Model Bridging: To Interpretable Simulation Model From Neural Network

Keiichi Kisamori
Data Science Laboratories
NEC Coopration
Nakahara-ku, Kawasaki, Japan
k-kisamori@ak.jp.nec.com

Keisuke Yamazaki
AI Research Center
National Institute of Advanced Industrial Science and Technology
Koto-ku, Tokyo, Japan
k.yamazaki@aist.go.jp

Abstract

The interpretability of machine learning, particularly for deep neural networks, is strongly required when performing decision-making in a real-world application. There are several studies that show that interpretability is obtained by replacing a non-explainable neural network with an explainable simplified surrogate model. Meanwhile, another approach to understanding the target system is simulation modeled by human knowledge with interpretable simulation parameters. Recently developed simulation learning based on applications of kernel mean embedding is a method used to estimate simulation parameters as posterior distributions. However, there was no relation between the machine learning model and the simulation model. Furthermore, the computational cost of simulation learning is very expensive because of the complexity of the simulation model. To address these difficulties, we propose a "model bridging" framework to bridge machine learning models with simulation models by a series of kernel mean embeddings. The proposed framework enables us to obtain predictions and interpretable simulation parameters simultaneously without the computationally expensive calculations associated with simulations. In this study, we investigate a Bayesian neural network model with a few hidden layers serving as an un-explainable machine learning model. We apply the proposed framework to production simulation, which is important in the manufacturing industry.

1 Introduction

The interpretability of machine learning, especially for deep neural networks, is strongly required when decision-making is required in a real-world application. In recent years, there are many studies that have addressed the interpretability of neural networks [5, 3, 13]. One of the approaches is to replace a un-interpretable machine learning model with a simplified surrogate model. This approach is considered to be a type of model compression. For example, Hara et al. [6] introduced a method to replace a un-interpretable random forest model with a simple decision tree model, with information criterion as a model selection problem; however, there is no method for neural networks. As another example, "distillation" of a neural network model [7] is one of the representative methods for model compression to replace a complex model with a simplified model; meanwhile, there is no

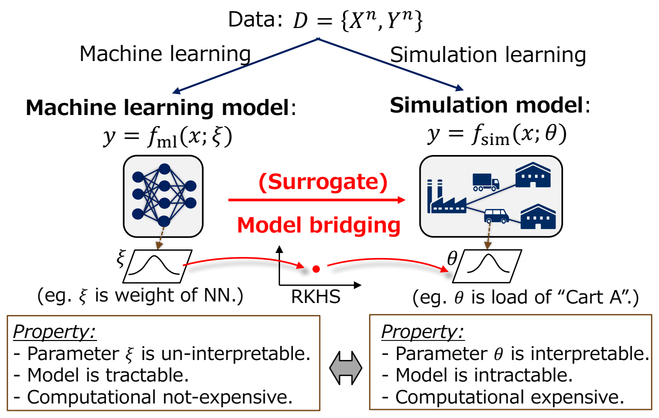

Data: $D = \{X^n, Y^n\}$
Machine learning
Simulation learning
Machine learning model: $y = f_{\mathrm{ml}}(x;\xi)$
Simulation model: $y = f_{\mathrm{sim}}(x;\theta)$
(Surrogate)
Model bridging
ξ
θ
(eg. ξ is weight of NN.)
RKHS
(eg. θ is load of "Cart A".)
Property:
- Parameter ξ is un-interpretable.
- Model is tractable.
- Computational not-expensive.
Property:
- Parameter θ is interpretable.
- Model is intractable.
- Computational expensive.

Figure 1: Overview of the model-bridging framework.

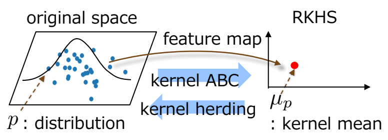


Figure 2: Schematics of kernel mean embedding as a tool of model bridging.

interpretability for a small surrogate neural network model. These methods do not provide a clear pathway toward obtaining interpretability of a neural network.

Another approach to understanding the target system is conducting a simulation that may be outside the scope of conventional machine learning. Here, we assume a simulation such as multi-agent simulation, traffic simulation, production simulation, or simulation of the dynamics of a physical system, which are widely used in social and industrial society. Simulation modeling is implemented to describe the basic law of the objective system, using human knowledge with interpretable simulation parameters. Recently developed "simulation learning" [9] is a method in which simulation parameters are estimated as posterior distributions in the context of machine learning. The simulation model is treated as an intractable or non-differentiable regression function in simulation learning. The challenge of simulation learning is a computational cost that is often more expensive than that of the machine learning model because of the complexity of the simulation model. Thus, if we have a simulation model for the objective system, we now have two ways to reproduce real data: machine learning with a statistical model and simulation learning with a simulation model. However, before, there was no way to relate a simulation model with a machine learning model, such as a neural network model.

We propose a "model bridging" framework to bridge the un-interpretable aspect of the machine learning model and the interpretable aspect of the simulation model (Fig. 1). A model-bridging framework enables us to not only predict a new dataset with high accuracy using a machine learning model but also obtain interpretable simulation parameters simultaneously without the expensive calculation of a simulation model.

Let us consider the example of production simulation to predict the efficiency of manufacturing production, implementing a series of processes for production (Example in Sec. 4 and Fig. 4). Assume that we obtain a dataset in which input $X^n = \{X_1, ..., X_n\}$ is the number of products to be manufactured in unit time and output $Y^n = \{Y_1, ..., Y_n\}$ is the elapsed time to manufacture, which directly affects the efficiency of production. The simulation parameter θ is the elapsed time of each process, a function of the failure time, which undergoes probabilistic behavior. Note that, θ is interpretable and helpful in understanding the system and decision making. Thus, if we would like to understand the objective system, we need to obtain not only prediction $\hat{Y}_{n+1}$ for new data X_{n+1} but also obtain interpretable simulation parameters, such as θ, representing the elapsed time of process, which provides information of where the "bottleneck processes" occur. The detailed assumption is described in Sec. 3.1.

This paper is organized as follows. We briefly review a series of applications of kernel mean embedding as the building blocks of the proposed framework. Subsequently, we propose the model-bridging framework. Afterwards, we confirm the accuracy of the proposed method for an application of the production simulation.

2 Related Works

We briefly introduce simulation learning and distribution regression based on kernel mean embedding as a building block of the proposed framework.

2.1 Simulation Learning

"Simulation learning" [9] is a method in which the simulation model is treated as a regression function $f_{\mathrm{sim}}(x;\theta)$ by combining a series of kernel mean embedding methods. Simulation learning is considered to be an example of data assimilation. Conventional statistical methods of parameter estimation are not applicable to data assimilation owing to the properties of the likelihood function: intractable or nondifferentiable. If Gaussian noise is employed with regression function $f_{\mathrm{sim}}(x;\theta)$, the likelihood is expressed as $p(y|x,\theta) = \frac{1}{\sqrt{2\pi\sigma_0^2}} \exp\left\{-\frac{1}{2\sigma_0^2}\left\|y - f_{\mathrm{sim}}(x;\theta)\right\|^2\right\}$, where $\sigma_0 > 0$ is a constant of observation noise. This likelihood function is nondifferentiable, owing to the simulation model $f_{\mathrm{sim}}(x;\theta)$. The posterior mean to be obtained is formulated as $p(\theta|X^n, Y^n) = p(Y^n|X^n,\theta)\pi(\theta)/Z(X^n, Y^n)$, where $\pi(\theta)$ is the prior distribution, and $Z(X^n, Y^n)$ is the regularization constant. In this application, simulation learning estimates the simulation parameter θ as a kernel mean of the posterior distribution by using kernel approximated Bayesian computation (kernel ABC) [15, 4]. After obtaining the kernel mean of the posterior distribution, a posterior sample is obtained by kernel herding [1].

2.2 Application of Kernel Mean Embedding

As an application of kernel mean embedding [14], we briefly review the kernel ABC and kernel herding. Kernel mean embedding is a framework to map distributions into a reproducing kernel Hilbert space (RKHS) $\mathcal{H}$ as a feature space. Kernel herding is a sampling method from the embedded distribution in RKHS that has the opposite operation of kernel mean embedding. Figure 2 shows a schematic of the relation of kernel ABC and kernel herding.

Kernel ABC: Kernel ABC [15, 4] is a method to compute the kernel mean of the posterior distribution from a sample of parameter θ, generated by the prior distribution $\pi(\theta)$. The assumption is that the explicit form of the likelihood function is unavailable, while the sample from the likelihood is available. The kernel ABC allows us to calculate the kernel mean of the posterior distribution as follows: First, sample $\{\theta_1, ..., \theta_m\}$ is generated from prior distribution $\pi(\theta)$ and pseudo-data $\{\bar{Y}_1^n, ..., \bar{Y}_m^n\}$ as a sample from $p(y|x,\theta_j)$ for $j = 1, ..., m$. Next, the empirical kernel mean of the posterior distribution

$$\hat{\mu}_{\theta|YX} = \sum_{j=1}^{m} w_j k_\theta(\cdot, \theta_j) \tag{1}$$

is calculated, where k_θ is a kernel of θ. Weight w_j is calculated by

$$\mathbf{w} = (w_1, ..., w_m)^T = (G_y + n\delta I)^{-1}\mathbf{k}_y(Y^n) \in \mathbb{R}^m \tag{2}$$

$$G_y = \{k_y(\bar{Y}_j^n, \bar{Y}_{j'}^n)\}_{j,j'=1}^m \in \mathbb{R}^{m\times m} \tag{3}$$

$$\mathbf{k}_y(Y^n) = (k_y(\bar{Y}_1^n, Y^n), ..., k_y(\bar{Y}_m^n, Y^n)) \in \mathbb{R}^m. \tag{4}$$

Kernel $k_y(Y^n, \bar{Y}^n)$ indicates the "similarity" between real data Y^n and pseudo-data $\bar{Y}_j^n$. The calculation of the kernel mean corresponds to the estimation of the posterior distribution as an element in function space $\mathcal{H}$.

Kernel Herding: Kernel herding [1] is a method used to sample data from the kernel mean representation of a distribution, which is an element of the RKHS. Kernel herding can be considered as an opposite operation to that of kernel ABC. Kernel herding greedily obtains samples $\{\theta_1, ..., \theta_m\}$ by updating Eqs.(1) and (2) in Chen et al. [1].

2.3 Distribution Regression

Distribution regression is a regression for d_x-dimensional "distributions" represented by samples. Meanwhile normal regression is regression for d_x-dimensional "point". There are several studies

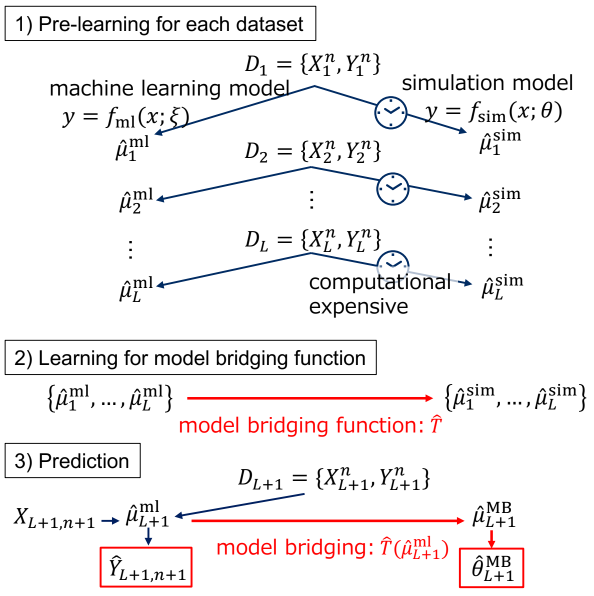

1) Pre-learning for each dataset
$D_1 = \{X_1^n, Y_1^n\}$
machine learning model
$y = f_{\text{ml}}(x;\xi)$
simulation model
$y = f_{\text{sim}}(x;\theta)$
$\hat{\mu}_1^{\text{ml}}$
$\hat{\mu}_1^{\text{sim}}$
$D_2 = \{X_2^n, Y_2^n\}$
$\hat{\mu}_2^{\text{ml}}$
$\hat{\mu}_2^{\text{sim}}$
$D_L = \{X_L^n, Y_L^n\}$
$\hat{\mu}_L^{\text{ml}}$
computational expensive
$\hat{\mu}_L^{\text{sim}}$
2) Learning for model bridging function
$\{\hat{\mu}_1^{\text{ml}}, \dots, \hat{\mu}_L^{\text{ml}}\}$
$\{\hat{\mu}_1^{\text{sim}}, \dots, \hat{\mu}_L^{\text{sim}}\}$
model bridging function: $\hat{T}$
3) Prediction
$D_{L+1} = \{X_{L+1}^n, Y_{L+1}^n\}$
$X_{L+1,n+1}$
$\hat{\mu}_{L+1}^{\text{ml}}$
$\hat{\mu}_{L+1}^{\text{MB}}$
model bridging: $\hat{T}(\hat{\mu}_{L+1}^{\text{ml}})$
$\hat{Y}_{L+1,n+1}$
$\hat{\theta}_{L+1}^{\text{MB}}$

Figure 3: Overview of model-bridging framework.

Algorithm 1: Model bridging

```
*1) Pre-learning for each dataset* :
**Input**: Dataset $\{X_l^n, Y_l^n\}_{l=1}^L$,
    machine learning model $f_{\text{ml}}(x,\xi)$
    and simulation model $f_{\text{sim}}(x,\theta)$
**Output**: $\{\hat{\mu}_1^{\text{ml}}, ..., \hat{\mu}_L^{\text{ml}}\}$ and $\{\hat{\mu}_1^{\text{sim}}, ..., \hat{\mu}_L^{\text{sim}}\}$
**for** $l = 1$ to $L$ **do**
    Estimation for $\hat{\mu}_l^{\text{ml}}$ by Eq. (14)
    Estimation for $\hat{\mu}_l^{\text{sim}}$ by Eq. (1)
**end for**
*2) Learning for model-bridging function $\hat{T}$:*
**Input**: $\{\hat{\mu}_1^{\text{ml}}, ..., \hat{\mu}_L^{\text{ml}}\}$ and $\{\hat{\mu}_1^{\text{sim}}, ..., \hat{\mu}_L^{\text{sim}}\}$
**Output**: Model-bridging function $\hat{T}$
Learning for $\hat{T}$ by Eq. (5)
*3) Prediction:*
**Input**: Dataset $\{X_{L+1}^n, Y_{L+1}^n\}$ and $X_{L+1,n+1}$
**Output**: $\hat{Y}_{L+1,n+1}$ and $\hat{\theta}_{L+1}$
Estimation for $\xi_{L+1}$ and $\hat{\mu}_{L+1}^{\text{ml}}$
Prediction for $\hat{Y}_{L+1,n+1}$ by $f_{\text{ml}}(x;\xi_{L+1})$
Estimation for $\hat{\mu}_{L+1}^{\text{MB}} = \hat{T}(\hat{\mu}_{L+1}^{\text{ml}})$ by Eq. (7)
Sampling for $\hat{\theta}_{L+1}^{\text{MB}}$ by Eq. (11)
```

of distribution regression, including distribution-to-distribution regression [17] and distribution-to-point regression [18, 11]. Oliva et al. [17] employ the idea of approximating a density function by kernel density estimation, rather than using a reproducing kernel. Szabó et al. [18] propose the distribution-to-point with kernel ridge regression method; however, there are no methods for distribution-to-distribution regression.

3 Proposed Framework: Model Bridging

We propose a new framework to bridge the un-interpretable machine learning model and the interpretable simulation model. In this study, we assume a machine learning model, such as a Bayesian neural network (BNN) [16] with a few hidden layers. Meanwhile, this proposed framework is applicable to any model. In this section, first, we confirm the problem setting and framework of model bridging. Second, we propose the algorithm of distribution regression for proposed method. Afterwards, we propose the formulation of the input of distribution-to-distribution regression for the parametric model, assuming BNN, and the non-parametric model, assuming Gaussian process. Gaussian process is considered to be BNN with an infinite unit for one hidden layer [16]. Figure 3 and Alg. 1 shows an overview of the framework.

3.1 Problem Setting, Assumption, and Usage of Model Bridging

We define the problem setting of the model-bridging framework. Let L be dataset $\{X_1^n, Y_1^n, ..., X_L^n, Y_L^n\}$ ($X_l^n \in \mathbb{R}^{n\times d_x}, Y_l^n \in \mathbb{R}^{n\times d_y}$), given in the pre-learning phase. The purpose is to predict $\hat{Y}_{L+1,n+1}$ and simultaneously obtain interpretable simulation parameter $\hat{\theta}_{L+1}^{\text{MB}}$ to reproduce $Y_{L+1,n+1} = f_{\text{sim}}(X_{L+1,n+1}; \hat{\theta}_{L+1}^{\text{MB}})$ without the expensive calculation of simulation model $f_{\text{sim}}(x;\theta)$ when we obtain new dataset $\{X_{L+1}^n, Y_{L+1}^n\}$. The assumptions of the problem setting are as follows. These assumptions are a typical setting for use case of a simulation.

- Existing simulation model $f_{\text{sim}}(x;\theta)$ with interpretable simulation parameter $\theta \in \mathbb{R}^{d_\theta}$ and a machine learning model $f_{\text{ml}}(x;\xi)$ that is sufficiently accurate to predict a typical regression problem while having un-interpretable parameter $\xi \in \mathbb{R}^{d_\xi}$.
- Cost of simulation learning is much higher than that of learning from the machine learning model. For example, it takes more than 1 day for simulation learning of one dataset $\{X_l^n, Y_l^n\}$ while learning of BNN takes less than minute.

- Dataset $\{X_l^n, Y_l^n\}$ has dependency of parameter θ_l for each $l = 1, ..., L$. Let us assume the following situation: $\{X_l^n, Y_l^n\}$ is obtained in one day with the same conditions, described as parameter θ_l, while conditions are changed for the following day, described as θ_{l+1}.
- Time for off-line calculation of simulation learning is sufficient, while time for prediction is restricted.

Once we obtain model-bridging function $\hat{T}$ as a mapping from the machine learning model to the simulation model, we can obtain an accurate prediction for $\hat{Y}_{L+1,n+1}$ by both the machine learning model and interpretable $\hat{\theta}_{L+1}^{\mathrm{MB}}$ by the simulation model for new dataset $\{X_{L+1}^n, Y_{L+1}^n\}$ without an expensive calculation from the simulation model.

3.2 Distribution-to-Distribution Regression with Kernel Ridge Regression

We present the regression algorithm between the kernel mean of the machine learning model μ^{ml} and that of the simulation model μ^{sim}, as a model-bridging function $\mu_l^{\mathrm{sim}} = T(\mu_l^{\mathrm{ml}})$. We develop the distribution-to-distribution regression algorithm based on kernel ridge regression. The target of regression is the kernel mean of the posterior distribution of θ_l, which is the output of simulation learning. This is the extension of the distribution-to-point regression method proposed by Szabó et al. [18] for the distribution output.

3.2.1 Kernel Ridge Regression for Kernel Mean

The formulation to be solved is as follows as an analogy of normal kernel ridge regression:

$$\hat{T} = \arg\max_{T \in \mathcal{H}} \frac{1}{L} \sum_{l=1}^{L} \|\hat{\mu}_l^{\mathrm{sim}} - T(\hat{\mu}_l^{\mathrm{ml}})\|_{\mathcal{H}}^2 + \lambda \|T\|_{\mathcal{H}}^2, \tag{5}$$

where $\lambda > 0$ is a regularization constant. The difference from normal kernel ridge regression is that the inputs and outputs are kernel means. Therefore, we define kernel κ as a function of kernel mean $\mu \in \mathcal{H}$. We employ a Gaussian-like kernel as

$$\kappa(\mu, \mu') = \exp\left\{-\frac{1}{2\sigma_\mu^2} \|\mu - \mu'\|_{\mathcal{H}}^2\right\} \in \mathcal{H}, \tag{6}$$

where constant $\sigma_\mu > 0$ is the width of kernel κ. The kernel κ is also a positive definite kernel [2].

Following the representer theorem of kernel ridge regression [10], the estimated model-bridging function $\hat{T}$ for new $\hat{\mu}_{L+1}^{\mathrm{ml}}$ is described as

$$\hat{\mu}_{L+1}^{\mathrm{MB}} = \hat{T}(\hat{\mu}_{L+1}^{\mathrm{ml}}) = \sum_{l=1}^{L} v_l \hat{\mu}_l^{\mathrm{sim}} \quad \in \mathcal{H}, \tag{7}$$

where

$$\mathbf{v} = (v_1, ..., v_L)^T = (G_\mu + \lambda L I)^{-1} \mathbf{k}_\mu(\hat{\mu}_{L+1}^{\mathrm{ml}}) \quad \in \mathbb{R}^L. \tag{8}$$

I is an identity matrix. Gram matrix G_μ and the vector $\mathbf{k}_\mu(\hat{\mu}_{L+1}^{\mathrm{ml}})$ are described as follows:

$$G_\mu = \left\{\kappa(\hat{\mu}_l^{\mathrm{ml}}, \hat{\mu}_{l'}^{\mathrm{ml}})\right\}_{l,l'=1}^{L} \quad \in \mathbb{R}^{L \times L} \tag{9}$$

$$\mathbf{k}_\mu(\hat{\mu}_{L+1}^{\mathrm{ml}}) = \left(\kappa(\hat{\mu}_1^{\mathrm{ml}}, \hat{\mu}_{L+1}^{\mathrm{ml}}), ..., \kappa(\hat{\mu}_L^{\mathrm{ml}}, \hat{\mu}_{L+1}^{\mathrm{ml}})\right)^T \quad \in \mathbb{R}^L. \tag{10}$$

3.2.2 Kernel Herding from Kernel Mean $\hat{\mu}_l^{\mathrm{MB}}$

After obtaining the kernel mean of $\hat{\mu}_{L+1}^{\mathrm{MB}}$, kernel herding can be applied to sample $\hat{\theta}_{L+1}^{\mathrm{MB}} = \{\hat{\theta}_{L+1,1}, ..., \hat{\theta}_{L+1,m}\}$ where $\hat{\theta}_{L+1,j} \in \mathbb{R}^m$. The explicit form of the update equation for sample $j = 1, ..., m$ iteration of kernel herding with kernel mean $\hat{\mu}_{L+1}^{\mathrm{MB}}$ is as follows:

$$\hat{\theta}_{L+1,j} = \arg\max_{\theta} \sum_{l=1}^{L} \sum_{j'=1}^{m} v_l w_{l,j'} k_\theta(\theta, \theta_{l,j'}) + \frac{1}{m+1} \sum_{j'=1}^{m} k_\theta(\theta, \theta_{j'}) \in \mathbb{R}^{d_\theta}, \tag{11}$$

for $j = 2, ..., m$. For initial state $j = 1$, the update equation is only the first term of Eq. (11). The weight of $w_{l,j}$ is calculated by kernel ABC for dataset $\{X_l^n, Y_l^n\}$ in Eq.(2).

3.3 Target of Distribution-to-Distribution Regression: $\hat{\mu}_l^{\mathrm{sim}}$

As a target of the distribution-to-distribution regression, simulation learning directly provides the empirical kernel mean of the posterior distribution of the parameter θ_l for the regression problem as $\hat{\mu}_l^{\mathrm{sim}} = \sum_{j=1}^{m} w_{l,j} k_\theta(\cdot, \theta_{l,j}) \in \mathcal{H}$, where $\theta_{l,j}$ for $j = 1, ..., m$ represents the parameter samples from the prior $\pi(\theta)$.

3.4 Input of Distribution-to-Distribution Regression: $\hat{\mu}_l^{\mathrm{ml}}$

As an input of the distribution-to-distribution regression, we present the explicit formulation to calculate the kernel mean of machine learning model $\hat{\mu}_l^{\mathrm{ml}}$. First, as a useful application of the parametric Bayesian model, we present the formulation of BNN. Second, as a non-parametric Bayesian model, we present the formulation for Gaussian process regression. The equivalence between the BNN with one hidden layer with infinite nodes and the Gaussian process is well known [16]. Furthermore, a recent study reveals the kernel formulation that is equivalent with multi-layered BNN as an extension of the Gaussian process [12]. Thus, we consider the Gaussian process regression as a non–parametric alternative to BNN.

3.4.1 Parametric Model: Bayesian Neural Network

We assume BNN with a few hidden layers $f_{\mathrm{ml}}(x; \xi)$. We can obtain the posterior distribution of ξ_l for $l = 1, ..., L$ by the Markov Chain Monte Carlo (MCMC) method or variational approximation. The $j = 1, ..., m$ is the number of parameter samples. Then, the empirical kernel mean of the posterior distribution is represented as $\hat{\mu}_l^{\mathrm{ml}} = \sum_{j=1}^{m} k_\xi(\cdot, \xi_{l,j}) \in \mathcal{H}$ for $l = 1, ..., L$ dataset.

We employ Gaussian-like kernel κ as an function of $\hat{\mu}_l^{\mathrm{ml}} \in \mathcal{H}$ as

$$\kappa(\hat{\mu}_l^{\mathrm{ml}}, \hat{\mu}_{l'}^{\mathrm{ml}}) = \exp\left\{-\frac{1}{2\sigma_\mu^2} \left\|\hat{\mu}_l^{\mathrm{ml}} - \hat{\mu}_{l'}^{\mathrm{ml}}\right\|_{\mathcal{H}}^2\right\} \in \mathcal{H} \tag{12}$$

$$= \exp\left\{-\frac{1}{\sigma_\mu^2}\left(1 - \sum_{j=1}^{m}\sum_{j'=1}^{m'} k_\xi(\xi_{l,j}, \xi_{l',j'})\right)\right\}, \tag{13}$$

where constant $\sigma_\mu > 0$ is a width of kernel κ. The relation $\langle \hat{\mu}_l^{\mathrm{ml}}, \hat{\mu}_{l'}^{\mathrm{ml}} \rangle = \sum_{j=1}^{m}\sum_{j'=1}^{m'} k_\xi(\xi_{l,j}, \xi_{l',j'})$ is used, where $\langle \cdot, \cdot \rangle$ represents the inner product. We employ Gaussian kernel $k_\xi(\xi_{l,j}, \xi_{l',j'})$ with with constant $\sigma_\xi > 0$.

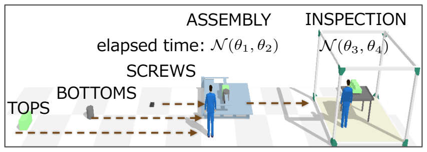
3.4.2 Non-Parametric Model: Gaussian Process Regression

As a non-parametric model, we use Gaussian process regression. Gaussian process regression is considered as an infinite parameter dimension of BNN. As a result of Gaussian process regression, we can express the mean of the predictive distribution for l-th dataset $\{X^n, Y^n\}$ as

$$y = \hat{\mu}_{Y|X,l}(x) = \sum_{i=1}^{n} u_{l,i}(x) k_y(\cdot, Y_{l,i}) \in \mathcal{H}, \tag{14}$$

where $u_{l,i}(x) = \{(G_x + n\delta I)^{-1}\mathbf{k}_x(x)\}_i$. The G_x is the Gramm matrix, δ is regularization constant, and k_x is Gaussian kernel. This formulation is clear if we remember the equivalence between Gaussian process regression and kernel ridge regression [8]. As a predictor of $\hat{Y}_{l,n+1}$ for new $X_{l,n+1}$, we can calculate $\hat{Y}_{l,n+1} = \hat{\mu}_{Y|X,l}(X_{l,n+1})$. Note that this $\hat{\mu}_{Y|X,l}$ is interpreted as the kernel mean of the Y_l^n conditioned by X_l^n. Thus, we can use $\hat{\mu}_{Y|X,l}$ as the input of the distribution-to-distribution regression, represented as $\hat{\mu}_l^{\mathrm{ml}}$. The precise relation between Gaussian process regression, kernel ridge regression, and kernel mean is reviewed by Kanagawa et al [8]. We employ Gaussian-like kernel κ as an function of $\hat{\mu}_l^{\mathrm{ml}}$ as

$$\kappa(\hat{\mu}_l^{\mathrm{ml}}, \hat{\mu}_{l'}^{\mathrm{ml}}) = \exp\left\{-\frac{1}{\sigma_\mu^2}\left(1 - \sum_{i=1}^{n}\sum_{i'=1}^{n'} u_{l,i} u_{l',i'} k_y(Y_{l,i}, Y_{l',i'})\right)\right\} \in \mathcal{H}, \tag{15}$$


ASSEMBLY
INSPECTION
elapsed time: $\mathcal{N}(\theta_1, \theta_2)$
$\mathcal{N}(\theta_3, \theta_4)$
SCREWS
BOTTOMS
TOPS
Figure 4: Illustration of the simulation model of production simulation for the experiment.

where $k_y(Y_{l,i}, Y_{l',i'})$ is Gaussian kernel with constant σ_y. There is an difference between the proposed non-parametric method and parametric method: In the parametric method, the distribution of parameter ξ_l is the input of the distribution-to-distribution regression, while in the non-parametric method, the distribution of data Y_l^n conditioned by X_l^n is the input.

4 Experiment

We apply a model-bridging framework for production simulation to investigate the efficiency of factory manufacturing of product. Not just for production simulation, this framework is widely applicable, e.g., multi-agent simulation, traffic simulation, and simulation of dynamics of physical systems such as fluid mechanics, thermomechanics, structural mechanics, and electro-magnetic mechanics, which are widely used in social and industrial society.

In this experiment, we confirm that model bridging enables us to predict $\hat{Y}_{L+1,n+1}$ for new $X_{L+1,n+1}$, using a machine learning model, and simultaneously obtain interpretable simulation parameter $\hat{\theta}_{L+1}^{\mathrm{MB}}$ without an expensive calculation of the simulation model when we obtain the $L+1$-th dataset $\{X_{L+1}^n, Y_{L+1}^n\}$. We also confirm the accuracy of the estimation of parameter $\hat{\theta}_{L+1}^{\mathrm{MB}}$ by model bridging.

4.1 Setting of the Experiment

A production simulator is widely used simulation software for discrete and interconnection systems to model various processes, such as production processes, logistics, transportation, and office work. We used a simulator called WITNESS, which is a commercially available and standard software package used in production simulation.

We examine the regression problem with a production simulation that has a simple four-dimensional simulation parameter $\theta \in \mathbb{R}^4$. We define simulation input $x = X_i$ as the number of products to be manufactured, output $Y_i = r(X_i, \theta)$ as the total time to manufacture all the X_i-th products, and parameter θ as the time for each procedure of the production line: time of "ASSEMBLY" process is $\mathcal{N}(\theta_1, \theta_2)$, and time of "INSPECTION" process is $\mathcal{N}(\theta_3, \theta_4)$, where $\mathcal{N}(\mu_{\mathrm{ND}}, \sigma_{\mathrm{ND}})$ is the normal distribution with mean μ_{ND} and standard deviation σ_{ND}. We assume that the elapsed time of each process will become much longer owing to an increasing load, if the number of products to be manufactured also increases. To create this situation artificially, we set different true parameters between the observed data region $\theta^{(0)}$ and the predictive region $\theta^{(1)}$. We set $\theta^{(0)} = (2, 0.5, 5, 1)$ if $x < 110$, and $\theta^{(1)} = (3.5, 0.5, 7, 1)$ if $x > 110$. A shift in parameters θ_1 and θ_3 between $\theta^{(0)}$ and $\theta^{(1)}$ is sigmoidal. For each l-th dataset, the observed data of size $n = 50$ are generated by $q_l(x) = \mathcal{N}(\chi_l, 5)$ where χ_l is uniform in $[70, 130]$ for $l = 1, ..., L$.

In practice, the effective hyperparameter for model-bridging function to be tuned is the regularization constant λ for distribution-to-distribution regression. Hyperparameter λ can stabilize the calculation of the inverse Gram matrix. This hyperparameter should be decided by cross-validation. Meanwhile, as a common hyperparameter of the kernel method, the width of the kernel must be selected to measure the similarity between the data. The typical setting of the width of the kernel in practice is the median of Euclid distance of the input data of a kernel. In the proposed method, we confirmed that this typical setting is performed well for all kernels $\sigma_y, \sigma_\xi, \sigma_\theta$, and σ_μ. The regularization constant $\lambda = 1.0 \times 10^{-6}$ in this experiment. We used a PC equipped with a 3.4-GHz, Intel Core i7, four-core processor, and 16-GB memory.

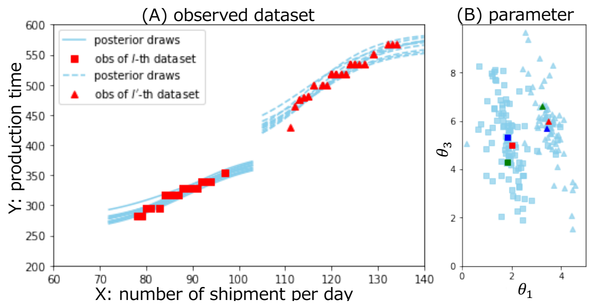


Figure 5: Result for two different datasets: l-th dataset as square markers and l'-th dataset as triangle markers. (A) Observed data and fitted result. (B) Estimated distribution of simulation parameters by model bridging. Red markers are true, green makers are estimated by simulation learning, and blue markers represent the distribution and mean of the model-bridging estimation.

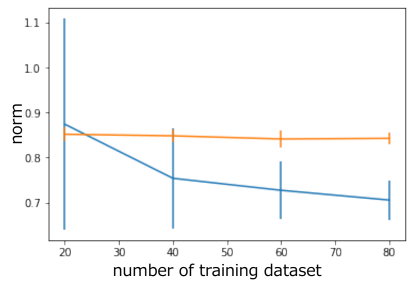


Figure 6: Estimation of norm for $\hat{\mu}$ by the model-bridging function for number of training datasets. Blue line represents the estimated result of the norm, and the orange line represents the norm with prior information only.

4.2 Result of the Experiment

Estimated simulation parameter $\hat{\theta}_l^{\mathrm{MB}}$ for two different datasets by model bridging is shown in Fig. 5. Figure 5 (A) shows the observed data for l-th dataset as red squares and the l'-th dataset as red triangles. The solid line and dashed line is the fitted result by BNN with variational approximation. BNN has two fully connected hidden layers with three nodes and bias nodes for each layer. The scatterplot in Fig. 5 (B) shows the distribution of parameter $\hat{\theta}_l^{\mathrm{MB}}$, estimated by model bridging. Blue markers show the mean of distribution, red markers show the true parameter, and green markers represent the estimated result of simulation learning. Each square is for l-th dataset, and each triangle is for l'-th dataset. The number of training datasets L is 80. These two figures show the reasonable estimation of model bridging framework in comparison with simulation learning for the two different datasets with two different true parameters θ. Furthermore, the elapsed time for prediction by model bridging is less than a minute in the presented computational environment, while simulation learning takes several hours.

We also investigate $\|\hat{\mu}_{L+1}^{\mathrm{MB}} - \hat{\mu}_{L+1}^{\mathrm{sim}}\|_{\mathcal{H}}^2$ which indicates the accuracy of the model-bridging framework. Detailed formulation for numerical calculation is presented in supplemental material. Figure 6 shows that the mean and standard deviation of 10 independent trials for the mean of the test dataset. The horizontal axis shows the number of training datasets with 20 test datasets. We can see the convergence for bias, which originates from simulation learning.

5 Discussion

There are many possible options to be discussed in the proposed framework for the individual use case. In this study, we assume the given observed dataset as problem setting (Sec. 3.1). Meanwhile, there are two other possible ways for problem setting with the assumption of data generation process: 1) Generate data from $f_{\mathrm{ml}}(x;\xi)$ and 2) generate data from $f_{\mathrm{sim}}(x;\theta)$. Considering another use case with these assumptions of data generation may be meaningful, e.g., when the real observed data is limited or when the simulation has high confidence. Another option to be discussed is the parametric or non-parametric regression model for model-bridging function $\hat{T}$. In this study, we employ the non-parametric kernel ridge regression model as distribution-to-distribution regression. Furthermore, in this study, we present the practical effectiveness of the model-bridging framework, while theoretical analysis for asymptotic behavior of the model-bridging framework is desired.

6 Conclusion

We propose a novel framework named "model bridging" to bridge from the un-interpretable machine learning model to the interpretable simulation model with interpretable parameters. The model-bridging framework enables us to not only obtain precise prediction from the machine learning model but also obtain the interpretable simulation parameter simultaneously without the expensive calculation of simulation. We confirm the effectiveness of the model-bridging framework and accuracy of the estimated simulation parameter with production simulation, which is widely used in the real-world manufacturing industry.

References

[1] Yutian Chen, Max Welling, and Alex Smola. 2010. Super-samples from kernel herding. *Proceedings of the Twenty-Sixth Conference Annual Conference on Uncertainty in Artificial Intelligence (UAI-10)* (2010), 109–116. arXiv:1203.3472 `http://arxiv.org/abs/1203.3472`

[2] Andreas Christmann and Ingo Steinwart. 2010. Universal Kernels on Non-Standard Input Spaces. (2010).

[3] Finale Doshi-Velez and Been Kim. 2017. Towards A Rigorous Science of Interpretable Machine Learning. (2017). `https://doi.org/10.1016/j.intell.2013.05.008` arXiv:1702.08608

[4] Kenji Fukumizu, Le Song, and Arthur Gretton. 2013. Kernel Bayes' Rule: Bayesian Inference with Positive Definite Kernels. *Journal of Machine Learning Research* 14 (2013), 3753–3783. arXiv:arXiv:1009.5736v4

[5] Riccardo Guidotti, Anna Monreale, Salvatore Ruggieri, Franco Turini, Dino Pedreschi, and Fosca Giannotti. 2018. A Survey Of Methods For Explaining Black Box Models. (2018). `https://doi.org/10.1145/3236009` arXiv:1802.01933

[6] Satoshi Hara and Kohei Hayashi. 2018. Making Tree Ensembles Interpretable: A Bayesian Model Selection Approach. (2018). `https://doi.org/10.1002/hbm.24063` arXiv:1606.09066

[7] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the Knowledge in a Neural Network. (2015), 1–9. arXiv:arXiv:1503.02531v1

[8] Motonobu Kanagawa, Philipp Hennig, Dino Sejdinovic, and Bharath K Sriperumbudur. 2018. Gaussian Processes and Kernel Methods: A Review on Connections and Equivalences. (2018). arXiv:arXiv:1807.02582v1

[9] Keiichi Kisamori and Keisuke Yamazaki. 2018. Intractable Likelihood Regression for Covariate Shift by Kernel Mean Embedding. (2018). arXiv:1809.08159

[10] S. Y. Kung. 2014. *Kernel Methods and Machine Learning*. Cambridge University Press. `https://doi.org/10.1017/CBO9781139176224`

[11] Ho Chung Leon Law, Dougal J. Sutherland, Dino Sejdinovic, and Seth Flaxman. 2017. Bayesian Approaches to Distribution Regression. *Proceedings of the 21st International Conference on Artificial Intelligence and Statistics (AISTATS)* (2017). arXiv:1705.04293

[12] Jaehoon Lee, Yasaman Bahri, Roman Novak, Samuel S Schoenholz, Jeffrey Pennington, and Jascha Sohl-Dickstein. 2018. DEEP NEURAL NETWORKS AS GAUSSIAN PROCESSES. *Proceedings of The International Conference on Learning Representations (ICLR)* (2018).

[13] Christoph Molnar. 2012. *Interpretable Machine Learning*. Christoph Molnar.

[14] Krikamol Muandet, Kenji Fukumizu, Bharath Sriperumbudur, and Bernhard Schölkopf. 2016. Kernel Mean Embedding of Distributions: A Review and Beyonds. *arXiv:1605.09522* (2016), 133. `https://doi.org/10.1561/2200000060` arXiv:1605.09522

[15] Shigeki Nakagome, Kenji Fukumizu, and Shuhei Mano. 2013. Kernel approximate Bayesian computation in population genetic inferences. *Statistical Applications in Genetics and Molecular Biology* 12, 6 (2013), 667–678. `https://doi.org/10.1515/sagmb-2012-0050` arXiv:1009.5736

[16] Radford M. Neal. 1996. *Bayesian Learning for Neural Networks*. Springer. `https://doi.org/10.1007/978-1-4612-0745-0`

[17] Junier B Oliva and Jeff Schneider. 2013. Distribution to Distribution Regression. (2013).

[18] Zoltan Szabo, Bharath Sriperumbudur, Barnabas Poczos, and Arthur Gretton. 2016. Learning Theory for Distribution Regression. *Journal of Machine Learning Research* 17 (2016), 1–40. arXiv:1411.2066

A. Detailed Setting of Experiment for Production Simulation

A production simulator is a general-purpose simulation software package for discrete and interconnection systems for modeling various processes such as production processes, logistics, transportation, and office work. The purpose of the production simulation in this experiment is to predict the total production time when the number of products to be manufactured is set. Figure 4 shows a typical assembly process for one product with four parts used in this experiment. Items with four parts consist of one "TOPS" part, one "BOTTOMS" part, and two "SCREWS." The products assembled in the "ASSEMBLY" machine are inspected by the "INSPECTION" machine before shipping. The "INSPECTION" machine starts when four assembled products arrive and is capable of inspecting four assembled products simultaneously. Parameters θ_1 and θ_2 represent the mean and standard deviation in a normal distribution of elapsed time for the "ASSEMBLY" machine, respectively. Parameters θ_3 and θ_4 represent the mean and standard deviation in a normal distribution of elapsed time for the "INSPECTION" machine, respectively.

B. Explicit Formulation of Norm of Empirical Kernel Mean

We present the explicit formulation of $\|\hat{\mu}^{\mathrm{MB}}_{L+1} - \hat{\mu}^{\mathrm{sim}}_{L+1}\|^2_{\mathcal{H}}$. The key for calculation is the relation $\langle \hat{\mu}, \hat{\mu}' \rangle = \sum\sum k_\theta(\theta, \theta')$ for θ kernel. The norm between estimated $\hat{\mu}^{\mathrm{MB}}_{L+1}$ and target $\hat{\mu}^{\mathrm{sim}}_{L+1}$ is described as follows:

$$\begin{aligned} \|\hat{\mu}^{\mathrm{MB}}_{L+1} - \hat{\mu}^{\mathrm{sim}}_{L+1}\|^2_{\mathcal{H}} &= 2\left\{1 - \langle \hat{\mu}^{\mathrm{MB}}_{L+1}, \hat{\mu}^{\mathrm{sim}}_{L+1} \rangle\right\} && (16) \\ &= 2\left\{1 - \sum_{l=1}^{L}\sum_{j=1}^{m}\sum_{j'=1}^{m} v_l w_{l,j} w_{L+1,j'} k_\theta(\theta_{L+1,j'}, \theta_{L+1,j})\right\}. && (17) \end{aligned}$$

Then, the norm of the empirical kernel mean is obtained.