Convolutional Neural Networks based Lung Nodule Classification: A Surrogate-Assisted Evolutionary Algorithm for Hyperparameter Optimization

Miao Zhang, Huiqi Li, *Senior Member, IEEE,* Shirui Pan, *Member, IEEE,* Juan Lyu, Steve Ling, *Senior Member, IEEE* and Steven Su, *Senior Member, IEEE*

***Abstract*—This paper investigates deep neural networks (DNNs) based lung nodule classification with hyperparameter optimization. Hyperparameter optimization in DNNs is a computationally expensive problem, and a surrogate-assisted evolutionary algorithm has been recently introduced to automatically search for optimal hyperparameter configurations of DNNs, by applying computationally efficient surrogate models to approximate the validation error function of hyperparameter configurations. Different from existing surrogate models adopting stationary covariance functions (kernels) to measure the difference between hyperparameter points, this paper proposes a non-stationary kernel that allows the surrogate model to adapt to functions whose smoothness varies with the spatial location of inputs. A multi-level convolutional neural network (ML-CNN) is built for lung nodule classification, and the hyperparameter configuration is optimized by the proposed non-stationary kernel-based Gaussian surrogate model. Our algorithm searches with a surrogate for optimal setting via a hyperparameter importance based evolutionary strategy, and the experiments demonstrate our algorithm outperforms manual tuning and several well-established hyperparameter optimization methods, including random search, grid Search, the Tree-structured Parzen Estimator Approach (TPE), Gaussian processes (GP) with stationary kernels, and the recently proposed Hyperparameter Optimization via RBF and Dynamic coordinate search (HORD).**

***Index Terms*—Lung nodule classification, hyperparameter optimization, AutoML, non-stationary kernel, evolutionary algorithm**

I. Introduction

LUNG cancer is a notoriously aggressive cancer. Sufferers have an average 5-year survival rate of 18% with a mean survival time of fewer than 12 months [45], [58], hence early diagnosis is very important to improve the survival rate. Recently, deep learning has shown its superiority in computer vision [14], [26], [56], and an increasing number of researchers have tried to diagnose lung cancers with deep neural networks such as computer aided diagnosis (CAD) systems [1], [15], [48], [63], [40], [42], [10] to assist early diagnosis. In our previous work [34], a multi-level convolutional neural network (ML-CNN) is proposed to handle lung nodule malignancy classification, which extracts multi-scale features through different convolutional kernel sizes. Our ML-CNN [34] achieves competitive accuracy in both binary and ternary classification (92.21% and 84.81% accuracy, respectively) without any preprocessing. However, the experiments also indicate the performance is very sensitive to hyperparameters, especially the number of feature maps in each convolutional layer. We obtain near-optimal hyperparameter configuration through trial and error, which is a difficult and time-consuming task [35], [11].

Automatic hyperparameter optimization, which is an important branch of AutoML, is a very crucial step for deep learning algorithms in practical applications, and several methods including grid search [27], random search [4], the Tree-structured Parzen Estimator Approach (TPE) [3] and Bayesian optimization[46], [60] have shown their superiority over manual tuning in hyperparameters optimization. Hyperparameter optimization in deep neural networks is a global optimization with a black-box and expensive function, where evaluating a hyperparameter setting may cost several hours or even days. It is a computationally expensive problem, and a popular solution is to employ a probabilistic surrogate, such as Gaussian processes (GP) and Tree-structured Parzen Estimator (TPE), to approximate the expensive error function to guide the optimization process. A stationary covariance function (kernel) is usually used in these surrogates to measure the difference between hyperparameter points based on spatial distance without considering its spatial locations. Such a covariance function that employs constant smoothness throughout the hyperparameter search space violates the intuition that most points away from the optimal point all attain similarly poor performance even though them have large spatial distance.

In this paper, a deep neural network for lung nodule classification is built based on multi-level convolutional neural networks (ML-CNN), in which three levels of CNNs with the same structure but different convolutional kernel sizes are designed to extract the multi-scale features of input with variable nodule sizes and morphologies. Then the hyperparameter optimization in a deep convolutional neural network

M. Zhang is with the School of Information and Electronics, Beijing Institute of Technology, Beijing 100081, China, and with the Faculty of Engineering and Information Technology, University of Technology Sydney, NSW 2007, Australia, and also with the Faculty of Information Technology, Monash University, VIC 3800, Australia. E-mail: miaozhang1991@gmail.com.

H. Li is with the School of Information and Electronics, Beijing Institute of Technology, Beijing 100081, China. E-mail: huiqili@bit.edu.cn

S. Pan is with the Faculty of Information Technology, Monash University, VIC 3800, Australia. E-mail: shirui.pan@monash.edu

J. Lyu is with the Information and Communication Engineering, Harbin Engineering University (HEU), Harbin 150001, China. E-mail: lvjuan@hrbeu.edu.cn

S. Ling and S. Su are with the Faculty of Engineering and Information Technology, University of Technology Sydney, NSW 2007, Australia. E-mail: steve.ling@uts.edu.au, steven.su@uts.edu.au

Corresponding authors: Huiqi Li and Shirui Pan.

Manuscript received March 15, 2020; revised July 22, 2020.

is formulated as an expensive optimization problem, and a Gaussian surrogate model is built to approximate the error function of hyperparameter configurations, which is a novel attempt to handle hyperparameter optimization in CNN-based lung nodule classification. Appropriately measuring the distance between the hyperparameter settings is the key point for the hyperparameter optimization of DNNs. In the hyperparameter optimization of DNN, all the points far away from the optimal point all usually perform poorly and objective functions are often much more sensitive near the optimal point. To take these non-stationary characters into consideration, this paper proposes a non-stationary kernel, which utilizes spatial location transformation and input warping, to allow the model to adapt its smoothness variations with the inputs. Our algorithm searches the surrogate via a hyperparameter importance based evolutionary strategy and finds the near-optimal hyperparameter setting in limited function evaluations.

We name our algorithm **H**yperparameter **O**ptimization with s**U**rrogate-a**S**sisted **E**volutionary **S**trategy, HOUSES for short. We compare our algorithm with several well-established hyperparameter optimization algorithms, namely random search, grid search, the Tree-structured Parzen Estimator (TPE), the Gaussian process with stationary kernels, and Hyperparameter Optimization via RBF and Dynamic coordinate search (HORD) [20]. The main contribution of our paper is summarized in four folds: (1)

1) A multi-level convolutional neural network is adopted for lung nodule malignancy classification, whose hyperparameter optimization is formulated as a computationally expensive optimization problem.
2) A surrogate-assisted evolutionary strategy is introduced to solve hyperparameter optimization for ML-CNN, which utilizes hyperparameter importance-based mutation as the sampling method for efficient candidate points generation.
3) A non-stationary kernel is proposed to define the relationship between different hyperparameter configurations, which allows the model to adapt spatial dependent structure to vary with location. Unlike the commonly used GP model, which has invariant smoothness throughout the whole sampling region, our non-stationary GP regression model is able to satisfy the assumption that the correlation function is no longer dependent on distance only and is dependent on their relative locations to the optimal point. An input-warping method is also adopted which makes covariance functions more sensitive near the hyperparameter optimums.
4) Extensive experimental results illustrate the superiority of the proposed HOSUES for the hyperparameter optimization of deep neural networks.

We organize this paper as follows: Section II introduces the background to lung nodule classification, hyperparameter optimization in deep neural networks, and surrogate-assisted evolutionary algorithm. Section III describes the proposed non-stationary covariance function for hyperparameter optimization in deep neural network, and also the framework and details of HOUSES for ML-CNN. The experiment design is described in Section IV, and we detail the experiment results and discuss the state-of-the-art hyperparameter optimization approaches in Section V. We conclude and propose future work in Section VI.

All the mathematical symbols used in this paper are listed in Table I.

TABLE I
SYMBOLS AND THEIR MEANINGS.

Symbol	Meaning
Z_{train} and Z_{val}	Training and validation datasets
w	Learning weights for DNN
f^* and $\hat{f}$	The true and approximated fitness value
$\xi(x)$	Error function of true and approximated fitness values
Tr	Queried points with true fitness values
k	Kernel function
$\boldsymbol{N}(\mu, \sigma)$	Gaussian distribution with mean μ and covariance σ
θ_c	Noise parameter
K	Covariance matrix
$B(\alpha_d, \beta_d)$	Beta function with parameters α_d, and β_d
$\alpha_{\mathbf{PI}}$	Acquisition function of Probability of Improvement
$\alpha_{\mathbf{EI}}$	Acquisition function of Expected Improvement
$\alpha_{\mathbf{UCB}}$	Acquisition function of Upper Confidence Bound
$\tilde{f}(\theta_s)$	The component function for hyperparameter θ_s
$\mathbb{V}_s$	Variance of response performance for θ_s
$\mathbb{I}_s$	Importance of hyperparameter θ_s

II. PRELIMINARIES

A. Lung Nodule Classification with deep neural network

Deep neural networks have shown their superiority in relation to conventional algorithms in computer vision, and many researchers have employed DNNs in medical imaging diagnosis areas. The work in [50] presents several deep learning algorithms in lung cancer diagnosis, including the stacked denoising autoencoder, the deep belief network, and the convolutional neural network, which obtain a binary classification accuracy of 79.76%, 81.19%, and 79.29%, respectively. Most state-of-the-art works on DNN-based lung nodule classification are inspired by the fact that, when pathologists examine an image to determine whether an image is cancerous or not, they often zoom in and out with the microscope to look at the details as well as the context. In order to understand the details as well as the context, these works design neural network models to take in medical images with different scales. Shen et al. [43] proposed Multi-scale Convolutional Neural Networks (MCNNs) which utilizes multi-scale nodule patches to sufficiently quantify nodule characteristics, which obtained a binary classification accuracy of 86.84%. In MCNN, three CNNs that took different nodules as inputs were assembled in parallel and concatenated the output of each fully connected layer as its resulting output. The experiment results show that multi-scale inputs help CNN learn a set of discriminative features. In 2017, they extended their research and proposed a multi-crop CNN (MC-CNN) [44], which automatically extracted nodule features by adopting a multi-crop pooling strategy, and obtained 87.14% binary classification and 62.46% ternary classification accuracy. Similar to MC-CNN, the work in [63] proposed a multi-resolution convolutional neural network (MRC) to extract the features of lung nodules with different resolutions.

MRC merged feature maps with different resolutions after pooling layers as the final feature maps to train the classifier. Liu and Kang [31] proposed a multi-view convolutional neural network (MV-CNN) for both two-class and three-class lung nodule classification. MV-CNN utilizes multiple views as input channels for CNN based lung nodule classification which obtain a 94.59% and 86.09% accuracy for binary and ternary classification, respectively. In our previous work [34], a ML-CNN was proposed to extract multi-scale features through different convolutional kernel sizes. It also designed three CNNs with the same structure but different convolutional kernel sizes to extract multi-scale features with variable nodule sizes and morphologies. Our ML-CNN achieves state-of-the-art accuracy both in binary and ternary classification (92.21% and 84.81%, respectively) without any additional hand-craft preprocessing. Even though these deep learning methods were end-to-end machine learning architectures and had shown their superiority over conventional methods, the structure design and hyperparameter configuration were based on expert experience through a trial and error search guided by expert intuition, which was a difficult and time-consuming task [35], [11].

B. Hyperparameter optimization in DNN

Setting correct hyperparameters is often critical for reaching the full potential of the chosen or designed deep neural network; otherwise, it may severely hamper the performance of the deep neural networks. Hyperparameter optimization is an important branch of AutoML [57]. Hyperparameter optimization in DNN is a global optimization to find a D-dimensional hyperparameter setting x that minimizes the validation error f of a DNN with learned weights w. The optimal x could be obtained through optimizing f as follows:

$$\begin{aligned} &\min_{x \subseteq \mathbb{R}^D} \quad f(x, w^*; Z_{val}) \\ &s.t. \quad w = \arg\min_{w} \; f(x, w; Z_{train}), \end{aligned} \tag{1}$$

where Z_{train} and Z_{val} are the training and validation datasets respectively. Solving Eq.(1) is very challenging for the high complexity of function f and it is usually accomplished manually in the deep learning community, which largely depends on the expert's experience or intuition. It is also difficult to reproduce similar results when this configuration is applied to different datasets or problems.

There are several systematic approaches to tune hyperparameters in the machine learning community, such as grid search, random search, and Bayesian optimization methods. Grid search is the most common strategy in hyperparameter optimization [27], and it is simple to be implemented with parallelization, which makes it reliable in low dimensional spaces (e.g., 1-d, 2-d). However, grid search suffers from the curse of dimensionality because the search space grows exponentially with the number of hyperparameters. Random search [4] proposes to randomly sample points from the hyperparameter configuration space. Although this approach looks simple, it can find a comparable hyperparameter configuration to grid search with less computation time. Hyperparameter optimization in deep neural networks is a computationally expensive problem as evaluating a hyperparameter choice may cost several hours or even days. This property also makes it unrealistic to sample enough points to be evaluated in grid and random search. One popular approach uses efficient surrogates to approximate the computationally expensive fitness functions to guide the optimization process. Bayesian optimization [46] builds a probabilistic Gaussian model surrogate to estimate the distribution of computationally expensive validation errors. A hyperparameter configuration space is usually modeled smoothly, which means that knowing the quality of certain points might help infer the quality of their nearby points, and Bayesian optimization [3], [41], [5] utilizes the above smoothness assumption to assist the search of hyperparameters. The Gaussian process with a stationary kernel function, e.g., the squared exponential covariance function, is one of the most commonly used methods in Bayesian optimization due to its practicality, simplicity and efficiency. However, the smoothness of the covariance function often varies over the input space in many real-world applications, and the stationarity does not hold when mapping the original input space to a new space with the covariance functions [37], [16]. Therefore, the GP with stationary kernel functions could hardly handle these problems. How to incorporate the non-stationarity into the covariance functions poses a challenge to the GP based hyperparameter optimization in DNN.

C. Gaussian Process based Bayesian Optimization

The Gaussian process[38], [53] uses a generalization of the Gaussian distribution to describe a function, defined by mean μ, and covariance function σ:

$$\hat{f}(x) \sim \boldsymbol{N}(\mu(x), \sigma(x)). \tag{2}$$

Given training data that consists of n D-dimensional inputs and outputs, $\{x_{1:n}, f_{1:n}\}$, where $x_i \subseteq \mathbb{R}^D$ and $f_i = f(x_i)$. The predictive distribution based on the Gaussian process at an unknown input, x^*, is calculated by the following:

$$\mu(x^*) = K_*(K + \theta_c^2 I)^{-1} f_{i:n}, \tag{3}$$

$$\sigma(x^*) = K_{**} - K_*((K + \theta_c^2 I)^{-1}) K_*^T, \tag{4}$$

where $K_* = [k(x^*, x_1), ..., k(x^*, x_n)]$ and $K_{**} = k(x^*, x^*)$, θ_c is a noise parameter, K is the associated covariance matrix which is built as:

$$K = \begin{bmatrix} k(x_1, x_1) & \dots & k(x_1, x_n) \\ \vdots & \ddots & \vdots \\ k(x_n, x_1) & \dots & k(x_n, x_n) \end{bmatrix}, \tag{5}$$

where k is a covariance function that defines the relationship between points in the forms of a kernel. A commonly used kernel is the automatic relevance determination (ARD) squared exponential covariance function:

$$k(x_i, x_j) = \theta_f \, \mathbf{exp} \sum_{d=1}^{D} \frac{-(x_i^d - x_j^d)^2}{2\theta_d^2}. \tag{6}$$

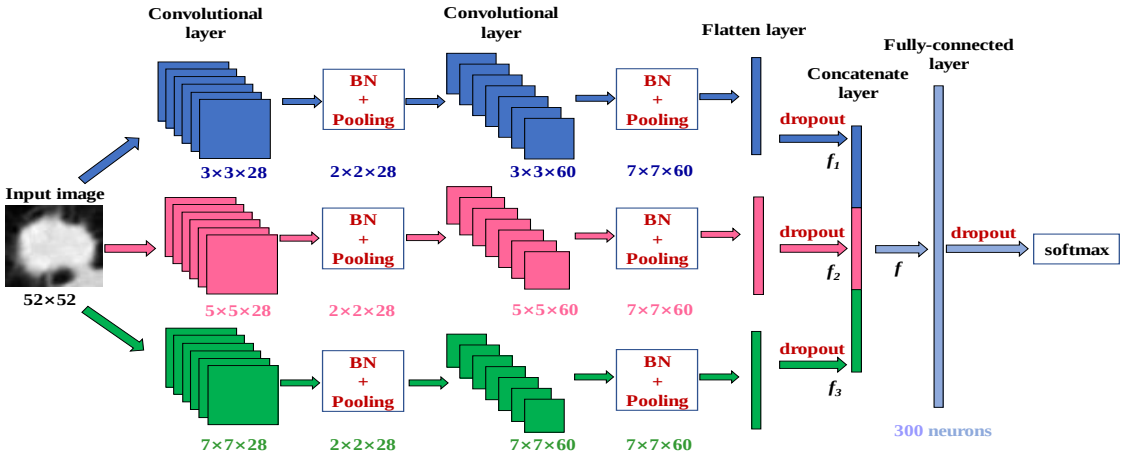


Fig. 1. The structure of the proposed ML-CNN for lung nodule malignancy classification.

After building a surrogate model, Bayesian optimization uses the acquisition function to determine the next querying point in each iteration. There are several acquisition functions to determine the next promising points in GP, including Probability of Improvement (PI), Expected Improvement (EI), Upper Confidence Bound (UCB), and the Predictive Entropy Search (PES) [46], [17]. We applied the three different acquisition functions for Gaussian process (GP)-based hyperparameter optimization:

- Probability of Improvement

$$\alpha_{\mathbf{PI}}(\mathbf{x}) = \Phi(\gamma(\mathbf{x})), \quad \gamma(\mathbf{x}) = \frac{f(\mathbf{x_{best}}) - \mu(\mathbf{x})}{\sigma(\mathbf{x})}. \tag{7}$$

  where $\Phi(z) = (2\pi)^{-\frac{1}{2}} \int_z^{-\infty} \mathbf{exp}(\frac{-t^2}{2})dt$.
- Expected Improvement

$$\alpha_{\mathbf{EI}}(\mathbf{x}) = \sigma(\mathbf{x})(\gamma(\mathbf{x})\Phi(\gamma(\mathbf{x})) + \boldsymbol{N}(\mu(x))), \tag{8}$$

  where $\boldsymbol{N}(z)$ is the variable z which has a Gaussian distribution with $z \sim \boldsymbol{N}(0, 1)$.
- and Upper Confidence Bound

$$\alpha_{\mathbf{UCB}}(\mathbf{x}) = \mu(\mathbf{x}) + r \cdot \sigma(\mathbf{x}) \tag{9}$$

  with a tunable r to balance the exploitation against exploration [20].

D. Surrogate-assisted evolutionary algorithm

Evolutionary algorithms are generic population-based meta-heuristic optimization algorithm for many tasks [23], [12], [55]. A surrogate-assisted evolutionary algorithm is designed to solve expensive optimization problems whose fitness function is highly computationally expensive [22], [23], [12], [6], [54], [30], [29], [51], [24]. It usually utilizes computationally efficient models, also called surrogates, to approximate the fitness function. The surrogate model is built as follows:

$$\hat{f}(x) = f^*(x) + \xi(x), \tag{10}$$

where f^* is the true fitness value, $\hat{f}$ is the approximated fitness value, and ξ is the error function to be minimized by the built surrogate. The surrogate-assisted evolutionary algorithm uses one or several surrogate models $\hat{f}$ to approximate true fitness value f^* and uses the computationally cheap surrogate to guide the search process [61]. The iteration of the surrogate-assisted evolutionary algorithm is described as: 1) learn surrogate model $\hat{f}$ based on previously truly evaluated points $(x, f(x))$; 2) utilize $\hat{f}$ to evaluate new mutation-generated points and find the most promising individual x^*; 3) evaluate the true fitness value of additional points$(x^*, f(x^*))$; 4) update training set.

In this paper, we focus on the Gaussian process to build the surrogate. As described in Sec. II-B, the common stationary kernels are hard to capture the non-stationarity in the hyperparameter optimization of DNN. On the other hand, the non-stationary kernels constructed by non-stationary extensions of stationary kernels with input-dependent length-scales [37], [16] or input warping [47], are good options to bring the non-stationary properties to the input space transitions for modeling. This paper devises a non-stationary kernel function-based Gaussian process as the surrogate model, which allows the model to adapt a spatial dependent structure with varying locations to satisfy our assumption that the hyperparameter configuration performs well near the optimal points but poorly away from the optimal point. Then the evolutionary strategy is used to search the near-optimal hyperparameter configuration. The next section presents the details of HOUSES for ML-CNN.

III. HYPERPARAMETER OPTIMIZATION WITH SURROGATE-ASSISTED EVOLUTIONARY STRATEGY

In our previous work [34], a ML-CNN is proposed for lung nodule classification, which applies different kernel sizes in three parallel levels of CNNs to effectively extract different features of each lung nodule with different sizes and various morphologies. Fig. 1 presents the structure of ML-CNN, which

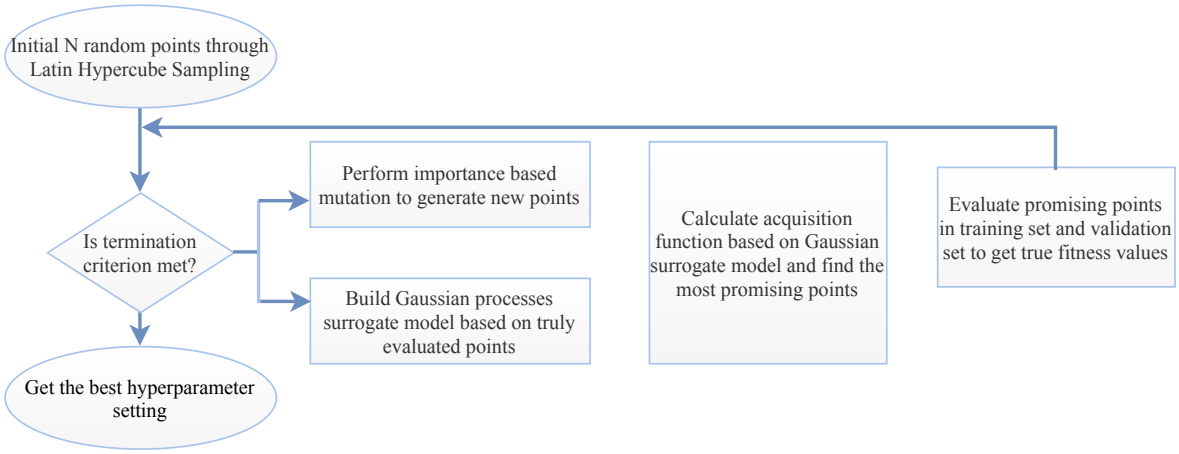


Fig. 2. General flowchart of the proposed **H**yperparameter **O**ptimization with s**U**rrogate a**S**sisted **E**volutionary **S**trategy (**HOUSES**)

contains three levels of CNNs, and each having the same structure and different kernel sizes. As suggested in our previous work, the number of feature maps in each convolutional layer has a significant impact on the performance of ML-CNN, so as do the dropout rates. The hyperparameter configuration of ML-CNN in [34] is based on trial and error search, which is time-consuming for researchers. In this section, we introduce HOUSES to our ML-CNN for lung nodule classification, which is able to automatically find a competitive or even better hyperparameter configuration than manual search method without too much computational cost. The framework of the proposed HOUSES for ML-CNN is presented in **Algorithm 1**. In our hyperparameter optimization method, a non-stationary kernel is proposed as a covariance function to define the relationship between different hyperparameter configurations, which allows the model to adapt a spatial dependent structure which varies with a function of location, and the algorithm searches for the most promising hyperparameter values based on the surrogate model through the evolutionary strategy. In the proposed HOUSES, several initial hyperparameter configuration points are randomly generated using Latin Hypercube Sampling (LHS) [21] methods to keep the diversity of the initial population. These initial points are truly evaluated and used as the training set $Tr_0\{(\mathbf{x}_i, f_i)\}_{i=1}^{n_0}$ to build the initial surrogate model. Then the evolutionary strategy generates a group of new points that are evaluated according to the acquisition function of the surrogate model. Several most promising individuals x^* are found from those newly generated points based on the acquisition function and are then truly evaluated. The most promising points with true fitness value$(x^*, f(x^*))$ are added to the training set to update the surrogate model. Fig. 2 gives the general flowchart of HOUSES, which we describe it in detail in the following paragraphs.

Algorithm 1 General Framework of HOUSES

Input:Initial population size n_0, Maximum generation g_{max}, Mutation rate p_m, number of new generated points every generation m, $Dataset$, DNN model.
Output: best hyperparameter configuration c_{best}.

1. Divide dataset into Training, Validation and Testing sets
2. Initialization a hyperparameter configuration population pop_0 is randomly generated through Latin Hypercube Sampling. These hyperparameter points are used to train DNN model in Training set, and truly evaluated in the Validation set to get true fitness values $Tr_0 = \{(\mathbf{x}_i, f_i)\}_{i=1}^{n_0}$.
for $i = 1, 2, ..., g_{max}$ **do**
  1. Use Tr to fit or update the Gaussian surrogate model $\hat{f}$ according Eq.(2);
  2. $\boldsymbol{pop_{selected}}$**= select** $(\boldsymbol{pop_g})$// *select individuals with good performance and diversity for mutation*;
  3. pop_m**= mutation** $(\boldsymbol{pop_{selected}})$// *apply mutation operation to selected points to generate* m *new points*;
  4. Calculate $\{(\mathbf{x}_i, \hat{f}_i)\}_{i=1}^{m}$ for m new generated points based on Gaussian surrogate model and acquisition functions Eq.(14)(15)(16);
  Set $\mathbf{x}^* =$ **argmin**$\{\hat{f}_i\}_{i=1}^{m}$;
  5. Truly evaluate $f(\mathbf{x}^*)$ in Training set and Validation set to get true fitness values;
  6. Update $Tr_{g+1} = \{Tr_{g+1} \cup (\mathbf{x}^*, f(\mathbf{x}^*))\}$;
end for

A. Non-stationary Covariance Function for Hyperparameter Optimization in DNNs

1) Spatial location transformation: In the hyperparameter optimization of DNNs, two far away hyperparameter points usually perform poorly when they are a distance away from the optimal points. This property means that the correlation of two hyperparameter configurations depends not only on the distance between them, but also the points' spatial locations. Those stationary kernels, such as the Gaussian kernel, clearly do not satisfy this property of hyperparameter optimization in DNNs. To account for this non-stationarity, we propose a non-stationary covariance function, where the relative distance to the optimal point is used to measure the spatial location difference of two hyperparameter points. The relative distance

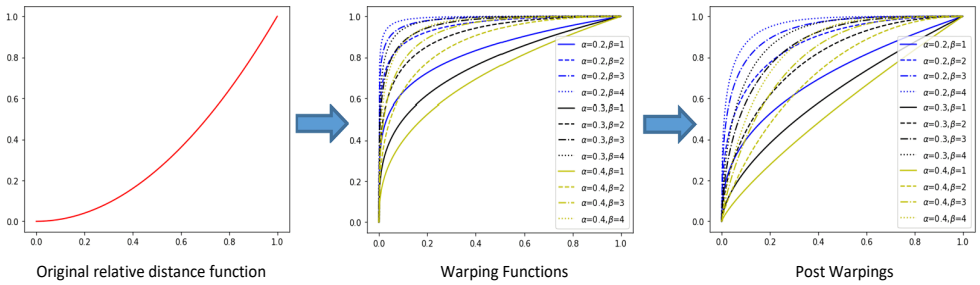


Fig. 3. Example of how Kumaraswamy cumulative distribution function transforming a concave function into a convex function, which makes the kernel function is much more sensitive to small inputs.

based kernel is defined as:

$$k(x_i, x_j) = \theta_f \mathbf{exp} \sum_{d=1}^{D} \frac{-(\left|x_i^d - \boldsymbol{s}^d\right| - \left|x_j^d - \boldsymbol{s}^d\right|)^2}{2\theta_d^2}, \quad (11)$$

where $\boldsymbol{s}$ is the assumed optimal point. It is also easy to prove this relative distance based covariance function $k(x_i, x_j)$ is a kernel based on **Theorem** 1. Eq.(11) can be obtained by set $\psi(\mathbf{x}) = |\mathbf{x} - \mathbf{s}|$ and k' as the Gaussian kernel. This relative distance based kernel is no longer a function of distance between two points but depends on their own spatial locations to the optimal point.

Theorem 1: if ψ is an $\mathbb{R}^D$-valued function on $\boldsymbol{X}$ and k' is a kernel on $\mathbb{R}^D \times \mathbb{R}^D$, then

$$k(\mathbf{x}, \mathbf{z}) = k'(\psi(\mathbf{x}), \psi(\mathbf{z})) \quad (12)$$

is also a kernel.

Proof: $k' : \mathbb{R}^D \times \mathbb{R}^D \to \mathbb{R}$, $\psi : \mathbb{R}^D \to \mathbb{R}^D$, k' is a valid kernel, then we have

$$k'(\mathbf{x}, \mathbf{z}) = \varphi(\mathbf{x})^{\mathbf{T}} \varphi(\mathbf{z})$$

so that

$$k(\mathbf{x}, \mathbf{z}) = \varphi(\psi(\mathbf{x}))^{\mathbf{T}} \varphi(\psi(\mathbf{z}))$$

is a kernel.

2) Input Warping: In the hyperparameter optimization of machine learning models, objective functions are usually more sensitive near the optimal hyperparameter settings but they are much less sensitive when they are far away from the optimum. For example, if the optimal learning rate is 0.05, it is supposed to obtain a 50% performance increase when the learning rate changes from 0.04 to 0.05, whereas there is only a 5% increase from 0.25 to 0.24. Traditionally, most researchers often use the logarithm function to transform the input space and then search in the transformed space, which is effective only when the input space's non-stationary property is known in advance. Recently, a beta cumulative distribution function was proposed as the input warping transformation function [47], [52],

$$w_d(\mathbf{x}_d) = \int_0^{\mathbf{x}_d} \frac{u^{\alpha_d - 1}(1-u)^{\beta_d - 1}}{B(\alpha_d, \beta_d)} \mathbf{d}u, \quad (13)$$

where $B(\alpha_d, \beta_d)$ is the beta function, which adjusts the shape of the input warping function to the original data based on parameters α_d, and β_d.

Different from [47], [52], we take the relative distance to local optimum as inputs to be warped to make the kernel function more sensitive to small inputs and less sensitive to large ones. We take the Kumaraswamy cumulative distribution function as the substitute, not only for computational reasons, but also because it is easier to fulfill the non-stationary property of our kernel function after spatial location transformation,

$$w_d(\mathbf{x}_d) = 1 - (1 - \mathbf{x}_d^{\alpha_d})^{\beta_d}. \quad (14)$$

Similar to Eq.(12), it is easy to prove that $k(\mathbf{x}, \mathbf{x}') = k'(w(\psi(\mathbf{x})), w(\psi(\mathbf{x}')))$ is a kernel. Fig.3 illustrates input warping example with different shaped parameters α_d, and β_d input warping functions. The final kernel for HOUSES is defined as:

$$\begin{aligned} k(x_i, x_j) = \theta_f \ \mathbf{exp} \sum_{d=1}^{D} \frac{-(w_d \left|x_i^d - \boldsymbol{s}^d\right| - w_d \left|x_j^d - \boldsymbol{s}^d\right|)^2}{2\theta_d^2} \\ + \theta_k \ \mathbf{exp} \sum_{d=1}^{D} \frac{-(w_d \left|x_i^d - x_j^d\right|)^2}{2\gamma_d^2} \end{aligned}. \quad (15)$$

Eq.(15) is also proven to be a kernel based on **Theorem** 2. This non-stationary kernel satisfies the assumption that the correlation function of two hyperparameter configuration depends on their distances and their relative locations to the optimal point. However, it is impossible to determine the optimal point in advance, so we use the hyperparameter configuration with the best performance in the training set and update it in every iteration in the proposed HOUSES.

Theorem 2: If k_1 is a kernel on $\mathbb{R}^D \times \mathbb{R}^D$ and k_2 is also a kernel on $\mathbb{R}^D \times \mathbb{R}^D$, then

$$k(\mathbf{x}, \mathbf{z}) = k_1(\mathbf{x}, \mathbf{z}) + k_2(\mathbf{x}, \mathbf{z}) \quad (16)$$

is also a kernel.

Proof: This is because if $k_1(\mathbf{x}, \mathbf{z})$ and $k_2(\mathbf{x}, \mathbf{z})$ are valid kernels on $\mathbb{R}^D \times \mathbb{R}^D \to \mathbb{R}$, then we have $k_1(\mathbf{x}, \mathbf{z}) = \varphi^{\mathbf{T}}(\mathbf{x})\varphi(\mathbf{z})$ and $k_2(\mathbf{x}, \mathbf{z}) = \psi^{\mathbf{T}}(\mathbf{x})\psi(\mathbf{z})$, we may define

$$\theta(\mathbf{x}) = \varphi(\mathbf{x}) \oplus \psi(\mathbf{x}) = [\varphi(\mathbf{x}), \psi(\mathbf{x})]^{\mathbf{T}}$$

so that

$$k(\mathbf{x}, \mathbf{z}) = \theta(\mathbf{x})^{\mathbf{T}} \theta(\mathbf{z})$$

is a kernel.

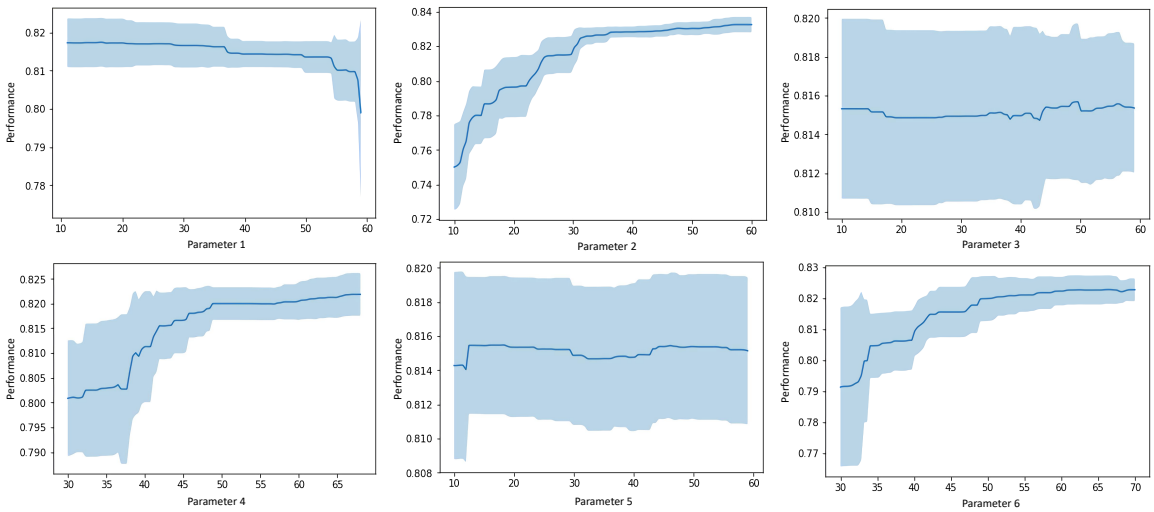


Fig. 4. Marginal response performance of the number of feature maps of all convolutional layers in three different levels of ML-CNN. The first two parameters are for the two convolutional layers in the first level, the middle two are for the second level, and the last two are for the third level. Results show that the latter ones in the three-level bring more effects to the performance, while there is no significant difference among all possible configurations for the previous feature maps number in each level of ML-CNN. We plot marginal response performance with standard deviations (mean $\pm$ standard deviations, also 68% confidence level) calculated by fANOVA.

B. Hyperparameter Importance based Mutation for Candidate Hyperparameter Points Generation

The mutation aims to generate better individuals by mutating selected excellent individuals, which is a crucial step in the optimization in evolutionary strategy [9]. To maintain the diversity of the population, a grid strategy-based mutation is adopted. We first divide every dimension into M uniformed grids [59] and the point with the highest fitness in every dimensional grid is selected for mutation. In this way, $D * M$ individuals are selected and the polynomial mutation is applied to every selected individual to generate n_d candidate hyperparameter points, respectively. These $D * M * n_d$ points are evaluated based on acquisition function, and the most promising point is selected for true evaluation and added into the training set to update the surrogate model.

However, as suggested by several recent works on Bayesian based hyperparameter optimization [18], [4], most hyperparameters are truly unimportant while some hyperparameters are much more important than others. Fig. 4 demonstrates ML-CNN's marginal performance variation with the number of feature maps, which clearly shows that the number of feature maps in the last convolutional layer in every layer is much more crucial to ML-CNN than previous ones. The work in [18] proposes the use functional analysis of variance (fANOVA[1]) to measure the importance of the hyperparameters in machine learning problems. fANOVA is a statistical method for prominent data analysis, which partitions the observed variation of a response value (CNN performance) into components due to each of its inputs (hyperparameter setting). fANOVA illustrates how response performance changes with input hyperparameters. It first accumulates the response function values of all subsets of its inputs N:

$$\check{y}(\theta) = \sum_{U \subseteq N} \check{f}_U(\theta_U), \tag{17}$$

where the component $\check{f}_U(\theta_U)$ is defined as:

$$\check{f}_U(\theta_U) = \begin{cases} \check{f}_\emptyset & \textbf{if } U = \emptyset \\ \check{a}_U(\theta_U) - \sum \check{f}_W(\theta_W) & \textbf{otherwise} \end{cases}, \tag{18}$$

where the constant $\check{f}_\emptyset$ is the mean value of the function over its domain, $\check{a}_U(\theta_U)$ is the marginal predicted performance defined as $\check{a}_U(\theta_U) = \frac{1}{\|\Theta_T\|} \int \check{y}(\theta_{N|U}) d\theta_T$. The subset $|U| > 1$ captures the interaction between all the hyperparameters in subset U, while we only consider the separate hyperparameter importance in this paper and set $|U| = 1$. The component function $\check{f}_U(\theta_U)$ is then calculated as:

$$\check{f}(\theta_s) = \check{a}(\theta_s) = \frac{1}{\|\Theta_T\|} \int \check{y}(\theta_{N|s}) \mathbf{d}\theta_T, \tag{19}$$

where θ_s is the single hyperparameter, $T = N \backslash s$, $\Theta_T = \Theta \backslash \theta_s$, $\Theta = \theta_1 \times \cdots \times \theta_D$. The variance of the response performance of $\check{y}$ across its domain Θ is

$$\mathbb{V} = \sum_{i=1}^{n} \mathbb{V}_s, \ \mathbb{V}_s = \frac{1}{\|\theta_s\|} \int \check{f}(\theta_i)^2 \mathbf{d}\theta_s. \tag{20}$$

The importance of each hyperparameter could thus be quantified as:

$$\mathbb{I}_s = \mathbb{V}_s / \mathbb{V}. \tag{21}$$

When the polynomial mutation operator is applied to individuals, genes corresponding to different hyperparameters have different mutation probabilities in terms of hyperparameter importance, where genes with greater importance are supposed to have higher mutation probabilities of generating more offspring. In this way, our evolutionary strategy emphasizes those subspaces of important hyperparameters and finds better hyperparameter settings.

[1]https://github.com/automl/fanova.

IV. Experimental Design

A. Synthetic Function and DNN Problems

To examine the optimization performance of the proposed HOUSES for hyperparameter optimization, three sets of experiments are conducted.

1) First Experiment Set: There are several local optimal points in the hyperparameter optimization of DNNs, and the correlation between two hyperparameter configuration usually depends on the distance and the spatial locations. In this case, we consider the trimodal and Branin functions [60] to simulate the hyperparameter optimization of DNNs, which both contain several local optimums with intrinsic dimension $d_e = 2$. We first conduct comparison experiments with baselines on the trimodal and Branin function [60], [32]. The trimodal function is defined as:

$$\begin{aligned} f(\mathbf{x}) = g(\mathbf{u}) = \log(0.1 \times \text{mvnpdf}(\mathbf{u}, \mathbf{c}_1, \sigma^2) + \\ 0.8 \times \text{mvnpdf}(\mathbf{u}, \mathbf{c}_2, \sigma^2) + 0.1 \times \text{mvnpdf}(\mathbf{u}, \mathbf{c}_3, \sigma^2)), \end{aligned} \tag{22}$$

where $\sigma^2 = 0.01 d_e^{0.1}$ and $\text{mvnpdf}(\mathbf{x}, \mu, \sigma^2)$ are multivariate Gaussian distributions and c_i are the fixed centers in $\mathbb{R}^{d_e}$. The global maximum is $f(\mathbf{x}^*) = g(\mathbf{u}^*) = g(\mathbf{c}_2) = 2.4748$ at center $\mathbf{c}_2$ with the highest probability of 0.8. The Branin function is defined as:

$$f(\mathbf{x}) = g(\mathbf{u}) = a(x_j - bx_i^2 + cx_i - r) + s(1-t)cos(x_i) + s \tag{23}$$

where i, j are two randomly selected dimensions, $a = 1$, $b = \frac{5.1}{4\pi^2}$, $c = \frac{5}{\pi}$, $r - 6$, $s = 10$, $t = \frac{1}{8\pi}$, and the global minimum is $g(\mathbf{u}^*) = 0.397887$ at $\mathbf{u}^* = (-\pi, 12, 275), (\pi, 2.275)$ and $(9.42478, 2.475)$.

2) Second Experiment Set: There are three DNN problems in the second experiment set, the first being the MLP network applied to MNIST, which consists of three dense layers with ReLU activation and a dropout layer between them and SoftMax at the end. This problem has five parameters to be optimized, including the number of units in three dense layers and two dropout rates in the dropout layer. This paper describes this problem as 5-MLP. The second DNN problem is LeNet5 applied to MNIST with seven hyperparameters to be optimized, described as 7-CNN. The 7-CNN contains two convolutional blocks, each containing one convolutional layer with batch normalization, followed by $ReLU$ activation and 2×2 max-pooling, and three fully connected layers with two dropout layers among them are followed at the end. The optimizing parameters in 7-CNN contain the number of feature maps in the two convolutional layers and units in the first two fully connected layers, and also the dropout rates in two dropout layers. The third DNN problem is to optimize the hyperparameters of AlexNet applied to the CIFAR-10 dataset. There are nine parameters: feature numbers in five convolutional layers, numbers of units in two fully-connected layers, and the dropout rate of the dropout layer after them. This is described as the 9-CNN problem in this paper.

3) Third Experiment Set: For the third set, the purpose is to find an optimal hyperparameter configuration of ML-CNN applied to lung nodule classification. We evaluate HOUSES on ML-CNN applied to lung nodule classification. There are nine hyperparameters to be optimized, consisting of the number of feature maps of two convolutional layers for the three levels, the number of units in the full connected layer, and the dropout rate of two dropout layers. This hyperparameter optimization problem is denoted as 9-ML-CNN in this paper. The lung nodule images in this experiment are from the Lung Image Database Consortium (LIDC) and Image Database Resource Initiative (IDRI) database [2], [39], containing 1,018 cases from 1,010 patients and are annotated by 4 radiologists. The malignancy suspiciousness of each nodule in the database is rated from 1 to 5 by four radiologists, where scores 1 and 2 are benign nodules, score 3 is an indeterminate nodule, and scores 4 and 5 are malignant nodules. The diagnosis of nodules is labeled to the class with the highest frequency, or is indeterminate when more than one class has the highest frequency. The nodules are cropped according to the contour annotations of four radiologists and resized by 52×52 as the input of our multi-level convolutional neural networks.

All the experiments were performed using Nvidia Quadro P5000 GPU (16.0 GB Memory, 8873 GFLOPS). Our experiments are implemented in the Python 3.6 environment, and Tensorflow [2] and Tensorlayer [3] are used to build deep neural networks. The following subsections present a brief introduction of peer algorithms, evaluation budgets, and the experiment settings.

B. Peer Algorithm

We compare HOUSES against random search, grid search, Tree-structured Parzen Estimator (TPE), Gaussian processes (GP) with Gaussian kernel, and Hyperparameter Optimization via RBF and Dynamic coordinate search (HORD). We also compare three different acquisition functions for Gaussian processes (GP) based hyperparameter optimization: HOUSES with Expected Improvement (HOUSE-EI), HOUSES with Probability of Improvement (HOUSE-PI), and HOUSES with Upper Confidence Bound (HOUSE-UCB).

C. Evaluation Budget and Experimental Setting

Hyperparameter configuration evaluation is typically computationally expensive, being the highest computation cost in the DNN hyperparameter optimization problem. For a fair comparison, we set the number of function evaluations as 200 for all the compared algorithms. The number of training iterations for the MNIST dataset is set as 100, and CIFAR-10 and LIDC-IDRI are set as 200 and 500, respectively.

We implement the random search and TPE with the open-source HyperOpt library [4]. We use the public sklearn library [5] to build the Gaussian processes-based surrogate model. Details of the implementation of HORD are available at [6]. The code for hyperparameter importance assessment based on functional ANOVA is available at [7]. The code for HOUSES is also available at [8]. We run each algorithm for 10 independent runs.

[2]https://github.com/tensorflow/tensorflow.
[3]https://github.com/tensorflow/tensorflow.
[4]http://hyperopt.github.io/hyperopt/
[5]https://scikit-learn.org/stable/modules/gaussian_process.html
[6]bit.ly/hord-aaai
[7]https://github.com/automl/fanova
[8]https://github.com/MiaoZhang0525/code-for-houses

TABLE II
EMPIRICAL COMPARISONS ON SYNTHETIC FUNCTIONS WITH BASELINES.

DNN Problems	Trimodel		Branin	
	d=5	d=10	d=5	d=10
Random Search	0.328 ± 0.024	0.300 ± 0.025	0.301 ± 0.305	0.374 ± 0.275
Grid Search	0.249 ± 0.132	0.478 ± 0.213	0.254 ± 0.112	0.342 ± 0.189
TPE	$\mathbf{0.014 \pm 0.013}$	0.131 ± 0.113	$\mathbf{0.032 \pm 0.029}$	0.135 ± 0.076
GP-EI	0.154 ± 0.056	0.034 ± 0.036	0.150 ± 0.234	0.271 ± 0.161
GP-PI	0.078 ± 0.004	0.434 ± 0.062	0.209 ± 0.101	0.374 ± 0.256
GP-UCB	0.123 ± 0.044	0.534 ± 0.162	0.234 ± 0.188	0.361 ± 0.180
HOUSES-EI-G	0.024 ± 0.008	0.152 ± 0.162	0.074 ± 0.013	0.232 ± 0.083
HOUSES-PI-G	0.068 ± 0.021	0.053 ± 0.051	0.135 ± 0.045	0.302 ± 0.156
HOUSES-UCB-G	0.083 ± 0.172	0.214 ± 0.162	0.087 ± 0.079	0.274 ± 0.189
HOUSES-EI	$\mathbf{0.023 \pm 0.007}$	$\mathbf{0.003 \pm 0.002}$	0.077 ± 0.021	0.104 ± 0.007
HOUSES-PI	0.072 ± 0.006	0.183 ± 0.058	0.067 ± 0.045	0.121 ± 0.136
HOUSES-UCB	0.063 ± 0.002	0.267 ± 0.234	$\mathbf{0.057 \pm 0.049}$	$\mathbf{0.097 \pm 0.067}$

V. EXPERIMENT RESULTS AND DISCUSSION

A. Experiments on the First Experiment Set

In this section, the HOUSES is compared with baselines on a numerical case, where the trimodel and Branin functions are used to simulate the hyperparameter optimization in DNNs. We further augment the number of dimensions to $d = 5$ and $d = 10$ with dummy variables since most hyperparameters are unimportant in the hyperparameters optimization of DNNs. Table. II provides the comparison results of the simple regrets of the proposed HOUSES and compares the baselines on trimodal and Branin functions with a different number of dummy variables. We consider three different acquisition functions for the Gaussian processes-based methods. We could find that our HOUSES achieves excellent performance for four different cases, which shows the superiority of our approach. Specifically, TPE achieves the best results for low-dimension $d = 5$ that, 0.014 ± 0.013 for trimodal and 0.032 ± 0.029 for Branin. HOUSES also obtains competitive results in this case that, HOUSES-EI obtains 0.023 ± 0.007 for Trimodal and HOUSES-UCB obtains 0.057 ± 0.049 for Branin. When we augment more dummy variables where $d = 10$, we find that HOUSES performs the best on the two synthetic functions, achieving 0.003 ± 0.002 with HOUSES-EI on trimodal function and 0.097 ± 0.067 with HOUSES-UCB on the Branin function. Compared with other peer algorithms, HOUSES is more robust with the dimension.

To investigate the impacts of the proposed hyperparameter importance based mutation, we compare our proposed method with HOUSES-G, which adopts the baseline mutation (a grid strategy-based mutation). The comparison results are presented in Table. II. We find that HOUSES performs better than HOUSES without our proposed mutation strategy in all scenarios. These results verify that it is essential to identify these important hyperparameters and force the algorithm to search in those dimensions with a limited computational budget, significantly helping the following optimization process.

We conducted an ablation study to investigate the impacts of the proposed non-stationary kernel function. We compare GP and HOUSES-G in Table. II. In this table, the only difference between the compared methods is the kernel function. GP is the baseline Bayesian optimization with the stationary kernel function, and HOUSES-G is the Bayesian optimization with the proposed non-stationary kernel function. We compare GP and HOUSES-G with different acquisition functions and dimensions on the two synthetic functions, with 12 cases. From Table. II, we observe that HOUSES-G outperforms GP in 11 of the 12 cases, demonstrating the effectiveness of the proposed non-stationary kernel function.

B. Experiments on the Second Experiment Set

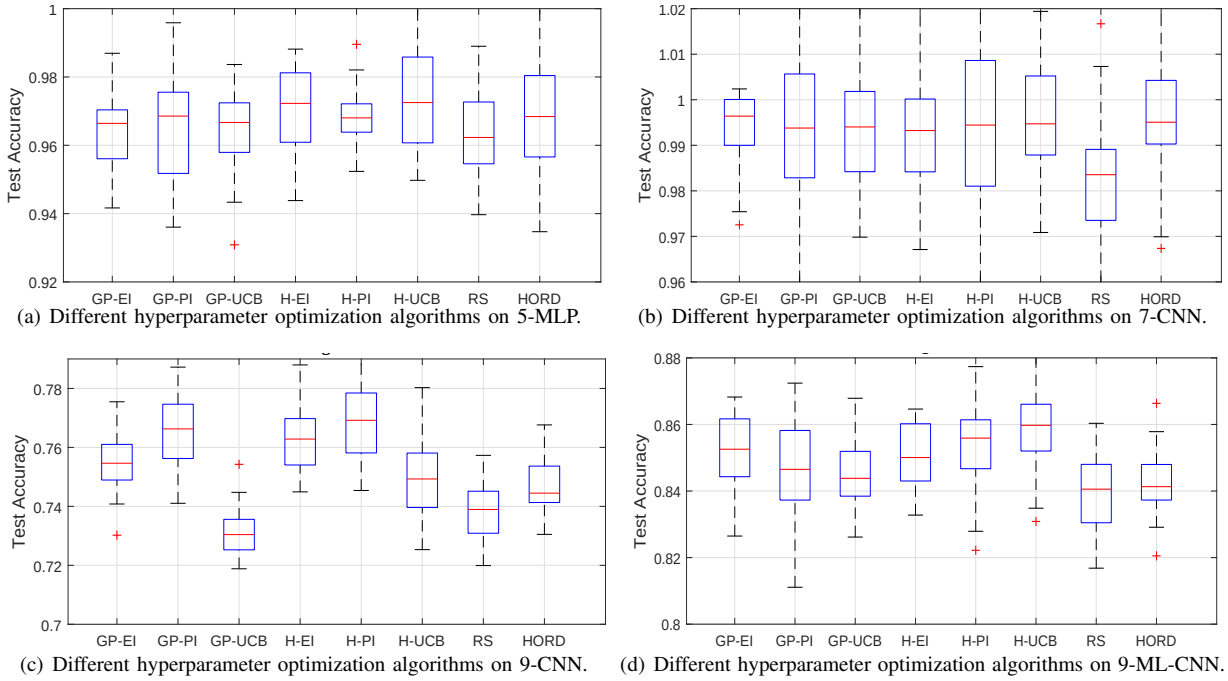
In this section, we evaluate these peer hyperparameter optimization algorithms on 3 DNN problems, including the MLP applied to MNIST (5-MLP), and LeNet network to MNIST (7-CNN), and AlexNet applied to CIFAR10 (9-CNN).

For the 5-MLP problem, Table III (Column 2) and Fig. 5 (a) show the test results obtained by the different methods and Fig. 6 (a) plots the average accuracy over epochs of the obtained hyperparameter configurations from different hyperparameter optimization methods. One surprising observation from Table III and Fig. 5 is that the simplest random search method obtains satisfied results, and sometimes even outperforms some Bayesian optimization based methods (GPs and HORD). This phenomenon suggests that, for low-dimensional hyperparameter optimization, the simple random search algorithm could perform very well, which is also in line with [4]. Furthermore, we also find from Table III (Column 2) and Fig. 5 (a) that, with the same experiment settings, our proposed non-stationary kernel clearly performs better than the stationary Gaussian kernel with all three acquisition functions in the 5-MLP problem. This also demonstrates that incorporating expert intuition-based priors into Bayesian optimization and designing a non-stationary kernel is necessary for Gaussian processes based hyperparameter optimization.

In the 7-CNN problem, we find that most hyperparameter optimization algorithms obtain satisfactory results, and test errors are less than the best result in the 5-MLP problem (see Column 3 of Table III). These results demonstrate that a better neural network structure significantly improves the performance and is more robust to hyperparameter configuration, where there is not much significant difference between these hyperparameter optimization methods in the 7-CNN problem. Designing an appropriate neural network structure

TABLE III
EXPERIMENTAL ACCURACY OF COMPARING ALGORITHMS ON 4 DNN PROBLEMS

DNN Problems	5-MLP	7-CNN	9-CNN	9-ML-CNN
Random Search	0.973 ± 0.015	0.9947 ± 0.005	0.743 ± 0.016	0.840 ± 0.007
HORD	0.968 ± 0.015	0.9929 ± 0.006	0.747 ± 0.017	0.841 ± 0.009
GP-EI	0.964 ± 0.014	0.9934 ± 0.006	0.754 ± 0.015	0.852 ± 0.009
GP-PI	0.964 ± 0.016	0.9937 ± 0.006	0.765 ± 0.015	0.847 ± 0.009
GP-UCB	0.963 ± 0.014	0.9942 ± 0.005	0.732 ± 0.015	0.846 ± 0.009
HOUSES-EI	0.970 ± 0.014	0.9931 ± 0.006	0.764 ± 0.016	0.851 ± 0.010
HOUSES-PI	0.969 ± 0.016	0.9949 ± 0.005	$\mathbf{0.768 \pm 0.014}$	0.854 ± 0.008
HOUSES-UCB	$\mathbf{0.974 \pm 0.017}$	0.9937 ± 0.004	0.749 ± 0.014	$\mathbf{0.855 \pm 0.008}$
Manual Tuning	-	-	-	0.848



(a) Different hyperparameter optimization algorithms on 5-MLP.
(b) Different hyperparameter optimization algorithms on 7-CNN.
(c) Different hyperparameter optimization algorithms on 9-CNN.
(d) Different hyperparameter optimization algorithms on 9-ML-CNN.

Fig. 5. Results of different Hyperparameter optimization algorithms on four DNN problems.

is the priority, which is also why we design a multi-level convolutional neural network for lung nodule classification.

As for the more complicated DNN problem 9-CNN, GPs found significantly better hyperparameters than the random search algorithm, except GP-UCB, which may be due to the improper weighting setting in the UCB acquisition function (see Column 4 of Table III, Fig. 5 (c), Fig. 6 (c)). These results show the random search algorithm performs significantly worse than other hyperparameter optimization algorithms on the 9-CNN problem, and suggest that hyperparameter optimization is required for complicated DNN problems, which helps the deep neural network to reach its full potential. Fig. 5 (c) and Fig. 6 (c) show the performance of HOUSES and GP with different acquisition functions. We can find HOUSES-PI and GP-PI obtain better results than the other two acquisition functions, demonstrating the importance of the acquisition function. Furthermore, similar to the results for the 5-MLP problem, the results for the 9-CNN again show the superiority of our proposed non-stationary kernel for hyperparameter optimization in CNN, where the non-stationary kernel always outperforms the standard Gaussian kernel.

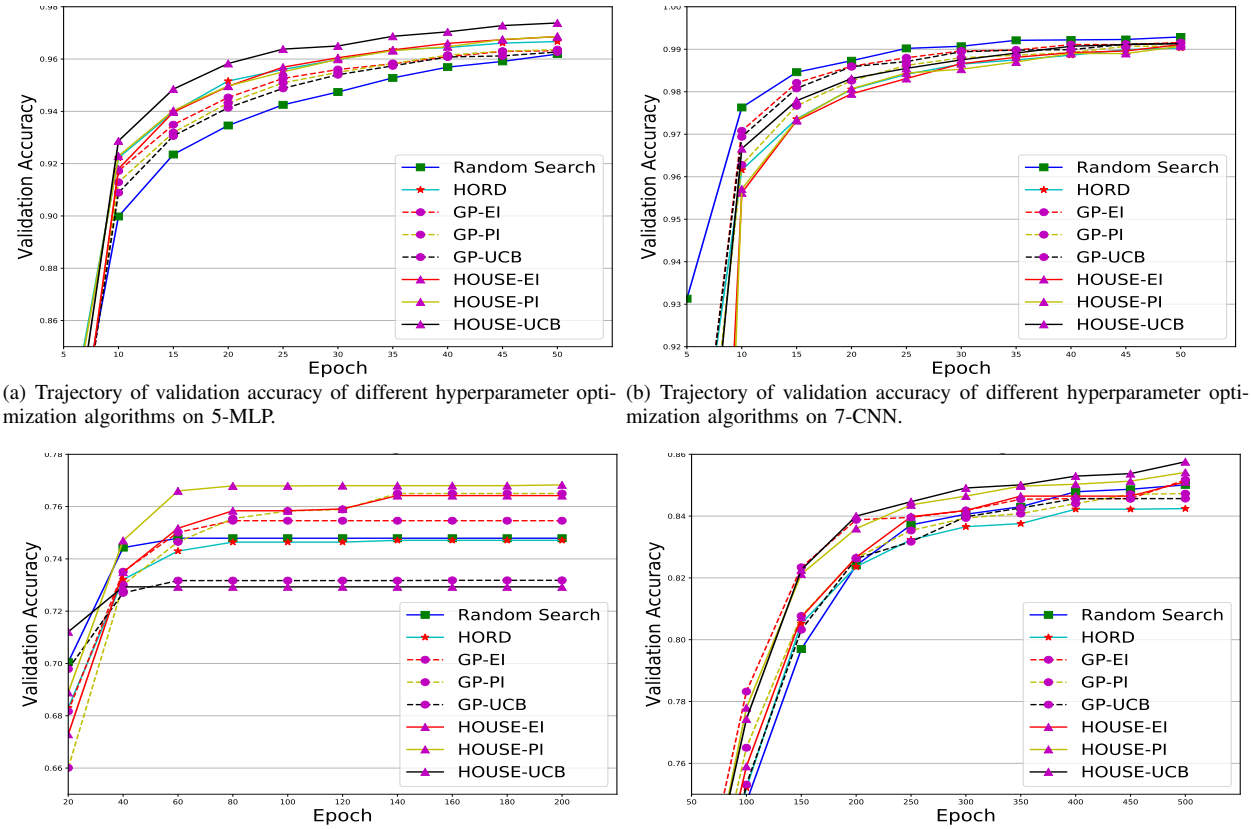
Table IV summarizes the mean sensitivity and specificity (accuracy was presented in Table III) of hyperparameter configuration obtained by all the compared algorithms for 5-MLP, 7-CNN, and 9-CNN. We also calculate the area under curve (AUC) [7] as the assessment criteria for the receiver operating characteristic (ROC) curve. As shown in Table IV, the proposed HOUSES approach outperforms random search, HORD, and normal kernel-based Gaussian processes in terms of the three metrics on the 7-CNN and 9-CNN problems. For the 5-MLP problem, random search achieves remarkable incredible results, which suggests that the simple random search algorithm is able to perform very well in low-dimensional hyperparameter optimization. There is no statistical difference between the compared algorithms in the results of 7-CNN, indicating a better neural network structure could significantly improve the performance and relieve the work associated with hyperparameter optimization works.

C. Experiments on the Third Experimental Set

In this section, we first compare the existing works on CNN based ternary lung nodule classification, and the results are

TABLE IV
SENSITIVITY, SPECIFICITY, AND AUC COMPARISON RESULTS IN THE SECOND EXPERIMENTAL SET.

Algorithm	5-MLP			7-CNN			9-CNN		
	Sensitivity	Specificity	AUC	Sensitivity	Specificity	AUC	Sensitivity	Specificity	AUC
Random Search	0.95054	0.99521	0.97588	0.98809	0.99569	0.99339	0.7590	0.97130	0.8617
HORD	0.95085	0.99458	0.97272	0.98398	0.99832	0.99111	0.76690	0.97410	0.87050
GP-EI	0.95474	0.99502	0.97488	0.95576	0.99840	0.99182	0.76800	0.97420	0.87110
GP-PI	0.93838	0.99324	0.96581	0.98706	0.99857	0.99281	0.7571	0.9730	0.86505
GP-UCB	0.93604	0.99298	0.96451	0.98511	0.99842	0.99207	0.7609	0.97343	0.86717
HOUSES-EI	0.93642	0.99300	0.96472	0.98414	0.99855	0.99519	0.76940	0.97438	0.87200
HOUSES-PI	0.94486	0.99395	0.96940	0.98517	0.99857	0.99377	**0.7798**	**0.97553**	**0.87767**
HOUSES-UCB	**0.96161**	**0.99578**	**0.97870**	0.98578	0.99852	0.99355	0.7609	0.9343	0.86717



(a) Trajectory of validation accuracy of different hyperparameter optimization algorithms on 5-MLP.

(b) Trajectory of validation accuracy of different hyperparameter optimization algorithms on 7-CNN.

(c) Trajectory of validation accuracy of different hyperparameter optimization algorithms on 9-CNN.

(d) Trajectory of validation accuracy of different hyperparameter optimization algorithms on 9-ML-CNN.

Fig. 6. Validation accuracy on four DNN problems over epochs.

shown in Table V. We observe that our work is the only one that handles hyperparameter optimization in CNN-based lung nodule classification through the automatic approach. Our ML-CNN achieves competitive classification accuracy of 84.8%, and with the proposed hyperparameter optimization method HOUSES (taking the HOUSES-UCB as an example), ML-CNN could obtain better results of 85.5%. More interesting, our recently proposed ML-xResNet [33] adds more skip-connections among residual layers in different levels based on ML-CNN, and it also achieves better results than ML-CNN. ML-xResNet is more related to another branch of AutoML, neural architecture search (NAS), and these experimental results demonstrate that hyperparameter optimization and architecture design are two ways to improve the performance of existing CNN based lung nodule classification.

We then evaluate HOUSES and all the compared algorithms applied to 9-ML-CNN (multi-level convolutional neural network applied to lung nodule classification with nine hyperparameters to be optimized), and the results are shown in Table III Column 4, Fig.5(d) and Fig.6(d). As expected, the performance of conventional hyperparameter optimization methods degrades significantly in complicated and high dimensional search space, while HOUSES continues to achieve satisfying results and outperforms the Gaussian process with stationary kernels. Similar to the results discussed in the previous subsection, we find that UCB achieves the best result

TABLE V
COMPARISON RESULTS WITH THE STATE-OF-THE-ART CNN BASED TERNARY LUNG NODULE CLASSIFICATION METHODS.

Method	**Accuracy**	**Sensitivity**	**Specificity**	**HO method**
Ensenble SVM [25]	0.8336	0.8259	0.9117	Manual Tuning
Ensenble RF [25]	0.8489	0.8311	0.9209	Manual Tuning
MV-CNN with BN [31]	0.8129	-	-	Manual Tuning
MC-CNN [44]	0.6246	-	-	Manual Tuning
DenseNet [19]	0.6890	-	-	Manual Tuning
MoDenseNet [8]	0.8545	-	-	Manual Tuning
ML-xResNet [33]	0.8588	0.8456	0.9248	Manual Tuning
ML-CNN [33]	0.848	0.8275	0.9137	Manual Tuning
HOUSES based ML-CNN	0.8550	0.8503	0.9236	HOUSES

TABLE VI
COMPARISON RESULTS OF LUNG NODULE CLASSIFICATION PROBLEM FOR EACH CLASS.

Algorithm	Benign			Indeterminate			Malignant		
	Sensitivity	Specificity	AUC	Sensitivity	Specificity	AUC	Sensitivity	Specificity	AUC
Random Search	0.79260	0.92026	0.86643	0.83593	0.85256	0.85925	0.81320	0.92327	0.87814
HORD	0.83457	0.92735	0.88096	0.83447	0.90906	0.87226	0.86095	0.92947	0.89521
GP-EI	0.83395	0.92615	0.88005	0.83790	0.91208	0.87499	0.85685	0.92640	0.89160
GP-PI	0.81913	0.93726	0.89239	0.85314	0.91601	0.87604	0.85385	0.91874	0.88955
GP-UCB	0.82407	0.93155	0.88707	0.85314	**0.91994**	**0.87740**	0.85385	0.92364	0.89792
HOUSES-EI	0.84259	**0.94116**	0.88262	0.83406	0.89758	0.87536	0.87219	0.92702	0.89043
HOUSES-PI	0.84753	0.93966	0.87940	0.83608	0.89819	0.87109	0.86035	0.92180	0.88871
HOUSES-UCB	**0.85000**	0.93546	**0.89273**	0.82328	0.91516	0.86919	**0.87745**	0.92000	**0.89371**
Manual Tuning	0.80617	0.93455	0.87036	**0.87751**	0.85468	0.86610	0.79882	**0.95216**	0.87549

of the three acquisition functions, which also suggests that UCB may be the most appropriate acquisition function in 9-ML-CNN hyperparameter optimization.

We present the test accuracy over iterations of the obtained hyperparameter configurations for the 9-ML-CNN problem from the different hyperparameter optimization methods in Fig. 5 and Fig. 6 (d). Our spatial location based non-stationary kernel outperforms the stationary Gaussian kernel with three different acquisition functions, indicating a non-stationary kernel is especially necessary for complicated CNN hyperparameter optimization. The 9-ML-CNN problem again shows that using a non-stationary kernel significantly improves the convergence of the hyperparameter optimization, especially for high-dimensional and complicated deep neural networks.

We observe that HOUSES-UCB reaches better validation accuracy in only 250 epochs compared to the manual tuning method [34]. Table VI presents the ability of ML-CNN with hyperparameter configurations obtained by different hyperparameter optimization methods to classify different type of malignant nodules, and shows the sensitivity, specificity, and AUC on three types of malignant nodules. Our hyperparameter optimization method HOUSES is able to relieve the trivial work of tuning hyperparameters and obtain better results in terms of accuracy compared with manual tuning [34]. The experimental results from above show that the non-stationary assumption is non-trivial for hyperparameter optimization in DNN with Bayesian methods, and incorporating expert intuition based priors into the Bayesian optimization framework improves optimization effectiveness.

VI. CONCLUSION

In this paper, a **H**yperparameter **O**ptimization with s**U**rrogate-a**S**sisted **E**volutionary **S**trategy, named HOUSES, is proposed for CNN hyperparameter optimization. A non-stationary kernel is devised and adopted as a covariance function to define the relationship between different hyperparameter configurations to build the Gaussian processes model, which allows the model to adapt spatial dependent structure which varies with a function of location. Our previously proposed multi-level convolutional neural network (ML-CNN) is developed for lung nodule malignancy classification, whose hyperparameter configuration is optimized by our HOUSES. The experiment results on several deep neural networks and datasets validate that our non-stationary kernel-based approach achieves better hyperparameter configuration than other approaches, such as grid search, random search, Tree-structured Parzen Estimator (TPE), Hyperparameter Optimization via RBF and Dynamic coordinate search (HORD), and stationary kernel based Gaussian kernel Bayesian optimization. The experimental results suggest that, even though random search is a simple and effective way to undertake CNN hyperparameter optimization, it is difficult to find a satisfactory configuration for high-dimensional and complex deep neural networks, and incorporating expert intuition-based priors into a conventional Bayesian optimization framework improves optimization effectiveness. Furthermore, the results also demonstrate that devising a suitable network structure is essential to improve performance, while hyperparameter optimization could help achieve the network's potential.

Our future research will focus on extending HOUSES to deep neural networks architecture search in light of the

promising initial research results. Several works have been proposed to automatically search for well-performing CNN architectures via hill-climbing procedure [13], Q-Learning [62], sequential model-based optimization (SMBO), genetic programming approach [49], and so on [36]. However, few works utilize the surrogate model to reduce the expensive complexity required by the neural architecture search (NAS). Moreover, a simple evolutionary strategy is not an appropriate method to search the surrogate for optimal architecture design, as it is a variable-length optimization problem [28]. Future works will also investigate other evolutionary algorithms for hyperparameter optimization, and the quality-diversity-based evolutionary algorithm may provide a solution.

REFERENCES

[1] M Anthimopoulos, S Christodoulidis, L Ebner, A Christe, and S Mougiakakou. Lung pattern classification for interstitial lung diseases using a deep convolutional neural network. *IEEE Transactions on Medical Imaging*, 35(5):1207–1216, 2016.

[2] Samuel G Armato III, Geoffrey McLennan, Luc Bidaut, Michael F McNitt-Gray, Charles R Meyer, Anthony P Reeves, Binsheng Zhao, Denise R Aberle, Claudia I Henschke, Eric A Hoffman, et al. The lung image database consortium (lidc) and image database resource initiative (idri): a completed reference database of lung nodules on ct scans. *Medical physics*, 38(2):915–931, 2011.

[3] James Bergstra and Yoshua Bengio. Algorithms for hyper-parameter optimization. In *International Conference on Neural Information Processing Systems*, pages 2546–2554, 2011.

[4] James Bergstra and Yoshua Bengio. Random search for hyper-parameter optimization. *Journal of Machine Learning Research*, 13(1):281–305, 2012.

[5] J Bergstra, D Yamins, and D. D Cox. Making a science of model search. 2012.

[6] Xiwen Cai, Liang Gao, and Xinyu Li. Efficient generalized surrogate-assisted evolutionary algorithm for high-dimensional expensive problems. *IEEE Transactions on Evolutionary Computation*, 24(2):365–379, 2019.

[7] Y. Chien. Pattern classification and scene analysis. *IEEE Transactions on Automatic Control*, 19(4):462–463, 2003.

[8] Raunak Dey, Zhongjie Lu, and Yi Hong. Diagnostic classification of lung nodules using 3d neural networks. In *ISBI 2018)*, pages 774–778. IEEE, 2018.

[9] Weiping Ding, Chin-Teng Lin, and Zehong Cao. Deep neuro-cognitive co-evolution for fuzzy attribute reduction by quantum leaping pso with nearest-neighbor memeplexes. *IEEE transactions on cybernetics*, 49(7):2744–2757, 2018.

[10] Weiping Ding, Chin-Teng Lin, Mukesh Prasad, Zehong Cao, and Jiandong Wang. A layered-coevolution-based attribute-boosted reduction using adaptive quantum-behavior pso and its consistent segmentation for neonates brain tissue. *IEEE Transactions on Fuzzy Systems*, 26(3):1177–1191, 2017.

[11] Hongbin Dong, Tao Li, Rui Ding, and Jing Sun. A novel hybrid genetic algorithm with granular information for feature selection and optimization. *Applied Soft Computing*, 65, 2018.

[12] Dominique Douguet. e-lea3d: a computational-aided drug design web server. *Nucleic Acids Research*, 38(Web Server issue):615–21, 2010.

[13] Thomas Elsken, Jan-Hendrik Metzen, and Frank Hutter. Simple and efficient architecture search for convolutional neural networks. *arXiv preprint arXiv:1711.04528*, 2017.

[14] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *2014 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 00, pages 580–587, June 2014.

[15] Rotem Golan, Christian Jacob, and Jörg Denzinger. Lung nodule detection in ct images using deep convolutional neural networks. In *International Joint Conference on Neural Networks*, pages 243–250, 2016.

[16] Markus Heinonen, Henrik Mannerström, Juho Rousu, Samuel Kaski, and Harri Lähdesmäki. Non-stationary gaussian process regression with hamiltonian monte carlo. In *Artificial Intelligence and Statistics*, pages 732–740. PMLR, 2016.

[17] Matthew W. Hoffman and Zoubin Ghahramani. Predictive entropy search for efficient global optimization of black-box functions. In *International Conference on Neural Information Processing Systems*, pages 918–926, 2014.

[18] Holger Hoos and Kevin Leyton-Brown. An efficient approach for assessing hyperparameter importance. In *International Conference on Machine Learning*, pages 754–762, 2014.

[19] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *CVPR*, pages 4700–4708, 2017.

[20] Ilija Ilievski, Taimoor Akhtar, Jiashi Feng, and Christine Shoemaker. Efficient hyperparameter optimization for deep learning algorithms using deterministic rbf surrogates, 2017.

[21] Ronald L. Iman. *Latin Hypercube Sampling*. John Wiley & Sons, Ltd, 2008.

[22] Yaochu Jin. Surrogate-assisted evolutionary computation: Recent advances and future challenges. *Swarm and Evolutionary Computation*, 1(2):61–70, 2011.

[23] Yaochu Jin and B Sendhoff. A systems approach to evolutionary multiobjective structural optimization and beyond. *Computational Intelligence Magazine IEEE*, 4(3):62–76, 2009.

[24] Yaochu Jin, Handing Wang, Tinkle Chugh, Dan Guo, and Kaisa Miettinen. Data-driven evolutionary optimization: An overview and case studies. *IEEE Transactions on Evolutionary Computation*, 23(3):442–458, 2018.

[25] Aydın Kaya and Ahmet Burak Can. A weighted rule based method for predicting malignancy of pulmonary nodules by nodule characteristics. *Journal of biomedical informatics*, 56:69–79, 2015.

[26] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In *International Conference on Neural Information Processing Systems*, pages 1097–1105, 2012.

[27] Yann Lecun, Leon Bottou, Genevieve B. Orr, and Klaus Robert Müller. Efficient backprop. *Neural Networks Tricks of the Trade*, 1524(1):9–50, 1998.

[28] Joel Lehman and Kenneth O Stanley. Evolving a diversity of virtual creatures through novelty search and local competition. In *Proceedings of the 13th annual conference on Genetic and evolutionary computation*, pages 211–218. ACM, 2011.

[29] Bo Liu, Qingfu Zhang, and Georges GE Gielen. A gaussian process surrogate model assisted evolutionary algorithm for medium scale expensive optimization problems. *IEEE Transactions on Evolutionary Computation*, 18(2):180–192, 2013.

[30] Guanfeng Liu, Yi Liu, Kai Zheng, An Liu, Zhixu Li, Yang Wang, and Xiaofang Zhou. Mcs-gpm: multi-constrained simulation based graph pattern matching in contextual social graphs. *IEEE Transactions on Knowledge and Data Engineering*, 30(6):1050–1064, 2017.

[31] Kui Liu and Guixia Kang. Multiview convolutional neural networks for lung nodule classification. *Plos One*, 12(11):12–22, 2017.

[32] Daniel James Lizotte. *Practical bayesian optimization*. University of Alberta, 2008.

[33] Juan Lyu, Xiaojun Bi, and Sai Ho Ling. Multi-level cross residual network for lung nodule classification. *Sensors*, 20(10):2837, 2020.

[34] Juan Lyu and Sai Ho Ling. Using multi-level convolutional neural network for classification of lung nodules on ct images. In *EMBC*, pages 686–689. IEEE, 2018.

[35] Renato Negrinho and Geoff Gordon. Deeparchitect: Automatically designing and training deep architectures. 2017.

[36] Renato Negrinho and Geoff Gordon. Deeparchitect: Automatically designing and training deep architectures. *arXiv preprint arXiv:1704.08792*, 2017.

[37] Christopher Paciorek and Mark Schervish. Nonstationary covariance functions for gaussian process regression. *Advances in neural information processing systems*, 16:273–280, 2003.

[38] Carl Edward Rasmussen. Gaussian processes in machine learning. In *Summer School on Machine Learning*, pages 63–71. Springer, 2003.

[39] Anthony P. Reeves, Alberto M. Biancardi, Tatiyana V. Apanasovich, Charles R. Meyer, Heber Macmahon, Edwin J. R. Van Beek, Ella A. Kazerooni, David Yankelevitz, Michael F. Mcnittgray, and Geoffrey Mclennan. The lung image database consortium (lidc): pulmonary nodule measurements, the variation, and the difference between different size metrics. In *Medical Imaging 2007: Computer-Aided Diagnosis*, pages 1475–1485, 2007.

[40] Ahmed Shaffie, Ahmed Soliman, Luay Fraiwan, Mohammed Ghazal, Fatma Taher, Neal Dunlap, Brian Wang, Victor van Berkel, Robert Keynton, Adel Elmaghraby, et al. A generalized deep learning-based diagnostic system for early diagnosis of various types of pulmonary nodules. *Technology in cancer research & treatment*, 17:1533033818798800, 2018.

[41] Bobak Shahriari, Kevin Swersky, Ziyu Wang, Ryan P. Adams, and Nando De Freitas. Taking the human out of the loop: A review of bayesian optimization. *Proceedings of the IEEE*, 104(1):148–175, 2015.

[42] Shiwen Shen, Simon X Han, Denise R Aberle, Alex A Bui, and William Hsu. An interpretable deep hierarchical semantic convolutional neural network for lung nodule malignancy classification. *Expert Systems with Applications*, 128:84–95, 2019.

[43] W. Shen, M. Zhou, F. Yang, C. Yang, and J. Tian. Multi-scale convolutional neural networks for lung nodule classification. *Inf Process Med Imaging*, 24:588–599, 2015.

[44] Wei Shen, Mu Zhou, Feng Yang, Dongdong Yu, Di Dong, Caiyun Yang, Yali Zang, and Jie Tian. Multi-crop convolutional neural networks for lung nodule malignancy suspiciousness classification. *Pattern Recognition*, 61(61):663–673, 2017.

[45] R. L. Siegel, K. D. Miller, S. A. Fedewa, D. J. Ahnen, R. G. Meester, A Barzi, and A Jemal. Colorectal cancer statistics, 2017. *Ca Cancer J Clin*, 67(3):104–17, 2017.

[46] Jasper Snoek, Hugo Larochelle, and Ryan P. Adams. Practical bayesian optimization of machine learning algorithms. In *International Conference on Neural Information Processing Systems*, pages 2951–2959, 2012.

[47] Jasper Snoek, Kevin Swersky, Rich Zemel, and Ryan Adams. Input warping for bayesian optimization of non-stationary functions. In *International Conference on Machine Learning*, pages 1674–1682, 2014.

[48] Q. Song, L. Zhao, X. Luo, and X. Dou. Using deep learning for classification of lung nodules on computed tomography images. *J Healthc Eng.*, 2017(1):1–7, 2017.

[49] Masanori Suganuma, Shinichi Shirakawa, and Tomoharu Nagao. A genetic programming approach to designing convolutional neural network architectures. pages 497–504, 2017.

[50] Wenqing Sun, Bin Zheng, and Wei Qian. Computer aided lung cancer diagnosis with deep learning algorithms. In *Medical Imaging 2016: Computer-Aided Diagnosis*, 2016.

[51] Yanan Sun, Handing Wang, Bing Xue, Yaochu Jin, Gary G Yen, and Mengjie Zhang. Surrogate-assisted evolutionary deep learning using an end-to-end random forest-based performance predictor. *IEEE Transactions on Evolutionary Computation*, 24(2):350–364, 2019.

[52] Kevin Jordan Swersky. *Improving Bayesian Optimization for Machine Learning using Expert Priors*. PhD thesis, 2017.

[53] Karthick Thiyagarajan, Sarath Kodagoda, Linh Van Nguyen, and Sathira Wickramanayake. Gaussian markov random fields for localizing reinforcing bars in concrete infrastructure. In *ISARC*, volume 35, pages 1–7. IAARC Publications, 2018.

[54] Hao Tong, Changwu Huang, Jialin Liu, and Xin Yao. Voronoi-based efficient surrogate-assisted evolutionary algorithm for very expensive problems. In *2019 IEEE Congress on Evolutionary Computation (CEC)*, pages 1996–2003. IEEE, 2019.

[55] Liang Wang, Zhiwen Yu, Qi Han, Dingqi Yang, Shirui Pan, Yuan Yao, and Daqing Zhang. Compact scheduling for task graph oriented mobile crowdsourcing. *IEEE Transactions on Mobile Computing*, 2020.

[56] Naiyan Wang, Siyi Li, Abhinav Gupta, and DitYan Yeung. Transferring rich feature hierarchies for robust visual tracking. *Computer Science*, 2015.

[57] Quanming Yao, Mengshuo Wang, Yuqiang Chen, Wenyuan Dai, Hu Yi-Qi, Li Yu-Feng, Tu Wei-Wei, Yang Qiang, and Yu Yang. Taking human out of learning applications: A survey on automated machine learning. *arXiv preprint arXiv:1810.13306*, 2018.

[58] Guobin Zhang, Zhiyong Yang, Li Gong, Shan Jiang, Lu Wang, Xi Cao, Lin Wei, Hongyun Zhang, and Ziqi Liu. An appraisal of nodule diagnosis for lung cancer in ct images. *Journal of medical systems*, 43(7):181, 2019.

[59] Miao Zhang and Huiqi Li. A reference direction and entropy based evolutionary algorithm for many-objective optimization. *Applied Soft Computing*, 70:108–130, 2018.

[60] Miao Zhang, Huiqi Li, and Steven Su. High dimensional bayesian optimization via supervised dimension reduction. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, pages 4292–4298. AAAI Press, 2019.

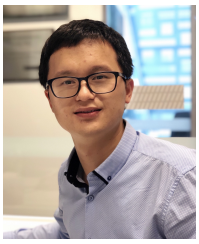
[61] Qingfu Zhang, Wudong Liu, Edward Tsang, and Botond Virginas. Expensive multiobjective optimization by moea/d with gaussian process model. *IEEE Transactions on Evolutionary Computation*, 14(3):456–474, 2010.

[62] Zhao Zhong, Junjie Yan, Wei Wu, Jing Shao, and Cheng-Lin Liu. Practical block-wise neural network architecture generation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2423–2432, 2018.

[63] Wangxia Zuo, Fuqiang Zhou, Zuoxin Li, and Lin Wang. Multi-resolution cnn and knowledge transfer for candidate classification in lung nodule detection. *Ieee Access*, 7:32510–32521, 2019.

Miao Zhang received a Ph.D. from Beijing Institute of Technology (BIT), China. He is also pursuing a dual-degree of Ph.D. at University of Technology Sydney (UTS). His major research interests are AutoML, neural architecture search, Bayesian optimization, continual learning, and deep learning.

Huiqi Li received Ph.D. degree from Nanyang Technological University, Singapore in 2003. She is currently a professor at Beijing Institute of Technology. Her research interests are image processing and computer-aided diagnosis.



Shirui Pan received a Ph.D. in computer science from the University of Technology Sydney (UTS), Ultimo, NSW, Australia. He is currently a lecturer with the Faculty of Information Technology, Monash University, Australia. Prior to this, he was a Lecturer with the School of Software, University of Technology Sydney. His research interests include data mining and machine learning. To date, Dr Pan has published over 80 research papers in top-tier journals and conferences, including the IEEE Transactions on Neural Networks and Learning Systems (TNNLS), IEEE Transactions on Knowledge and Data Engineering (TKDE), IEEE Transactions on Cybernetics (TCYB), KDD, AAAI, and CVPR.

Juan Lyu received the B.E degree in Communication Engineering from Harbin Institute of Technology at Weihai, in 2014. She has a successive master and PhD program since 2014 in Harbin Engineering University (HEU). She is now a PhD candidate in Information and Communication Engineering in HEU. She had studied in the University of Technology Sydney as a joint doctoral student from 2017 to 2019. She is interested in and working on the medical imaging using deep learning, especially the convolutional neural networks (CNNs).

Steve Ling received Ph.D. degree in Electronic and Information Engineering from the Hong Kong Polytechnic University (HKPU). He is currently a Senior Lecturer in University of Technology Sydney. His research interest are Machine Learning, Medical Imaging, and Bio-signal processing.

Steven Su received Ph.D. degree in Control Engineering from RSISE the Australian National University (ANU). He is currently an Associate Professor in University of Technology Sydney (UTS). His research interest are system modeling and control, machine learning, wearable health monitoring, and rehabilitation engineering.