be in class c.

Our proposed algorithm that solves the ***explicit* high** dimensional **LSE** problem using BNN, **ExpHLSE**, is presented in **Algorithm** 1.

Remark 1. *It is worth noting that instead of using the mutual information, other criteria such as variation ration (Freeman 1965) and entropy (Shannon 1948) can also be used to construct the acquisition function.*

The Implicit LSE Problem

In the *implicit* LSE problem, the goal of the sampling process is to select the data point with two goals: 1) learn the unknown threshold level most accurately, and, 2) classify the data points into the super- and sub-level sets corresponding to the estimated threshold level most accurately. To unify these two goals, we aim to sample the data point for which if its function value is known, the estimated super- and sub-level set are most accurate. For $\forall x \in \mathcal{X}$, we denote $\tilde{f}(x)$ as a random variable denoting all the possible values of $f(x)$. We then define a random variable $\tilde{H}(x)$ as,

$$\tilde{H}(x) = \{x' \in \mathcal{X} \mid \hat{f}_x(x') > l \max_{x'\in\mathcal{X}} \hat{f}_x(x')\}, \quad (3)$$

where

$$\hat{f}_x(x') = \begin{cases}\mathbb{E}_{\hat{f}(x')\sim p(\hat{f}(x')|\mathcal{D}_t)}\hat{f}(x') & \text{if } x' \neq x\\ \hat{f}(x) & \text{if } x' = x\end{cases}, \ \forall x' \in \mathcal{X},$$

with $\hat{f}(x')$ is a random variable representing the estimated value of $f(x')$. With this definition, for $\forall x \in \mathcal{X}$, $\hat{f}_x(x')$ represents all the possible functions $f(x')$ *when the value of f(x) varies*, thus, the random variable $\hat{H}(x)$ represents all the possible super-level sets H *when the value of* $f(x)$ *varies*. To obtain an accurate estimated super(sub)-level set within a minimal number of sampled data, the goal is to sample the data point x for whom the random variable $\tilde{H}(x)$ is most

uncertain. As $\tilde{H}(x)$ is a random variable representing sets of data points, evaluating which $\tilde{H}(x)$ being the most uncertain is intractable. Thus, we simplify by defining a new random variable $\tilde{G}(x)$ representing the cardinality of the random variable $\tilde{H}(x)$, i.e. $\tilde{G}(x) = |\tilde{H}(x)|$. Then we can sample the data point x whose the corresponding random variable $\tilde{G}(x)$ is most uncertain. To characterize the uncertainty of the random variable $\tilde{G}(x)$, we compute its mutual information (Houlsby et al. 2011; Gal, Islam, and Ghahramani 2017). Thus, the acquisition function can be formulated as,

$$\begin{aligned}\alpha_{\text{imp}}(x; \mathcal{D}_t) &= \mathbb{I}[\tilde{G}(x); \omega | x, \mathcal{D}_t] \\ &= \mathbb{H}[\tilde{G}(x)|x, \mathcal{D}_t] - \mathbb{E}_{p(\omega|\mathcal{D}_t)}[\mathbb{H}[\tilde{G}(x)|x, \omega]],\end{aligned} \tag{4}$$

where $\mathbb{I}[.]$ is the mutual information operator and $\mathbb{H}[.]$ is the entropy operator. The value of $\alpha_{\text{imp}}(x; \mathcal{D}_t)$ is highest when the BNN is most uncertain in estimating the value of $\tilde{G}(x)$.

Firstly, we show how to approximate the first term in Eq. (4), $\mathbb{H}[\tilde{G}(x)|x, \mathcal{D}_t]$. Using the entropy formula, $\mathbb{H}[\tilde{G}(x)|x, \mathcal{D}_t]$ can be computed as,

$$\begin{aligned}&\mathbb{H}[\tilde{G}(x)|x, \mathcal{D}_t] \\ &= -\sum\nolimits_{q \in \mathcal{Q}(x)} p(\tilde{G}(x) = q|x, \mathcal{D}_t) \log p(\tilde{G}(x) = q|x, \mathcal{D}_t),\end{aligned}$$

where $\mathcal{Q}(x)$ is a set consisting of all the possible values of $\tilde{G}(x)$ when $f(x)$ varies. Using the property $p(\tilde{G}(x) = q|x, \mathcal{D}_t) = \int p(\tilde{G}(x) = q|x, \omega) p(\omega|\mathcal{D}_t)\, \mathrm{d}\omega$ and the stochastic forward passes $\{\hat{\omega}_j\}_{j=1}^M$ generated by MC-dropout, we can approximate $p(\tilde{G}(x) = q|x, \mathcal{D}_t)$ as $1/M \sum_{j=1}^M p(\tilde{G}(x) = q|x, \hat{\omega}_j)$. Therefore, the term $\mathbb{H}[\tilde{G}(x)|x, \mathcal{D}_t]$ can finally be approximated as,

$$\begin{aligned}\mathbb{H}[\tilde{G}(x)|x, \mathcal{D}_t] \approx -\sum_{q \in \mathcal{Q}(x)} &\Big(\frac{1}{M} \sum\nolimits_{j=1}^M p(\tilde{G}(x) = q|x, \hat{\omega}_j)\Big) \\ &\times \log\Big(\frac{1}{M} \sum\nolimits_{j=1}^M p(\tilde{G}(x) = q|x, \hat{\omega}_j)\Big),\end{aligned} \tag{5}$$

Similarly, the second term $\mathbb{E}_{\omega \sim p(\omega|\mathcal{D}_t)}[\mathbb{H}[\tilde{G}(x)|x, \omega]]$ in Eq. (4) can be approximated as,

$$\begin{aligned}&\mathbb{E}_{\omega \sim p(\omega|\mathcal{D}_t)}[\mathbb{H}[\tilde{G}(x)|x, \omega]] \approx \frac{1}{M} \sum\nolimits_{j=1}^M \mathbb{H}[\tilde{G}(x)|x, \hat{\omega}_j)]] \\ &\approx \frac{1}{M} \sum_{\substack{j=1,\dots,M \\ q \in \mathcal{Q}(x)}} p(\tilde{G}(x) = q|x, \hat{\omega}_j)) \log p(\tilde{G}(x) = q|x, \hat{\omega}_j)).\end{aligned} \tag{6}$$

Our proposed ***implicit*** **h**igh dimensional **LSE** algorithm with BNN, **ImpHLSE**, is presented in **Algorithm** 2.

Remark 2. *If there are multiple data points with the same mutual information $\mathbb{I}[\tilde{G}(x); \omega | x, \mathcal{D}_t]$, we select the data point with the smallest set $\bigcap_{j=1}^M \tilde{H}_j(x)$, i.e. the data point that makes the cardinality $|\bigcap_{j=1}^M \tilde{H}_j(x)|$ to be smallest. This is to ensure that the selected data point is also the data point that causes the set $\tilde{H}(x)$ to be most uncertain.*

Algorithm 2 ImpHLSE: Implicit High Dimensional LSE via Bayesian Neural Network

Input: Function f, domain $\mathcal{X}$, threshold ratio l, Bayesian neural network $\mathcal{M}$, acquisition functions $\alpha_{\text{imp}}(x)$, initial observations $\mathcal{D}_{init}$, evaluation budget T.
Output: Estimated super- and sub-level sets $\hat{H}$ and $\hat{L}$.
Initialize $\mathcal{D}_0 = \mathcal{D}_{init}$,
for $t = 1, 2, \dots, T$ **do**
  Train the MC-Dropout BNN using $\mathcal{D}_{t-1}$
  Generate M forward passes $\{\hat{\omega}_j\}_{j=1}^M$ from the BNN
  Compute all the possible values of the random variable $\tilde{G}(x)$ for each data point $x \in \mathcal{X}$ using $\{\hat{\omega}_j\}_{j=1}^M$
  Compute the probability distribution of $\tilde{G}(x)$
  Compute $x^* = \text{argmax}\ \alpha_{\text{imp}}(x; \mathcal{D}_{t-1})$ using Eq. (4)
  Update observations $\mathcal{D}_t = \mathcal{D}_{t-1} \cup \{x^*, f(x^*)\}$
end for
Train the BNN using $\mathcal{D}_T$
Compute the estimations of the super- and sub-level sets as $\hat{H} = \{x \in \mathcal{X} \mid \mathcal{A}(x) > l \max_{x \in \mathcal{X}} \mathcal{A}(x)\}$ and $\hat{L} = \{x \in \mathcal{X} \mid \mathcal{A}(x) \leq l \max_{x \in \mathcal{X}} \mathcal{A}(x)\}$ where $\mathcal{A}(x)$ is the BNN prediction for the data point x.

Remark 3. *Similar to the explicit LSE problem, instead of using the mutual information, other criteria such as variation ration (Freeman 1965) and entropy (Shannon 1948) can also be used to construct the acquisition function.*

BNN Hyper-parameters Tuning

In general, training a neural network requires a variety of hyper-parameters which contribute to their estimation accuracies. Since our proposed methods belong to the active learning scheme, the data keeps growing, and thus the network hyper-parameters need to be updated to avoid issues such as under-fitting, improper parameter optimization. In the sequel, we will denote the hyper-parameters corresponding to the BNN architecture such as the architecture type, number of layers or neurons as major hyper-parameters and other hyper-parameters such as learning rate, drop-out rate as minor hyper-parameters. We then use the incremental neural architecture search (iNAS) method developed in (Geifman and El-Yaniv 2019) to tune the major hyper-parameters and a grid-search method to tune the minor hyper-parameters. In particular, at iteration t, we split the current observed data $\mathcal{D}_t$ into a training and a validation set, and initialize the BNN architecture with fixed major and minor hyper-parameters. We then generate a set of potential candidates for the major hyper-parameters provided by iNAS given the initial major hyper-parameters, and a grid for the minor hyper-parameters. We then choose the major and minor hyper-parameters that optimize the mean square error on the validation dataset. The whole current observed data $\mathcal{D}_t$ is then trained again with these optimal hyper-parameters to generate the BNN. Finally, note that when training the BNN, we use the standard loss function for regression problem, i.e. the loss function is the mean square error between the model predicted output and the ground-truth value.

Time Complexity of The Proposed Algorithms

The Proposed Explicit LSE Algorithm The time complexity of the proposed **ExpHLSE** acquisition fucntion developed in Eqs. (1), (2) is $\mathcal{O}(M|\mathcal{X}|)$, where M is the number of stochastic forward passes by MC-dropout, and $|\mathcal{X}|$ is the cardinality of the function domain $\mathcal{X}$.

The Proposed Implicit LSE Algorithm The time complexity of the proposed **ImpHLSE** acquisition function in Eqs. (4), (5), (6) is $\mathcal{O}(4|\mathcal{X}| + 4M|\mathcal{X}|)$ where M is the number of stochastic forward passes by MC-dropout, and $|\mathcal{X}|$ is the number of instances in $\mathcal{X}$. This time complexity is computed by splitting the computation of the acquisition function $\alpha_{\text{imp}}(x; \mathcal{D}_t)$ into 3 steps. The first step, which is to compute all the possible values of $\tilde{G}(x)$ for $\forall x \in \mathcal{X}$ when $\tilde{f}(x)$ varies, has the time complexity of $\mathcal{O}((M+3)|\mathcal{X}|)$. The second step, which is to compute the probability distribution of $\tilde{G}(x)$, has the time complexity of $\mathcal{O}((M+1)|\mathcal{X}|)$. The final step, which is to compute the two entropy terms, which has the time complexity of $\mathcal{O}(2M|\mathcal{X}|)$.

Experimental Results

In this section, we evaluate our proposed methods on three benchmark synthetic functions and three real-world problems. The details about the synthetic functions and the real-world problems are in the subsequent sections. For all the problems with dimension d, the optimization process is initialized with an initial $3d$ points (for synthetic functions), and $5d$ points (for real-world problems) sampled following a latin hypercube sample scheme (Jones 2001). For all the tasks, the experiments were repeated 5 times for the synthetic functions and 3 times for the real-world experiments. All the experiments are running on multiple servers where each server has multiple Tesla V100 SXM2 32GB GPUs.

We compare our proposed methods, **ExpHLSE** and **ImpHLSE**, with the state-of-the-art LSE methods. Specifically, for the *explicit* LSE problem, three baselines we compare against are: (1) **Straddle** in (Bryan et al. 2005); (2) $\textbf{LSE}_{\textbf{exp}}$ in (Gotovos et al. 2013); and (3) **TruVAR** in (Bogunovic et al. 2016). For the *implicit* LSE problem, we compare against the method $\textbf{LSE}_{\textbf{imp}}$ in (Gotovos et al. 2013) as this is the only method developed to solve the *implicit* LSE problem. For the $\textbf{LSE}_{\textbf{exp}}$ and $\textbf{LSE}_{\textbf{imp}}$ methods, as suggested in (Gotovos et al. 2013), the exploration-exploitation parameters $\beta_t^{1/2}$ is chosen to be 3 and the accuracy parameter ϵ is chosen to increase exponentially from 2% to 20% of the maximum value of each dataset. For the **TruVAR** method, we use same setting as suggested in (Bogunovic et al. 2016), i.e. we set $\beta_t^{1/2} = \log(|\mathcal{X}|t^2)$, $\eta_{(1)} = 1$, $r = 0.1$ and $\bar{\delta} = 0$. For GP-based methods, we use the Matérn 5/2 kernel for the GP and we fit the GP using the Maximum Likelihood Estimation method. We also use multi-start to ensure the solution of the optimizer does not stuck in a local minima. For our proposed methods, the BNN we use is a feedforward neural network (FNN). The major hyper-parameters for the iNAS tuning process are the number of layer and the number of neurons per layer whilst the minor hyper-parameters are the learning rate and the drop-out rate. The iNAS tuning process is initialized with a FNN with 1 layer and 256 neurons/layer. More details of the hyper-parameter tuning process are in the supplementary material. Finally, for the GP-based methods (except **TruVAR**), the batch size is set to 1 and for our proposed BNN based methods and **TruVAR**, the batch size is set to $10d$ with d being the dimension of the LSE problem. The batch size is the number of data points selected by the acquisition function at each iteration. Setting batch size as $10d$ means we select $10d$ data points with highest acquisition function values. Note that this setting is in favour of the GP-based methods, as the smaller the batch size, the better the performance due to early feedback. Our source code is publicly available at *https://github.com/HuongHa12/HighdimLSE*.

Synthetic Functions

We evaluate the performance of the methods on three ten dimensional benchmark test functions: Ackley10, Levy10 and Alpine10. As their original function domains are continuous, thus we follow the common practice in (Gotovos et al. 2013; Bogunovic et al. 2016) which is to randomly select a large number of data points in the continuous domains to generate the corresponding discrete domains $\mathcal{X}$. In particular, for each problem with dimension d, we select $10000d$ data points. For the *explicit* LSE problem, for each function, the threshold h is set in such a way that results in the volume of the super-level set being 20% of the domain X. For the *implicit* LSE problem, we set the implicit threshold ratio l in such a way that makes $l \max_{x \in \mathcal{X}} f(x)$ to be equal to the threshold h in the *explicit* LSE problem. This is to enable us to compare the two settings. Same with previous works, to measure the effectiveness of the LSE methods, we use the F1-score (i.e., the harmonic mean of precision and recall) with respect to the true super- and sub-level sets. The method resulting in the higher F1-score is the better method.

In Figure 1, we compare the performance of our proposed methods **ExpHLSE** and **ImpHLSE** and other baselines. Note that we do not have the performance of **TruVAR** as its computation time is prohibitive in these cases. Besides, it is also worth noting that the $\textbf{LSE}_{\text{imp}}$ method has a special stopping condition, thus, it only runs for a limited number of iterations. For the *explicit* LSE problem, it can be seen that **ExpHLSE** outperforms all baselines on all three functions, especially on the function Ackley10. For the *implicit* LSE problem, our method **ImpHLSE** also outperforms $\textbf{LSE}_{\text{imp}}$ significantly on Levy10 and Ackley10, and performs similarly on Alpine10. It is interesting to observe that all the baseline methods perform poorly on the function Ackley10 for both LSE problems. This can be explained by the low representational power of GP, in particular, the number of basic kernels (e.g. RBF, Matérn, polynomial) for GP is limited, and the kernel methods generally encounter the curse of dimensionality issue when learning complex functions.

Material Design

Alloys are produced by mixing many elements to achieve properties that are not possible by a single element. We consider a special alloy that consists of 16 elements and the sum

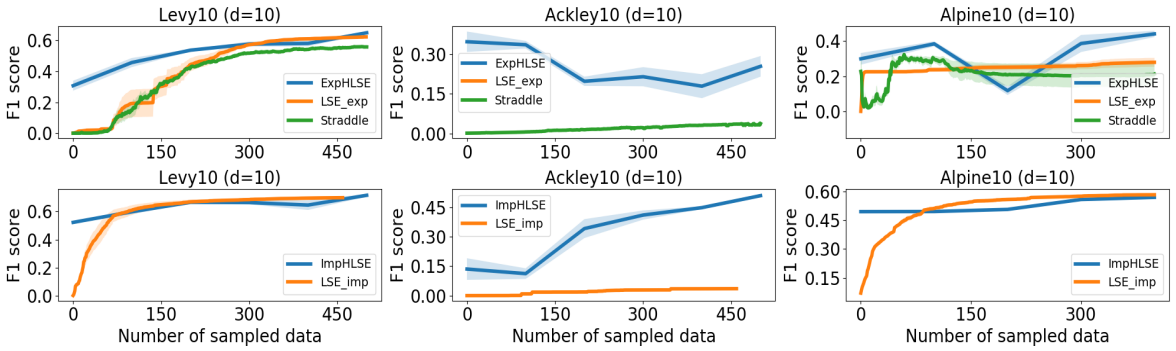

Figure 1: Three synthetic benchmark functions: F1-score for super-/sub-level set for *explicit* LSE (top row) and *implicit* LSE (bottom row). Our proposed methods are in blue while baselines are in other colors. Plotting mean and standard error over 5 repetitions. Method with higher F1-score is better.

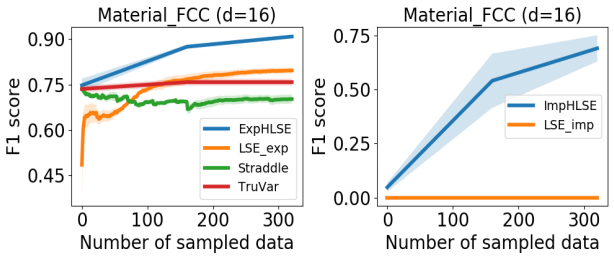

Figure 2: Material Design: F1-score for super-/sub-level set for *explicit* LSE (left plot) and *implicit* LSE (right plot). Our proposed methods are in blue while baselines are in other colors. Plotting mean and standard error over 3 repetitions. Method with higher F1-score is better.

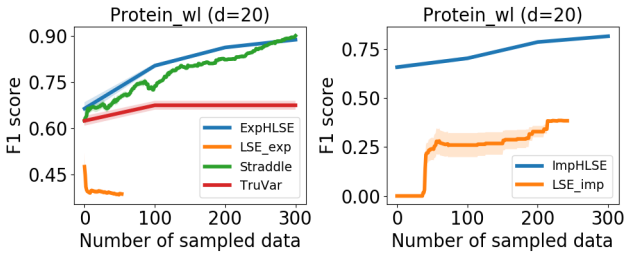

Figure 3: Protein Selection: F1-score for super-/sub-level set for *explicit* LSE (left plot) and *implicit* LSE (right plot). Our proposed methods are in blue while baselines are in other colors. Plotting mean and standard error over 3 repetitions. Method with higher F1-score is better.

of all elements is 100%. We aim to find the element compositions that achieve the desired properties, that is, when cast a room temperature (27^o), these composition results in an Face-Centered-Cubic (FCC) proportion of at least 95%. This problem can be formulated as an *explicit* or *implicit* LSE problem for which the explicit threshold level h is set as 0.95 or the implicit threshold ratio l is set as 0.95. Same with the current practice in material design, we use the Thermo-Calc software to assist in computing the thermochemical heterogeneous phase equilibria and conducting analysis for evaluating the FCC score of each element composition.

In Figure 2, we evaluate our proposed methods and other baselines using this alloy design problem. For the *explicit* LSE problem (left plot), our proposed method **ExpHLSE** outperforms other baseline methods significantly. In particular, **ExpHLSE** can achieve an F1-score of 90% within 300 sampled data while all the baseline methods can only achieve F1-scores of approximately $65\% - 80\%$ within the same number of sampled data. For the *implicit* LSE problem (right plot), our proposed method **ImpHLSE** also outperforms $\textbf{LSE}_{\text{imp}}$ by a very high margin. In this particular problem, because of its designed classification methodology, $\textbf{LSE}_{\text{imp}}$ does not identify any data point belonging to the super-level set, and hence, results in an F1-score of 0.

Protein Selection

In biotechnology, one of the important tasks is to construct new functional proteins by artificially modifying amino acid sequences of proteins (Karasuyama et al. 2018; Inatsu et al. 2019). Bio-engineers need to identify the region in the protein space where the protein satisfies the required functional properties. For this task, we use the Rhodopsin-family protein dataset provided in (Karasuyama et al. 2018). This family of proteins are commonly used in optogenetics as it can absorb a light with some certain wavelengths. The goal of this experiment is to estimate the region in the protein feature space where the absorption wavelength is sufficiently large for optogenetics usage. This dataset contains 796 proteins with each protein having an amino acid sequence vector and a scalar absorption wavelength output. Our goal is to identify the region of protein that (a) has the absorption wavelength larger than 562 (*explicit* LSE), or, (b) has the absorption wavelength larger than 90.4% of the maximum absorption wavelength (*implicit* LSE).

In Figure 3, we compare the performance of our proposed methods **ExpHLSE** and **ImpHLSE** with other baselines using this protein selection problem. For the *explicit* LSE problem, our proposed method **ExpHLSE** outperforms **TruVAR** and $\textbf{LSE}_{\text{exp}}$ significantly whilst performs slightly better than

$\textbf{LSE}_{\text{imp}}$. Note that for this problem, $\textbf{LSE}_{\text{exp}}$ completes the level set classification process very early (within a small number of sampled data), however, its classification accuracy is low. This can be explained that the GP approximation is not too accurate, and hence, it classifies the data points wrongly and also causes the early termination of $\textbf{LSE}_{\text{exp}}$. For the *implicit* LSE problem, $\textbf{LSE}_{\text{imp}}$ also stops the classification process early and it can achieve a maximum F1-score of 40% whilst our proposed method **ImpHLSE** can achieve a maximum F1-score of 80% within 300 sampled data points.

Algorithmic Assurance

We consider the algorithmic assurance problem, which is to find the domain of inputs where a given machine learning model $\mathcal{M}$ will perform as users expected (Sawade et al. 2010). Specifically, given $\mathcal{X}$ being the domain where the machine learning model $\mathcal{M}$ is applied, the problem is to find the domain $L_h = \{x \in \mathcal{X} \mid |y_x - \mathcal{M}(x)| \leq h\}$ or $L_l = \{x \in \mathcal{X} \mid |y_x - \mathcal{M}(x)| \leq l \max|y_x - \mathcal{M}(x)|\}$, where y_x is the true label of the data point x, $\mathcal{M}(x)$ is the predicted label by the machine learning model, and h, l are user-specified thresholds. This problem corresponds to the *explicit* and *implicit* LSE problem where the black-box function $f(x)$ is the deviation between the machine learning model output and the ground truth. The goal here is to estimate the domain L_h or L_l using the least number of ground truth sampled from the applied domain $\mathcal{X}$. For the *explicit* LSE problem, the threshold h is set in such a way that results in the volume of the sub-level set L_h being 80% of the domain X. For the *implicit* LSE problem, the implicit threshold ratio l is set in such a way that makes $l \max_{x \in \mathcal{X}} f(x)$ to be equal to the threshold h in the *explicit* LSE problem.

For this algorithmic assurance problem, we consider the problem of predicting the performance of highly configurable software systems. We aim to perform an algorithmic assurance analysis on a state-of-the-art machine learning model of predicting software performance, namely DeepPerf (Ha and Zhang 2019). We want to estimate the sub-level sets where DeepPerf performs as expected on different datasets (i.e. applied domain $\mathcal{X}$). We use two benchmark datasets published in (Siegmund et al. 2015) for the software performance prediction problem: HSMGP (3456 data points) and HIPACC (13485 data points). The input dimensions of HSMGP and HIPACC are 14, and 33, respectively.

In Figure 4, we show the performance of our proposed methods and other baselines. For the *explicit* LSE problem, our method **ExpHLSE** outperforms all baselines on all datasets. It especially performs better than **Straddle** and **TruVar** by a high margin. For the *implicit* LSE problem, our method **ImpHLSE** also outperforms the state-of-the art $\textbf{LSE}_{\text{imp}}$ significantly for both datasets.

Scope and Limitations

Low Dimensional LSE The proposed BNN approaches might be worse than the GP based methods when the dimension of the LSE problem is low or when the objective function is very simple. We ran the experiments with the functions Branin (d=2), Hartman3 (d=3) and the results show

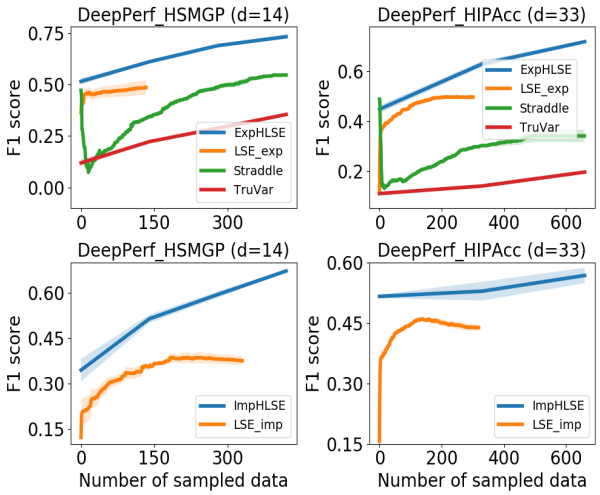

Figure 4: Algorithmic Assurance: F1-score for super-/sub-level set for *explicit* LSE (top row) and *implicit* LSE (bottom row). Our proposed methods are in blue while baselines are in other colors. Plotting mean and standard error over 3 repetitions. Method with higher F1-score is better.

that the GP-based approaches are better than the BNN approaches. Our rule of thumb is that when the input dimension is higher than 10, the BNN-based approaches perform better than the GP-based approaches and vice versa.

Run Time The run time of the BNN-based approaches, which includes both the hyper-parameter tuning and the BNN training process, is approximately 5-16 hours for each experiment. Specifically, it takes 1-4 hours each time tuning hyper-parameters and several minutes each time training the BNN; and for each experiment, with the batch size of $10d$, we only need to tune and train 3-5 times for the evaluation budget used. On the other hand, the run time of the GP-based methods, which includes the GP hyperparameter tuning and fitting process, is approximately 1-24 hours for each experiment. The slow run time of the GP-based methods is due to two reasons: 1) the batch sizes of the GP-based methods are set as 1 to achieve the best performance, and, 2) the sample budgets are large (as the input is high dimensional) and the GP fitting process is slow when the sampled data is large (GP complexity is cubic in the number of sampled data) (Krauth and E. Bonilla 2017).

Conclusion

This paper proposes novel methods to solve the *implicit* and *explicit* LSE problems using Bayesian Neural Networks. Using the uncertainty provided by the BNN, we derive new acquisition functions to sample data points and estimate the super- and sub-level sets in the most efficient manner. We also suggest a practical methodology to efficiently tune the network hyper-parameters to achieve high model accuracy. Numerical experiments on both synthetic and real-world datasets show that our proposed methods can achieve better results compared to state-of-the-art approaches.

Acknowledgements

The authors would like to thank Prof. Matthew Barnett and Ms. Manisha Senadeera for the dataset of the material design experiment. This research was partially funded by the Australian Government through the Australian Research Council (ARC). Prof. Venkatesh is the recipient of an ARC Australian Laureate Fellowship (FL170100006).

References

Bogunovic, I.; Scarlett, J.; Krause, A.; and Cevher, V. 2016. Truncated Variance Reduction: A Unified Approach to Bayesian Optimization and Level-Set Estimation. In *Advances in Neural Information Processing Systems (NeurIPS)*, 1515–1523.

Bryan, B.; and Schneider, J. 2008. Actively Learning Level-Sets of Composite Functions. In *Proceedings of the 25th International Conference on Machine Learning (ICML)*, 80–87.

Bryan, B.; Schneider, J.; Nichol, R.; Miller, C.; Genovese, C.; and Wasserman, L. 2005. Active Learning for Identifying Function Threshold Boundaries. In *Advances in Neural Information Processing Systems (NeurIPS)*, 163–170.

Freeman, L. 1965. *Elementary applied statistics*. John Wiley and Sons.

Gal, Y.; and Ghahramani, Z. 2016. Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning. In *Proceedings of The 33rd International Conference on Machine Learning (ICML)*, 1050–1059.

Gal, Y.; Islam, R.; and Ghahramani, Z. 2017. Deep Bayesian Active Learning with Image Data. In *Proceedings of the 34th International Conference on Machine Learning (ICML)*, 1183–1192.

Garnett, R.; Krishnamurthy, Y.; Xiong, X.; Schneider, J.; and Mann, R. 2012. Bayesian Optimal Active Search and Surveying. In *Proceedings of the 29th International Coference on International Conference on Machine Learning (ICML)*, 843–850.

Geifman, Y.; and El-Yaniv, R. 2019. Deep Active Learning with a Neural Architecture Search. In *Advances in Neural Information Processing Systems (NeurIPS)*, 5976–5986.

Gotovos, A.; Casati, N.; Hitz, G.; and Krause, A. 2013. Active Learning for Level Set Estimation. In *Proceedings of the Twenty-Third International Joint Conference on Artificial Intelligence*, IJCAI'13, 1344–1350.

Ha, H.; and Zhang, H. 2019. DeepPerf: Performance Prediction for Configurable Software with Deep Sparse Neural Network. In *International Conference on Software Engineering (ICSE)*.

Houlsby, N.; Huszár, F.; Ghahramani, Z.; and Lengyel, M. 2011. Bayesian Active Learning for Classification and Preference Learning. URL *https://arxiv.org/abs/1112.5745*.

Inatsu, Y.; Karasuyama, M.; Inoue, K.; and Takeuchi, I. 2019. Active learning for level set estimation under cost-dependent input uncertainty. URL *https://arxiv.org/abs/1909.06064*.

Iwazaki, S.; Inatsu, Y.; and Takeuchi, I. 2019. Bayesian Experimental Design for Finding Reliable Level Set under Input Uncertainty. URL *https://arxiv.org/abs/1910.12043*.

Jones, D. 2001. A Taxonomy of Global Optimization Methods Based on Response Surfaces. *Journal of Global Optimization* 21(4): 345–383.

Karasuyama, M.; Inoue, K.; Nakamura, R.; Kandori, H.; and Takeuchi, I. 2018. Understanding Colour Tuning Rules and Predicting Absorption Wavelengths of Microbial Rhodopsins by Data-Driven Machine-Learning Approach. *Nature Scientific Report* 8: 15580.

Krauth, K.; and E. Bonilla, K. Cutajar, M. F. 2017. AutoGP: Exploring the Capabilities and Limitations of Gaussian Process Models. In *Proceedings of the Thirty-Third Conference on Uncertainty in Artificial Intelligence (UAI)*.

MacKay, D. 1992. A Practical Bayesian Framework for Backpropagation Networks. *Neural Computation* 4(3): 448–472.

Neal, R. 1995. *Bayesian learning for neural networks*. Ph.D. thesis, University of Toronto.

Rasmussen, C.; and Williams, C. 2006. *Gaussian Processes for Machine Learning*. MIT Press.

Sawade, C.; Landwehr, N.; Bickel, S.; and Scheffer, T. 2010. Active Risk Estimation. In *Proceedings of the 27th International Conference on International Conference on Machine Learning (ICML)*, 951–958.

Shahriari, B.; Swersky, K.; Wang, Z.; Adams, R. P.; and de Freitas, N. 2016. Taking the Human Out of the Loop: A Review of Bayesian Optimization. *Proceedings of the IEEE* 104(1): 148–175.

Shannon, C. E. 1948. A mathematical theory of communication. *The Bell System Technical Journal* 27(3): 379–423.

Shekhar, S.; and Javidi, T. 2019. Multiscale Gaussian Process Level Set Estimation. In *Proceedings of the 22nd International Conference on Artificial Intelligence and Statistics (AISTATS)*, 3283–3291.

Siegmund, N.; Grebhahn, A.; Apel, S.; and Kästner, C. 2015. Performance-influence Models for Highly Configurable Systems. In *Proceedings of the 2015 10th Joint Meeting on Foundations of Software Engineering*, ESEC/FSE 2015, 284–294.

Snoek, J.; Larochelle, H.; and Adams, R. 2012. Practical Bayesian Optimization of Machine Learning Algorithms. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2951—-2959.

Sui, Y.; Gotovos, A.; Burdick, J.; and Krause, A. 2015. Safe Exploration for Optimization with Gaussian Processes. In *Proceedings of the 32nd International Conference on International Conference on Machine Learning (ICML)*, 997–1005.

Sui, Y.; Zhuang, V.; Burdick, J.; and Yue, Y. 2018. Stagewise Safe Bayesian Optimization with Gaussian Processes. In *Proceedings of the 35th International Conference on Machine Learning (ICML)*, 4781–4789.

Zanette, A.; Zhang, J.; and Kochenderfer, M. 2019. Robust Super-Level Set Estimation Using Gaussian Processes. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, 276–291.