Proceedings of the 2018 Winter Simulation Conference
M. Rabe, A. A. Juan, N. Mustafee, A. Skoogh, S. Jain, and B. Johansson, eds.

DESIGNING IMPORTANCE SAMPLERS TO SIMULATE MACHINE LEARNING PREDICTORS VIA OPTIMIZATION

Zhiyuan Huang

Department of IOE
University of Michigan
1205 Beal Avenue
Ann Arbor, MI 48105, USA

Henry Lam

Department of IEOR
Columbia University
500 W. 120th Street
New York, NY 10027, USA

Ding Zhao

Department of Mechanical Engineering
Carnegie Mellon University
5000 Forbes Avenue
Pittsburgh, PA 15213, USA

ABSTRACT

We study the problem of designing good importance sampling (IS) schemes to simulate the probability that a sophisticated predictor, built for instance from an off-the-shelf machine learning toolbox, gives a prediction that exceeds a large threshold. This problem is motivated as a step towards building good learning algorithms that takes into account the extremal risks of the prediction. We provide a framework to design IS for two common machine learning models, namely random forest and a basic neural network. Our approach utilizes some available mathematical programming formulations to optimize over these models and a simple "cutting plane" idea to look for dominating points under Gaussian input distributions.

1 INTRODUCTION

We consider the problem of estimating $P(g(X) > \gamma)$ via simulation, where $X \in \mathbb{R}^d$ is a random input and $\gamma \in \mathbb{R}$ is a high threshold, so that the probability is small and leads to a rare-event simulation problem. We are interested particularly in $g(\cdot)$ that is the response of a sophisticated predictor, built for instance from an off-the-shelf machine learning toolbox. This problem is motivated as a step towards building good learning methods that take into account the extremal risks of machine learning prediction.

To be more concrete, suppose we are interested in training the parameter θ of a predictor, say $Y_\theta(X)$, by minimizing a risk criterion $E[L(Y_\theta(X), Y(X))]$, where $Y(\cdot)$ is the true response function. Suppose that this loss function L exerts a big value in some "hidden region" where the occurrence of X is rare. The data in this case may not reveal satisfactorily this hidden risk. An approach to learn a good risk-conscious predictor in this situation is to fit the data using some probability distribution, and use it to train the predictor. A judicious choice of the distribution allows one to generate Monte Carlo samples, and to use importance sampling (IS) to populate more samples in the rare-event set. Of course, this approach requires various aspects of considerations, both statistically and practically. However, one step towards handling such type of problems involves how to speed up simulation for a machine learning model, which is precisely what we address in this paper.

As a motivating example, autonomous vehicle (AV) design often uses predictions of the movements of surrounding cars to decide its own behavior (e.g. Jain et al. 2015). Predictions that seriously deviate from the reality can post substantial danger and catastrophic failure. However, unusual surrounding driving movements that lead to serious deviations of the predictions are rare. In this case, one can represent the driving environment as a stochastic model (the X), which provides inputs for the prediction model (the $Y_\theta(X)$) (e.g. Wang et al. 2017; Huang et al. 2017), and a risk-sensitive predictive training could involve rare-event simulation on functionals associated with $Y_\theta(X)$.

2 PROBLEM SETTING

We state more precisely our problem setting. Suppose we are given a prediction model $g(\cdot)$, with the input $X \in \mathbb{R}^d$ and the output $g(X) \in \mathbb{R}$. Suppose that the input follows a standard Gaussian distribution, i.e, $X \sim N(0, I_d)$, where I_d is the $d \times d$ identity matrix. We want to estimate the probability $p = P(g(X) \geq \gamma)$, where $\gamma \in \mathbb{R}$ is a threshold that triggers a rare event. We note that the Gaussian assumption can be relaxed without much difficulty in our framework to, for instance, mixtures of Gaussians, which can expand our scope of applicability. For simplicity, however, we will concentrate on the standard Gaussian distribution in this paper.

When p is small, estimation using crude Monte Carlo is challenging due to large variance. A common approach to speed up simulation in such contexts is to use IS (see, e.g. the surveys Bucklew 2013; Asmussen and Glynn 2007; Rubinstein and Kroese 2016; Glasserman 2013; Juneja and Shahabuddin 2006; Blanchet and Lam 2012, among others). Suppose X has a density f. The basic idea of IS is to change the sampling distribution to say $\tilde{f}$, and output

$$Z = I(g(X) \geq \gamma) \frac{f(X)}{\tilde{f}(X)}, \tag{1}$$

where X is now sampled from $\tilde{f}$. This output is unbiased if f is absolutely continuous with respect to $\tilde{f}$ over the rare-event set $\{x : g(x) \geq \gamma\}$. Moreover, by choosing $\tilde{f}$ appropriately, one can substantially reduce the simulation variance.

In the case of Gaussian input distributions, finding a good $\tilde{f}$ is particularly handy and one approach to devise good IS distributions uses the notion of so-called dominating point. Starting from a standard Gaussian distribution, a well-known IS scheme is as follows (see, e.g., Sadowsky and Bucklew 1990; Dieker and Mandjes 2006; Blanchet and Li 2011; Adler et al. 2012):

1. If there exists a dominating point a, i.e., a point a such that $a = \arg\min_x\{\|x\|^2 : g(x) \geq \gamma\}$ and $\{x : g(x) \geq \gamma\} \subseteq \{x : a'(x-a) \geq 0\}$, then we use a Gaussian distribution with mean at a as the IS distribution $\tilde{f}$.
2. If we can split $\{x : g(x) \geq \gamma\}$ into $\mathscr{R}_1, ..., \mathscr{R}_r$, and for each $\mathscr{R}_i, i = 1, ..., r$ there exists a dominating point a_i such that $a_i = \arg\min_x\{\|x\|^2 : x \in \mathscr{R}_i\}$ and $\mathscr{R}_i \subseteq \{x : a_i'(x-a_i) \geq 0\}$, then we use a Gaussian mixture distribution with r components as the IS distribution $\tilde{f}$, where the ith component has mean a_i.

Note that the quantity $\arg\min_x\{\|x\|^2 : g(x) \geq \gamma\}$ is equivalent to $\arg\max_x\{\phi(x) : g(x) \geq \gamma\}$, where ϕ is the standard Gaussian density. The condition $2a'(x-a) \geq 0$ is the first order condition of optimality for the optimization $\min_x \|x\|^2$ over a convex set for x, and so the first situation above occurs if $\{x : g(x) \geq \gamma\}$ is a convex set (though it can be more general). The proposals above guarantee the so-called asymptotic efficiency or logarithmic efficiency (e.g., Asmussen and Glynn 2007) of the IS, meaning that the relative error, measured by the ratio of standard deviation of a single IS output over the probability of interest, grows at most polynomially in the location of the rare-event set.

Given the proposal above, one can follow Algorithm 1 to obtain the dominating points $a_1, ..., a_r$ to build an efficient IS distribution in our prediction model context. The procedure uses a sequential "cutting plane" approach to exhaustively look for all dominating points, by reducing the search space at each iteration via taking away the regions covered by the existing dominating points. The set A in the procedure serves to store the dominating points we have found throughout the procedure. At the end of the procedure, one obtain a set A that contains all the dominating points $a_1, ..., a_r$.

Algorithm 1: Procedure to find all dominating points for the set $\{x : g(x) \geq \gamma\}$.

Input: Prediction model $g(x)$, threshold γ.

Output: Dominating-points set A.

Start with $A = \emptyset$;

While $\{x : g(x) \geq \gamma, a_i'(x - a_i) < 0,\ \forall a_i \in A\} \neq \emptyset$ **do**

Find a dominating point a by solving the optimization problem

$$\begin{aligned} a = \arg\min_{x} \quad & \|x\|^2 \\ s.t. \quad & g(x) \geq \gamma \\ & a_i'(x - a_i) < 0, \text{ for } \forall a_i \in A \end{aligned} \tag{2}$$

and update $A \leftarrow A \cup \{a\}$;

End

To apply Algorithm 1 to find all dominating points, the key is to be able to solve the optimization problems (2). We will investigate this for two popular prediction models, random forest and neural network (NN). Section 3 studies the tractable formulations using some recent work on optimization over these models, while Section 4 applies them to design IS schemes and demonstrate some numerical results.

3 TRACTABLE FORMULATION FOR PREDICTION MODELS

We discuss how to formulate the optimization problems in Algorithm 1 as a mixed integer program (MIP) with quadratic objective function and linear constraints. Sections 3.1 and 3.2 focus on a basic NN and random forest respectively. Section 3.3 then further discusses solving larger-scale problems using a simple bisection method that leverages existing techniques in Benders decompositions in the random forest case.

3.1 Tractable Formulation for Neural Network

We consider a NN with L layers and for each layer, the number of neurons is $n_1, ..., n_L$. (As we focus on output $g(x) \in \mathbb{R}$, we have $n_L = 1$.) We use rectified linear unit (ReLU) for each neuron, i.e., the activation function for each neuron is $\max\{0, x\}$. The input of the jth neuron in layer i is weighted from the output of the previous layer by a vector $w_i^j \in \mathbb{R}^{n_{i-1}}$ and is added by a bias $b_i^j \in \mathbb{R}$. We use $s_i \in \mathbb{R}^{n_i}, i = 1, ..., L$ to represent the output of the ith layer. At the ith layer, given the output from the $(i-1)$th layer s_{i-1}, we have $s_i \in \mathbb{R}^{n_i}$, where the jth element of s_i is given by $s_i^j = \max\{w_i^{jT} s_{i-1} + b_i^j, 0\}$. For further details on such type of NNs, see, e.g., Goodfellow et al. (2016).

Here we reformulate (2) by replacing $g(x) \geq \gamma$ with constraints that represent the structure of the NN:

$$\begin{aligned} & s_L \geq \gamma \\ & s_i^j = \max\{w_i^{jT} s_{i-1} + b_i^j, 0\},\ i = 1, ..., L,\ j = 1, ..., n_i \\ & s_0 = x. \end{aligned}$$

Tjeng and Tedrake (2017) discussed one approach to express ReLU as a mixed integer model, by introducing auxiliary binary variables z. For $y = max(x, 0)$, if we have $l \leq x \leq u$, where $l \leq 0$ and $u \geq 0$ are the smallest and largest possible values of x, then we set $z = I(x \geq 0)$. We can reformulate $y = max(x, 0)$ as a set of linear constraints: $y \leq x - l(1 - z);\ y \geq x;\ y \leq uz;\ y \geq 0;\ z \in \{0, 1\}$.

Using this approach, we rewrite problem (2) with $A = \emptyset$ as

$$\min_{x, s_0, ..., s_L, z_1 ..., z_L} \quad \|x\|^2 \tag{3}$$

$$
\begin{aligned}
s.t.\ \ & s_L \geq \gamma \\
& s_i \leq W_i^T s_{i-1} + b_i - l(1-z_i),\ i = 1,...,L \\
& s_i \geq W_i^T s_{i-1} + b_i,\ i = 1,...,L \\
& s_i \leq u z_i,\ i = 1,...,L \\
& s_i \geq 0,\ i = 1,...,L \\
& z_i \in \{0,1\}^{n_i},\ i = 1,...,L \\
& s_0 = x,
\end{aligned}
$$

where $W_i = [w_i^1,...,w_i^{n_i}] \in \mathbb{R}^{n_{i-1} \times n_i}$, $b_i = [b_i^1,...,b_i^{n_i}]^T \in \mathbb{R}^{n_i}$, and $s_i = [s_i^1,...,s_i^{n_i}]^T \in \mathbb{R}^{n_i}$.

Note that this optimization has a quadratic objective and linear constraints. Similarly, we can formulate (2) by adding linear constraints $a_i'(x-a_i) < 0,\ \forall a_i \in A$ to (3), which arrives at the same optimization class. Medium-size instances of these problems can be handled by standard solvers.

3.2 Tractable Formulation for Random Forest

To look for dominating points in a random forest or tree ensemble, we follow the route in Mišic (2017) that studies optimization over these models. We consider a random forest as follows. The input x has d dimensions. Suppose the model consists of T trees $f_1,...,f_T$. In each tree f_t, we use $a_{i,j}$ to denote the jth unique split point for the ith dimension of the input x, such that $a_{i,1} < a_{i,2} < ... < a_{i,K_i}$, where K_i is the number of unique split points for the ith dimension of x.

Following the notations of Mišic (2017), let $\mathbf{leaves}(t)$ be the set of leaves (terminal nodes) of tree t and $\mathbf{splits}(t)$ be the set of splits (non-terminal nodes) of tree t. In each split s, we let $\mathbf{left}(t)$ be the set of leaves that are accessible from the left branch (the query at s is true), and $\mathbf{right}(t)$ be the set of leaves that are accessible from the right branch (the query at s is false). For each node s, we use $\mathbf{V}(s) \in \{1,...,d\}$ to denote the dimension that participate in the node and $\mathbf{C}(s) \in \{1,...,K_{\mathbf{V}(s)}\}$ to denote the set of values of dimension i that participate in the split query of s ($\mathbf{C}(s) = \{j\}$ and $\mathbf{V}(s) = \{i\}$ indicates the query $x_i \leq a_{i,j}$).

We use λ_t to denote the weight of tree t ($\sum_{t=1}^T \lambda_t = 1$). For each $l \in \mathbf{leaves}(t)$, $p_{t,l}$ denotes the output for the lth leaf in tree t.

To formulate the random forest optimization as an MIP, we introduce binary decision variables $z_{i,j}$ and $y_{t,l}$. Firstly, we have

$$z_{i,j} = I(x_i \leq a_{i,j}),\ i = 1,...,d,\ j = 1,...,K_i. \tag{4}$$

We then use $y_{t,l} = 1$ to denote that tree t outputs the prediction value $p_{t,l}$ on leaf l, and $y_{t,l} = 0$ otherwise. We use $\mathbf{z}, \mathbf{y}$ to represent the vectors of $z_{i,j}$ and $y_{t,l}$ respectively. For the input x, we assume that $x \in [-B,B]^d$ and $|a_{i,j}| \leq B$. Then (4) is represented by the following constraints

$$
\begin{aligned}
& x_i \leq a_{i,j} + 2(1-z_{i,j})B \\
& x_i > a_{i,j} - 2z_{i,j}B.
\end{aligned}
$$

Now we formulate (2) with $A = \emptyset$ as the following MIP

$$
\begin{aligned}
\min_{x,\mathbf{y},\mathbf{z}}\ & \|x\|^2 \\
s.t.\ & \sum_{t=1}^T \sum_{l \in \mathbf{leaves}(t)} \lambda_t p_{t,l} y_{t,l} \geq \gamma \\
& \sum_{l \in \mathbf{leaves}(t)} y_{t,l} = 1
\end{aligned} \tag{5}
$$

$$\sum_{l\in\textbf{left}(t)} y_{t,l} \le \sum_{j\in\textbf{C}(s)} z_{\textbf{V}(s),j},\ \forall t\in\{1,...,T\},\ s\in\textbf{splits}(t)$$
$$\sum_{l\in\textbf{right}(t)} y_{t,l} \le 1-\sum_{j\in\textbf{C}(s)} z_{\textbf{V}(s),j},\ \forall t\in\{1,...,T\},\ s\in\textbf{splits}(t)$$
$$z_{i,j}\le z_{i,j+1},\ \forall i\in\{1,...,d\},\ j\in\{1,...,K_i-1\}$$
$$z_{i,j}\in\{0,1\},\ \forall i\in\{1,...,d\},\ j\in\{1,...,K_i\}$$
$$y_{t,l}\ge 0,\ \forall t\in\{1,...,T\},\ l\in\textbf{leaves}(t)$$
$$x_i\le a_{i,j}+2(1-z_{i,j})B$$
$$x_i > a_{i,j}-2z_{i,j}B.$$

This formulation again has a quadratic objective function and linear constraints. Similarly, we can formulate (2) with $A\neq\emptyset$ by adding linear constraints $a_i'(x-a_i)<0,\ \forall a_i\in A$ to (5).

3.3 Bisection Algorithm and the Benders Decomposition for Solving Larger-scale Problems

The MIPs (5) are already tractable for small- and medium-size problems. Nonetheless, because of the special structure of (5), we can obtain, for larger-scale problems, an efficient algorithm based on a bisection on a "dual" form of the problem. The latter is a linear MIP and can be solved by the Benders decomposition considered in Mišic (2017).

We consider the "dual" form problem of (2) with $A=\emptyset$:

$$\begin{aligned} h(\eta) = \max_x\ & g(x) \\ s.t.\ & \|x\|^2\le\eta. \end{aligned} \tag{6}$$

Note that (6) is non-decreasing in η. Moreover, we know that if η is above the optimal objective value for (2) with $A=\emptyset$, then (6) will output an objective value at least γ. On the other hand, if η is below that, then (6) will output a value less than γ. This motivates us to propose a bisection method that does a line search on the smallest value of η that gives $g(x)\ge\gamma$. This is summarized in Algorithm 2.

Note that Algorithm 2 converges to the solution of (5) as ε goes to 0. Also, the number of iterations we need is given by $N_{iter}<\log\frac{M}{\varepsilon}/\log 2$. Therefore, as long as we can solve (6) efficiently, the same is true for the dominating point problem. But, because the quadratic constraint in (6) only results in eliminating some of the dummy binary variables in the MIP formulation, we recover the formulation suggested in Mišic (2017), but with less variables, to represent (6). This means that (6) is equivalent to

$$\max_{x,\mathbf{y},\mathbf{z}}\ \sum_{t=1}^{T}\sum_{l\in\textbf{leaves}(t)}\lambda_t p_{t,l} y_{t,l} \tag{10}$$
$$s.t.\ \sum_{l\in\textbf{leaves}(t)} y_{t,l}=1$$
$$\sum_{l\in\textbf{left}(t)} y_{t,l} \le \sum_{j\in\textbf{C}(s)} z_{\textbf{V}(s),j},\ \forall t\in\{1,...,T\},\ s\in\textbf{splits}(t)$$
$$\sum_{l\in\textbf{right}(t)} y_{t,l} \le 1-\sum_{j\in\textbf{C}(s)} z_{\textbf{V}(s),j},\ \forall t\in\{1,...,T\},\ s\in\textbf{splits}(t)$$
$$\|x\|^2\le\eta \tag{11}$$
$$z_{i,j}\le z_{i,j+1},\ \forall i\in\{1,...,d\},\ j\in\{1,...,K_i-1\} \tag{12}$$
$$z_{i,j}\in\{0,1\},\ \forall i\in\{1,...,d\},\ j\in\{1,...,K_i\} \tag{13}$$
$$x_i\le a_{i,j}+2(1-z_{i,j})B \tag{14}$$

Algorithm 2: Bisection algorithm for solving (2) **with** $A = \emptyset$**.**

Input: Prediction model $g(x)$, threshold γ, tolerance parameters ε.
Output: Optimal solution a.

Set $\eta_1 = 0$ and $\eta_2 = M$, where M is large enough (e.g. $M = dB^2$);
Solve

$$\gamma_1 = h(\eta_1) \tag{7}$$

and

$$\gamma_2 = h(\eta_2); \tag{8}$$

While $\eta_2 - \eta_1 \geq \varepsilon$ **do**
    Update $\eta_m = \frac{\eta_1+\eta_2}{2}$;
    Solve

$$\gamma_m = h(\eta_m); \tag{9}$$

    If $\gamma_m \geq \gamma$ **do**
        Update $\eta_2 \leftarrow \eta_m$;
    Else do
        Update $\eta_1 \leftarrow \eta_m$;
    End;
End;
Update a using the optimal solution x from (8) ;

$$x_i > a_{i,j} - 2z_{i,j}B \tag{15}$$
$$y_{t,l} \geq 0,\ \forall t \in \{1,...,T\},\ l \in \textbf{leaves}(t).$$

In (10), let us consider $x, \mathbf{z}$ as one set of variables and $\mathbf{y}$ as the other set of variables, where $\mathbf{y}$ can be further partitioned as $\mathbf{y} = (\mathbf{y}_1, ..., \mathbf{y}_T)$ and $\mathbf{y}_T$ consists of $y_{t,l}$'s. We observe that for any two trees $t \neq t'$, $\mathbf{y}_t$ and $\mathbf{y}_{t'}$ does not appear together in any constraints and are only linked through $\mathbf{z}$. This observation allows us to use the Benders decomposition as Mišic (2017) suggests.

We rewrite problem (10) as follows:

$$\max_{x,\mathbf{z}} \ \sum_{t=1}^{T} \lambda_t G_t(\mathbf{z}) \tag{16}$$
$$s.t. \ \text{ constraints (11)-(15),}$$

where $G_t(\mathbf{z})$ is the optimal value of the following subproblem:

$$G_t(\mathbf{z}) = \max_{\mathbf{y},\mathbf{z}} \ \sum_{l \in \textbf{leaves}(t)} \lambda_t p_{t,l} y_{t,l} \tag{17}$$
$$s.t. \ \sum_{l \in \textbf{leaves}(t)} y_{t,l} = 1$$
$$\sum_{l \in \textbf{left}(t)} y_{t,l} \leq \sum_{j \in \mathbf{C}(s)} z_{\mathbf{V}(s),j},\ \forall t \in \{1,...,T\},\ s \in \textbf{splits}(t)$$
$$\sum_{l \in \textbf{right}(t)} y_{t,l} \leq 1 - \sum_{j \in \mathbf{C}(s)} z_{\mathbf{V}(s),j},\ \forall t \in \{1,...,T\},\ s \in \textbf{splits}(t)$$
$$y_{t,l} \geq 0,\ \forall t \in \{1,...,T\},\ l \in \textbf{leaves}(t).$$

The dual of the linear problem (17) is

$$\min_{\alpha_t,\beta_t,\zeta_t} \ \sum_{s \in \textbf{splits}} \alpha_{t,s} \left[\sum_{j \in \mathbf{C}(s)} z_{\mathbf{V}(s),j} \right] + \sum_{s \in \textbf{splits}} \beta_{t,s} \left[1 - \sum_{j \in \mathbf{C}(s)} z_{\mathbf{V}(s),j} \right] + \zeta_t \tag{18}$$

$$s.t. \quad \sum_{s:l\in\textbf{left}(t)} \alpha_{t,s} + \sum_{s:l\in\textbf{right}(t)} \beta_{t,s} + \zeta_t \geq p_{t,l},\ \forall l \in \textbf{leaves}(t)$$
$$\alpha_{t,s}, \beta_{t,s} \geq 0,\ s \in \textbf{splits}.$$

We use $\mathscr{D}_t$ to denote the set of feasible $(\alpha_t, \beta_t, \zeta_t)$ for subproblem (18) of tree t. We write (16) in terms of the dual variables $(\alpha_t, \beta_t, \zeta_t)$:

$$\max_{x,\mathbf{z},\theta} \quad \sum_{t=1}^{T} \lambda_t \theta_t \tag{19}$$

$$s.t. \quad \sum_{s\in\textbf{splits}} \alpha_{t,s} \left[\sum_{j\in\mathbf{C}(s)} z_{\mathbf{V}(s),j} \right] + \sum_{s\in\textbf{splits}} \beta_{t,s} \left[1 - \sum_{j\in\mathbf{C}(s)} z_{\mathbf{V}(s),j} \right] + \zeta_t \geq \theta_t,\ \forall (\alpha_t, \beta_t, \zeta_t) \in \mathscr{D}_t, t \in \{1,...,T\} \tag{20}$$

constraints (11)-(15).

With problem (19), we can use a constraint generating scheme. We start with solving problem (19) using a subset of $\bar{\mathscr{D}}_t \subseteq \mathscr{D}_t$ in the constraint (20). We then solve (18) for each tree t to check if there exists a solution $(\alpha_t, \beta_t, \zeta_t)$ for which the constraint (20) is violated. If so, we add the constraint to (19) and solve it again; otherwise, we obtain the optimal solution. For a quick method to solve problem (18), please refer to Proposition 5 in Mišic (2017).

The decomposition of (2) with non-empty A follows the same route as above and the resulting formulation is very similar. The dual of the subproblem is exactly the same as (18), while the main problem is (19) with a set of additional constraints $a_i'(x - a_i) < 0,\ \forall a_i \in A$.

4 APPLICATIONS TO IMPORTANCE SAMPLING AND NUMERICAL EXPERIMENTS

In this section, we use several sets of simple experiments to illustrate the IS scheme using dominating points generated from Algorithm 1, for the described NN and random forest. In the first set of problems, we consider an example where there is only one dominating point. In the second set of problems, we solve a problem with multiple dominating points.

To illustrate the efficiency of the proposed IS scheme, we compare with a naive use of a uniform IS estimator as follows. Consider a problem where X follows a distribution $f(x)$, and the set $\{x : g(x) \geq \gamma\}$ is known to lie inside $[l,u]^d$ where d is the dimension of the input variable X. The uniform IS estimator is given by:

$$Z_{uniform} = I(g(X) \geq \gamma) f(X) (u-l)^d,$$

where X is generated from a uniform distribution on $[l,u]^d$. This estimator has a polynomially growing relative efficiency as the magnitude of the dominating points grow, but the efficiency also depends significantly on the size of the bounded set, i.e., l, u, d.

4.1 Neural Network Example: IS with A Single Dominating Point

We recall the problem setting that the input X follows a standard Gaussian distribution, and we are interested in estimating the probability $P(g(X) \geq \gamma)$. Here $g(x)$ represents the output of a NN prediction at x and γ is a real-valued threshold. The NN has 3 layers with 100 neurons in each of the 2 hidden layers, and all neurons are ReLU. We consider only X in the region $[0,5]^2$, so that $g(x)$ can be thought of as being set to 0 outside this box.

The NN is trained as follows. We generate 2,601 samples using a uniform grid over the space $[0,5]^2$ with a mesh of 0.1 on each coordinate. For the input $x = [x_1, x_2]$, we use the function

$$y(x) = (x_1 - 5)^3 + (x_2 - 4.5)^3 + (x_1 - 1)^2 + x_2^2 + 500 \tag{21}$$

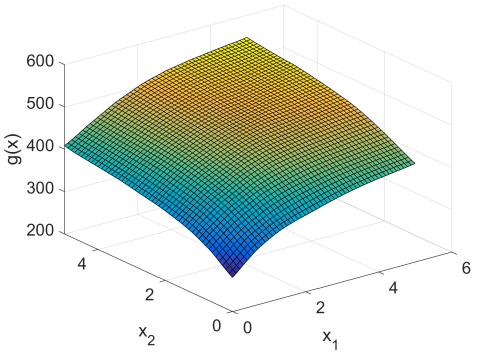

Figure 1: Response surface of the neural network.

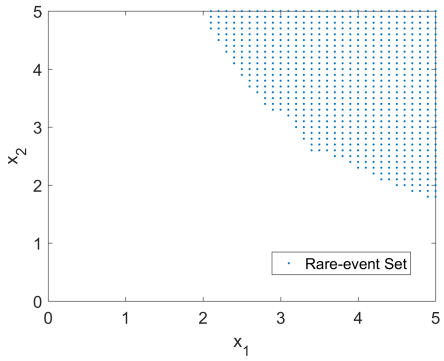

Figure 2: Rare-event set $\{x : g(x) \geq \gamma\}$ of the neural network.

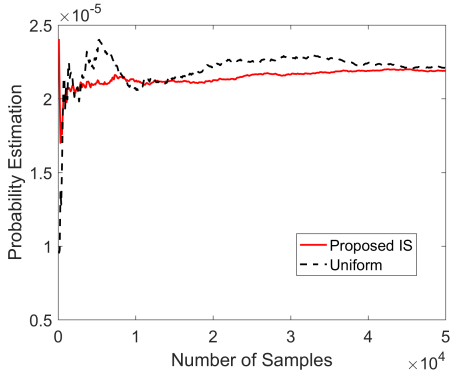

Figure 3: Probability estimation with different numbers of samples.

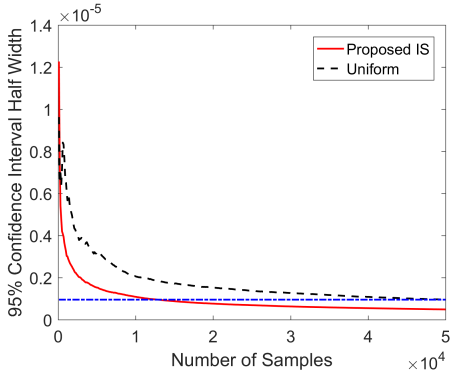

Figure 4: 95% confidence interval half-width with different numbers of samples.

to generate output value as the "truth" . We obtain the dataset $D = \{(X_n, Y_n)\}$ and use it to train the NN with gradient descent. We present the response surface of $g(x)$ in Figure 1. We use $\gamma = 500$ in this example and the shape of the rare-event set $\{x : g(x) \geq \gamma\}$ in this case is presented in Figure 2. We observe that the set is roughly convex and should have a single dominating point. By solving (3), we obtain the dominating point for the set at $(3.3676, 2.6051)$.

We use the obtained dominating point to construct the IS estimator described in (1). In Figures 3 and 4, we compare the performance of the proposed IS scheme with that of the uniform IS estimator. Figure 3 shows the estimated probabilities of the two estimators as the number of samples increases in a single sample path. We observe that the uniform IS (black dash line) has more fluctuations than the proposed IS (red solid line), which indicates that the proposed IS gives more stable estimates. This observation is confirmed in Figure 4 that shows the half-width of the 95% confidence intervals of the two estimators as the number of samples varies. Our IS appears to have shorter confidence intervals and it only takes about 12,000 samples for our IS to reach the accuracy that the uniform IS reaches with 50,000 samples.

4.2 Neural Network Example: IS with Multiple Dominating Points

We now consider true output values generated according to the function

$$y(x) = 10 \times e^{-\left(\frac{x_1-5}{3}\right)^2 - \left(\frac{x_2-5}{4}\right)^2} + 10 \times e^{-{x_1}^2 - (x_2-4.5)^2}. \tag{22}$$

We use a uniform grid over $[0,5]^2$ with a mesh of 0.1 on each coordinate to train a neural network with 2 hidden layers, 100 neurons in the first hidden layer and 50 neurons in the second hidden layer. All neurons in the neural network are ReLU. The response surface of the trained model $g(x)$ is shown in Figure 5. We

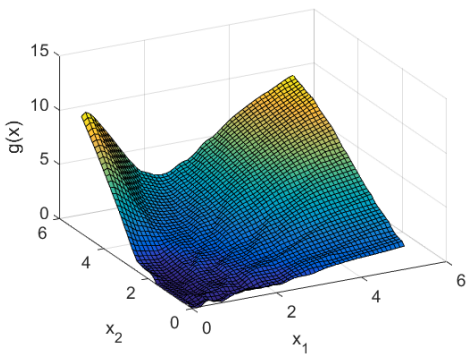

Figure 5: Response surface of the neural network.

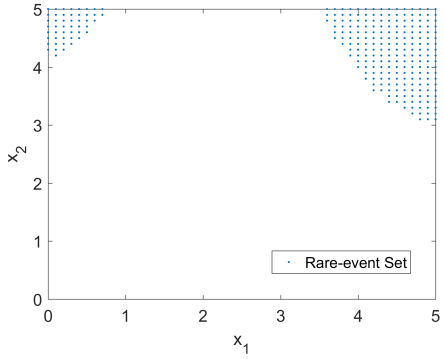

Figure 6: Rare-event set $\{x : g(x) \geq \gamma\}$ of the neural network.

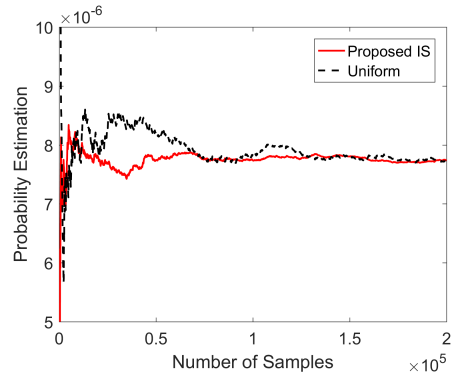

Figure 7: Probability estimation with different numbers of samples.

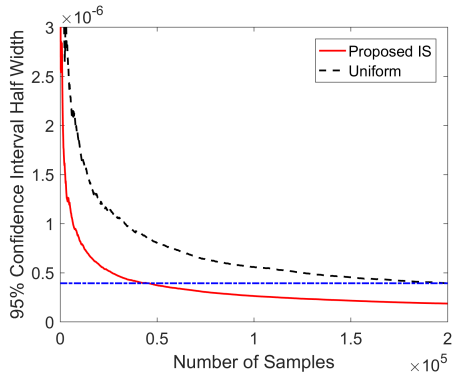

Figure 8: 95% confidence interval half-width with different numbers of samples.

set $\gamma = 8$. The shape of the rare-event set is shown in Figure 6. We observe that the set now consists of two separate regions and therefore we expect to obtain multiple dominating points. Using Algorithm 1 with the formulation in Section 3.1, we obtain two dominating points, $(0.113, 4.162)$ and $(4.187, 3.587)$. We use these dominating points to construct a mixture distribution, as discussed in Section 2, as the IS distribution.

The comparison between the proposed IS scheme and the uniform IS is shown in Figures 7 and 8. In Figure 7, the probability estimates of the proposed IS (red solid line) appear more stable than that of the uniform IS (black dash line). Figure 8 further shows that the confidence intervals of the proposed IS are shorter. The efficiency of the proposed IS is roughly 4 times better than the uniform IS, considering that the confidence interval half-width of the proposed IS at 50,000 samples is similar to the uniform IS at 200,000 samples.

Thus, comparing with uniform IS, the estimation accuracy of the proposed IS is better in both experiments, demonstrating the efficiency of the IS scheme obtained in the formulation in Section 3.1.

4.3 Random Forest Example: IS with A Single Dominating Point

Consider now that $g(x)$ represents a random forest. The random forest is trained from samples generated uniformly over the space $[0,5]^2$ with a mesh 0.25 on each coordinate, with output values from (21). The trained random forest model consists of 3 trees with 563, 535, 565 nodes respectively. The response surface of the model $g(x)$ is shown in Figure 9. Here we use $\gamma = 500$ and the shape of the rare-event set $\{x : g(x) \geq \gamma\}$ in presented in Figure 10. We use Algorithm 2 to obtain the dominating point of the

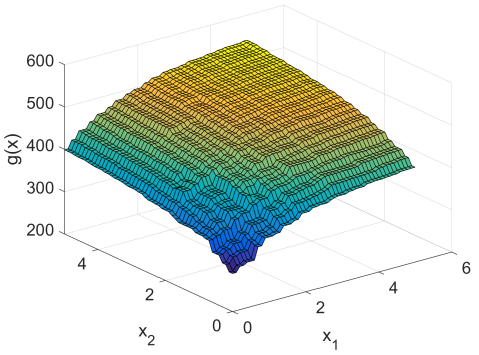

Figure 9: Response surface of the random forest.

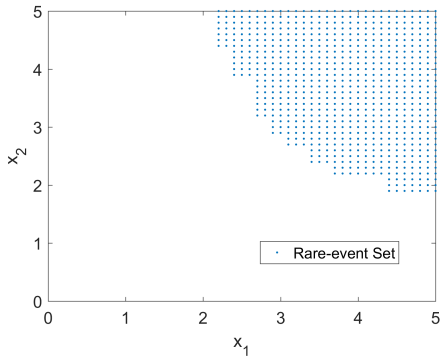

Figure 10: Rare-event set $\{x : g(x) \geq \gamma\}$ of the random forest.

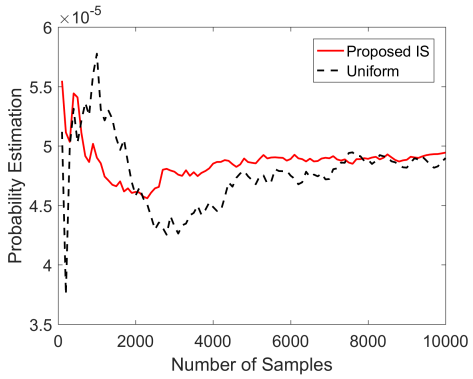

Figure 11: Probability estimation with different numbers of samples.

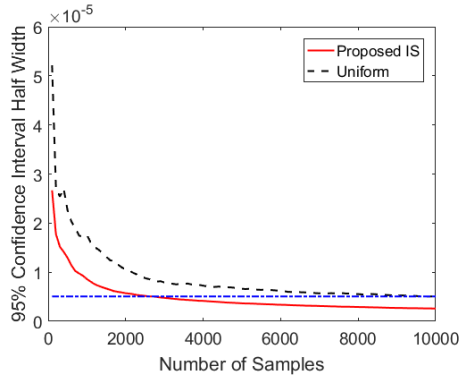

Figure 12: 95% confidence interval half-width with different numbers of samples.

rare-event set. In the algorithm, we choose the tolerance parameters to be $\varepsilon = 0.01$. This gives us the dominating point for the set at $(3.00001, 2.62501)$.

We use the proposed IS and the uniform IS to estimate the probability $P(g(x) \geq \gamma)$. In Figure 11, we observe that the estimates from the uniform IS (black dash line) are relatively unstable, compared to those from the proposed IS (red solid line). In Figure 12, the confidence intervals of the uniform IS (black dash line) appear wider than the proposed IS. These comparisons are similar to the neural network case.

4.4 Random Forest Example: IS with Multiple Dominating Points

We now generate input samples uniformly over $[0,5]^2$ with a mesh 0.2 on each coordinate, with the output values drawn from (22). The random forest has 2 trees in this example. The trained model using the generated dataset has 865 nodes in the first tree and 835 nodes in the second tree. Figure 13 shows the response surface of the random forest. We consider $\gamma = 8$ and the shape of the rare-event set is shown in Figure 14, which shows that the rare event set consists of two separate regions.

We find dominating points by implementing Algorithm 1 combined with Algorithm 2. Again, we use $\varepsilon = 0.01$ and $\delta = 0.01$ for the tolerance level. Note that the tolerance level we choose could generate an error on the dominating point we obtain. For $\varepsilon = 0.01$, the error is given by $\|a\|^2 - \|\hat{a}\|^2 \leq 0.02$, where a is the true dominating point and $\hat{a}$ is the dominating point we obtain from the bisection. This error might bring an issue to Algorithm 1. That is, the constraint $\hat{a}(x - \hat{a}) < 0$ we add may not cut off the corresponding half-space. This can possibly lead to an infinite loop for Algorithm 1. Here we resolve this issue by modifying the additional constraint as $\hat{a}(x - \hat{a}) < -\delta$, where $\delta > 0$ is approximately in the scale of $\sqrt{\varepsilon}$. In this case, we use $\delta = 0.1$. We obtain two dominating points, $(0.1, 3.9)$ and $(4.1, 3.3)$.

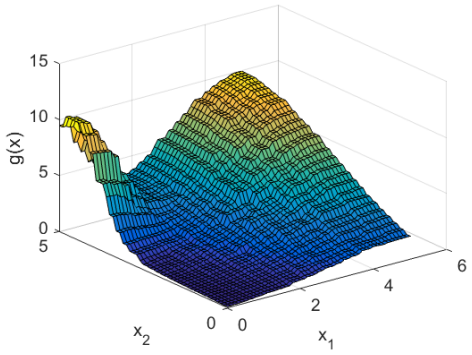

Figure 13: Response surface of the random forest.

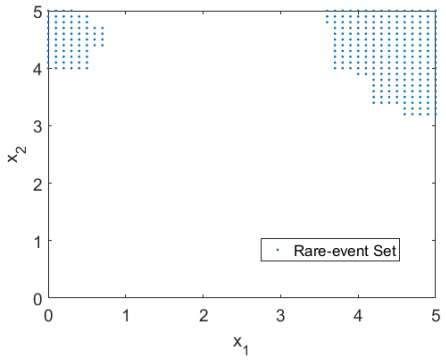

Figure 14: Rare-event set $\{x : g(x) \geq \gamma\}$ of the random forest.

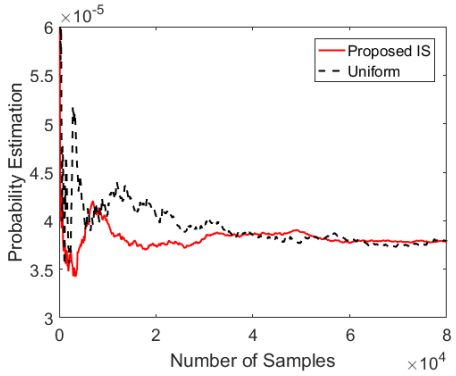

Figure 15: Probability estimation with different numbers of samples.

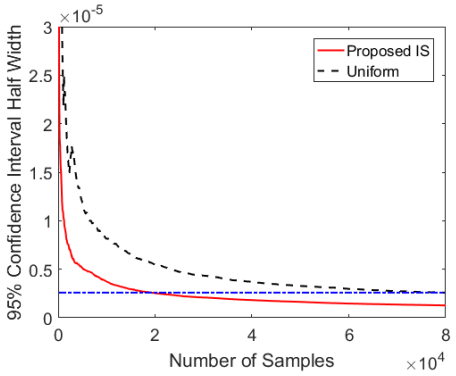

Figure 16: 95% confidence interval half-width with different numbers of samples.

The obtained dominating points allow us to implement the proposed IS. Compared to the uniform IS, Figures 15 and 16 show that the proposed approach is more efficient as in the previous experiments. The estimates of the proposed approach are more stable (red solid line in Figure 15) and the confidence intervals are shorter (red solid line in Figure 16). The uniform IS (black dash line in Figure 16) requires roughly 3 times more samples to achieve the same level of accuracy as the proposed IS.

ACKNOWLEDGMENTS

We gratefully acknowledge support from the National Science Foundation under grants CMMI-1542020 and CAREER CMMI-1653339/1834710.

REFERENCES

Adler, R. J., J. H. Blanchet, J. Liu et al. 2012. "Efficient Monte Carlo for High Excursions of Gaussian Random Fields". *The Annals of Applied Probability* 22(3):1167–1214.

Asmussen, S., and P. W. Glynn. 2007. *Stochastic Simulation: Algorithms and Analysis*, Volume 57. New York: Springer Science & Business Media.

Blanchet, J., and C. Li. 2011. "Efficient Simulation for the Maximum of Infinite Horizon Discrete-time Gaussian Processes". *Journal of Applied Probability* 48(2):467–489.

Blanchet, J., and H. Lam. 2012. "State-dependent Importance Sampling for Rare-event Simulation: An Overview and Recent Advances". *Surveys in Operations Research and Management Science* 17(1):38–59.

Bucklew, J. 2013. *Introduction to Rare Event Simulation*. New York: Springer Science & Business Media.

Dieker, A. B., and M. Mandjes. 2006. "Fast Simulation of Overflow Probabilities in A Queue with Gaussian input". *ACM Transactions on Modeling and Computer Simulation (TOMACS)* 16(2):119–151.

Glasserman, P. 2013. *Monte Carlo Methods in Financial Engineering*, Volume 53. New York: Springer Science & Business Media.

Goodfellow, I., Y. Bengio, A. Courville, and Y. Bengio. 2016. *Deep Learning*, Volume 1. Massachusetts: MIT press Cambridge.

Huang, Z., H. Lam, D. J. LeBlanc, and D. Zhao. 2017. "Accelerated Evaluation of Automated Vehicles Using Piecewise Mixture Models". *IEEE Transactions on Intelligent Transportation Systems*:1–11.

Jain, A., H. S. Koppula, B. Raghavan, S. Soh, and A. Saxena. 2015. "Car That Knows Before You Do: Anticipating Maneuvers via Learning Temporal Driving Models". In *Proceedings of the IEEE International Conference on Computer Vision*, 3182–3190. Piscataway, New Jersey: IEEE.

Juneja, S., and P. Shahabuddin. 2006. "Rare-event Simulation Techniques: An Introduction and Recent Advances". *Handbooks in Operations Research and Management Science* 13:291–350.

Mišic, V. V. 2017. "Optimization of Tree Ensembles". *Working Paper: arXiv preprint arXiv:1705.10883*.

Rubinstein, R. Y., and D. P. Kroese. 2016. *Simulation and the Monte Carlo Method*, Volume 10. New Jersey: John Wiley & Sons.

Sadowsky, J. S., and J. A. Bucklew. 1990. "On Large Deviations Theory and Asymptotically Efficient Monte Carlo Estimation". *IEEE Transactions on Information Theory* 36(3):579–588.

Tjeng, V., and R. Tedrake. 2017. "Verifying Neural Networks with Mixed Integer Programming". *Working Paper: arXiv preprint arXiv:1711.07356*.

Wang, W., D. Zhao, J. Xi, D. J. LeBlanc, and J. K. Hedrick. 2017. "Development and Evaluation of Two Learning-Based Personalized Driver Models for Car-following Behaviors". In *Proceedings of American Control Conference (ACC), 2017*, 1133–1138. Piscataway, New Jersey: IEEE.

AUTHOR BIOGRAPHIES

ZHIYUAN HUANG is a third-year Ph.D. student in the Department of Industrial and Operations Engineering at the University of Michigan, Ann Arbor. His research interests include simulation and stochastic optimization. His email address is zhyhuang@umich.edu.

HENRY LAM is an Associate Professor in the Department of Industrial Engineering and Operations Research at Columbia University. His research focuses on Monte Carlo simulation, uncertainty quantification, risk analysis, and stochastic and robust optimization. His email address is khl2114@columbia.edu.

DING ZHAO is an Assistant Professor in the Department of Mechanical Engineering at Carnegie Mellon University. His research focuses on Connected and Automated Vehicles (CAVs) using synthesized approaches rooted in advanced statistics, modeling, optimization, dynamic control, and big data analysis. His email address is dingzhao@cmu.edu.