Research Article

An Extreme Learning Machine Based on the Mixed Kernel Function of Triangular Kernel and Generalized Hermite Dirichlet Kernel

Senyue Zhang and Wenan Tan

College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing, Jiangsu 211106, China

Correspondence should be addressed to Senyue Zhang; 35487300@qq.com

Received 2 June 2016; Accepted 16 August 2016

Academic Editor: Xing-Gang Yan

Copyright © 2016 S. Zhang and W. Tan. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

According to the characteristics that the kernel function of extreme learning machine (ELM) and its performance have a strong correlation, a novel extreme learning machine based on a generalized triangle Hermitian kernel function was proposed in this paper. First, the generalized triangle Hermitian kernel function was constructed by using the product of triangular kernel and generalized Hermite Dirichlet kernel, and the proposed kernel function was proved as a valid kernel function of extreme learning machine. Then, the learning methodology of the extreme learning machine based on the proposed kernel function was presented. The biggest advantage of the proposed kernel is its kernel parameter values only chosen in the natural numbers, which thus can greatly shorten the computational time of parameter optimization and retain more of its sample data structure information. Experiments were performed on a number of binary classification, multiclassification, and regression datasets from the UCI benchmark repository. The experiment results demonstrated that the robustness and generalization performance of the proposed method are outperformed compared to other extreme learning machines with different kernels. Furthermore, the learning speed of proposed method is faster than support vector machine (SVM) methods.

1. Introduction

The kernel extreme learning machine (KELM) is proposed by Huang et al. in 2010 by applying the kernel functions to ELM algorithm [1, 2] and where the random hidden layer feature mapping in ELM is substituted by the kernel mapping [3]. It effectively improves the undesirable generalization performance and stability caused by the stochastic nature of hidden layer output matrix and greatly reduces computational complexity. In KELM, optimization of the number of hidden layer nodes is avoided and the least square optimal solution can be obtained. Compared with SVM [4] and basic ELM [5, 6], it can provide more stable and better generalization performance. Hence, KELM has been widely applied in classification and regression problems [7, 8].

It is well known that the learning ability and generalization performance of ELM mainly depend on the kernel function, and different kernel functions or same kernel function with different parameters has different influence on the generalization performance. Besides, the time required to search for the optimal kernel parameters is different using various kernel functions. Normally, the selection and optimization of kernel parameters are much tedious and time-consuming [9, 10]. Liu et al. [11] pointed out that the common Gauss kernel function and polynomial kernel function are very sensitive to the changes of kernel parameters, so the selection range of kernel parameters is large with small stepsize leading to high computation complex issue. In order to solve this issue, a series of SVM kernel functions based on generalized orthogonal polynomials have been constructed in recent years [12–16]. Ozer et al. [12] introduced a set of Chebyshev kernel functions derived from the generalized Chebyshev polynomials. The test results show that the generalized Chebyshev kernel approaches the minimum support vector number for classification in general. Furthermore, Zhang et al. constructed a series of novel SVM kernel functions, such as Laguerre kernel function from generalized Laguerre polynomial and Hermite kernel function from
generalized Hermite polynomial [13–15]. Those algorithms can shorten the time of parameter optimization; however, the processing to the parameters of weight function is so simple that the influence of structure information of sample data on the generalization performance is neglected. Tian and Wang [17] verified that Gaussian Hermite kernel function can achieve the highest classification accuracy in the binary classification problem compared to the rest of above orthogonal polynomial kernel functions; however, the efficiency of its training and the robustness are relatively lower.

According to above analysis, based on Hermite orthogonal polynomials, a mixed kernel function called Generalized Triangular Hermite kernel function is constructed by using the product of triangular kernel and generalized Hermite Dirichlet kernel. This kernel function has only one parameter chosen from a small range of integer numbers, thus the parameter optimization is facilitated greatly, and more structure information of sample data is retained. It is proved that generalized triangle Hermite kernel can be used as an allowed kernel function of extreme learning machine in theory. The effectiveness of the proposed method for regression and binary, multiclass classification problems is demonstrated by performing numerical experiments on a number of real-world datasets from the UCI benchmark repository and comparing their results with SVM and other extreme learning machines with different kernels.

2. Kernel Extreme Learning Machine

2.1. Introduction to Kernel Extreme Learning Machine. A training set \( \mathcal{N} = \{ (\mathbf{x}_i, t_i) \mid \mathbf{x}_i \in \mathbb{R}^d, t_i \in \mathbb{R}, i = 1, \ldots, N \} \) is given, the output function of hidden node is chosen as \( G(\mathbf{a}_i, \mathbf{b}_i, \mathbf{x}) \) with \( L \) hidden nodes, and ELM algorithm can be written as follows.

Step 1. Randomly generate hidden node parameters \( (\mathbf{a}_i, \mathbf{b}_i), i = 1, \ldots, L \).

Step 2. Calculate the hidden layer output matrix \( \mathbf{H} \) (ensure \( \mathbf{H} \) to be full rank):

\[
\mathbf{H} = \begin{bmatrix}
\mathbf{h}(\mathbf{x}_1) \\
\vdots \\
\mathbf{h}(\mathbf{x}_N)
\end{bmatrix} = \begin{bmatrix}
G(\mathbf{a}_1, \mathbf{b}_1, \mathbf{x}_1) & \cdots & G(\mathbf{a}_L, \mathbf{b}_L, \mathbf{x}_1) \\
\vdots & \ddots & \vdots \\
G(\mathbf{a}_1, \mathbf{b}_1, \mathbf{x}_N) & \cdots & G(\mathbf{a}_L, \mathbf{b}_L, \mathbf{x}_N)
\end{bmatrix}_{N \times L}
\] (1)

Step 3. Calculate the output weight vector \( \mathbf{\beta} \):

\[
\mathbf{\beta} = \mathbf{H}^T \left( \frac{1}{\lambda} + \mathbf{H} \mathbf{H}^T \right)^{-1} \mathbf{T},
\] (2)

where \( \lambda \) is the regularization coefficient and \( \mathbf{T} = [t_1 \cdots t_N]_{N \times 1} \).

The output function of ELM is

\[
f(\mathbf{x}) = \mathbf{h}(\mathbf{x}) \mathbf{H}^T \left( \frac{1}{\lambda} + \mathbf{H} \mathbf{H}^T \right)^{-1} \mathbf{T}.
\] (3)

If the hidden layer feature mapping \( \mathbf{h}(\mathbf{x}) \) is unknown, a kernel matrix can be defined to replace \( \mathbf{H} \mathbf{H}^T \) by using Mercer’s condition. Thus, KELM algorithm is generated as follows:

\[
\mathbf{\Omega}_{\text{ELM}} = \mathbf{H}^T \mathbf{H} : \mathbf{\Omega}_{\text{ELM},ij} = \mathbf{h}(\mathbf{x}_i) \cdot \mathbf{h}(\mathbf{x}_j)
\]

where \( \mathbf{h}(\mathbf{x}) \) is written as follows.

\[
\mathbf{h}(\mathbf{x}) = \begin{bmatrix}
K(\mathbf{x}_1, \mathbf{x}_1) & \cdots & K(\mathbf{x}_1, \mathbf{x}_N) \\
\vdots & \ddots & \vdots \\
K(\mathbf{x}_N, \mathbf{x}_1) & \cdots & K(\mathbf{x}_N, \mathbf{x}_N)
\end{bmatrix}
\] (4)

Finally, the output function of KELM is defined as

\[
f(\mathbf{x}) = \left( \frac{1}{\lambda} + \mathbf{\Omega}_{\text{ELM}} \right)^{-1} \mathbf{H}^T \left( \frac{1}{\lambda} + \mathbf{H} \mathbf{H}^T \right)^{-1} \mathbf{T}.
\] (5)

2.2. Introduction to the Property of KELM Kernel Function. Some of the well-known common KELM kernel functions are

(1) polynomial kernel: \( K(\mathbf{x}, \mathbf{z}) = (\mathbf{x}^T \mathbf{z} + 1)^n \);
(2) Gaussian kernel: \( K(\mathbf{x}, \mathbf{z}) = \exp\left(-\|\mathbf{x} - \mathbf{z}\|^2/2\sigma^2\right) \);
(3) Laplacian kernel: \( K(\mathbf{x}, \mathbf{z}) = \exp\left(-\|\mathbf{x} - \mathbf{z}\|/\sigma\right) \).

In addition to the above kernel functions, a new kernel function also can be constructed according to the property of the kernel function.

Property 1. Assume that \( K_1(\mathbf{x}, \mathbf{z}) \) and \( K_2(\mathbf{x}, \mathbf{z}) \) are valid kernel functions on \( \mathbb{R}^d \times \mathbb{R}^d \); then kernel functions \( K(\mathbf{x}, \mathbf{z}) = K_1(\mathbf{x}, \mathbf{z}) + K_2(\mathbf{x}, \mathbf{z}) \) and \( K(\mathbf{x}, \mathbf{z}) = K_1(\mathbf{x}, \mathbf{z}) \times K_2(\mathbf{x}, \mathbf{z}) \) also are valid on \( \mathbb{R}^d \times \mathbb{R}^d \).

Theorem 1. Let \( f : \mathbb{R}^d \to \mathbb{R} \) be an integrable bounded continuous function. Then the necessary and sufficient condition for the translation invariant function \( K(\mathbf{x}, \mathbf{z}) = f(\mathbf{x} - \mathbf{z}) \) to be a kernel function is \( f(0) > 0 \) and Fourier transform is

\[
\hat{F}[K](\omega) = (2\pi)^{-d/2} \int_{\mathbb{R}^d} \exp(-i\omega \cdot \mathbf{x}) K(\mathbf{x}) \, d\mathbf{x} \geq 0.
\] (6)

Like SVM, a function is an allowed KELM kernel function as long as it satisfies Mercer’s condition.
Mercer Theorem [18]. Assume that, for $X \subset \mathbb{R}^n$, $K(x, z)$ is a continuous symmetric real value function on $X \times X$ such that the following integration should always be nonnegative for every $f \in L_2(X)$:

$$
\iint_{X \times X} K(x, z) f(x) f(z) dx dz \geq 0.
$$

Then $K(x, z)$ must be a valid kernel function.

3. Triangular Hermite Kernel Extreme Learning Machine

3.1. Construction of Triangular Kernel Function. Laplace kernel function $K(x, z) = \exp(-\|x - z\|/\sigma)$ also is a radial basis function. Its classification performance is nearly equivalent to the Gaussian kernel $K_{RBF}(x, z) = \exp(-\|x - z\|^2/2\sigma^2)$, but it is less sensitive for the changes of parameter $\sigma$. The Laplace kernel can be used as an alternative when using the Gaussian becomes too expensive.

While $x \to 0$, using the Taylor formula, Laplace kernel can be simplified as

$$
K_{\text{mac}}(x, z) = \left(1 - \frac{\|x - z\|}{\sigma}\right) \times e\left(1 - \frac{\|x - z\|}{\sigma}\right). 
$$

Figure 1 shows the function curve of (7) and $K_{RBF}(x, z)$, when $\sigma = 1, 2$, where $x \in [-7, 7]$ and $z = 0$.

Seen from Figure 1, two types of kernel functions are quite different, and $K_{\text{mac}}(x, z)$ is relatively less sensitive to changes of the parameter $\sigma$. A typical triangle is presented on its function curve; thus, it is called triangular kernel function [19].

Given a sample set $X = \{\bar{x}\}_{i=1}^N$, where $\bar{x}$ is the sample mean and $N$ is the number of training samples, in order to further simplify (7), let $\sigma_0$ [16] be twice as long as the maximum distance of all sample points to sample mean such that formula holds with probability one. Thus, the simplified triangular kernel function is obtained:

$$
K_{\text{Tri}}(x, z) = 1 - \frac{\|x - z\|}{\sigma_0},
$$

where $\sigma_0 = 2 \times \max \{\|x - \bar{x}\| \leq r, x \in X, \bar{x} = (1/N) \sum_{i=1}^{N} x_i\}$.

Equation (8) is a translation invariant function; according to Theorem 1, $K_{\text{Tri}}(x, z)$ is strictly proved to be a valid KELM kernel function in the following proof.

Proof. $K(x) = f(x) = 1 - \|x\|/\sigma_0$; it is well known that $f(x)$ is an integrable bounded continuous function on $\mathbb{R}^n$ by function analysis and $f(0) = 1 > 0$ such that the Fourier transform of $K(x)$ is

$$
F[K](\omega) = (2\pi)^{-n/2} \int_{\mathbb{R}^n} \exp(-i\omega x) K(x) dx
$$

$$
= (2\pi)^{-n/2} \int_{\mathbb{R}^n} \exp(-i\omega x) \left(1 - \frac{\|x\|}{\sigma_0}\right) dx = (2\pi)^{-n/2}
$$

$$
\cdot \left[\int_{-\infty}^{\infty} \exp(-i\omega x) \times 1 dx - \frac{1}{\sigma_0} \int_{-\infty}^{\infty} \exp(-i\omega x) \times \|x\| dx \right] = (2\pi)^{-n/2}
$$

$$
\cdot \left[2\pi \delta(\omega) + \frac{2}{\sigma_0 \omega^2}\right] \geq 0.
$$

The proof is completed. □

References [12–16] constructed the weighting function as $K_{\text{Gau}}(x, z) = \exp(-\|x - z\|^2/d)$, where $d$ denotes the dimension of vector $x, z$, in which since the parameter $\sigma$ of Gaussian kernel $K_{RBF}$ is directly set with $\sqrt{\delta/2}$, the data structure information is lost although the parameter optimization is simplified. However, the setting of parameter $\sigma$ in triangular kernel function proposed in this paper just makes up for the shortcoming of it.

In order to reflect the difference more intuitively between $K_{\text{Gau}}(x, z)$ and $K_{\text{Tri}}(x, z)$, Figures 2 and 3 show the graph of the above two kernel functions, where $x \in [-0.5, 0.5], [-1, 1], [-1.5, 1.5], \text{and} [-2, 2]$, $z = 0, 0.2, 0.4$.

As is shown in Figures 2 and 3, both the vectors $x$ and $z$ are one-dimensional ($d = 1$). When $z$ takes a fixed value, the graph of kernel function is invariably constant, but the graph of kernel function changes as the value interval of $x$ changes. It is well known that the choice of different kernel functions is to select different criteria to measure the similarity and the degree of similarity [18].

Consequently, for the same point $(x, z)$, the similarity in different intervals should be different, and the value of $K_{\text{Tri}}(x, z)$ increases as the expansion of the interval, while the value of $K_{\text{Gau}}(x, z)$ in the four intervals always remains changeless. Therefore, it can be said that the parameter $\sigma_0$

Hermite polynomial [16] is a kind of orthogonal polynomials with respect to the weighting function $e^{-x^2}$ between the intervals $(-\infty, +\infty)$, which is defined as

\[
H_n(x) = (-1)^n e^{-x^2} \frac{d^n}{dx^n} (e^{x^2})
\]

\[
= \sum_{k=0}^{n/2} \left(-1\right)^k \frac{n!}{k! (n-2k)!} (2x)^{n-2k},
\]

\[n = 0, 1, 2, \ldots\]

It satisfies the orthogonal relationship:

\[
\int_{-\infty}^{\infty} e^{-x^2} H_m(x) H_n(x) \, dx = \begin{cases} 0, & m \neq n, \\ 2^{n+1} n! \sqrt{\pi}, & m = n. \end{cases}
\]

It has a recursive relationship:

\[
H_0(x) = 1,
\]

\[
H_1(x) = 2x,
\]

\[
H_{n+1}(x) = 2x H_n(x) - 2n H_{n-1}(x), \quad n = 1, 2, \ldots
\]

Owing to the orthogonality, variability, and universal approximation function capability of Hermite polynomial, a general Hermite kernel function can be constructed as a good alternative to other common kernel functions (Gaussian kernel, polynomial kernel, etc.). For this purpose, let the scalar variable $x$ be instead of row vector $x$ and $x^{n+1}$ be substituted as follows correspondingly:

\[
x^{n+1} \rightarrow \begin{cases} \left(xx^T\right)^{(n+1)/2}, & n = 2k + 1, \\ \left(xx^T\right)^{n/2} x, & n = 2k, \end{cases}
\]

\[k = 0, 1, 2, \ldots,
\]

where $x^T$ is the transpose of $x^{n+1}$.

Therefore, for vector input, it can define the generalized Hermite polynomial as

\[
H_0(x) = 1,
\]

\[
H_1(x) = 2x,
\]

\[
H_{n+1}(x) = 2x H_n(x) - 2n H_{n-1}(x), \quad n = 1, 2, \ldots
\]

By using generalized Hermite polynomial, this paper defines generalized $n$th order Hermite Dirichlet kernel [12] as

\[
K_{Hem}(x, z) = \sum_{i=1}^{n} H_i(x) H_i^T(z).
\]

It can evaluate and verify the Mercer Theorem for $K_{Hem}(x, z)$ as follows by assuming that each element is independent from others:

\[
\int_{X \times X} \int_{Z} K(X, Z) f(x) f(z) \, dx \, dz = \int_{X \times X} \left[ \sum_{i=1}^{n} H_i(x) H_i^T(z) \right] f(x) f(z) \, dx \, dz
\]

\[= \sum_{i=1}^{n} \int_{X \times Z} H_i(x) H_i^T(z) f(x) f(z) \, dx \, dz
\]
\[ K_{\text{Tri-H}}(\mathbf{x}, \mathbf{z}) = \left( 1 - \frac{\|\mathbf{x} - \mathbf{z}\|}{\sigma_0} \right) \sum_{k=1}^{n} H_k(\mathbf{x}) H_k^T(\mathbf{z}). \]  

(17)

Generalized Triangular Hermite kernel combines the advantages of triangular kernel and generalized Hermite Dirichlet kernel, which not only retains more distance similarity information of sample data, but also just chooses natural numbers to its parameter. Accordingly, it can greatly shorten the time of parameter optimization. Although it has two kernel parameters, that can be determined quickly and easily, which greatly reduces parameter optimization cost.

The Generalized Triangular Hermite kernel function up to the 3rd order is listed as in Table 1.

Figures 4 and 5 show the Generalized Triangular Hermite kernel output up to the 3rd order at two different coordinate scales of vertical axis, where changes are within the range of \([-2, 2]\) and \([0, 2]\) and are fixed at a constant value. Figure 4 shows the kernel function, while Figure 5 shows the value, where the 0th and 1st orders correspond to the left vertical axis, and the 2nd and 3rd orders correspond to the right vertical axis.

Finally, the kernel \(K_{\text{Tri-H}}(\mathbf{x}, \mathbf{z})\) to KELM algorithm is introduced; as a result, the Generalized Triangular Hermite kernel extreme learning machine algorithm is obtained as follows.

Given a training set \(\mathcal{N} = \{(\mathbf{x}_i, t_i) | \mathbf{x}_i \in \mathbb{R}^d, t_i \in \mathbb{R}^m, i = 1, \ldots, N\}\), the output function is

\[ f(\mathbf{x}) = \left[ K(\mathbf{x}, \mathbf{x}_1), \ldots, K(\mathbf{x}, \mathbf{x}_N) \right]^T \left( \frac{1}{\lambda} + \Omega_{\text{ELM}} \right)^{-1} T, \]  

(18)

where \(K(\mathbf{x}, \mathbf{x}_j) = (1 - \|\mathbf{x}_i - \mathbf{x}_j\|/\sigma_0) \sum_{k=1}^{n} H_k(\mathbf{x}_i) H_k^T(\mathbf{x}_j)\).

4. Experiments and Analysis

In order to test the performance of Generalized Triangular Hermite kernel extreme learning machine (Tri-H KELM) algorithms, this section compares it concerning testing accuracy, training time, and regression determination coefficient with other various algorithms in bispiral dataset and real-world benchmark regression, binary, multiclass classification datasets. Table 2 lists the various algorithms used in the experiments and the range of corresponding kernel parameter value, which also includes Gaussian kernel (Gauss), polynomial kernel (Poly), Gaussian Hermite kernel (Gau-H) [15] extreme learning machine algorithms, and triangular Hermite kernel support vector machine algorithm (Tri-H SVM). In multiclass simulations, we use the LIBSVM toolbox available and train SVM for each class separately as one versus all. The SVM cost parameter value is 100 in each corresponding experiment. Better test results are given in boldface in Tables 2–9.
Table 1: List of the Generalized Triangular Hermite kernel function up to the 3rd order.

<table>
<thead>
<tr>
<th>n</th>
<th>( K_{\text{Tri}}(x,z) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>( 1 \cdot (1 -</td>
</tr>
<tr>
<td>1</td>
<td>( (1 + 4xz^T) \cdot (1 -</td>
</tr>
<tr>
<td>2</td>
<td>( [1 + 4 \cdot (2</td>
</tr>
<tr>
<td>3</td>
<td>( [1 + 4(2</td>
</tr>
</tbody>
</table>

Table 2: The various algorithms and the range of their kernel parameters.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Kernel parameter</th>
<th>Range</th>
<th>Stepsize</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gauss KELM</td>
<td>( \sigma )</td>
<td>1–10</td>
<td>0.2</td>
</tr>
<tr>
<td>Poly KELM</td>
<td>( n )</td>
<td>0–3</td>
<td>1</td>
</tr>
<tr>
<td>Gau-H KELM</td>
<td>( n )</td>
<td>0–3</td>
<td>1</td>
</tr>
<tr>
<td>Tri-H KELM</td>
<td>( n )</td>
<td>0–3</td>
<td>1</td>
</tr>
<tr>
<td>Tri-H SVM</td>
<td>( n )</td>
<td>0–3</td>
<td>1</td>
</tr>
</tbody>
</table>

Table 3: The maximum testing rate (%) and the corresponding kernel parameter value.

<table>
<thead>
<tr>
<th>KELM algorithm</th>
<th>Maximum rate and optimization time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gauss KELM</td>
<td>92.5 (( T = 7.2384 ) s)</td>
</tr>
<tr>
<td>Poly KELM</td>
<td>91.75 (( T = 2.6988 ) s)</td>
</tr>
<tr>
<td>Gau-H KELM</td>
<td>92.25 (( T = 0.9360 ) s)</td>
</tr>
<tr>
<td>Tri-H KELM</td>
<td>93 (( T = 0.8424 ) s)</td>
</tr>
</tbody>
</table>

Table 4: Specifications of binary classification cases.

<table>
<thead>
<tr>
<th>Datasets</th>
<th>Training data</th>
<th>Testing data</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Liver</td>
<td>230</td>
<td>115</td>
<td>6</td>
</tr>
<tr>
<td>Australian</td>
<td>460</td>
<td>230</td>
<td>6</td>
</tr>
<tr>
<td>Breast-cancer</td>
<td>200</td>
<td>77</td>
<td>9</td>
</tr>
<tr>
<td>Wpbc</td>
<td>128</td>
<td>70</td>
<td>33</td>
</tr>
<tr>
<td>Sonar</td>
<td>150</td>
<td>58</td>
<td>60</td>
</tr>
</tbody>
</table>

Table 5: Specifications of multiclass classification cases.

<table>
<thead>
<tr>
<th>Datasets</th>
<th>Training data</th>
<th>Testing data</th>
<th>Attributes</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Iris</td>
<td>100</td>
<td>50</td>
<td>4</td>
<td>3</td>
</tr>
<tr>
<td>Glass</td>
<td>142</td>
<td>72</td>
<td>6</td>
<td></td>
</tr>
<tr>
<td>Wine</td>
<td>118</td>
<td>60</td>
<td>9</td>
<td>6</td>
</tr>
<tr>
<td>Vehicle</td>
<td>564</td>
<td>282</td>
<td>18</td>
<td>4</td>
</tr>
<tr>
<td>Segment</td>
<td>1540</td>
<td>770</td>
<td>18</td>
<td>7</td>
</tr>
<tr>
<td>Waveform</td>
<td>800</td>
<td>400</td>
<td>21</td>
<td>3</td>
</tr>
</tbody>
</table>

4.1. Classification Performance Comparison on Banana Dataset. The Banana dataset is a well-known binary class dataset from the UCI benchmark repository used in many pattern recognition tests. Each vector of the dataset has 2 features. The training set consists of 800 randomly selected samples (400 for (+1) class and 400 for (-1) class); meanwhile, 400 samples are randomly chosen as a testing set, which contains 200 samples of each class. At each trial of simulation, regularization coefficient is assumed. The simulation results, the maximum testing accuracy, and the corresponding kernel parameter optimization time are given in Table 3.

Figures 6–9 show the boundaries of different KELM algorithms on benchmark Banana dataset. Seen from Table 3 and Figures 6–9, Tri-H KELM achieves the highest testing accuracy in Banana dataset compared with Gauss, Poly, and Gau-H KELM. Furthermore, for the process of parameter optimization, the kernel parameter of two types of Hermite KELM algorithms is only taken from 0 to 3, which greatly shortens the time.
Table 6: Specifications of multiclass classification cases.

<table>
<thead>
<tr>
<th>Datasets</th>
<th>Gauss KELM</th>
<th>Poly KELM</th>
<th>Gau-H KELM</th>
<th>Tri-H, KELM</th>
<th>Tri-H SVM</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Testing rate (%)</td>
<td>Training time (s)</td>
<td>Testing rate (%)</td>
<td>Training time (s)</td>
<td>Testing rate (%)</td>
</tr>
<tr>
<td>Liver (C = 1)</td>
<td>60 (σ = 7.4)</td>
<td>0.0037</td>
<td>72.17 (n = 3)</td>
<td>0.0062</td>
<td>59.13 (n = 0)</td>
</tr>
<tr>
<td>Breast cancer (C = 0.1)</td>
<td>81.82 (σ = 4.2)</td>
<td>0.0016</td>
<td>76.62 (n = 1)</td>
<td>0.0058</td>
<td>79.22 (n = 0)</td>
</tr>
<tr>
<td>Wpbc (C = 0.1)</td>
<td>77.14 (σ = 0.4)</td>
<td>0.0126</td>
<td>77.74 (n = 2)</td>
<td>0.003</td>
<td>80 (n = 1)</td>
</tr>
<tr>
<td>Sonar (C = 1)</td>
<td>84.48 (σ = 0.6)</td>
<td>0.001</td>
<td>84.48 (n = 3)</td>
<td>0.0024</td>
<td>81.03 (n = 2)</td>
</tr>
</tbody>
</table>
Table 7: Performance comparison of different algorithms in multiclass datasets.

<table>
<thead>
<tr>
<th>Datasets</th>
<th>Gauss KELM</th>
<th>Poly KELM</th>
<th>Gau-H KELM</th>
<th>Tri-H, KELM</th>
<th>Tri-H SVM</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Testing rate (%)</td>
<td>Training time (s)</td>
<td>Testing rate (%)</td>
<td>Training time (s)</td>
<td>Testing rate (%)</td>
</tr>
<tr>
<td>Iris (C = 0.1)</td>
<td>98 (σ = 2)</td>
<td>0.0005</td>
<td>97 (n = 4)</td>
<td>0.0012</td>
<td>98 (n = 0)</td>
</tr>
<tr>
<td>Glass (C = 1)</td>
<td>77.78 (σ = 0.4)</td>
<td>0.0026</td>
<td>72.22 (n = 3)</td>
<td>0.004</td>
<td>73.64 (n = 0)</td>
</tr>
<tr>
<td>Wine (C = 100)</td>
<td>80 (σ = 4)</td>
<td>0.0012</td>
<td>96.67 (n = 1)</td>
<td>0.0046</td>
<td>97 (n = 0)</td>
</tr>
<tr>
<td>Vehicle (C = 10)</td>
<td>80.5 (σ = 1.4)</td>
<td>0.0163</td>
<td>81.21 (n = 2)</td>
<td>0.0076</td>
<td>82.62 (n = 2)</td>
</tr>
<tr>
<td>Segment (C = 10)</td>
<td>87.4 (σ = 7)</td>
<td>0.2002</td>
<td>88.44 (n = 2)</td>
<td>0.0834</td>
<td>87.27 (n = 0)</td>
</tr>
<tr>
<td>Waveform (C = 1)</td>
<td>86.75 (σ = 10)</td>
<td>0.0361</td>
<td>84.25 (n = 1)</td>
<td>0.0254</td>
<td>86.25 (n = 0)</td>
</tr>
</tbody>
</table>
Table 8: Specifications of binary classification cases.

<table>
<thead>
<tr>
<th>Datasets</th>
<th>Training data</th>
<th>Testing data</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cloud</td>
<td>682</td>
<td>342</td>
<td>9</td>
</tr>
<tr>
<td>Autoprice</td>
<td>106</td>
<td>53</td>
<td>15</td>
</tr>
<tr>
<td>Housing</td>
<td>337</td>
<td>169</td>
<td>13</td>
</tr>
<tr>
<td>Body fat</td>
<td>168</td>
<td>84</td>
<td>14</td>
</tr>
<tr>
<td>NIR</td>
<td>50</td>
<td>10</td>
<td>401</td>
</tr>
</tbody>
</table>

Figure 8: The boundaries of Gau-H KELM algorithms in Banana dataset.

4.2. Classification Performance Comparison on UCI Benchmark Datasets. In order to extensively verify the performance of different algorithms, wide types of datasets have been tested in simulations, which include 5 binary classification cases and 6 multiclass classification cases. Most of the datasets are taken from UCI Machine Learning Repository. The specifications of binary classification and multiclass classification cases are listed in Tables 4 and 5. Comparing Tri-H KELM with Gauss, Poly, and Gau-H KELM and Tri-H SVM, experiment results, including the maximum testing accuracy and corresponding kernel parameter value, and the training time are given in Tables 6 and 7.

1) Classification Performance Comparison on Binary Classification Cases. As demonstrated by the simulation results, Tri-H KELM achieves much better generalization performance in binary classification cases than other KELM algorithms (Gauss, Poly, and Gau-H KELM). More importantly, Tri-H KELM has achieved the highest classification accuracy just within a smaller range of integer numbers of kernel parameter, which greatly reduces the parameter optimization time. Besides, it is observed that Tri-H KELM can always achieve comparable performance with Tri-H SVM at much faster learning speed.

2) Classification Performance Comparison on Multiclass Classification Cases. The experiment results infer that Tri-H KELM performs better than other KELM algorithms and Tri-H SVM on average for multiclass classification cases. Its biggest advantage is that the parameter selection time is short, because the best classification performance is achieved with given kernel parameters. A further study reveals that all KELM algorithms have better scalability and run at much faster learning speed than traditional SVM.

4.3. Regression Performance Comparison on UCI Benchmark Datasets. This subsection selects 5 regression cases from UCI Machine Learning Repository; data is described in Table 8. The simulations of different algorithms on all the regression datasets were carried out. The regression performance of ELM was evaluated by the coefficient of determination $R^2$ which is defined within the interval of $[0, 1]$ and is closer to 1. The simulations results ($R^2$ and the training time) are given in Table 9.

As Table 9 shows, in comparison to other several KELM algorithms, Tri-H KELM obtains the maximum coefficient of determination in most regression datasets, and it has all achieved the maximum with respect to kernel parameter. It has better generalization performance for regression problem. Note that Tri-H KELM achieves similar regression performance as SVM at much faster learning speeds.

5. Conclusion

In this work, a novel extreme learning machine based on the mixed kernel function of triangular kernel and generalized Hermite Dirichlet kernel (Tri-H KELM) has been put forward, which introduces the triangular Hermite kernel function to kernel extreme learning machine algorithm. Because the presented kernel has only one parameter chosen from a small set of integers, the parameter optimization is facilitated greatly. Besides, more structure information of sample data is retained in the proposed kernel. Numerical experiments have been performed with different algorithms (Tri-H SVM, Gauss, Poly, Gau-H, and Tri-H KELM) on bisprial benchmark dataset and a number of real-world benchmark datasets and their results have been compared with Tri-H SVM and Gauss, Poly, and Gau-KELM for regression and binary, multiclass classification. Comparable
Table 9: Performance comparison of different algorithms in regression datasets.

<table>
<thead>
<tr>
<th>Datasets</th>
<th>Gauss KELM</th>
<th>Training time (s)</th>
<th>Testing rate (%)</th>
<th>Poly KELM</th>
<th>Training time (s)</th>
<th>Testing rate (%)</th>
<th>Algorithms</th>
<th>Gau-H KELM</th>
<th>Training time (s)</th>
<th>Testing rate (%)</th>
<th>Tri-H, KELM</th>
<th>Training time (s)</th>
<th>Testing rate (%)</th>
<th>Tri-H SVM</th>
<th>Training time (s)</th>
<th>Testing rate (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cloud</td>
<td></td>
<td>0.0221</td>
<td>0.9646 (n = 1)</td>
<td>0.0184</td>
<td>0.9658 (n = 2)</td>
<td>0.0779</td>
<td></td>
<td>0.9799</td>
<td>(n = 2)</td>
<td>0.0364</td>
<td></td>
<td>0.9842 (n = 2)</td>
<td>0.0307</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>0.4168</td>
<td>(σ = 2)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>0.8251</td>
<td>(σ = 10)</td>
<td>0.0016</td>
<td>0.8661 (n = 2)</td>
<td>0.0018</td>
<td>0.9183 (n = 1)</td>
<td>0.0012</td>
<td></td>
<td>0.9194</td>
<td>(n = 1)</td>
<td>0.0011</td>
<td></td>
<td>0.9057 (n = 1)</td>
<td>0.0616</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>0.8661</td>
<td>(σ = 3)</td>
<td>0.0051</td>
<td>0.8994 (n = 3)</td>
<td>0.0134</td>
<td>0.9141 (n = 1)</td>
<td>0.007</td>
<td></td>
<td>0.9162</td>
<td>(n = 1)</td>
<td>0.0071</td>
<td></td>
<td>0.8969 (n = 1)</td>
<td>0.1914</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>0.8936</td>
<td>(σ = 5.6)</td>
<td>0.001</td>
<td>0.9791 (n = 3)</td>
<td>0.0034</td>
<td>0.9643 (n = 2)</td>
<td>0.0014</td>
<td></td>
<td>0.9865</td>
<td>(n = 1)</td>
<td>0.0013</td>
<td></td>
<td>0.9725 (n = 1)</td>
<td>0.0455</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>0.2784</td>
<td>(σ = 10)</td>
<td>0.0034</td>
<td>0.9371 (n = 1)</td>
<td>0.0049</td>
<td>0.9452 (n = 0)</td>
<td>0.0077</td>
<td></td>
<td>0.9614</td>
<td>(n = 0)</td>
<td>0.0022</td>
<td></td>
<td>0.9678 (n = 0)</td>
<td>0.0638</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Discrete Dynamics in Nature and Society
generalization and robustness performance of the proposed approach with the rest of the methods considered at a much faster learning speed than Tri-H SVM indicate its usefulness and effectiveness. Future work will be on the study of Tri-H ELM in its practical applications.

Competing Interests

The authors declare that they have no competing interests.

References


