



An active-learning probabilistic neural network for feasibility classification of constrained engineering optimization problems

Dengjian Fang¹ · Taotao Zhang¹ · Fan Wu²

Received: 18 February 2021 / Accepted: 4 June 2021 / Published online: 29 June 2021
© The Author(s), under exclusive licence to Springer-Verlag London Ltd., part of Springer Nature 2021

Abstract

Constrained optimization problems (COPs) with multiple computational expensive constraints are commonly encountered in simulation-based engineering designs. During the optimization process, the feasibility analysis of the intermediate solutions depends on the computational simulations will be computationally prohibitive. To relieve this computational burden, an active-learning probabilistic neural network (AL-PNN) classification modeling approach is proposed to build a classifier for quickly analyzing the feasibility of the intermediate solutions. In the proposed AL-PNN approach, an interesting region tracking strategy is developed to locate the regions that may contain part of the constraint boundary. The judge rule of the interesting region is based on whether the predicted class labels of pseudo points are different in subregions, which is generated by dividing the design space with the K-means cluster algorithm. Once the interesting region is located, the newly infill sample point used to update the PNN classification model can be obtained by a distance screening criterion. Seven numerical cases and the design of the rocket interstage section are used to demonstrate the performances of the proposed approach. The results illustrate that the proposed AL-PNN approach can provide more accurate classification results than the compared four state-of-the-art algorithms.

Keywords Computational expensive constraints · Probabilistic neural network · Active learning · Constrained optimization problems

1 Introduction

Practical engineering design optimization problems usually contain multiple constraints that are dependent on evaluating the computational expensive simulations, e.g., finite element analysis (FEA) and computational fluid dynamic (CFD) models [1]. It will be computational prohibitive if directly relying on these simulations for feasibility analysis of the intermediate solutions in the design space [2–4]. For example, Ford Motor Company spends 36–160 h running a crash simulation for a full passenger car. A promising way to address this issue is to adopt machine-learning techniques to replace the computational expensive simulation model. Among these machine-learning techniques, most of them

are prediction models, also called the metamodel or surrogate models, e.g., polynomial response surface (PRS) model [5, 6], Kriging model [7–11], support vector regression (SVR) model [12, 13], and radial basis function (RBF) model [14–16]. Specifically, the prediction models provide the estimated values of each computational expensive constraint. Then, the predicted values are compared with the threshold values to evaluate the feasible status of the intermediate solutions during the optimization process. For example, Vasu et al. [6] used the least-squares PRS model to accelerate the numerical simulation of the residual stress prediction in laser peening; Qian et al. [8] developed a general sequential constraint updating approach based on the confidence intervals from the Kriging surrogate model; Karamichailidou et al. [14] successfully applied a RBF model for quickly predicting the wind turbine power curve; Zhai et al. [17] used the Kriging surrogate model to analyze the feasible solution in the collision between the missile and the adapter; Asadi et al. [18] demonstrated the effectiveness of the SVR model to predict the thermophysical properties, heat transfer performance, and pumping power of MWCNT/

✉ Taotao Zhang
zhangtaotaonudt@163.com

¹ College of Weapons Engineering, Naval University of Engineering, Wuhan 430033, China

² College of Naval Architecture and Ocean Engineering, Naval University of Engineering, Wuhan 430033, China

ZnO–engine oil hybrid nanofluid; Jiang et al. [19] utilized the Kriging model to classify the failure state of the random candidates.

The other approach to handle multiple computational expensive constraints is introducing the classification model to analyze the feasibility status of the intermediate solutions, e.g., support vector machine (SVM) [20–23], RBF neural network (RBFNN) [24], adaptive neural fuzzy inference system (ANFIS) [25], and probabilistic neural network (PNN) [26, 27]. For example, Singh et al. [22] extended the LOLA–Voronoi algorithm to sequential update the SVM for reducing the required samples; Basudhar et al. [23] developed an improved adaptive sampling technique by automatically selecting and modifying the polynomial degrees for refining SVM boundaries; Harandizadeh et al. [27] applied PNN for the classifying of air-overpressure induced by blasting; Patel et al. in the applied PNN for reliability analysis of the stiffest structure design for a hydrogen storage tank. When comparing with the prediction models, the classification model has more potential to deal with multiple computational expensive constraints [20]. The main features and advantages are (a) it is more suitable for dealing with optimization problem with discontinuous feasible region, (b) the multiple computationally expensive constraints can be reduced to one single classification model, and (c) the classification model can easily considering the correlation between each constraint.

Among these classification models for feasible status analysis, PNN is easier to be implemented due to its advantages of simple form and solid statistical foundation in Bayes decision theory. Although under a limited computational budget, the existing PNN classification modeling approaches are based on the one-shot sampling that usually leads to the false judgment of the points nearby the boundary of the constraint. Generally, an optimum could make some constraints being active. Moreover, all these existing sequential classification modeling methods are developed based on the SVM approaches; investigating the active-learning PNN classification modeling for handling multiple computational expensive constraints will be meaningful to expand the arsenal of constrained optimization problems. To address the above-mentioned issues, an active-learning probabilistic neural network (AL-PNN) classification modeling approach is proposed, aiming at quickly analyzing the feasible status of design solutions. Specifically, an interesting region tracking strategy is developed to locate the regions that may contain part of the boundary of the computational expensive constraints. The performance of the proposed approach will be demonstrated on seven numerical cases and the design of the rocket interstage section.

The remainder of the paper is organized as follows; Sect. 2 simply introduces the basic theory of PNN. Section 3 presents the motivation and details of the proposed

approach. The performance of the proposed method is tested on several numerical and engineering cases in Sect. 4. Some conclusions and future work are provided in Sect. 5.

2 The basic theory of probabilistic neural network

The probabilistic neural network proposed by Specht [28] has been widely used in the field of classification and pattern recognition, such as image recognition, signal processing, and online control [29–31]. When compared with other neural network architectures, PNN is easy to implement and the network-learning process is simple and fast [32]. PNN consists of four layers: the input layer, the pattern layer, the summation layer, and the decision layer, which are shown in Fig. 1.

“Bayes strategies” is used widely in decision rules for classification to minimize the “the expected risk” in the pattern classification. Consider a case of a two-category classification, the problem is to decide whether a pattern $X_t = [X_{t1}, X_{t2}, \dots, X_{tp}]$ belongs to either of the two classes θ_A or θ_B . The Bayes decision rule can be represented by [26]

$$d(x) = \theta_A \text{ if } h_A l_A f_A(x) > h_B l_B f_B(x), \tag{1}$$

$$d(x) = \theta_B \text{ if } h_A l_A f_A(x) < h_B l_B f_B(x), \tag{2}$$

where $f_A(x)$ and $f_B(x)$ are the probability density functions (PDF) for categories A and B, respectively. l_A is the loss function associated with the decision $d(x) = \theta_B$ when the true is θ_A , l_B is the loss function associated with the decision $d(x) = \theta_B$ when the true is θ_B . h_A and h_B are the prior probability of occurrence of patterns from category A and B, respectively.

The main problem with Bayes’ classification method is the estimation of the probability density functions about

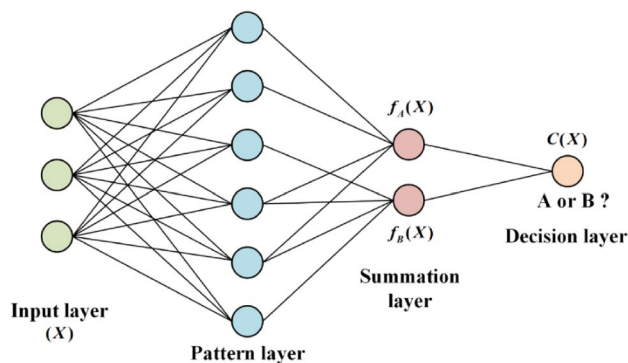


Fig. 1 The general architecture of the probabilistic neural network

each class. This mission is usually completed using a set of training patterns with known classification [33],

$$f_A(X) = \frac{1}{(2\pi)^{p/2}\sigma^p} \frac{1}{m} \times \sum_{i=1}^m \exp \left[-\frac{(X - X_{Ai})^T(X - X_{Ai})}{2\sigma^2} \right], \tag{3}$$

where X is the vector to be classified, $f_A(X)$ is the value of the PDF of class A at point X , X_{Ai} is the neuron vector, m is the number of patterns in category A , p denotes the dimension of the training vectors, and σ is the smoothing parameter. If the smoothing parameter is near zero, the PNN model acts as the nearest neighbor classifier. As it becomes larger, several nearby design vectors are taken into consideration during the model construction process. After the preliminary test, the smoothing parameter is set to 1 in all of the test cases in this manuscript.

Assuming that a priori probabilities for each class are the same and the losses function associated with making wrong decisions for each class are the same, the class can be determined in the decision layer based on the Bayes’s decision rule by [34]

$$c(X) = \operatorname{argmax}\{f_A(x)\}, k = 1, 2, \dots, n, \tag{4}$$

where $c(X)$ denotes the estimated category of the pattern X and n is the number of classes in the training samples.

3 The proposed approach

3.1 Motivation

An engineering design optimization problem with multiple constraints can be generally expressed as:

$$\begin{aligned} &\text{Find } X \\ &\min f(X) \\ &\text{s.t. } g_j(X) \leq 0; \quad i = 1, 2, \dots, J, \end{aligned} \tag{5}$$

where X is the design vector, $f(X)$ denotes the objective function, and $g_j(X)$ is the j th constraints.

When the constraints are evaluated by computationally expensive simulations, it will be impractical to directly use these simulation models to evaluate the feasibility of a large number of intermediate solutions when solving the above optimization problem. As illustrated in [26], the PNN classification model instead of the actual constraints can be used to improve the effectiveness of solving this problem. Even though the PNN classification model has brought many benefits in quickly analyzing the feasibility of the intermediate solutions, a challenge is that it is difficult to construct a PNN classification model with desirable accuracy under a limited computational budget. As shown in Fig. 1a, $g_1(x)$ and $g_2(x)$

denote two computationally expensive constraints, the black circles denote the uniform distributed samples, the black dotted line denotes the PNN classification model that is constructed based on the uniform distributed samples. There are four unobserved points A, B, C, and D marked with triangles to be predicted. Points A and B are feasible design solutions, while C and D are infeasible ones. Since the PNN classification model can capture the general trend of the actual feasible boundary, the predicted feasible status of points B and D, which are far from the feasible boundary, are true. While the predicted feasible status of points A and C, which are near or located on the actual feasible boundary, being totally opposite comparing with their actual status. This is because there exist large prediction errors along the boundary of the constraint with only small amounts of sample points in the neighborhood of the boundary of the constraint.

On the contrary, more sample points are located along the boundary of the constraint in Fig. 2b resulting in a more accurate PNN classification model. The predicted feasible status of all four points is consistent with the actual status. Of course, it should be mentioned that the points are allocated in the design space artificially. In practice, the actual constraints boundary cannot be expressed explicitly (it is a black-box function as it is based on the computationally expensive simulations). This brings a problem on how to allocate the limited sample points in the design space to construct an accurate PNN classification model for quickly analyzing the feasibility of the intermediate solutions. To address this challenge, an active-learning PNN classification modeling approach is proposed, which will be described in the following section.

3.2 The developed interesting region tracking strategy

The proposed active-learning PNN classification modeling approach is based on an iterative-learning process. An initial sample set is generated as a start. Then, the computational simulation models are conducted at these sample points. Based on the sampling data with labels (feasible and nonfeasible), an initial PNN classification model approach is constructed. If the available computational budget is not achieved, new sample points are selected and added to the current sample set for each iteration to update the PNN classification model.

Assuming that the available computational budget is n and the set X^δ with size δ sample points are selected in each iteration process. X^δ is evaluated using computationally expensive simulations to obtain the class label Y^δ . Therefore, the updated process of the training set S can be represented as

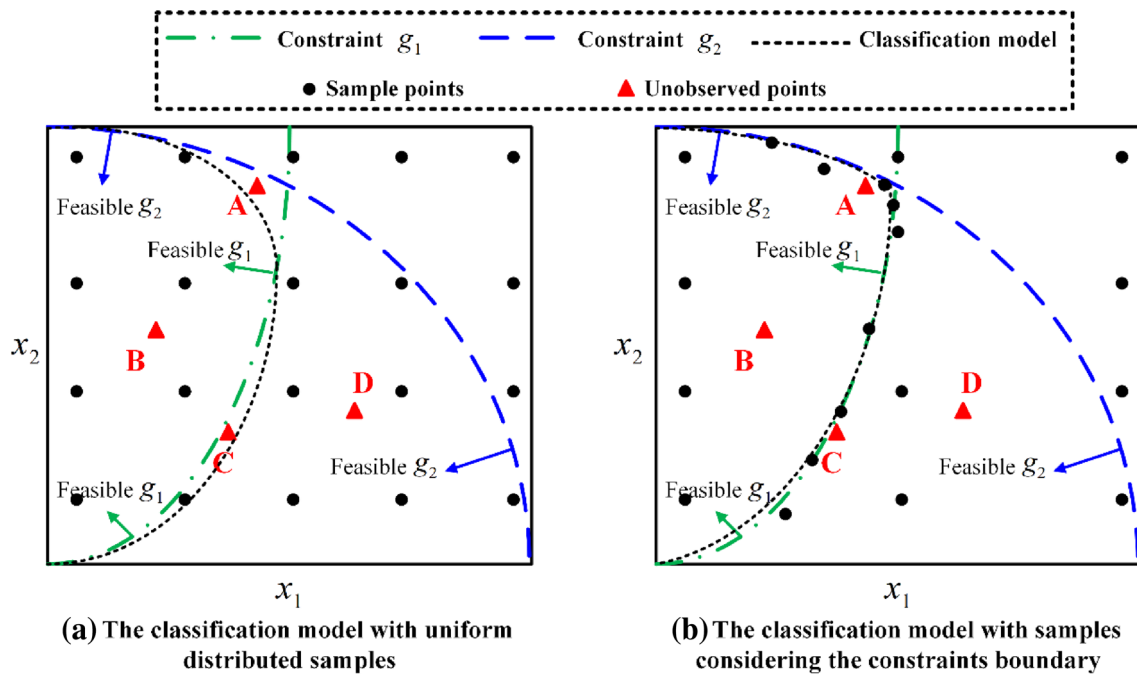


Fig. 2 The schematic diagram of motivation of the proposed approach

$$\begin{aligned}
 Y^\delta &:= f(X^\delta), \\
 S &:= S \cup (X^\delta, Y^\delta).
 \end{aligned}
 \tag{6}$$

The novelty of the proposed approach lies in its way of selecting infill sample points based on a developed interesting region tracking strategy in the design space.

3.3 The coarse interesting region

Before describing the proposed interesting region tracking strategy, the definition of the interesting region is given. As mentioned in Sect. 3.1, with more sample points in the neighborhood of the boundary of the constraint, a more accurate PNN classification model will be. Considering this, a subregion is defined as an interest region if the predicted labels of points in one subregion are different. The basic idea behind this definition is that if one subregion includes different class labels (feasible and nonfeasible samples), it means that this subregion may contain part of the constraint boundary. Therefore, we should pay close attention to this region. As illustrated in Fig. 3, according to the definition, region I can be regarded as an interesting region, while region II is not.

Considering an example with m dimension input, the training set is S in the input space $X \in R^d$ and the output space is $Y \in \{0, 1\}$. The “0” means the design solution is nonfeasible while “1” denotes that the design solution is feasible. The training set is $S = (X, Y) \in X \times Y$ where X is the input variable and consists of n sample points represented

as vectors $\{x_1, x_2, \dots, x_n\}$. The PNN classification model h is defined as $y = h(x)$ which predicts the label of the design solution x .

To determine the interesting region in the whole design space, a set of points randomly distributed in the design space is generated. Then K-means clustering algorithm is introduced to divide these points into k clusters, which can be expressed as

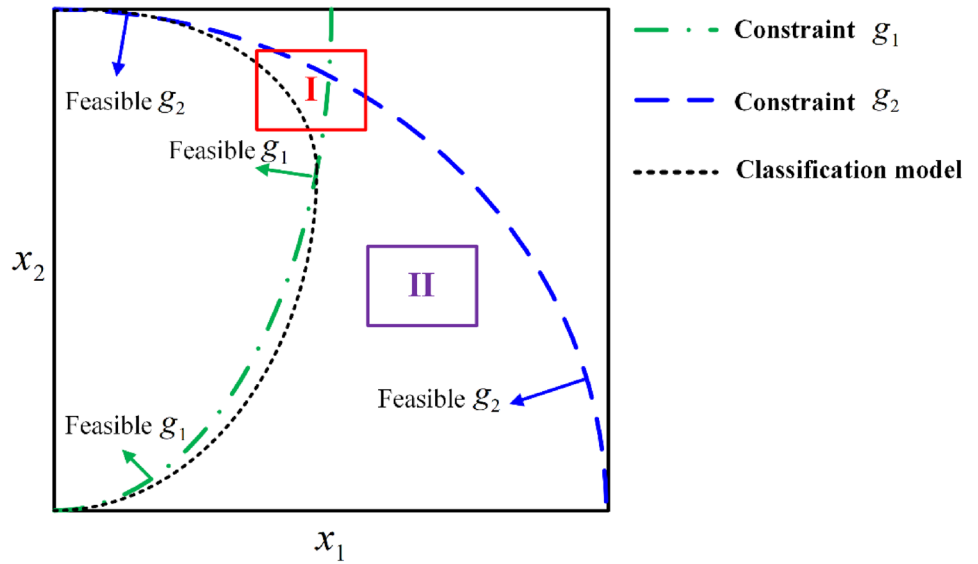
$$[X_1^T, X_2^T, \dots, X_k^T]^T = X.
 \tag{7}$$

Then, the labels of points in each cluster are predicted using the current PNN. These points are termed pseudo points as their actual feasible status is not analyzed by evaluating the computational expensive constraints. According to the definition of interesting regions, once the predicted labels in each cluster are obtained, the coarse interesting region is located.

3.4 The refined interesting region

It is important to point out that the scope and numbers of the interesting region are heavily dependent on the distribution of the pseudo points. Therefore, it is necessary to refine the coarse interesting region, aiming to shrinking the interesting region and focusing more on the constraint boundary. A set of points, in which the number of points is smaller than that in the first round are generated in the coarse interesting

Fig. 3 The definition of the interesting region



regions. To distinguish these points and those in the first round, the points in this stage are termed as refined-pseudo points as they have a higher probability to locate into the neighborhood of the constraint boundary. These points are divided into l clusters and the feasible status of points in each cluster is predicted using the current PNN. The same as in determining the coarse interesting region, if the predicted labels of points in one cluster are different, then this cluster is a cluster of interest and this area is defined as a refined interesting region.

When selecting the sequential samples in the refined interesting region, an intuitive idea is that the newly selected samples should not be close to the existing samples. This is because the prediction error at an unobserved point near the existing sample point is expected to be small. Therefore, to prevent the cluster of sample points, an objective function is formulated to selected new sample points in the refined interesting regions,

$$\begin{aligned} & \max d(\mathbf{x}) \\ & \text{s.t. } \mathbf{x} \in \mathbf{D}_{\text{refined}} \end{aligned} \tag{8}$$

$$d(\mathbf{x}) = \min \left(\sqrt{(\mathbf{x} - \mathbf{x}_i)^T (\mathbf{x} - \mathbf{x}_i)} \right) \quad (i = 1, 2, \dots, N),$$

where \mathbf{x} denotes the potential sequential samples to be added, \mathbf{x}_i is the existing samples, and $\mathbf{D}_{\text{refined}}$ are the refined interesting regions in the design space. Therefore, $d(\mathbf{x})$ is the minimum Euclidean distance between the candidate sample point \mathbf{x} and the sample points in the existing sample set.

3.5 Steps of the proposed approach

The flowchart of the proposed approach is illustrated in Fig. 4. Details of the steps of the proposed approach are presented as follows.

Begin

Step 1: Generate the initial sample points and obtain the feasible status of them.

The proposed approach starts with generating the initial sample points. To obtain the sample points that are located uniformly in the design space, Latin Hypercube Sampling (LHS) is introduced to generate the initial sample points. Then, the initial sample points are evaluated by running the expensive simulator and the corresponding feasible status is obtained.

Step 2: Construct the initial PNN classification model based on the sampling set.

According to the feasible status of the samples, the labels for them are marked and divided into two classes, feasible and nonfeasible. An initial PNN classification model can be constructed based on the labeled samples.

Step 3: Determine the coarse interesting region.

In this step, the coarse interesting region will be determined by following the description in Sect. 3.2.1.

Step 4: Refine the interesting region.

The coarse interesting region is refined by following the description in Sect. 3.2.2.

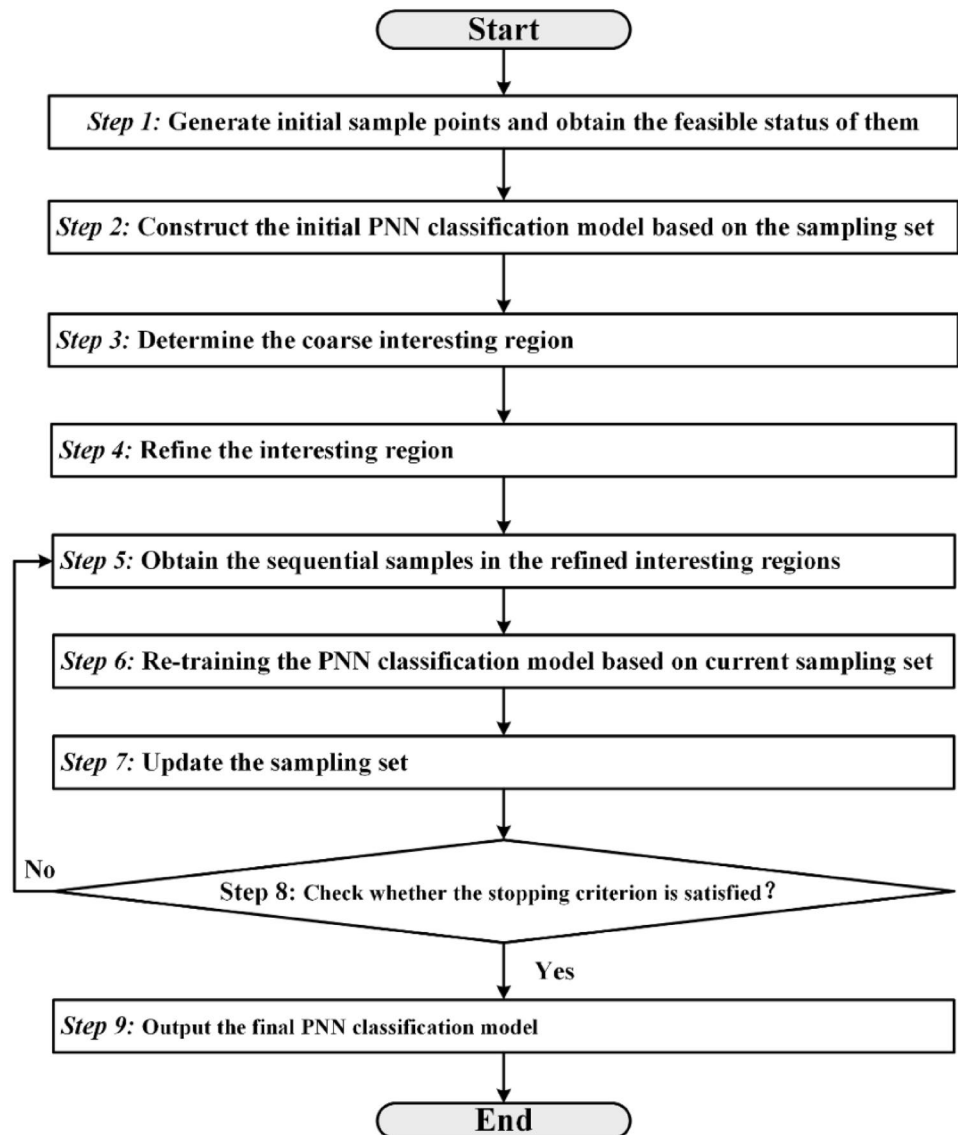
Step 5: Obtain the sequential samples in the refined interesting regions.

The newly added sample points can be obtained by Eq. (8).

Step 6: Update the sampling set.

After the sequential sample points are selected, the corresponding feasible status of the new samples is evaluated

Fig. 4 The framework of the proposed approach



by running the computational expensive simulations. Then, these samples with actual labels are added to the current sampling data.

Step 7: Re-training the PNN classification model based on the current sampling set.

The PNN classification model will be reconstructed according to the updated sampling set.

Step 8: Check whether the stopping criterion is satisfied: If yes, go to Step 9; otherwise, go back to Step 3.

Two stopping criteria can be used for judging whether the sequential process is stopped or not: (a) the prediction accuracy of the PNN classification model meets the preset requirements, and (b) the computational resources used to obtain the sample points reach the allowable upper limit. These two criteria are also used to measure the pros and cons of the classifier: (a) to achieve the required accuracy, which model requires the least computational resources; and (b) with the limited total

resources, which model can achieve the highest accuracy. The main difference between these two criteria is that criterion (a) needs a set of test set to verify the classification accuracy of the constructed PNN model, which is unaffordable in engineering design problems with computational expensive simulation models. What is more, the goal of the proposed approach is to provide a classification model for analyzing the feasible status of the design solutions for a given total computational budget, thus the total available computational budget is adopted as the stopping criterion in the proposed approach with the hope that the final PNN classification model can achieve the required classification accuracy.

Step 9: Output the final PNN classification model.

The final constructed PNN classification model is output when the stopping criterion is reached.

4 End

5 Examples and results

5.1 An illustrative example

A two-dimensional Gaussian function is demonstrated step by step to show how the proposed AL-PNN approach works. The mathematical expression is given in the following equation

$$f(x) = \exp\left(-\frac{(x_1-x'_1)^2+(x_2-x'_2)^2}{\sigma^2}\right), x_1, x_2 \in [-5, 5], \tag{9}$$

where the Gaussian function centered at $(x'_1, x'_2) = (0, 0)$, and the standard deviation σ is equal to $\sqrt{5}$.

The problem concerns finding the region in the design space whose response within 50% of the maximum of Gaussian function ($f_{\max} = 1$) and this region is called feasible region in this paper. Then, the feasible analysis problem is defined as:

$$y_i = \begin{cases} 1, & f(x_i) \in [0.5, \infty) \\ 0, & f(x_i) \in (-\infty, 0.5) \end{cases} \tag{10}$$

In the following subsection, it will be demonstrated step by step in the first iteration to show how the proposed AL-PNN approach selects sample points.

Iteration 1

Step 1: Generate 50 initial sample points and obtain the feasible status of them, the black points in Fig. 5 denote the generated sample points. It can be seen from Fig. 5 that only two points are feasible among the initial samples.

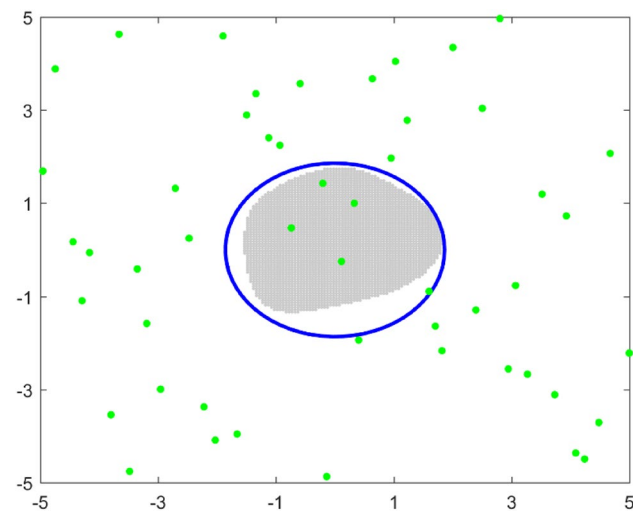


Fig. 5 PNN classification model of initial sample points

Step 2: Build the PNN classification model by the initial sample points. As illustrated in Fig. 5, the green points are the initial sample points generated by LHS. The blue line is the feasible boundary. The responses of the points in the blue circle are within 50% of the maximum Gaussian function. The gray line is the predicted feasible boundary based on the current sample points. It is concluded from Fig. 5 that the predicted feasible region is smaller than that of the actual one. The accuracy of the PNN classification model should be enhanced to satisfy the requirement of design optimization.

Step 3: Determine the coarse interesting region. For more intuitive display, the predicted region (the gray region) is replaced with gray discrete points. The other kinds of color sample points in Fig. 6 are a set of pseudo points generated randomly which is divided into 7 clusters using K-means clustering algorithm, and each color represents one cluster. Then, the constructed PNN classification model is used to estimate the labels of points in each cluster. It can be seen from Fig. 6 that only the blue points spread both inside and outside the blue circle. This indicates that the points in this cluster contain feasible and non-feasible points (with different labels). Therefore, the coarse interesting region is located by the distribution of the blue points. By counting the extremum of the coordinate axes among the blue points, the coarse interesting region is obtained. As shown in Fig. 6, the area surrounded by the white box is the interest region.

Step 4: Refine the interest region. The sample points in Fig. 7 are a new set of pseudo points that are located in the coarse interest region. K-means clustering algorithm is used these pseudo points into 3 clusters, which correspond to three kinds of colors. As in Step 3, if the cluster stretches across the feasible and nonfeasible regions, it will be an interesting subregion because it may contain part of the constraint boundary. It can be seen from Fig. 7, two of the three

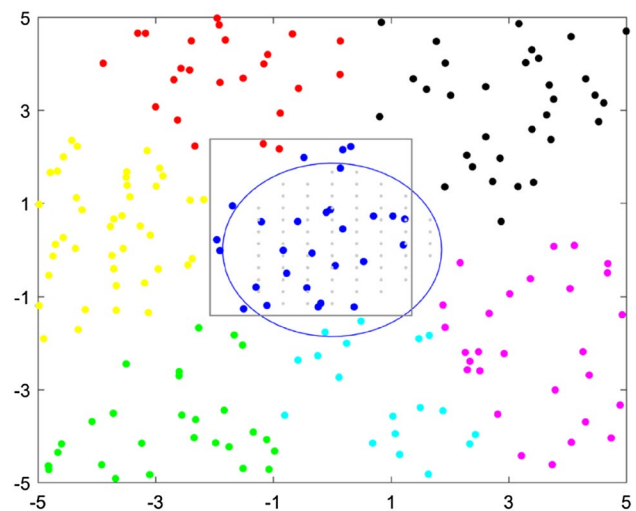


Fig. 6 Determination of the coarse interesting region

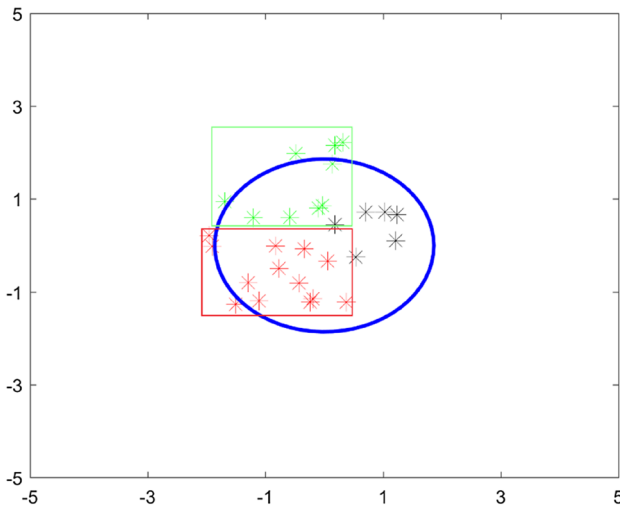


Fig. 7 The refined coarse interesting region

clusters contain different labels, so the two rectangular areas marked with red and green are the refined interest regions.

Step 5: Obtain the sequential samples in the refined interesting regions. As can be seen from Fig. 8, new sample points are obtained by Eq. (8). In this example, two sample points are selected in each refined interesting region. Therefore, four red points in Fig. 8 are obtained.

Steps 6, 7, 8: Update the sampling set and re-training the PNN classification model based on the current sampling set. Then checking the stopping criterion. Since the available computational budget is not used up, the updating process will continue.

The final sample points distribution and the obtained PNN classification model can be seen in Fig. 9. It can be observed that the new sample points are located near the

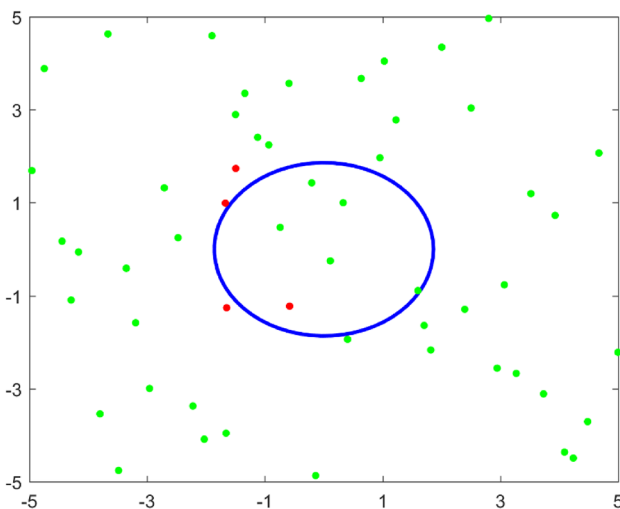
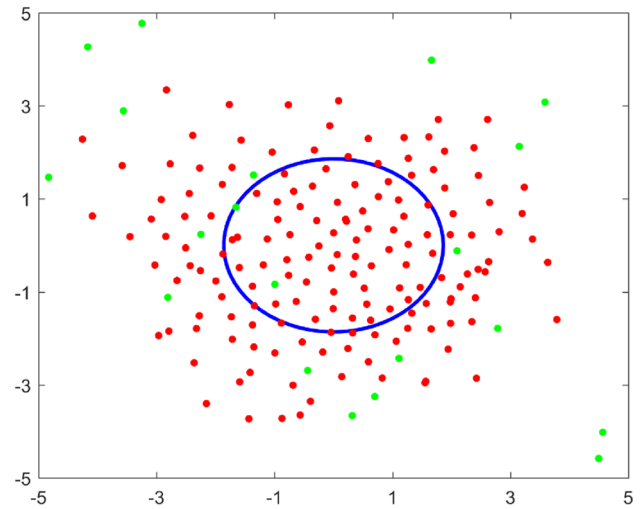
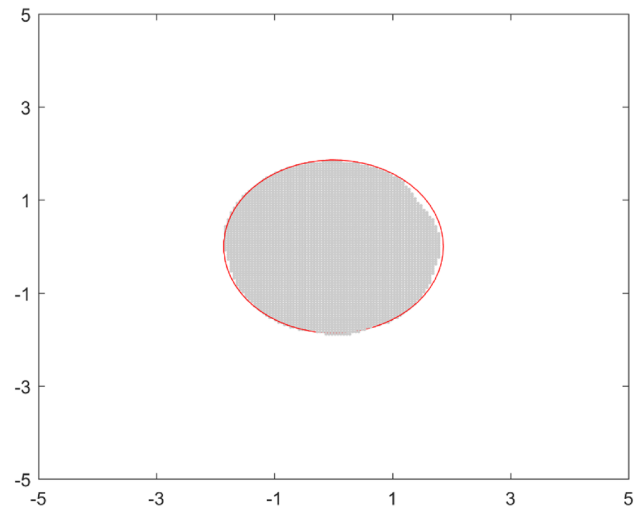


Fig. 8 The newly selected sample points in the first iteration



(a) The final distribution of the sample points



(b) The predicted feasible region

Fig. 9 The final sample points and the classification boundary predicted by the PNN classification model

boundary (the blue line) and the gray region in Fig. 9b is almost coincident with the true feasible region.

To further test the performance of the proposed method, the total number of sample points required when reaching the predefined classification accuracy of 99% is recorded. Five levels of the initial number of samples ranging from 20 to 70 are utilized. To isolate the influence of the distribution of initial samples, each case is duplicated 20 times and the boxplots of the results are plotted in Fig. 10. The upper and lower boundary of the box is the 75% and 25% quantile of the results, while the line within the box denotes the median value of the results. The red points denote the outliers.

It can be seen that the median values of the five boxes are very close, namely different numbers of initial samples result

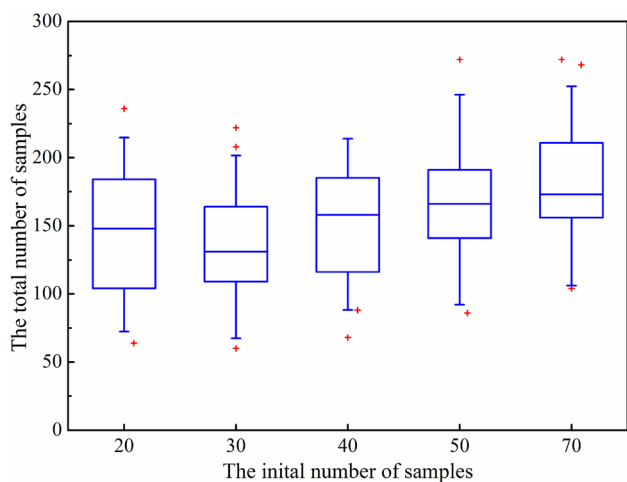


Fig. 10 The total number of samples used by the proposed method under different number of initial samples

in a very similar amount of the total simulation cost, therefore the proposed method is not very sensitive to the initial number of samples. On the other hand, the length of the box is long especially when the initial number of samples is 20; thus, the distribution of the samples has a vital influence on the performance of the proposed method. This influence is partially mitigated by the increasing number of initial samples. How to choose the initial sample points to facilitate the performance of the proposed method will be investigated in our future research.

5.2 Additional numerical examples

In this section, six well-known numerical test problems with different features are used to demonstrate the accuracy of the proposed approach. The description and expression of these numerical cases are described as below:

Disconnected feasible regions: the modified Branin function (P1):

$$f(x) = \left(x_2 - \frac{5.1}{4\pi^2}x_1^2 + \frac{5}{\pi}x_1 - 6\right)^2 + 10\left(1 - \frac{1}{8\pi}\right)\cos x_1 + 10;$$

$$x_1 \in [-5, 10], x_2 \in [0, 15]$$

$$y_i \begin{cases} 1, f(x_i) \in (-\infty, 5] \\ 0, f(x_i) \in (5, \infty) \end{cases} \quad (11)$$

Nonlinear region: two-dimensional function 1 (P2):

$$f(x) = x_2 - |\tan(x_1)| - 1;$$

$$x_1 \in [0, 6], x_2 \in [0, 7]$$

$$y_i \begin{cases} 1, f(x_i) \in (-\infty, 0) \\ 0, f(x_i) \in [0, \infty) \end{cases} \quad (12)$$

Nonlinear region: two-dimensional function 2 (P3):

$$f(x) = x_1x_2 - 4$$

$$x_1 \in [-25, 25], x_2 \in [-30, 30]$$

$$y_i \begin{cases} 1, f(x_i) \in (0, \infty) \\ 0, f(x_i) \in (-\infty, 0) \end{cases} \quad (13)$$

Nonlinear region: 4-dimensional vessel function (P4):

$$y_1 = -x_1 + 0.0193x_3;$$

$$y_2 = -x_2 + 0.00954x_3;$$

$$y_3 = -\pi x_3^2x_4 - \frac{4}{3}\pi x_3^3 + 1,296,000;$$

$$y_4 = x_4 - 240;$$

$$x_1, x_2 \in [0, 1.5], x_3 \in [30, 50], x_4 \in [160, 200]$$

$$y_i \begin{cases} 1, f(x_i) \in (-\infty, 0] \\ 0, f(x_i) \in (0, \infty) \end{cases} \quad (14)$$

Nonlinear region: six-dimensional function (P5):

$$y_1 = 2 - (x_1 + x_2);$$

$$y_2 = (x_1 + x_2) - 6;$$

$$y_3 = x_2 - x_1 - 2;$$

$$y_4 = x_1 - 3x_2 - 2;$$

$$y_5 = (x_3 - 3)^2 + x_4 - 4;$$

$$y_6 = 4 - [(x_5 - 3)^2 + x_6];$$

$$x_1, x_2, x_6 \in [0, 10], x_3, x_5 \in [1, 5], x_4 \in [0, 6]$$

$$y_i \begin{cases} 1, f(x_i) \in (-\infty, 0] \\ 0, f(x_i) \in (0, \infty) \end{cases} \quad (15)$$

Nonlinear region: modified 10-dimensional Zakharov function (P6):

$$y = \sum_{i=1}^n x_i^2 + \sum_{i=1}^n (0.5ix_i)^2 + \sum_{i=1}^n (0.5ix_i)^4$$

$$x_1, x_2, \dots, x_{10} \in [-5, 5] \quad (16)$$

$$y_i \begin{cases} 1, f(x_i) \in (-\infty, 5 \times 10^4] \\ 0, f(x_i) \in (5 \times 10^4, \infty) \end{cases}$$

For comparison, other three methods are used to test these numerical examples: LHS [35], Maximum Distance sampling [36], and LOLA–Voronoi sequential sampling [22]. It is mentioned that the LOLA–Voronoi sequential sampling method is scalable till approximately 5–6 dimensions, so

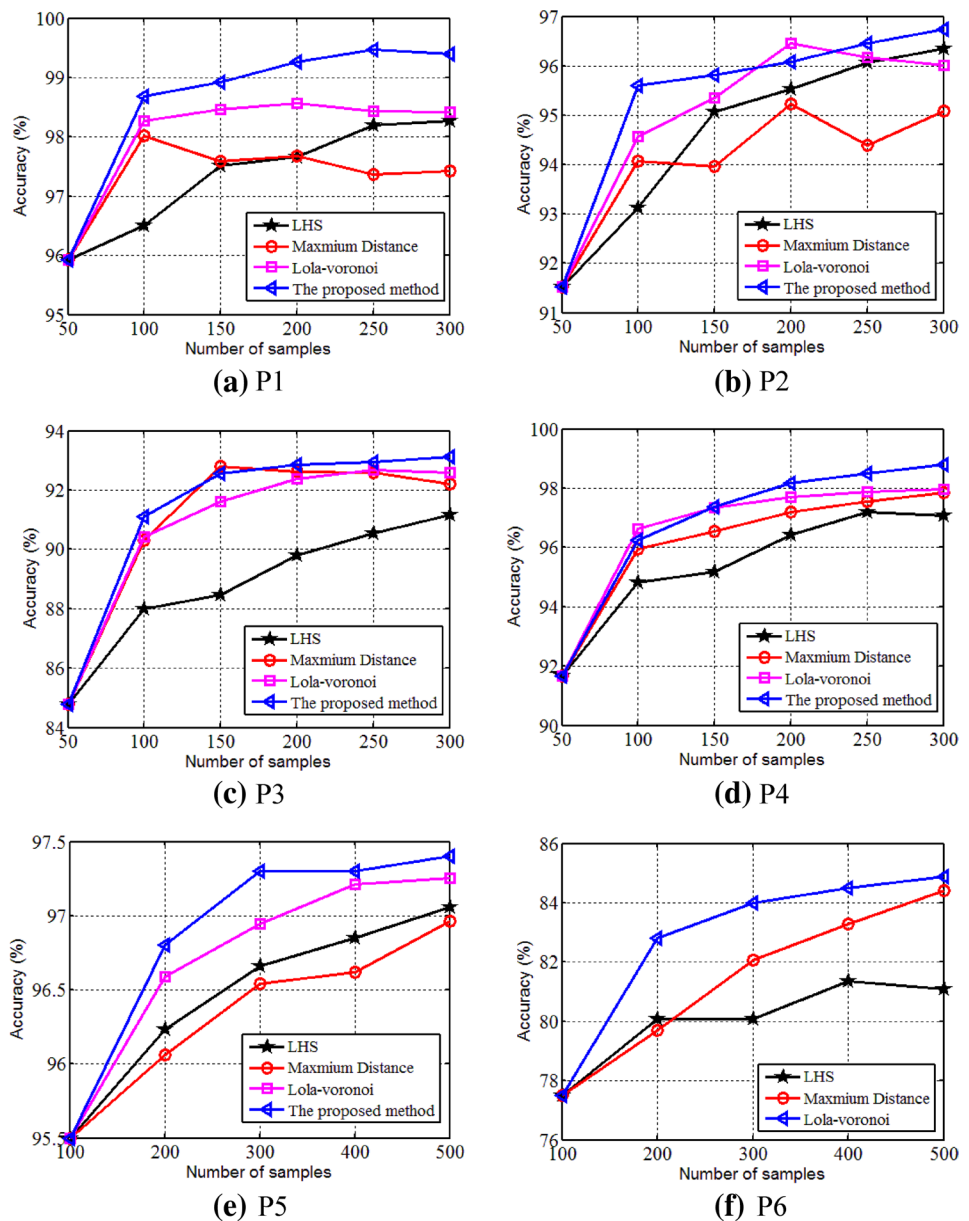
only the other three methods are adopted to the modified 10-dimensional Zakharov function.

Figure 11 demonstrates the classification accuracy of the six numerical examples among the four different classification models. Several intuitive conclusions can be drawn from Fig. 11, i.e. (1) The classification accuracy improves with the increase of the number of sample points in general, and classification accuracy increases fast when the number of sample points is from 50 to 100; (2) For the problem with disconnected feasible regions (P1), the Maximum Distance approach performs the worse in all the models, the proposed AL-PNN approach ranks first, followed by the LOLA–Voronoi approach. (3) When dealing with the problem with non-linear regions, the LOLA–Voronoi approach performs the best, the LHS approach performs the worst in three cases

(P3, P4, and P6). The performance of the LOLA–Voronoi approach is a little bit better than that of the Maximum Distance approach. Overall, for all the numerical cases, the proposed AL-PNN approach performs better than the other three approaches in terms of classification accuracy.

The number of clusters is an important factor that may have an influence on the capability of the proposed AL-PNN approach. To analyze the sensitivity of the number of clusters, four levels of the number of clusters, 3, 5, 7, and 9 clusters, are adopted to test on the four two-dimensional nonlinear numerical examples (Gaussian function, P1, P2, P3). Figure 12 shows the classification accuracy of the four two-dimensional numerical examples under four levels of clusters. As illustrated in Fig. 12, the classification accuracy improves gradually with the number of clusters increasing,

Fig. 11 Comparison results among the four classification models on the additional numerical examples



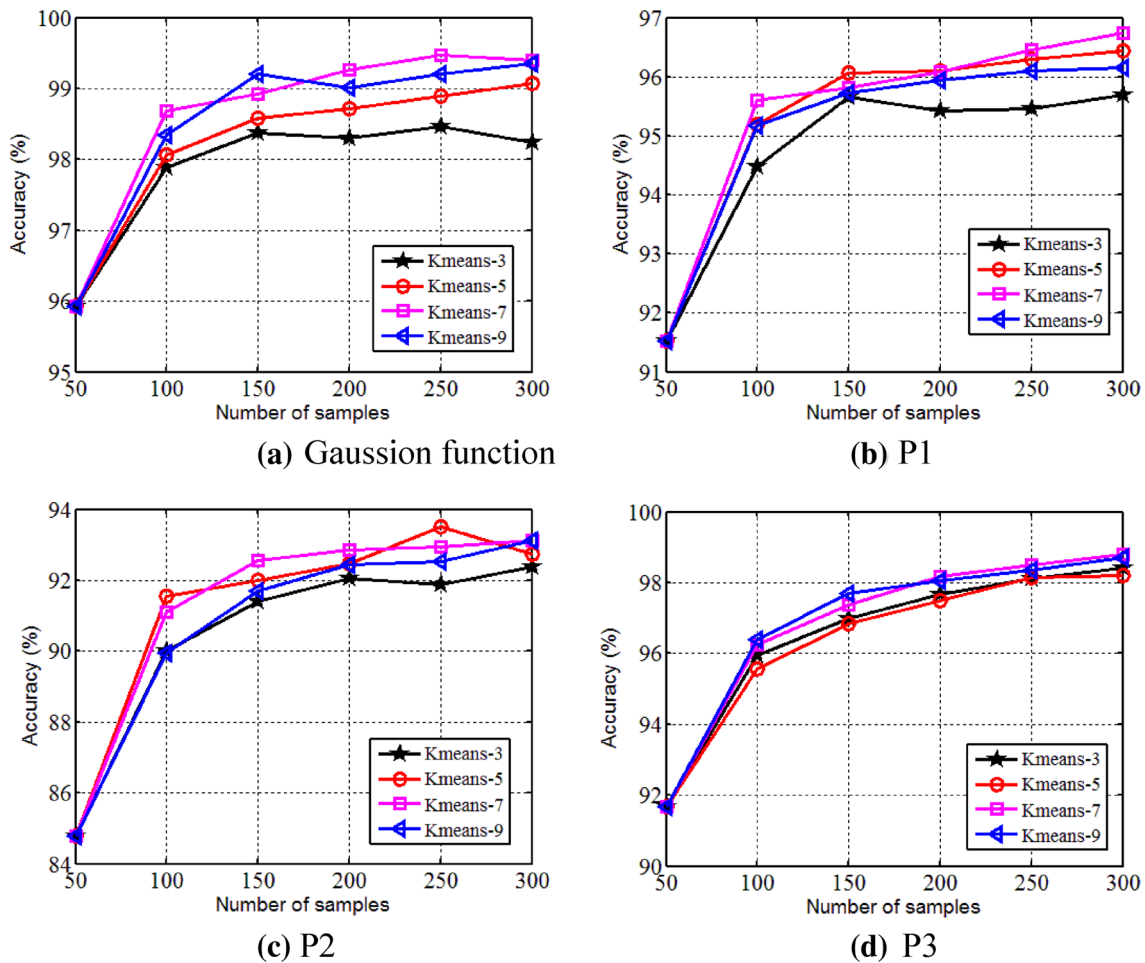


Fig. 12 The sensitive analysis of the number of clusters

while when the number of clusters exceeds 7, the classification accuracy improves slowly, even being reduced. Overall, when the number of clusters is larger than 5, the proposed AL-PNN approach is not sensitive to the numbers of the clusters.

5.3 Engineering case

In this section, the feasible status analysis of the design solutions for the rocket interstage section is performed by the proposed approach.

The interstage section is subjected to a shear force of 79,380 N and a bending moment of 190,000 N m. The elastic modulus of the material was $E=207$ GPa, Poisson’s ratio is $\mu = 0.3$. The goal of this design problem is to minimize the weight of the interstage section under the constraints of the allowable stress and deformation of the structure. The problem includes five consecutive design variables: height of the interstage section, radius of the small end, radius of the large end, the number of axial reinforcement, the number of hoop

reinforcement. The range of values for the five design variables is shown in Table 1.

The mathematical model of the lightweight optimization problem can be expressed as

$$\begin{aligned}
 &\text{Find: } \mathbf{x} = [R, r, h, a, b] \\
 &\text{min: } \text{Mass}(\mathbf{x}) \\
 &\text{s.t.: } \sigma_s(\mathbf{x}) \leq 110 \text{ Mpa} \\
 &\quad d(\mathbf{x}) \leq 0.60 \text{ mm.}
 \end{aligned}
 \tag{17}$$

Table 1 Range of values for design variables

Design variables	Range
height of the interstage section, h	100–1000 mm
radius of the small end, r	100–500 mm
radius of the large end, R	500–1000 mm
the number of axial reinforcement, a	2–30
the number of hoop reinforcement, b	1–10

It can be seen from Eq. (17) that the maximum stress σ_s and maximum deformation $d(x)$ of the interstage section cannot be directly calculated by an explicit function. Therefore, the proposed AL-PNN approach is used to handle these two expensive constraints by constructing a classification model between the design variables and the maximum stress and the maximum deformation. In this paper, the software MSC/NASTRAN and PATRAN are used to calculate the maximum stress and maximum deformation of the interstage section. The FEA model with the boundary condition is shown in Fig. 13. All the simulations are conducted on the computational platform with a 3.20 GHz Intel (R) Core (TM) i7 8700 CPU and 8 GB RAM. One simulation takes about 10 min. The plots of the displacement cloud and the stress cloud at a typical design solution are demonstrated in Fig. 14.

In this example, the total number of sample points is limited to 50 to build a classification model. First, 20 sample points are generated at one time through LHS, and the remaining 30 sample points are selected in the updating process. This study randomly selects 50 verification points to calculate the classification accuracy of the four compared approaches. A summary of the verification results is listed in Table 1. As illustrated in Table 1, the proposed AL-PNN classification model with a 96% classification accuracy only misjudges two points. While the LHS has a classification accuracy of less than 90%. The performance of the Local–Voronoi is comparable with that of maximum distance, which misjudges the status of five points. Taking the results from the LHS method as the baseline, the percentage of improvement from the maximum distance method, the Local–Voronoi method, and the AL-PNN methods are about 2.3%, 2.3%, and 9.1% respectively (Table 2).

6 Conclusions

In this paper, an active-learning probabilistic neural network classification model for feasibility analysis of the design solutions for problems with computationally expensive

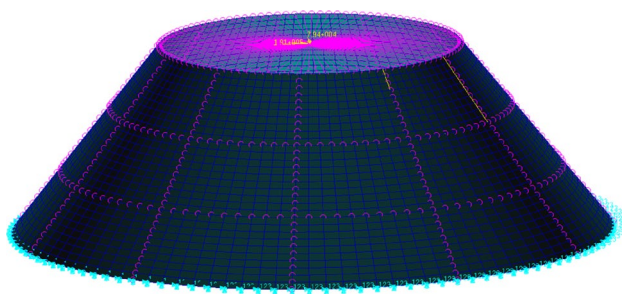
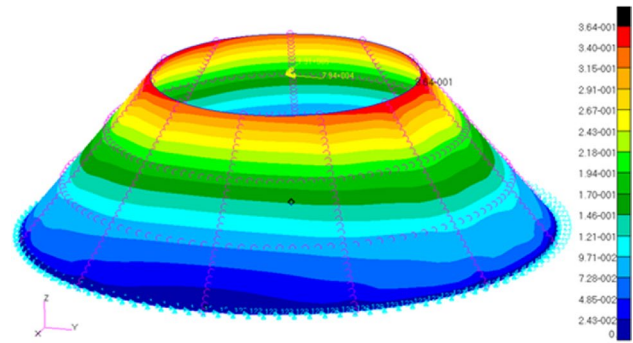
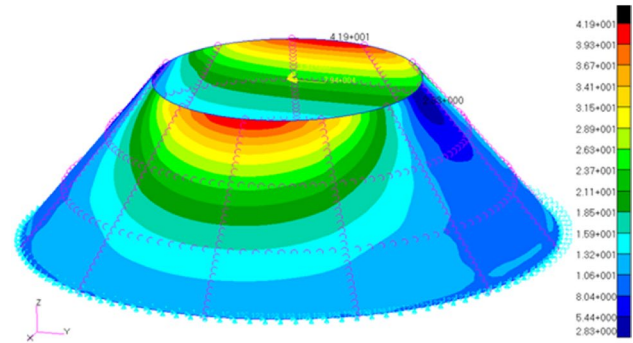


Fig. 13 The finite element model at typical design solution



(a) The displacement cloud



(b) The stress cloud

Fig. 14 The simulation result at typical design solution

constraints is proposed. In the proposed approach, an interesting region tracking strategy is developed to locate the regions that may contain part of the constraint boundary. These interesting regions are identified by judging whether the predicted class labels of the pseudo points are the same or not in each cluster. The newly updated sample points are obtained in the refined interesting regions using the maximum distance sampling method.

Seven numerical examples and an engineering case is used to compare the accuracy of the proposed method other four existing sampling methods. The comparison results demonstrate that (1) Under the limited computational budget, the proposed PNN classification model outperforms the other three existing methods in terms of classification accuracy; (2) The proposed PNN classification model is not sensitive to the number of clusters; (3) The proposed PNN classification model is applicable to practical engineering

Table 2 The comparison results among different approaches

	LHS	Maximin distance	Local–Voronoi	AL-PNN
Classification accuracy	88%	90%	90%	96%

optimization problems with computational expensive constraints.

With the increasing dimension of the design problems, the complexity of the classification boundary increases sharply and a huge number of sample points is required to construct a PNN model with the desired accuracy. Therefore, the proposed method is recommended to use in problems with less than ten design variables. As part of future work, extending the proposed method to the multifidelity situations to relieve the computational burden in high-dimensional problems will be investigated. What's more, the proposed approach only tested on deterministic simulations, extending the proposed approach for addressing stochastic simulations will be also addressed to enhance its engineering practicability.

References

- Chatterjee T, Chakraborty S, Chowdhury R (2019) A critical review of surrogate assisted robust design optimization. *Arch Comput Methods Eng* 26(1):245–274
- Wang H, Jin Y, Yang C, Jiao L (2020) Transfer stacking from low-to high-fidelity: a surrogate-assisted bi-fidelity evolutionary algorithm. *Appl Soft Comput* 92:106276
- Ruan X, Jiang P, Zhou Q, Hu J, Shu L (2020) Variable-fidelity probability of improvement method for efficient global optimization of expensive black-box problems. *Struct Multidiscip Optim* 62(6):3021–3052
- Zhou Q, Wu J, Xue T, Jin P (2021) A two-stage adaptive multifidelity surrogate model-assisted multi-objective genetic algorithm for computationally expensive problems. *Eng Comput* 37:623–639
- Zeng P, Li T, Chen Y, Jimenez R, Feng X, Senent S (2020) New collocation method for stochastic response surface reliability analyses. *Eng Comput* 36:1751–1762
- Vasu A, Grandhi RV (2014) Response surface model using the sorted k-fold approach. *AIAA J* 52(10):2336–2341
- Hu J, Zhou Q, McKeand A, Xie T, Choi SK (2021) A model validation framework based on parameter calibration under aleatory and epistemic uncertainty. *Struct Multidiscip Optim* 63(2):645–660
- Qian J, Yi J, Cheng Y, Liu J, Zhou Q (2020) A sequential constraints updating approach for Kriging surrogate model-assisted engineering optimization design problem. *Eng Comput* 36:993–1009
- Viana FAC, Simpson TW, Balabanov V, Toropov V (2014) Special section on multidisciplinary design optimization: metamodeling in multidisciplinary design optimization: how far have we really come? *AIAA J* 52(4):670–690
- Jiang C, Hu Z, Liu Y, Mourelatos ZP, Gorsich D, Jayakumar P (2020) A sequential calibration and validation framework for model uncertainty quantification and reduction. *Comput Methods Appl Mech Eng* 368:113172
- Jiang C, Qiu H, Yang Z, Chen L, Gao L, Li P (2019) A general failure-pursuing sampling framework for surrogate-based reliability analysis. *Reliab Eng Syst Saf* 183:47–59
- Zhou Q, Shao X, Jiang P, Zhou H, Shu L (2015) An adaptive global variable fidelity metamodeling strategy using a support vector regression based scaling function. *Simul Model Pract Theory* 59(12):18–35
- Pal M, Deswal S (2011) Support vector regression based shear strength modelling of deep beams. *Comput Struct* 89(13):1430–1439
- Karamichailidou D, Kaloutsas V, Alexandridis AJRE (2021) Wind turbine power curve modeling using radial basis function neural networks and tabu search. *Renew Energy* 163:2137–2152
- Safarpour M, Shirzadi A (2021) Numerical investigation based on radial basis function–finite-difference (RBF–FD) method for solving the Stokes–Darcy equations. *Eng Comput* 37:909–920
- Rashid K, Ambani S, Cetinkaya E (2013) An adaptive multi-quadratic radial basis function method for expensive black-box mixed-integer nonlinear constrained optimization. *Eng Optim* 45(2):185–206
- Zhai Z, Li H, Wang X (2020) An adaptive sampling method for Kriging surrogate model with multiple outputs. *Eng Comput*. <https://doi.org/10.1007/s00366-020-01145-1>
- Asadi A, Bakhtiyari AN, Alarifi IM (2020) Predictability evaluation of support vector regression methods for thermophysical properties, heat transfer performance, and pumping power estimation of MWCNT/ZnO–engine oil hybrid nanofluid. *Eng Comput*. <https://doi.org/10.1007/s00366-020-01038-3>
- Jiang C, Qiu H, Gao L, Wang D, Yang Z, Chen L (2020) Real-time estimation error-guided active learning Kriging method for time-dependent reliability analysis. *Appl Math Model* 77:82–98
- Basudhar A, Dribusch C, Lacaze S, Missoum S (2012) Constrained efficient global optimization with support vector machines. *Struct Multidiscip Optim* 46(2):201–221
- Basudhar A, Missoum S, Sanchez AH (2008) Limit state function identification using support vector machines for discontinuous responses and disjoint failure domains. *Probab Eng Mech* 23(1):1–11
- Singh P, Van Der Herten J, Deschrijver D, Couckuyt I, Dhaene T (2017) A sequential sampling strategy for adaptive classification of computationally expensive data. *Struct Multidiscip Optim* 55(4):1425–1438
- Basudhar A, Missoum S (2010) An improved adaptive sampling scheme for the construction of explicit boundaries. *Struct Multidiscip Optim* 42(4):517–529
- Suresh S, Dong K, Kim HJ (2010) A sequential learning algorithm for self-adaptive resource allocation network classifier. *Neurocomputing* 73(16–18):3012–3019
- Le Chau N, Dao T-P (2020) An efficient hybrid approach of improved adaptive neural fuzzy inference system and teaching learning-based optimization for design optimization of a jet pump-based thermoacoustic-stirling heat engine. *Neural Comput Appl* 32(11):7259–7273
- Patel J, Choi S-K (2012) Classification approach for reliability-based topology optimization using probabilistic neural networks. *Struct Multidiscip Optim* 45(4):529–543
- Harandizadeh H, Armaghani D (2020) Prediction of air-overpressure induced by blasting using an ANFIS–PNN model optimized by GA. *Appl Soft Comput* 99:106904
- Specht DF (1990) Probabilistic neural networks. *Neural Netw* 3(1):109–118
- Baliarsingh SK, Vipsita S, Gandomi AH, Panda A, Bakshi S, Ramasubbareddy S (2020) Analysis of high-dimensional genomic data using MapReduce based probabilistic neural network. *Comput Methods Prog Biomed* 195:105625
- Earp S, Curtis A (2020) Probabilistic neural network-based 2D travel-time tomography. *Neural Comput Appl* 32(22):17077–17095
- Sun Y, Chen J, Yuen C, Rahardja S (2017) Indoor sound source localization with probabilistic neural network. *IEEE Trans Ind Electron* 65(8):6403–6413

32. Zeinali Y, Story BA (2017) Competitive probabilistic neural network. *Integr Comput Aided Eng* 24(2):105–118
33. Yao Y, Wang N (2020) Fault diagnosis model of adaptive miniature circuit breaker based on fractal theory and probabilistic neural network. *Mech Syst Signal Process* 142:106772
34. Ahmadipour M, Hizam H, Othman ML, Radzi MAM, Murthy AS (2018) Islanding detection technique using slantlet transform and ridgelet probabilistic neural network in grid-connected photovoltaic system. *Appl Energy* 231:645–659
35. Crombecq K, Laermans E, Dhaene T (2011) Efficient space-filling and non-collapsing sequential design strategies for simulation-based modeling. *Eur J Oper Res* 214(3):683–696
36. Liu H, Ong Y-S, Cai J (2018) A survey of adaptive sampling for global metamodeling in support of simulation-based complex engineering design. *Struct Multidiscip Optim* 57(1):393–416