Supplementary Material for:

Bayesian Optimization with Pareto-Principled Training for Economical Hyperparameter Optimization

Yang Yang[1], Ke Deng[2] and Yu Zhu[3]

[1]*Nankai University,* [2]*Tsinghua University and* [3]*Purdue University*

S1 Proof of Theorem 1

For any $\mathbf{x} \in \mathcal{X}_e \setminus \mathcal{X}_c$, the joint distribution of $(y_c(\mathbf{x}), \mathbf{y}_c|\ \mathbf{y}_e, \hat{\boldsymbol{\psi}}, b_1, b_2)$ is multivariate truncated normal,

$$y_c(\mathbf{x}), \mathbf{y}_c|\ \mathbf{y}_e, \hat{\boldsymbol{\psi}}, b_1, b_2 \sim \mathcal{TN}_{n+1}(\hat{\rho}\mathbf{y}^*_{e_c} + \hat{\mu}_\delta \mathbf{1}_{n+1}, \boldsymbol{\Sigma}_\delta;\ \hat{\rho}\mathbf{y}^*_{e_c} + b_1\mathbf{1}_{n+1}, \hat{\rho}\mathbf{y}^*_{e_c} + b_2\mathbf{1}_{n+1}),$$

where $\mathbf{y}^*_{e_c} = (y_e(\mathbf{x}), \mathbf{y}_e)^T$, and $\boldsymbol{\Sigma}_\delta = \hat{\sigma}^2_\delta \begin{pmatrix} 1 & \hat{\mathbf{r}}^T_\delta \\ \hat{\mathbf{r}}_\delta & \hat{\mathbf{R}}_\delta \end{pmatrix}$ is the $(n+1)\times(n+1)$ covariance matrix of $\mathbf{y}^*_c = (y_c(\mathbf{x}), \mathbf{y}_c)^T$, with estimated length-scale parameters $\hat{\boldsymbol{\phi}}_\delta$. Next we will show that the distribution of $(y_c(\mathbf{x})|\ \mathbf{y}_c, \mathbf{y}_e, \hat{\boldsymbol{\psi}}, b_1, b_2)$ is truncated normal.

We consider the general case: suppose $\mathbf{w} = (\mathbf{w}_1, \mathbf{w}_2)$ is a multivariate truncated

Corresponding authors: Ke Deng, Department of Statistics and Data Science, Tsinghua University, Beijing 100084, China. E-mail: kdeng@tsinghua.edu.cn. Yu Zhu, Department of Statistics, Purdue University, West Lafayette, IN 47907, USA. E-mail: yuzhu@purdue.edu.

normal random vector with truncated condition $\mathbf{c}^l \leq \mathbf{w} \leq \mathbf{c}^u$,

$$\mathbf{w} \sim \mathcal{TN}_{n_1+n_2}(\boldsymbol{\mu}, \boldsymbol{\Sigma};\ \mathbf{c}^l, \mathbf{c}^u),$$

and $\mathbf{w}_1$ and $\mathbf{w}_2$ are two dependent multivariate truncated normal random vectors with truncated conditions $\mathbf{c}_1^l \leq \mathbf{w}_1 \leq \mathbf{c}_1^u$ and $\mathbf{c}_2^l \leq \mathbf{w}_2 \leq \mathbf{c}_2^u$, respectively,

$$\mathbf{w}_1 \sim \mathcal{TN}_{n_1}(\boldsymbol{\mu}_1, \boldsymbol{\Sigma}_{11};\ \mathbf{c}_1^l, \mathbf{c}_1^u), \quad \mathbf{w}_2 \sim \mathcal{TN}_{n_2}(\boldsymbol{\mu}_2, \boldsymbol{\Sigma}_{22};\ \mathbf{c}_2^l, \mathbf{c}_2^u),$$

where $\boldsymbol{\mu} = (\boldsymbol{\mu}_1, \boldsymbol{\mu}_2)^T$, $\boldsymbol{\Sigma} = \begin{pmatrix} \boldsymbol{\Sigma}_{11} & \boldsymbol{\Sigma}_{12} \\ \boldsymbol{\Sigma}_{21} & \boldsymbol{\Sigma}_{22} \end{pmatrix}$, $\mathbf{c}^l = (\mathbf{c}_1^l, \mathbf{c}_2^l)^T$, $\mathbf{c}^u = (\mathbf{c}_1^u, \mathbf{c}_2^u)^T$ and $\boldsymbol{\Sigma}_{12} = \mathrm{Cov}(\mathbf{w}_1, \mathbf{w}_2) = \boldsymbol{\Sigma}_{21}^T$, $\mathbf{c}^l, \mathbf{c}^u \in \mathbb{R}^{n_1+n_2}$, $\mathbf{c}_1^l, \mathbf{c}_1^u \in \mathbb{R}^{n_1}$, $\mathbf{c}_2^l, \mathbf{c}_2^u \in \mathbb{R}^{n_2}$. Next we will prove that the distribution of $\mathbf{w}_1|\mathbf{w}_2$ is a multivariate truncated normal distribution based on the conclusion from Horrace (2005), and give the analytical form of the density function of $\mathbf{w}_1|\mathbf{w}_2$ in detail.

The density function of $\mathbf{w}$ is given as follows

$$f_{\mathbf{w}}(\mathbf{w}, \boldsymbol{\mu}, \boldsymbol{\Sigma}, \mathbf{c}^l, \mathbf{c}^u) = \frac{\exp\{-\frac{1}{2}(\mathbf{w}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\mathbf{w}-\boldsymbol{\mu})\}}{\int_{\mathbf{c}^l}^{\mathbf{c}^u} \exp\{-\frac{1}{2}(\mathbf{w}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\mathbf{w}-\boldsymbol{\mu})\}d\mathbf{w}},\ \mathbf{c}^l \leq \mathbf{w} \leq \mathbf{c}^u. \quad \text{(S1.1)}$$

By rearranging the terms, we can obtain the following results

$$\exp\{-\frac{1}{2}(\mathbf{w}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\mathbf{w}-\boldsymbol{\mu})\} = g(\mathbf{w}_2) \cdot \exp\{-\frac{1}{2}(\mathbf{w}_1-\boldsymbol{\mu}_{1|2})^T\boldsymbol{\Sigma}_{1|2}^{-1}(\mathbf{w}_1-\boldsymbol{\mu}_{1|2})\},$$

where $\boldsymbol{\mu}_{1|2} = \boldsymbol{\mu}_1 + \boldsymbol{\Sigma}_{12}\boldsymbol{\Sigma}_{22}^{-1}(\mathbf{w}_2-\boldsymbol{\mu}_2)$, $\boldsymbol{\Sigma}_{1|2} = \boldsymbol{\Sigma}_{11} - \boldsymbol{\Sigma}_{12}\boldsymbol{\Sigma}_{22}^{-1}\boldsymbol{\Sigma}_{12}^T$, and $g(\mathbf{w}_2)$ is a function

relates only to $\mathbf{w}_2$. The marginal density functions of $\mathbf{w}_2$ can be expressed as

$$\begin{aligned} f_{\mathbf{w}_2}(\mathbf{w}, \boldsymbol{\mu}, \boldsymbol{\Sigma}, \mathbf{c}^l, \mathbf{c}^u) &= \int_{\mathbf{c}_1^l}^{\mathbf{c}_1^u} f_{\mathbf{w}}(\mathbf{w}, \boldsymbol{\mu}, \boldsymbol{\Sigma}, \mathbf{c}^l, \mathbf{c}^u) d\mathbf{w}_1,\ \mathbf{c}_2^l \le \mathbf{w}_2 \le \mathbf{c}_2^u, \\ &= g(\mathbf{w}_2) \cdot \int_{\mathbf{c}_1^l}^{\mathbf{c}_1^u} \exp\{-\frac{1}{2}(\mathbf{w}_1 - \boldsymbol{\mu}_{1|2})^T \boldsymbol{\Sigma}_{1|2}^{-1}(\mathbf{w}_1 - \boldsymbol{\mu}_{1|2})\} d\mathbf{w}_1 \end{aligned} \tag{S1.2}$$

Using the Bayes rule, we have

$$\begin{aligned} f_{\mathbf{w}_1|\mathbf{w}_2}(\mathbf{w}, \boldsymbol{\mu}, \boldsymbol{\Sigma}, \mathbf{c}^l, \mathbf{c}^u) &= \frac{f_{\mathbf{w}}(\mathbf{w}, \boldsymbol{\mu}, \boldsymbol{\Sigma}, \mathbf{c}^l, \mathbf{c}^u)}{f_{\mathbf{w}_2}(\mathbf{w}, \boldsymbol{\mu}, \boldsymbol{\Sigma}, \mathbf{c}^l, \mathbf{c}^u)} \\ &= \frac{\exp\{-\frac{1}{2}(\mathbf{w}_1 - \boldsymbol{\mu}_{1|2})^T \boldsymbol{\Sigma}_{1|2}^{-1}(\mathbf{w}_1 - \boldsymbol{\mu}_{1|2})\}}{\int_{\mathbf{c}_1^l}^{\mathbf{c}_1^u} \exp\{-\frac{1}{2}(\mathbf{w}_1 - \boldsymbol{\mu}_{1|2})^T \boldsymbol{\Sigma}_{1|2}^{-1}(\mathbf{w}_1 - \boldsymbol{\mu}_{1|2})\} d\mathbf{w}_1},\ \mathbf{c}_1^l \le \mathbf{w}_1 \le \mathbf{c}_1^u, \end{aligned} \tag{S1.3}$$

which is the density function of the multivariate truncated normal random vector

$$\mathbf{w}_1|\mathbf{w}_2 \sim \mathcal{TN}_{n_1}(\boldsymbol{\mu}_{1|2}, \boldsymbol{\Sigma}_{1|2};\ \mathbf{c}_1^l, \mathbf{c}_1^u). \tag{S1.4}$$

Applying the results in (S1.4) to $(y_c(\mathbf{x})|\ \mathbf{y}_c, \mathbf{y}_e, \hat{\boldsymbol{\psi}}, b_1, b_2)$, the results in Theorem 1 can be obtained. The mean and variance of $(y_c(\mathbf{x})|\ \mathbf{y}_c, \mathbf{y}_e, \hat{\boldsymbol{\psi}}, b_1, b_2)$ in Theorem 1 can be directly derived by applying the results of the truncated normal random variable provided by Johnson et al. (1994).

S2 Some Settings and Results of the Experiments

S2.1 Synthetic Black-Box Functions in Section 5.1

The Currin exponential example is defined as

$$\begin{aligned} y_c(\mathbf{x}) &= \Big[1-\exp\Big(-\frac{1}{2x_2}\Big)\Big]\frac{2300x_1^3+1900x_1^2+2092x_1+60}{100x_1^3+500x_1^2+4x_1+20}, \quad \mathbf{x}\in[0,1]^2, \\ y_e(\mathbf{x}) &= \frac{1}{4}y_c\big(x_1+0.05,x_2+0.05\big)+\frac{1}{4}y_c\big(x_1+0.05,\max(0,x_2-0.05)\big) \\ &\quad +\frac{1}{4}y_c\big(x_1-0.05,x_2+0.05\big)+\frac{1}{4}y_c\big(x_1-0.05,\max(0,x_2-0.05)\big). \end{aligned}$$

The Park example is defined as

$$\begin{aligned} y_c(\mathbf{x}) &= \frac{x_1}{2}\left[\sqrt{1+(x_2+x_3^2)x_4/x_1^2}-1\right]+(x_1+3x_4)\exp[1+\sin(x_3)], \\ y_e(\mathbf{x}) &= [1+\sin(x_1)/10]y_c(\mathbf{x})-2x_1+x_2^2+x_3^2+0.5, \quad \mathbf{x}\in[0,1]^4. \end{aligned}$$

The Rosenbrock example is defined as

$$\begin{aligned} y_c(\mathbf{x}) &= \sum_{i=1}^{d-1}100\left(x_{i+1}-x_i^2\right)^2+(1-x_i)^2, \\ y_e(\mathbf{x}) &= \sum_{i=1}^{d-1}50\left(x_{i+1}-x_i^2\right)^2+(2+x_i)^2-\sum_{i=1}^{d}0.5x_i, \quad \mathbf{x}\in[-2,2]^{10}. \end{aligned}$$

Since we have known the explicit expressions of $y_c(\mathbf{x})$ and $y_e(\mathbf{x})$ in advance for the Currin exponential example, Park example and Rosenbrock example, (b_1,b_2) can be determined precisely. We set $(b_1,b_2)=(-1,0.05)$ for the Currin exponential example, $(b_1,b_2)=(-1.5,3.5)$ for the Park example and $(b_1,b_2)=(-154,10)$ for the Rosenbrock example in the TGP-BOPT algorithm. For all of the three ML tasks in Sections 5.2-5.4,

we have prior knowledge that CT loss is often not large than ET loss, and that both ET loss and CT loss are bounded: $0 \leq y_c(\mathbf{x}) \leq y_e(\mathbf{x}) \leq 1$. Since $y_e(\mathbf{x})$ and $y_c(\mathbf{x})$ are at the same scale, we can roughly assume ρ is around 1, and then (b_1, b_2) can be set as $(-1, 0)$ for the TGP-BOPT algorithm. We have also tried simply fixing $\rho = 1$, and the results are similar to the previous case where ρ is estimated.

S2.2 Hyperparameters for the FFCNs and CNNs Models

The hyperparameters for the feed-forward neural networks (Section 5.3) and the convolutional neural networks (Section 5.4) are shown in Table 1 and Table 2, respectively. In Table 2, we consider two different types of optimizer, which are Adam (Kingma and Ba, 2015) and Stochastic Gradient Descent (SGD) (Bottou, 2012), respectively.

Table 1: Hyperparameters for the feed-forward neural networks and their ranges.

Hyperparameter	Range	Log-transform
# hidden layers	$\{1, 2, 3\}$	no
# neurons per layer	$[2^2, 2^6]$	yes
batch size	$[2^3, 2^8]$	yes
initial learning rate	$[10^{-6}, 10^{-2}]$	yes
exponential decay factor	$[0.5, 1]$	no
dropout rate	[0, 0.5]	no

Table 2: Hyperparameters for the convolutional neural network and their ranges.

Hyperparameter	Range	Log-transform
# hidden layers	$\{1, 2, 3, 4, 5\}$	no
# filters per layer	$[2^3, 2^6]$	yes
# fully connected neurons of last hidden layer	$[2^3, 2^8]$	yes
batch size	$[2^3, 2^8]$	yes
initial learning rate	$[10^{-6}, 10^{-2}]$	yes
dropout rate	$[0, 0.5]$	no
optimizer	{Adam, SGD}	no
momentum in SGD	$[0, 1]$	no

In Tables 1-2, these hyperparameters are discrete: the number of hidden layers x_{n_layer}, number of neurons per layer x_{n_neuron}, batch size x_{bs}, filters per layer x_{n_filter}, and choice of optimizer $x_{optimizer}$. In our experiments, x_{n_layer} is an integer variable, taking values from the set $\mathcal{A}_{n\_layer} = \{1, 2, 3\}$ or $\mathcal{A}_{n\_layer} = \{1, 2, 3, 4, 5\}$. Due to the relatively limited number of possible values for x_{n_layer}, we directly model the integer variable x_{n_layer} in its original space during the model fitting stage, and optimize the acquisition functions to determine the next hyperparameter configuration by exhaustively enumerating all possible values in $\mathcal{A}_{n\_layer}$ during the exploration stage. The hyperparameter $x_{optimizer}$ is a categorical variable taking values from {Adam, SGD}. We encode $x_{optimizer}$ as an integer variable, assigning Adam and SGD to integers 0 and 1, respectively. Following a similar logic as with x_{n_layer}, $x_{optimizer}$ can be optimized. However, for the integer variables like

the number of neurons per layer x_{n_neuron}, batch size x_{bs}, and filters per layer x_{n_filter}, which can take any integer value within a wide range like $[2^2, 2^8]$, exhaustive enumeration during the exploration stage becomes laborious. Therefore, we apply a log-transformation to x_{n_neuron}, x_{bs}, and x_{n_filter}, treating them as continuous variables when applying our methods to generate the candidates in the log-transformed space. We then back-transform those candidates to the nearest integers for use as hyperparameters during the training of FFNNs or CNNs.

S2.3 Definitions of ET runs and CT runs

To implement BOPT-HPO, we follow the strategies in Prechelt (2012) to define ET runs and CT runs for further optimization in Sections 5.2-5.4. Let P_e be the maximal number of iterations of an ET run, κ_e the length of training iteration strips, and p_e the threshold of measurement improvement in each ET run, respectively. If the model training procedure under one configuration reaches the maximal number of iterations P_e, or has the measurement improvement less than p_e in κ_e successive iterations, we would early stop the training process and refer to it as an ET run. Replacing (P_e, κ_e, p_e) with (P_c, κ_c, e_c), a set of more aggressive thresholds, rules to claim a CT run can be established in the same way. In practice, those thresholds are specified based on computational resources and the accuracy of the measurement required in the task.

To optimizing the support vector machines in Section 5.2, we recorded the validation

error of one configuration $\mathbf{x}$ during the training process under the setting $(P_e, \kappa_e, p_e) = (50, 5, 0.01)$ and $(P_c, \kappa_c, p_c) = (500, 5, 0)$ as the ET run and CT run, respectively. For the fully connected networks in Section 5.3, since there are a larger number of model parameters to be updated during the training process, training a neural network like a FFNN is more expensive than training a SVM. We reduce the maximal number of iterations and set $(P_e, \kappa_e, p_e) = (5, 3, 0.002)$ for ET runs and $(P_c, \kappa_c, p_c) = (40, 3, 0)$ for CT runs in this experiment. For the convolutional neural network in Section 5.4, The ET runs and CT runs are defined by the thresholds $(P_e, \kappa_e, p_e) = (5, 3, 0.001)$ and $(P_c, \kappa_c, p_c) = (50, 3, 0)$, respectively.

S2.4 Running Time of ET Runs and CT Runs in BOPT-HPO

In practice, the overall running time of BOPT-HPO denoted as T_{all} is composed of three components: time of the ET runs T_{ET} on configuration set $\mathcal{X}_e$, time of the CT runs T_{CT} on configuration set $\mathcal{X}_c$, and time of building the surrogate model T_{model}. Given the same hyperparameter configuration, an CT run always starts from the end of an ET run, so the running time of an CT run is the sum of the running time of the ET stage and the running time of the remaining stage. We denote $T_{\Delta CT}$ as the time of CT runs minus the time of ET runs on $\mathcal{X}_c$. In other words, we have $T_{all} = T_{ET} + T_{\Delta CT} + T_{model}$ for BOPT-HPO. For single-fidelity BO-based HPO methods with only CT runs, however, we have $T_{ET} = 0$ and $T_{all} = T_{CT} + T_{model}$. Here, we further investigate into the effectiveness of BOPT-HPO in

terms of the running time in different levels of training stages.

Figure 1 shows the boxplots of running time ratio between a CT run and the corresponding ET run under different hyperparameter configurations for three ML models in Sections 5.2-5.4. From the figure, we can see that the computation time of implementing a CT run in those experiments is about 3 times larger than that of implementing an ET run. Such a fact perfectly matches the Pareto principle, and suggests that it is reasonable to specify a relative small s, i.e., the number of ET runs per CT run, in the BOPT-HPO algorithm. A wide range of experiments suggest that setting $s = 2$ or 3 usually leads to satisfactory results.

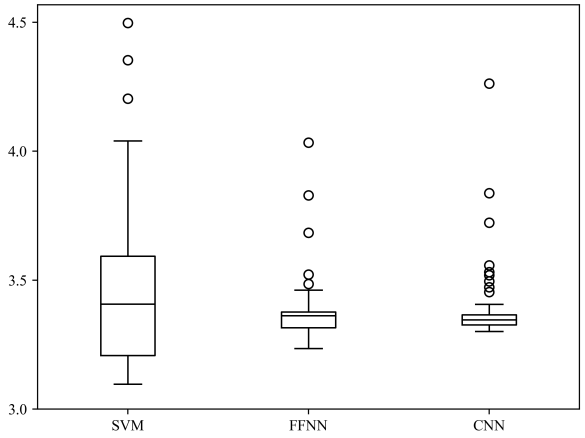

Figure 1: The boxplots of time ratio between each CT run and ET run in three ML tasks.

The proposed BOPT-HPO combines GP and TGP as the surrogate model, so we use the results of GP-BO (GP is the surrogate model) as the baseline to further compare their efficiency. We take the best validation error achieved by GP-BO as the benchmark,

and record the average running time of GP-BO and BOPT-HPO when reaching the same average validation error with five replicates in Table 3. We have experimentally found that the running time of building the surrogate model in GP-BO and BOPT-HPO is negligible compared to the running time of ET runs and CT runs in those ML tasks. By utilizing only 20%-30% of the overall running time for ET runs, BOPT-HPO achieves a reduction of approximately 34% in the running time of CT runs and improves the overall computation efficiency by 20% to 25% compared to GP-BO, when reaching the same best validation errors. Aggregating the results in Figures 5-7, if we allocate more computational resources, BOPT-HPO continues to improve the validation errors, while the validation errors of GP-BO remain almost unchanged.

Table 3: Running time (in seconds) of GP-BO and BOPT-HPO in three experiments.

	GP-BO				BOPT-HPO			
Experiment	Best error	T_{CT}	T_{model}	T_{all}	T_{ET}	$T_{\Delta CT}$	T_{model}	T_{all}
SVM	0.0153	31,204	393	31,507	7,369	16,552	381	24,302
FFNN	0.0323	39,387	763	40,150	9,083	22,918	899	32,900
CNN	0.28	36,673	662	37,335	8,207	19,248	845	28,300

S2.5 Area Under the Error-Cost Curve

The Area Under the Receiver Operating Characteristic Curve (ROC-AUC), is a metric commonly used to evaluate the performance of classification models in machine learning

(Hand, 2009). It quantifies the model's ability to distinguish between positive and negative instances. Given the comparable performance of DGP-BOPT and TGP-BOPT in our experiments, it is challenging to determine which method is superior. Therefore, drawing inspiration from the ROC-AUC, we introduce the concept of the Area Under the Error-Cost Curve (EC-AUC) in our work to provide a more precise evaluation of the performance of DGP-BOPT and TGP-BOPT. In all experiments, the EC-AUC is approximated using the trapezoidal rule, and the results are presented in Table 4.

Table 4: The EC-AUC of DGP-BOPT and TGP-BOPT in all experiments.

	Currin	Park	Rosenbrock	SVM	FFNN	CNN
DGP-BOPT	10.842	34.293	79.443	1034.659	1137.409	13715.513
TGP-BOPT	9.938	33.260	78.032	1035.200	1137.362	13695.850

Bibliography

Bottou, L. (2012). Stochastic gradient descent tricks. In *Neural Networks: Tricks of the Trade*, pp. 421–436.

Hand, D. J. (2009). Measuring classifier performance: a coherent alternative to the area under the roc curve. *Machine Learning 77*(1), 103–123.

Horrace, W. C. (2005). Some results on the multivariate truncated normal distribution. *Journal of Multivariate Analysis 94*(1), 209–221.

Johnson, N. L., S. Kotz, and N. Balakrishnan (1994). Continuous univariate distributions, Volume 1. *Wiley*.

Kingma, D. P. and J. L. Ba (2015). Adam: A method for stochastic optimization. In *The International Conference on Learning Representations*.

Prechelt, L. (2012). Early stopping-But when? In *Neural Networks: Tricks of the Trade*, pp. 53–67. Springer.