Can Bayesian Optimization be the Last Puzzle for Automatic Estimation of Neural Network Discrete Choice Models? An experiment

Rui Yao[1] and Renming Liu[1]

[1]Technical University of Denmark, Denmark

Short summary

This study investigates the performances of Bayesian optimization (BO) and random grid search methods for tuning neural network hyper-parameters in the context of discrete choice modeling. Specifically, the fully-connected feed-forward (FNN) and alternative-specific-utility neural networks (ASU) are tuned. Results show that BO outperforms random grid search for both FNN and ASU models in terms of out-of-sample log-likelihood. Furthermore, it is illustrated that BO has higher sample efficiency and is relatively more robust to different random initialization. Our experiments show that the Bayesian hyper-parameter tuning framework could accommodate and complement existing neural network models that are cast for automatic utility function specifications, and create a fully automatic estimation workflow.
Keywords: Bayesian optimization, Discrete choice modeling, Hyper-parameter tuning, Neural network models

1 Introduction

Discrete choice modeling typically requires prior knowledge of the utility functions (Han et al., 2022), which are often, however, specified using trial-and-error based on researcher's interpretations and experiences. Recent studies have adapted neural network (NN) methods for automatic utility function specifications and shown greater predictive power of NN models (e.g., Lee et al., 2018; Wang, Wang, & Zhao, 2020).

However, general-purpose neural networks tend to over-fitting. Consequently, their out-of-sample performances (Han et al., 2022), as well as their interpretability (Wang, Mo, & Zhao, 2020), could be poor. To tackle these issues, several studies have incorporated domain-knowledge-based regularization methods with NN models, by designing specific sparse NN architectures. For example, Sifringer et al. (2020) and Han et al. (2022) assign single-layer sparse (e.g., linear-in-parameter) neural network for the *interpretable* component of the systematic utility, and report improved predictive power and retained interpretability. Wang, Mo, & Zhao (2020) also show that their proposed alternative-specific-utility (ASU) NN modeling framework, in which only attributes associated with the same alternative are connected, generally improves model performances compared to fully-connected feed-forward NN. Yao & Bekhor (2022) applies the variational autoencoder neural network to estimate the implicit availability perception of alternatives and embed it in the utility computation.

Although these domain-specific NN for choice modeling have demonstrated the automatic feature-learning power of NN, their performances still heavily depend on the hyper-parameters. Wang, Mo, & Zhao (2020) shows that poorly tuned hyper-parameters can hinder the performances of NN models due to their large estimation error.

The challenges in selecting the NN hyper-parameters are two-fold: 1) Large number of hyper-parameters; 2) Bi-level problem structure. The lower-level model estimation depends on the upper-level hyper-parameter selection; whereas the upper-level hyper-parameter tuning typically requires evaluating the lower-level objective (i.e., out-of-sample log-likelihood in the context of choice modeling). In the case of large datasets, which is typical for applying data-driven methods, the lower-level model estimation could be time-consuming for complex NN models.

One classic approach for hyper-parameter tuning is to perform a random grid search on the hyper-parameter space and select the set of hyper-parameters with high out-of-sample log-likelihood (Bergstra & Bengio, 2012). Recently, Snoek et al. (2012) shows that, compared to random search, the Bayesian optimization (BO) method is more efficient and improves state-of-the-art performances of many machine-learning tasks.

The primary objective of BO is to identify the global optimum of an unknown function with only a small number of evaluations. This is achieved by modeling the unknown function as a Gaussian process (GP) and selecting the next sample point that maximizes an acquisition function derived from the GP. The acquisition function is designed to balance the trade-off between exploitation, where the mean is high, and exploration, where the uncertainty is high. BO is therefore employed for solving problems that are expensive to evaluate, have an unknown structure, such as concavity or linearity (i.e., the function is a black-box), and possess a continuous objective function (Frazier, 2018).
This study aims to evaluate and compare the performances of the Bayesian optimization method and random grid search for improving the model out-of-sample log-likelihood, which could provide a promising direction for a fully automatic neural network discrete choice model estimation pipeline.

2 Methodology

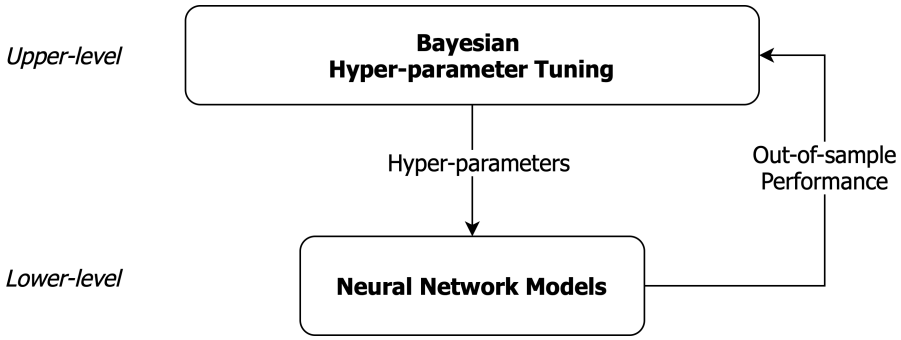

Figure 1. Bayesian hyper-parameter tuning framework

We show in Figure 1 the overall Bayesian hyper-parameter tuning framework. In the following subsections, we first detail the lower-level neural network models used in this study for discrete choice modeling. Next, the Bayesian hyper-parameter tuning method is introduced. Lastly, we briefly introduce the benchmark random grid search method.

Neural networks for choice modeling

Two neural network models, fully-connected feed-forward neural network (FNN) and alternative-specific utility neural network (ASU, Wang, Mo, & Zhao (2020)), are selected in this study for comparing the performances of Bayesian hyper-parameter tuning and random grid search in the context of choice modeling.

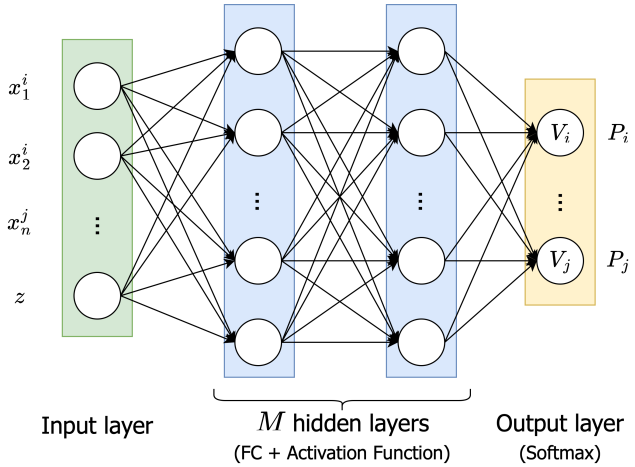

Figure 2. FNN architecture for discrete choice modeling

We show in Figure 2 the FNN architecture, in which there are two types of inputs, namely, alternative-specific attributes $\boldsymbol{x}^i$ for alternative i, and individual-specific attributes $\boldsymbol{z}$. The FNN

model connects all input attributes to the hidden layer neurons through fully-connected layers (FC) and activation functions. Correspondingly, the systematic utility of an alternative i, V_i, is function of attributes of *all* alternatives $\boldsymbol{x} = \{\boldsymbol{x}^i, \forall i\}$ and the individual-specific attributes $\boldsymbol{z}$. Mathematically, systematic utility V_i of FNN can be defined recursively as:

$$V_i = V(\boldsymbol{z}, \boldsymbol{x}) = w_i^\top \left(g_M \circ ... \circ g_1\right)(\boldsymbol{z}, \boldsymbol{x}) \tag{1}$$

where, w_i denote the weights on the output layer (i.e., readout) for alternative i before applying softmax (Logit function), $g_m(y) = \Phi(W_m^\top y)$ denote the hidden layer $m \in [1, M]$, $\Phi(\cdot)$ denote the activation function, W_m is the weight on the FC layer m, and $\circ$ denote function composition. We refer to Wang, Mo, & Zhao (2020) for detailed proof on the connection between FNN and utility function specification.

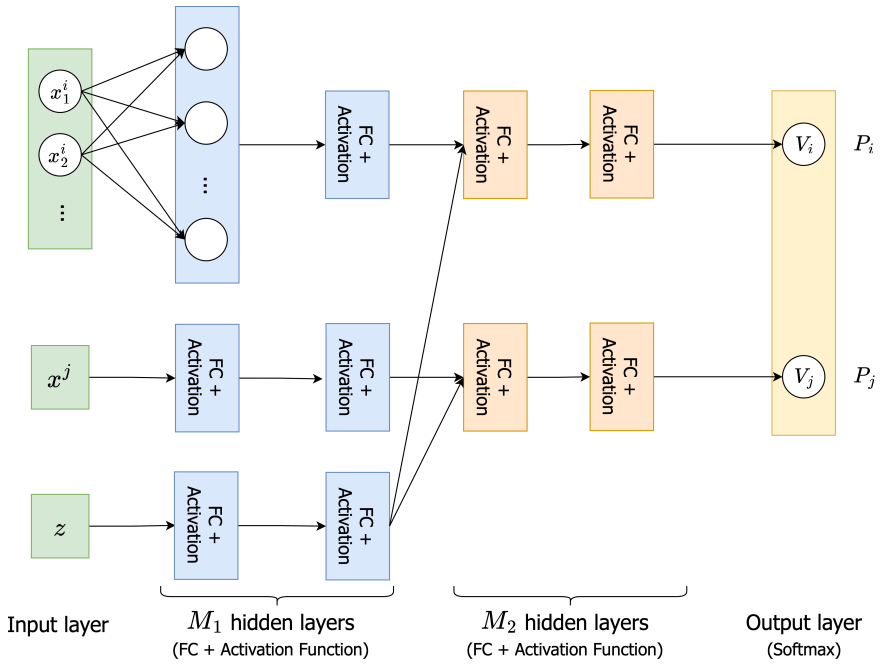

Figure 3. ASU architecture for discrete choice modeling
(adapted from Wang, Mo, & Zhao (2020))

The ASU architecture is shown in Figure 3. Different from the FNN architecture, the ASU model first transforms independently each set of alternative-specific attributes $\boldsymbol{x}^i$ and the individual-specific attributes $\boldsymbol{z}$ with separate NNs. Only after M_1 layers of transformation, $\boldsymbol{z}$ enter the systematic utility V_i. As a result, V_i, is only function of alternative-specific attributes $\boldsymbol{x}^i$ and the individual-specific attributes $\boldsymbol{z}$, which can be formally defined as:

$$V_i = V(\boldsymbol{z}, \boldsymbol{x}^i) = w_i^\top \left(g_{M_2}^i \circ ... \circ g_1^i\right)\left(\left(g_{M_1}^z \circ ... \circ g_1^z\right)(\boldsymbol{z}), \left(g_{M_1}^i \circ ... \circ g_1^i\right)\boldsymbol{x}^i\right) \tag{2}$$

where, $g_{m_1}^i$ and $g_{m_1}^z$ denote the hidden layers for separate transformation of alternative-specific attributes $\boldsymbol{x}^i$ and sociodemographic attributes $\boldsymbol{z}$, and $g_{m_2}^i$ denote the additional transformation after $\boldsymbol{z}$ enters the computation of V_i.
We summarize a list of selected FNN and ASU hyper-parameters for comparing the performances of Bayesian optimization and random grid search in Table 1:

Table 1. Hyper-parameters for FNN and ASU

FNN	
M	Number of FNN hidden layers
n	Width (number of neurons) of each FNN hidden layer
ASU	
M_1	Number of ASU hidden layers for separate transformations
M_2	Number of ASU hidden layers after $\boldsymbol{z}$ enters
n_1	Width of ASU M_1 hidden layer
n_2	Width of ASU M_2 hidden layer
Generic	
l_1	L1 regularization parameter to control model sparsity
l_2	L2 regularization parameter to control coefficient magnitudes
Dropout rate	Probability of dropping some coefficients for model sparsity
Learning rate	Step size in stochastic gradient descent (SGD)
Batch size	Number of observations per batch in SGD
Batch normalization	Normalizing each batch of observations in SGD

Note that, we set the number of iterations as 20,000 and employ an early-stopping strategy for training both FNN and ASU, for which the algorithm stops if the log-likelihood of the validation set does not improve in 50 consecutive iterations. Other hyper-parameters are set as recommended values in the literature.

Bayesian hyper-parameter tuning

A BO framework comprises two primary steps (Frazier, 2018). The first step involves updating a Bayesian statistical model which approximates the complex mapping from the hyper-parameters (denoted by $\boldsymbol{\theta}$), to the objective values (i.e., out-of-sample log-likelihood l). The second step is to select a candidate hyper-parameter vector that optimizes the acquisition function and evaluates its performance.

The Gaussian process is often chosen as the prior for the statistical model, because of its tractability in computing posterior and predictive distributions. The GP is characterized by its mean function $\mu_0(\boldsymbol{\theta})$, and its covariance kernel function, denoted by $k(\boldsymbol{\theta}, \boldsymbol{\theta}')$. Given a set $\mathcal{D}_m$ containing hyper-parameters and their corresponding objective values, that is, $\mathcal{D}_m = \{\boldsymbol{\theta}_{1:m}, l_{1:m}\}$, where subscript $1:m$ represents $||m||$ variables.

The joint distribution of $l_{1:m}$ is Gaussian:

$$l_{1:m} \ \sim \ \mathcal{N}(\mu_0(\boldsymbol{\theta}_{1:m}),\ K(\boldsymbol{\theta}_{1:m}, \boldsymbol{\theta}'_{1:m})), \tag{3}$$

where μ_0 is the prior mean function, which is usually set as a constant value (0 in this study), and $K(\boldsymbol{\theta}_{1:m}, \boldsymbol{\theta}'_{1:m})_{i,j} = k(\boldsymbol{\theta}_i, \boldsymbol{\theta}'_j)$, for $i,\ j \in \{1, 2, \ldots, m\}$, is the covariance matrix. We also assume k as the commonly used Matern kernel.

The posterior distribution of l_{m+1} can be computed using Bayes' theorem:

$$l_{m+1}|l_{1:m} \ \sim \ \mathcal{N}\Big(\mu(\boldsymbol{\theta}_{m+1}),\ \sigma^2(\boldsymbol{\theta}_{m+1})\Big), \tag{4}$$

where $\mu(\boldsymbol{\theta}_{m+1}) = \boldsymbol{k}^T \boldsymbol{K}^{-1} l_{1:m}$ and $\sigma^2(\boldsymbol{\theta}_{m+1}) = k(\boldsymbol{\theta}_{m+1}, \boldsymbol{\theta}_{m+1}) - \boldsymbol{k}^T \boldsymbol{K}^{-1} \boldsymbol{k}$.

The fitted GP, which serves as a surrogate model of the lower-level problem objective (i.e., out-of-sample log-likelihood in our paper), is capable of predicting the value of the objective function at unevaluated hyper-parameter locations based on previously collected data points $\mathcal{D}_m = \{\boldsymbol{\theta}_{1:m}, l_{1:m}\}$. The goal is to select the next vector of hyper-parameters with the highest gained value of information. Such value of information is measured by an acquisition function, which is a proxy function derived from the mean and variance of the objective function values and directs the next sample point. This study uses a popular acquisition function, Expected Improvement (EI), which computes the expected improvement with respect to the current maximum $l^* = \max l_{1:m}$, i.e.,

$$EI(\boldsymbol{\theta}) = \mathbb{E}([l_{m+1} - l^*]^+ | \boldsymbol{\theta}_{1:m}, l_{1:m}). \tag{5}$$

The next point to be evaluated is determined by:

$$\boldsymbol{\theta}_{m+1} = \arg\max_{\boldsymbol{\theta}} EI(\boldsymbol{\theta}) \tag{6}$$

This optimization problem can be efficiently solved by local solvers like L-BFGS-B (Liu & Nocedal, 1989) with multiple restarts.

Random grid search

For a given grid of hyper-parameter values, a random grid search selects random combinations of these values to train the models. The set of hyper-parameters with the best out-of-sample performance is chosen in a post-hoc manner. Although the independent sampling procedure suggests that random grid search can be performed in parallel, its efficiency (in terms of computational costs) and effectiveness (in terms of *best* performance) could be low (Snoek et al., 2012).

3 Results and discussion

Dataset and experiment setup

Our experiment is based on the swissmetro dataset (Bierlaire et al., 2001), for which 6,768 observations with trip purpose of *commute* and *business* are selected for model estimation. In the dataset, respondents choose among 3 alternative modes: train, swissmetro, and car. Data statistics of the attributes used for estimation are summarized in Table 2:

Table 2. Data statistics for the selected swissmetro dataset

Attribute	Mean	Std.
Train time [min]	166.63	77.35
Train cost [CHF]	514.34	1088.93
Swissmetro time [min]	87.47	53.55
Swissmetro cost [CHF]	670.34	1441.59
Car time [min]	123.80	88.71
Car cost [CHF]	78.74	55.26
Number of observations: 6,768		
Number of choices: Train (13.26%), Swissmetro (57.94%), Car (28.71%)		

The dataset is divided into training, validation, and out-of-sample sets in the ratio 4: 1: 1. A five-fold cross-validation is used for model selection. That is, each model (i.e. one set of hyper-parameters) is trained 5 times with different data folds, and evaluated with the out-of-sample set. The model performance is taken as the average of 5 out-of-sample log-likelihoods.
We consider the following hyper-parameter space for FNN and ASU (Table 3):

Table 3. Hyper-parameter space for FNN and ASU

Hyper-parameter	Values
FNN	
M	$\{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12\}$
n	$\{60, 120, 240, 360, 480, 600\}$
ASU	
M_1	$\{1, 2, 3, 4, 5, 6\}$
M_2	$\{0, 1, 2, 3, 4, 5, 6\}$
n_1	$\{10, 20, 40, 60, 80\}$
n_2	$\{10, 20, 40, 60, 80, 100\}$
Generic	
l_1	$[1^{-20}, 1.0]$
l_2	$[1^{-20}, 1.0]$
Dropout rate	$[1^{-20}, 0.1]$
Learning rate	$[1^{-5}, 0.5]$
Batch size	$\{50, 100, 200, 500, 1000\}$
Batch normalization	{True, False}
Invariant hyper-parameters	
Activation function	ReLU (Rectified linear unit) and Softmax (Logit)
Loss function	Log-likelihood
Weight initialization	He initialization (He et al., 2015)

Note that, we consider discrete architecture parameters for FNN and ASU, while selected parameters for the training algorithm, namely, l_1, l_2, dropout rate, and learning rate, are considered as continuous values. For the random grid search method, all discrete values are uniformly sampled, whereas log-uniform sampling is applied for the training parameters in order to draw values at different magnitudes.
Both FNN and ASU are implemented in PyTorch 1.12 with Adam optimizer (Kingma & Ba, 2014). The Bayesian optimization is implemented with the BoTorch package (Balandat et al., 2020). We set the number of hyper-parameter tuning iterations as 100 for ASU, and 80 for FNN (due to fewer hyper-parameters to be tuned). In addition, 2 replications with different random seeds are performed for both the BO and random grid search. That is, for each hyper-parameter tuning method, we estimate 200 ASU models and 160 FNN models.

Comparison of out-of-sample performances

We report out-of-sample performance and architecture parameters of the *best* models obtained by BO and random grid search methods in Table 4.

Table 4. Comparison of the *best* model of Bayesian optimization and random grid search

	Bayesian Optimization	Random grid search
FNN		
M	5	11
n	240	60
Out-of-sample performance	−741.17	−768.60
ASU		
M_1	1	3
M_2	1	1
n_1	40	10
n_2	80	20
Out-of-sample performance	−762.03	−787.94

As shown in Table 4, the BO outperforms random grid search in terms of out-of-sample log-likelihood for both the FNN and ASU models. Moreover, compared to random grid search, the BO method is able to find network architecture with fewer layers yet stronger predictive power on the out-of-sample dataset. This is consistent with the literature (e.g., Hillel, 2019; Han et al., 2022), for which shallower but wider NN empirically performs better in choice modeling tasks.
We further examine the hyper-parameter optimization iterations of BO and random grid search methods in Figure 4, where *iteration numbers* for the random grid search are sorted models.

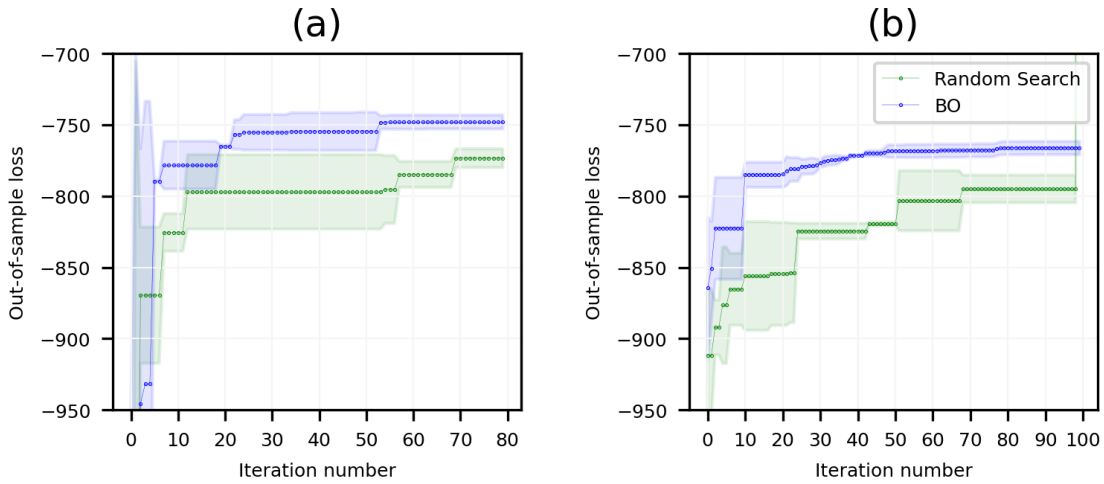

Figure 4. The optimization for hyper-parameters of (a) FNN and (b) ASU architectures (shaded area represents 95% confidence interval)

As shown in Figure 4, for both the FNN and ASU models, the BO method outperforms the random grid search after the initial 10 iterations. This suggests that, compared to random grid search, the BO method has higher sample efficiency, i.e., the BO candidate hyper-parameters have a higher potential to improve the out-of-sample performance. Furthermore, the variance of out-of-sample performance of BO iterations is smaller than random grid search (as indicated by the smaller shaded area of BO), which suggests the BO method could be more robust to (initial) randomization. In the following subsection, we further illustrate the sample efficiency of BO and random grid search.

Illustration of sample efficiency between Bayesian and random search methods

We next investigate the sample efficiency of random search and BO methods by illustrating the distributions of sample points for the ASU models, as shown in Figure 5 and 6. The diagonal plots show histograms for each hyper-parameter, and the lower triangle shows two-dimensional scatter plots of all sample points for each pair of the hyper-parameters. The red points represent the set

of hyper-parameters with the best out-of-sample performance (termed *optimal* hereinafter) found by each method.

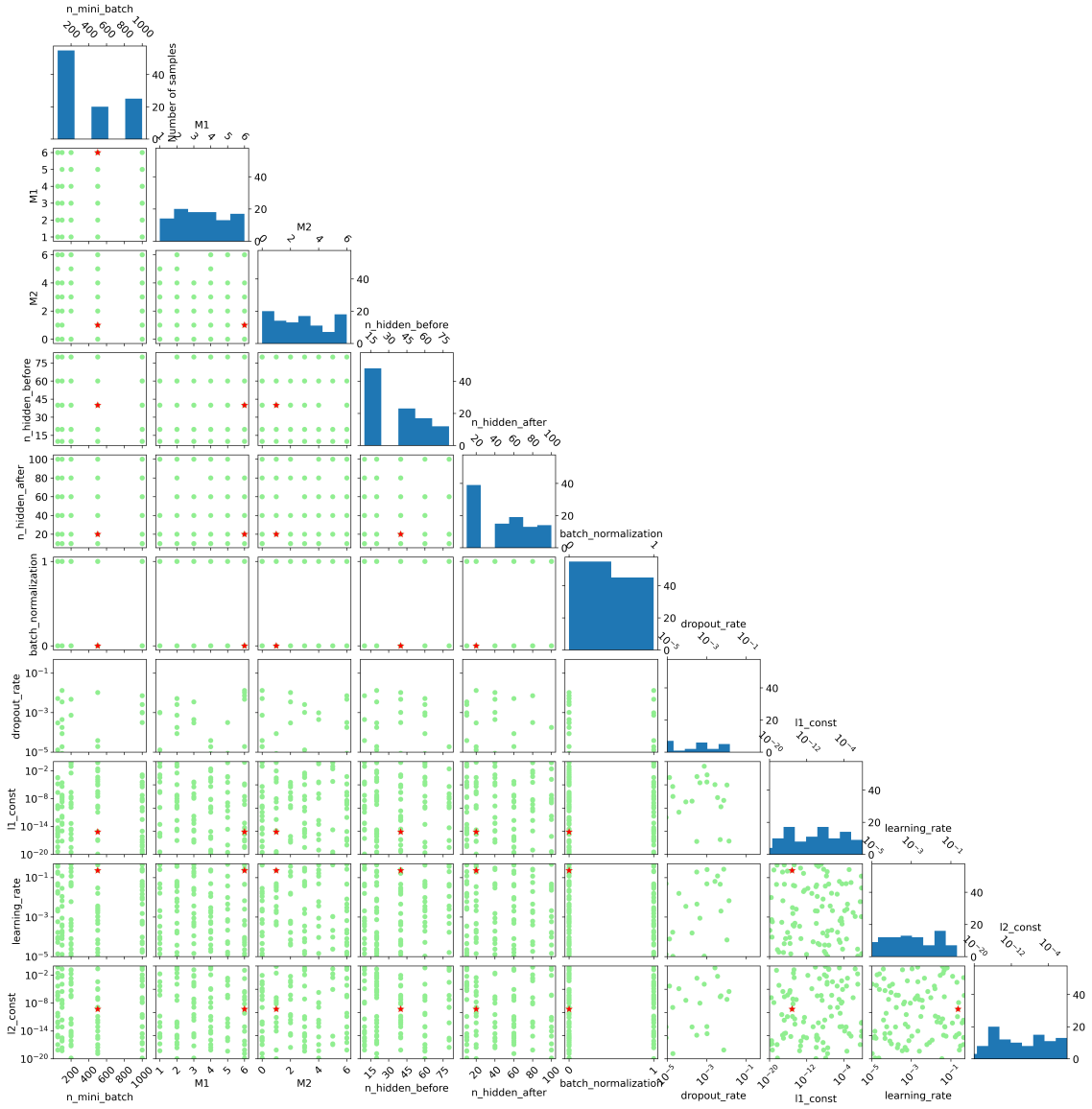

Figure 5. Sample points of the random search for ASU architecture.
(Red points represent the *optimal* hyper-parameters)

As shown in Figure 5 for the random grid search method, all sample points are distributed almost uniformly across the search space. Although random grid search has been proven to be more efficient than brutal-force full grid search (Bergstra & Bengio, 2012), this uniform distribution of sample points still suggests the out-of-sample performance might not be improved in consecutive iterations, resulting in relatively poorer sample efficiency. This can also be verified by the longer and flatter platoons of random grid search in Figure 4 for out-of-sample performance versus iterations.

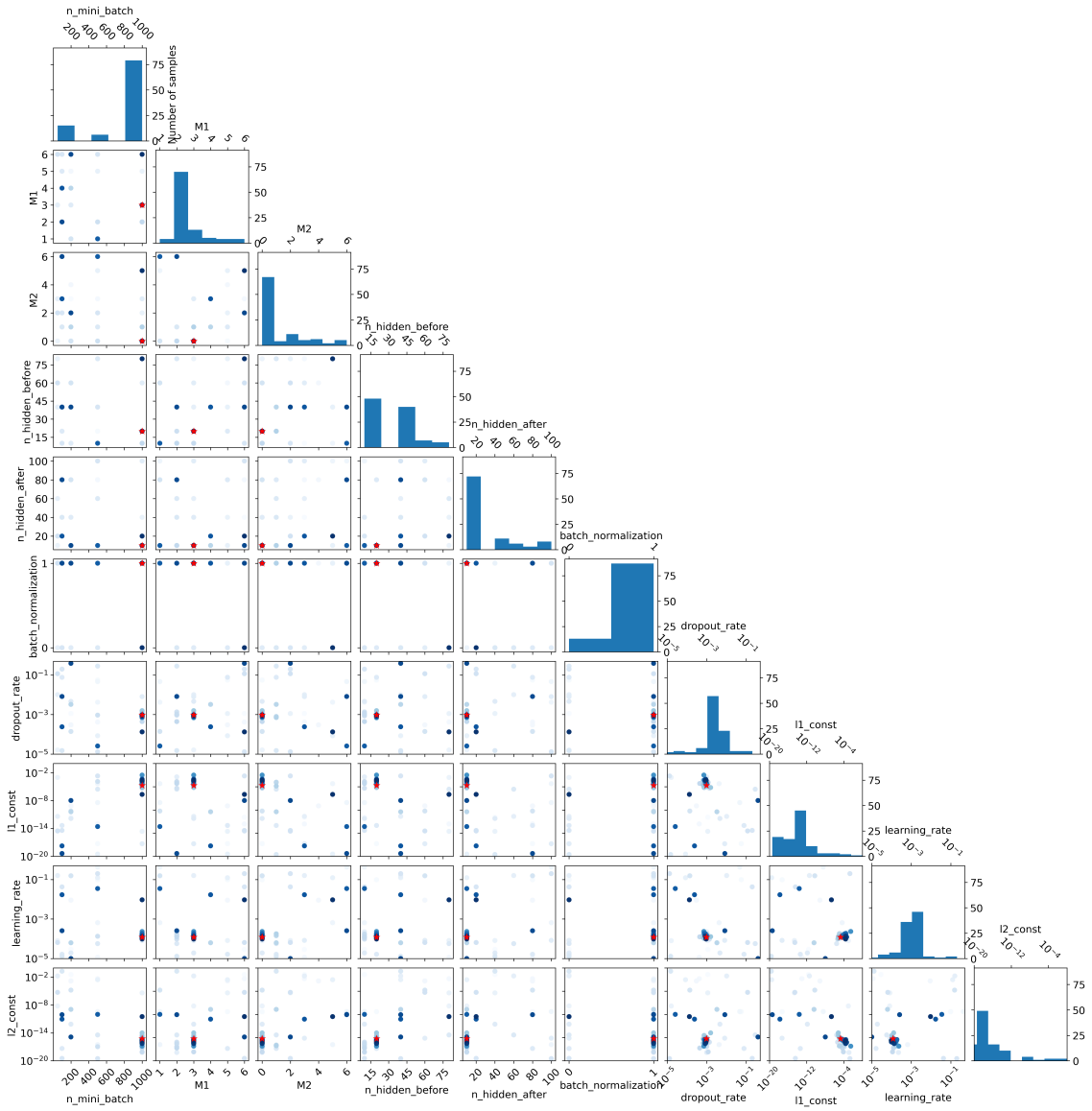

Figure 6. Acquisition points of the Bayesian optimization for ASU architecture. (Red points represent the *optimal* hyper-parameters, and darker colors correspond to later samples)

On the other hand, BO follows a different pattern. As shown in Figure 6, acquisition points obtained by BO gradually cluster around the *optimal* hyper-parameters, as indicated by the darker points centered around the red points. This clustering effect is also evident in the histograms, in which the highest frequencies are observed around the *optimal* hyper-parameters. These results indicate that the BO method has higher sample efficiency, compared to random grid search.
The behavior of the BO method can be explained by its optimization procedures. As more sample points are collected/evaluated, the surrogate model has higher confidence about its estimation of out-of-sample performance at unexplored hyper-parameter points. As a result, the associated acquisition function can direct more efficiently the search towards the *optimal* hyper-parameters with the improved surrogate model estimations.
Note that, similar patterns of sample points for random grid search and BO are also observed of FNN models, which suggests the BO method is expected to have relatively higher sample efficiency.

4 Conclusions

This study investigates the performances of Bayesian optimization (BO) and random grid search methods for tuning neural network hyper-parameters in the context of discrete choice modeling. Specifically, the fully-connected feed-forward (FNN) and alternative-specific-utility neural networks

(ASU) are tuned, and the out-of-sample performances as well as the sample efficiencies of the BO and random grid search methods are compared.
Results show that BO outperforms random grid search for both FNN and ASU models in terms of out-of-sample log-likelihood. Furthermore, it is illustrated that BO has higher sample efficiency and is relatively more robust to different random initialization.
Our experiments show that BO provides a promising direction for a fully automatic estimation workflow of neural network discrete choice models. Future research will extend the experiment to larger datasets, transferability of BO among datasets, as well as other data-driven machine learning models.

References

Balandat, M., Karrer, B., Jiang, D. R., Daulton, S., Letham, B., Wilson, A. G., & Bakshy, E. (2020). BoTorch: A Framework for Efficient Monte-Carlo Bayesian Optimization. In *Advances in neural information processing systems 33*. Retrieved from `http://arxiv.org/abs/1910.06403`

Bergstra, J., & Bengio, Y. (2012). Random search for hyper-parameter optimization. *Journal of machine learning research*, *13*(2).

Bierlaire, M., Axhausen, K., & Abay, G. (2001). The acceptance of modal innovation: The case of swissmetro. In *Swiss transport research conference.*

Frazier, P. I. (2018). A tutorial on bayesian optimization. *arXiv preprint*.

Han, Y., Pereira, F. C., Ben-Akiva, M., & Zegras, C. (2022). A neural-embedded discrete choice model: Learning taste representation with strengthened interpretability. *Transportation Research Part B: Methodological*, *163*, 166–186.

He, K., Zhang, X., Ren, S., & Sun, J. (2015). Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the ieee international conference on computer vision* (pp. 1026–1034).

Hillel, T. (2019). *Understanding travel mode choice: A new approach for city scale simulation* (Unpublished doctoral dissertation). University of Cambridge.

Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Lee, D., Derrible, S., & Pereira, F. C. (2018). Comparison of four types of artificial neural network and a multinomial logit model for travel mode choice modeling. *Transportation Research Record*, *2672*(49), 101–112.

Liu, D. C., & Nocedal, J. (1989). On the limited memory bfgs method for large scale optimization. *Mathematical programming*, *45*(1-3), 503–528.

Sifringer, B., Lurkin, V., & Alahi, A. (2020). Enhancing discrete choice models with representation learning. *Transportation Research Part B: Methodological*, *140*, 236–261.

Snoek, J., Larochelle, H., & Adams, R. P. (2012). Practical bayesian optimization of machine learning algorithms. *Advances in neural information processing systems*, *25*.

Wang, S., Mo, B., & Zhao, J. (2020). Deep neural networks for choice analysis: Architecture design with alternative-specific utility functions. *Transportation Research Part C: Emerging Technologies*, *112*, 234–251.

Wang, S., Wang, Q., & Zhao, J. (2020). Multitask learning deep neural networks to combine revealed and stated preference data. *Journal of choice modelling*, *37*, 100236.

Yao, R., & Bekhor, S. (2022). A variational autoencoder approach for choice set generation and implicit perception of alternatives in choice modeling. *Transportation Research Part B: Methodological*, *158*, 273–294.