A Novel Restricted Boltzmann Machine Training Algorithm With Dynamic Tempering Chains

XINYU LI, XIAOGUANG GAO, (Member, IEEE), AND CHENFENG WANG
School of Electronics and Information, Northwestern Polytechnical University, Xi’an 710129, China
Corresponding author: Xiaoguang Gao (cxg2012@nwpu.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 61573285, in part by the National Science Foundation for Young Scientists of China under Grant 62003267, and in part by the Natural Science Basic Research Program of Shaanxi under Grant 2020JQ-2020.

Abstract
Restricted Boltzmann machines (RBMs) are commonly used as pre-training methods for deep learning models. Contrastive divergence (CD) and parallel tempering (PT) are traditional training algorithms of RBMs. However, these two algorithms have shortcomings in processing high-dimensional and complex data. In particular, the number of temperature chains in PT has a significant impact on the training effect, and the PT algorithm cannot fully utilize parallel sampling from multiple temperature chains for the divergence of the algorithm. The training can quickly converge with fewer temperature chains, but this impacts the accuracy. More temperature chains can help PT achieve higher accuracy in theory, but severe divergence at the beginning of the training may ruin the training result. To exploit fully the advantages of PT and improve the ability of RBMs to process high-dimensional and complex models, this article proposes dynamic tempering chains (DTC). By dynamically changing the number of temperature chains during the training process, DTC starts training with fewer temperature chains and gradually increase the number of temperature chains with training going on, and finally get an accurate RBM. And one-step reconstruction error is proposed to measure the convergence, which can decrease the influence of the dynamic training strategy on reconstruction error. Experiments on MNIST, MNORB, Cifar 10, and Cifar 100 indicate that, compared with PT, the classification accuracy of DTC algorithm improved by up to 8%. DTC quickly converges in the early stage of training because of few exchanges among temperature chains and produces higher accuracy at the end for the global optimum model learned by more temperature chains, especially when learning high-dimensional and complex data. This proves that the DTC algorithm effectively utilizes parallel sampling of multiple temperature chains, overcomes divergence challenges, and further improves the training effect of the RBM.

Index Terms
Complex data, dynamic tempering chains, high-dimensional data, parallel tempering, restricted Boltzmann machine.

I. INTRODUCTION
The prototype of deep learning was proposed in 1967 [1]; however, the training problem was not solved until 2006 [2]. Subsequently, deep learning has gradually developed into the most widely used area of machine learning owing to its significant advantages over conventional machine learning models for processing complex data and constructing scale-free models. In recent years, deep learning has been widely used in tasks such as image recognition [3], speech recognition [4], and semantic analysis [5].

A restricted Boltzmann machine (RBM) [6] is an important basic model in deep learning. Multi-RBMs with different stacking strategies can contribute to a deep belief network (DBN) [7] and a deep Boltzmann machine (DBM) [8] as bipartite graphs. Compared to classic neural networks, the RBM-based deep learning model can easily complete network training by using a layer-by-layer greedy algorithm. RBM can also be used as a feature extracting method [9], and provide features of data to other models. Therefore, it can be used as a pre-training method for other deep learning models. The training accuracy of the RBM will directly impact the effectiveness of subsequent model training. However, the application range of the RBM is limited by the training algorithms. Training an RBM can be viewed as an optimization problem, whereby the minimization of a system’s energy is replaced with maximizing the posterior probability of activating...
a hidden neuron. This equivalence makes training more concise.

RBM were first trained using the Markov chain Monte Carlo (MCMC) method [10], [11]. However, they were not widely used at that time owing to the low training efficiency. RBMs can be trained much faster than before with contrastive divergence (CD) [12], persistent contrastive divergence (PCD) [13], or fast persistent contrastive divergence (FPD) [14], which has resulted in RBMs attracting widespread attention in the deep learning field. Parallel tempering (PT) [15]–[17] was subsequently proposed as a promising training algorithm. In recent years, dynamic Gibbs sampling (DGS) [18] and gradient fixing parallel tempering (GFPT) [19] have also been proposed as improvements to the RBM training algorithm.

While PT is a promising algorithm, it is more suited for lower-dimensional problems [20] and has low efficiency in training high-dimensional and complex data. To overcome the drawback of PT, this article proposes the dynamic tempering chains (DTC) algorithm. The proposed algorithm dynamically changes the number of tempering chains in the training process to fit the energy of the RBM. DTC converges quickly in learning high-dimensional and complex data, which leads to high training accuracy compared to the state-of-the-art algorithms. Additionally, this article proposes a new indicator to measure convergence, a one-step reconstruction error, which improves upon the classical reconstruction error. A one-step reconstruction error reduces the impact of various sampling techniques in different algorithms, especially for dynamic training strategy algorithms.

This article continues with a brief introduction to the theoretical background of RBMs (Section II) and an analysis of the conventional CD and PT training algorithms (Section III). Based on this analysis, the DTC algorithm is proposed (Section IV) and compared with the latest conventional algorithms, utilizing several indicators to analyze the performance (Section V). Finally, a summary of the work on the proposed algorithm is presented in the conclusion (Section VI).

The contributions of this article have been summarized as follows. 1) A novel algorithm is proposed to deal with high-dimensional and complex data in RBM training. 2) Dynamically changing the number of temperature chains during the training process result in rapid convergence in the early stage of training and produce higher accuracy at the end. 3) A new indicator is proposed to measure the convergence, which improves upon the classical reconstruction error.

II. THEORETICAL BACKGROUND

The first machine learning model with a similar structure to the RBM was named “Harmonium” [6] in 1986 and renamed by Geoffrey Hinton in the mid-2000s [21]. RBMs are energy-based stochastic neural networks composed of two layers of neurons. A typical RBM comprises a visible layer $v$ with $m$ units and a hidden layer $h$ with $n$ units. The weights between the two layers are saved in a real-valued matrix $W_{m \times n}$, in which the weights between the visible unit $v_i$ and hidden unit $h_j$ are saved as $w_{ij}$. Every unit in the hidden layer connects to all visible units, so each visible unit connects to all hidden units. However, the units in one layer do not connect to each other. The architecture of an RBM is depicted in Fig. 1.

![Architecture of an RBM](image)

Typically, both $v$ and $h$ are binary-valued units, which means $v \in \{0, 1\}^m$ and $h \in \{0, 1\}^n$. The biases of visible and hidden units are $a$ and $b$, respectively. The collection of these parameters is $\theta$, thus $\theta = \{a, b, W_{m \times n}\}$. As an energy-based model, the energy function of an RBM is defined as follows:

$$E(v, h) = -\sum_{i=1}^{m} a_i v_i - \sum_{j=1}^{n} b_j h_j - \sum_{i=1}^{m} \sum_{j=1}^{n} v_i h_j w_{ij},$$

where $v \in v$, $h \in h$, and $Z = \sum_{v, h} e^{-E(v, h)}$. $Z$ is called the partition function. When the training data or the state of a hidden layer is provided, the conditional probabilities are defined, respectively, as follows:

$$P(v_i = 1|h) = \phi \left( \sum_{j=1}^{n} w_{ij} h_j + a_i \right)$$

and

$$P(h_j = 1|v) = \phi \left( \sum_{i=1}^{m} w_{ij} v_i + b_j \right).$$

In (3) and (4), $\phi(x)$ represents the logistic-sigmoid function:

$$\phi(x) = \frac{1}{1 + e^{-x}}.$$

As previously mentioned, the training process of an RBM is the process of minimizing the system energy, i.e., finding the optimal $\theta$ to minimize $E(v, h)$. However, it is difficult to find the minimum directly. Instead, the parameters $v$ of the RBM can be modeled based on the distribution of underlying observed data. Hence, the parameter $\theta$ of the RBM can be learned from the training data $S = \{x_1, \cdots, x_n\}$ by using
maximum likelihood estimation. In practice, this training corresponds to performing stochastic gradient ascent on the log-likelihood of parameter $\theta$ for the training data.

There are three parameters in $\theta$. The $a$, $b$ and $W_{mxn}$ gradients are calculated separately with training sample $v$,
respectively, as follows:

$$\frac{\partial \ln P(v)}{\partial w_{ij}} = P(h_i = 1 | v) v_i - \sum_v P(v)P(h_i = 1 | v)v_j, \quad (6)$$

$$\frac{\partial \ln P(v)}{\partial a_i} = v_i - \sum_v P(v)v_i, \quad (7)$$

and

$$\frac{\partial \ln P(v)}{\partial b_j} = P(h_j = 1 | v) - \sum_v P(v)P(h_j = 1 | v). \quad (8)$$

It is usually difficult to determine this gradient analytically because the expectation under this distribution $P(v)$ is unknown and cannot be efficiently computed. As a result, it is difficult to train the RBM at the beginning.

III. RELATED WORK

The training algorithms of the RBM have been developed over a long period. The MCMC method [10], [11] was originally the only way to train RBMs, but it required numerous state transfer steps to ensure that the collected samples conformed to the target distribution, incurring a heavy computational burden.

The state of MCMC sampling can start from training samples because the goal is to fit the RBM to the distribution of training samples. Thus, the expectations are approximated by samples drawn from the corresponding distributions instead of a random state in MCMC. This sampling method is called Gibbs sampling. The basic idea is to update each variable, based on the conditional distribution of other given states, and then construct a Markov chain. In this manner, only a few state transitions are needed to approach the stationary distribution. Additional Gibbs sampling steps will also produce higher sampling and training accuracy. A CD training algorithm with Gibbs sampling [12] can learn the parameters of an RBM much faster than MCMC and greatly improve efficiency. Advanced CD algorithms such as PCD [13] and FPCD [14] improve the training speed and accuracy to an extent. DGS [18] is an advanced CD that dynamically increases the Gibbs sampling steps to accelerate the training process and improve the final training accuracy. Additionally, a dynamic learning rate [22] has been proposed to improve the RBM training.

CD significantly improves training efficiency; however, CD and DGS may fall into the local optimum with a single Gibbs sampling chain. To overcome this problem, Cho et al. [17] proposed PT in 2010. The PT sampling method is an improved variant of MCMC sampling that utilizes multiple Gibbs sampling chains with varying levels of temperatures. The temperature denotes the level of energy of the overall system. When the temperature is high, the samples collected by Gibbs sampling are more likely to move freely. PT is more likely than CD to produce the global optimum statement and higher learning accuracy. The Gibbs sampling component is the same in PT and CD, which means the sampling is a biased estimate. The accuracy of PT can be further improved by changing its sampling method; hence, GFPT [19] was proposed to reduce the estimated bias. Yin et al. [23] attempted to add a reconstruction error to the cost function to make the training more effective. Similarly, modified objective function is proposed to restrict the free energy value of the training data and reduce the model complexity [24]. Recently, Manukian et al. [25] proposed mode-assisted training, which combines a standard gradient update with an off-gradient direction, promoting faster training and stability. In addition to the advancement of training algorithms, there are also some innovations in the structure of RBM. Marc-Alexandre et al. proposed ordered Restricted Boltzmann Machine and they further proposed Infinite Restricted Boltzmann Machine [26], which obviates the need to specify the hidden layer size. And Graph regularized Restricted Boltzmann Machine is proposed by Dongdong Chen et al. [27]. The sparse and discriminative representations learned by this model can reflect data distributions while simultaneously preserving the local manifold structure of data. Saeed Pirmoradi et al. improved RBM further and proposed Self-Organizing Restricted Boltzmann Machine [28]. In this model, SCM algorithm is utilized to estimate the size of hidden layer in RBM. Another improved RBM model called Online RBM is proposed by Ramasamy Savitha et al. [29], this model can begin with a single neuron in hidden layer, progressively adds and suitably adapts the network to account for the variations in streaming data. Some improved RBMs are also exploited to model documents. Diversifying Restricted Boltzmann Machine [30] with diversified hidden units proposed by Pengtao Xie et al. can model the long-tail region in documents better. With these improved RBMs and training algorithms proposed, the application range of RBM is greatly expanded. More and more researches focus on how to simplify the RBM structure and accomplish the same work in a smaller size network. Indeed, the training algorithms applied in training these improved RBMs have little difference from conventional ones, the most commonly used algorithms are still CD.

CD is widely used to train RBMs because of its fast training speed. However, owing to the use of a single Gibbs sampling chain, falling into a local optimum can occur and the training accuracy is not ideal when processing high-dimensional and complex data. This defect also applies to other algorithms based on CD. In contrast, PT’s training accuracy is theoretically better than that of CD when using multiple Gibbs sampling chains at different temperatures. PT sampling with multi-temperature chains normally starts from a high energy state, where more state can transit. Thus, there are more states that PT can transit to at the beginning of the training. Additionally, multi-temperature chains create more uncertainty, which causes poor convergence at the beginning of PT. When using fewer temperature chains, PT can quickly
converge at the beginning. However, when the RBM energy drops to a low level, fewer temperature chains result in lower accuracy because there is less chance to jump out of the local optimum. In some cases, PT may develop a large divergence at the beginning and converge quite quickly for its exchange between these temperature chains, which can easily lead to overfitting. Using multi-temperature chains for PT sampling also takes more time than CD.

The main difference between PT and CD is that the temperature is introduced into the Gibbs sampling chain in PT, such that different temperature sampling chains exhibit different characteristics. In this case, the stability of the Markov chain with temperature \( T_r \) is as follows:

\[
P_r(v, h) = \frac{1}{Z_r} e^{-\frac{1}{T_r} E(v,h)}. \quad (9)
\]

The purpose of Gibbs sampling is to make the current state gradually approach the stable distribution. The higher the temperature \( T_r (T_r \geq 1) \), the closer the probability densities of the Gibbs distribution between the states. Conversely, the lower the temperature, the larger the difference in the probability densities between the states. In Fig. 2, the state distributions between normal and high temperature are different, which means,

\[
T_r = 1 \Rightarrow p_r(v, h) = \frac{1}{Z_r} e^{-E(v,h)} \quad (10)
\]

and

\[
T_r \rightarrow \infty \Rightarrow p_r(v, h) \rightarrow 0. \quad (11)
\]

Temperature is defined as \( 1 = T_1 < T_2 < \cdots < T_M \), where \( M \) denotes the number of tempering chains. In each step of the PT process, sampling of different temperature chains is required to generate samples \((v_1, h_1), \cdots, (v_M, h_M)\). Then, two neighboring Gibbs chains will exchange their samples \((v_r, h_r)\) and \((v_{r-1}, h_{r-1})\) with the Metropolis probability as in the following equation:

\[
\min \left\{ 1, \exp \left( \frac{1}{T_r} - \frac{1}{T_{r-1}} \right) (E(v_r, h_r) - E(v_{r-1}, h_{r-1})) \right\}. \quad (12)
\]

In the early stage of training, the selected training initial state is often distant from the target distribution (stationary distribution) for random initialization. The exchanges between the tempering chains of PT start from the high temperature chain, in which \( p_r(v, h) \rightarrow 0 \). Therefore, the difference of probability densities between these high temperature states is quite small, which makes the sampling of the high temperature chains more random at the beginning of training. At the same time, the exchanges between temperature chains are carried out according to a certain probability, and obvious divergence may occur. Even GFPT, which uses an improved sampling method to reduce the sampling bias, cannot avoid the divergence caused by temperature chain exchanging. In the later stage of the training process, these temperature chains lead to the high mixing rate of PT. Therefore, it is easier to jump out of the local optimum, such as from \( B' \) to \( A' \) in Fig. 2, and attain a higher training accuracy. When learning high-dimensional and complex datasets, the divergence in the early training process has a greater influence on the result. A brief experiment is designed to show this phenomenon. We set different number of temperature chains in PT to train RBM. The data set, structure, and parameters of RBM in this experiment are as follows.

As the result shown in Fig. 3, more temperature chains led to poorer results and increased training time when training high-dimensional and complex data.

\[ \text{TABLE 1. Network Structures and Parameters.} \]

| Number of temperature chains | Structure | Learning rate \( \eta \) | Batch size |
|------------------------------|-----------|----------------|----------|
| 3                            | 1024x700  | 0.03           | 50       |
| 5                            | (1,1024,1,1111) | 0.03           | 50       |
| 10                           | (1,1024,1,1111,1,1765,1,2500) | 0.03           | 50       |
| 15                           | (1,1024,1,1111,1,1765,1,2500,1,3333,1,4286,1,5385,1,6667,1,8182) | 0.03           | 50       |
| 20                           | (1,1024,1,1111,1,1765,1,2500,1,3333,1,4286,1,5385,1,6667,1,8182,2,2222,2,5000,2,8571,3,3333,4,5,6,6666,1,10,20) | 0.03           | 50       |

IV. DYNAMIC TEMPERING CHAINS

To improve the convergence effect and accuracy of PT, a novel algorithm is proposed based on PT, called dynamic...
tempering chains (DTC), to deal with high-dimensional and complex data in RBM training. The divergence of PT will not greatly affect the result when learning simple datasets, but will lead to excessive calculations, slower divergence, and decreased training efficiency and accuracy when learning high-dimensional and complex datasets such as large and complex images. Sambridge [20] demonstrated that PT is promising for lower-dimensional problems. To give full consideration to the advantages of PT in training high-dimensional and complex datasets, the DTC is designed to reduce divergence in the early stages and maintain higher accuracy in the later stages of training. According to the analysis based on Fig. 3, it is necessary to reduce the impact of the high temperature chains to reduce the divergence of PT early in the process. Obviously, omitting the high temperature chains is the most direct method. In practice, the sampling results of the normal temperature chains can be gradually increased after the parameters of the RBM quickly converge in the early stage. Experiments show that as long as the reconstruction error of the RBM is significantly reduced, the timing of the increase in the number of temperature chains will have minimal impact on the results of the algorithm. Therefore, the number of temperature chains can be gradually increased. After the temperature chains increase, parameter updates are as follows:

\[
a_{i}^{T} = a_{i}^{T} + \eta(v_{r,i}^{(0)} - v_{r,j}^{(k)}) \\
= a_{i}^{T} + \eta \nabla a_{i}^{T}, \quad (13)\\
b_{j}^{T} = b_{j}^{T} + \eta[P_r(h_{r,j} = 1|v_{r}^{(0)}) - P_r(h_{r,j} = 1|v_{r}^{(k)})] \\
= b_{j}^{T} + \eta \nabla b_{j}^{T}, \quad (14)\\
\]

and

\[
w_{i,j}^{T} = w_{i,j}^{T} + \eta[P_r(h_{r,j} = 1|v_{r}^{(0)})v_{r,j}^{(0)} - P_r(h_{r,j} = 1|v_{r}^{(k)})v_{r,j}^{(k)}] \\
= w_{i,j}^{T} + \eta \nabla w_{i,j}^{T}. \quad (15)\\
\]

On the premise that the hidden and visible layer variables, \(h\) and \(v\), respectively, do not change as \(T_r\) increases, \(\nabla a_{i}^{T}, \nabla b_{j}^{T}\), and \(\nabla w_{i,j}^{T}\) will become smaller than the original \(\nabla a_{i}, \nabla b_{j}\), and \(\nabla w_{i,j}\), respectively. This will also reduce the updating speed of the parameter. Therefore, setting the number of tempering chains and temperatures too high will diverge the algorithm and reduce the training efficiency, especially for high-dimensional and complex data sets.

The DTC algorithm gradually increases the number of tempering chains in the middle of the training process to accelerate the training and reduce the divergence. Appropriately increasing the number of temperature chains ensures a fast convergence rate in the middle of training and prevents the RBM from falling into a local optimum. The number of temperature chains is maximized at the end of the training process when the algorithm is trained and well converged. This approach allows the proposed algorithm to use more temperature chains to improve accuracy further.

The proposed DTC is as follows:

\[
\textbf{Algorithm 1: Dynamic Tempering Chains Algorithm}\\
\textbf{Input:} \text{RBM } (V_1, \cdots, V_m, H_1, \cdots, H_n), \text{ training batch } S, \text{ current state } v_r \text{ of Markov chain with stationary distribution } p_r \text{ for } r = 1, \cdots, M \\
\textbf{Output:} \text{gradient approximation } \Delta w_{ij}, \Delta b_j \text{ and } \Delta c_i \text{ for } i = 1, \cdots, n, j = 1, \cdots, m \n1 \text{ Init } \Delta w_{ij} = \Delta b_j = \Delta c_i = 0 \text{ for } i = 1, \cdots, n, j = 1, \cdots, m \n2 \text{ For } \text{batch } = 1, \cdots, \text{batchnum} \n3 \text{ If batch } \leq m_1 \text{ M } = M_1 \n4 \text{ If batch } \leq m_2 \text{ M } = M_2 \n5 \text{ else } M = M_3 \n6 \text{ For all } v \in S \text{ do } \n7 \quad \text{For } r = 1, \cdots, M \text{ do } \n8 \quad \quad v_r^{(0)} \leftarrow v_r \n9 \quad \quad \text{For } i = 1, \cdots, n \text{ do sample } h_{r,i}^{(0)} \sim p(h_{r,i}|v_r^{(0)}) \n10 \quad \quad \text{For } t = 0, \cdots, k - 1 \text{ do } \n11 \quad \quad \quad \text{For } j = 1, \cdots, m \text{ do sample } v_r^{(t+1)} \sim p(v_{r,j}|h_{r,j}^{(t)}) \n12 \quad \quad \quad \quad h_{r,i}^{(t+1)} \sim p(h_{r,i}|v_r^{(t+1)}) \n13 \quad \quad v_r \leftarrow v_r^{(k)} \n14 \quad \text{For } r \in [s|2 \leq s \leq M \text{ and } s \mod 2 = 0] \text{ do } \n15 \quad \quad \text{swap } (v_r^{(k)}, h_{r,j}^{(k)}) \text{ and } (v_{r-1}^{(k)}, h_{r-1,j}^{(k)}) \text{ with probability given by (13)} \n16 \quad \text{For } r \in [s|3 \leq s \leq M \text{ and } s \mod 2 = 1] \text{ do } \n17 \quad \quad \text{swap } (v_r^{(k)}, h_{r,j}^{(k)}) \text{ and } (v_{r-1}^{(k)}, h_{r-1,j}^{(k)}) \text{ with probability given by (13)} \n18 \quad \text{For } i = 1, \cdots, n, j = 1, \cdots, m \text{ do } \n19 \quad \quad \Delta w_{ij} \leftarrow \Delta w_{ij} + p(H_i = 1|v_{r}^{(0)}) \cdot v_{r}^{(0)} - p(H_i = 1|v_{r}^{(k)}) \cdot v_{r}^{(k)} \n20 \quad \quad \Delta b_j \leftarrow \Delta b_j + v_{r}^{(0)} - v_{r}^{(k)} \n21 \quad \quad \Delta c_i \leftarrow \Delta c_i + p(H_i = 1|v_{r}^{(0)}) - p(H_i = 1|v_{r}^{(k)}) \n22 \text{ For } i = 1, \cdots, n \text{ do } \n23 \]

V. METHODOLOGY AND RESULTS
CD [12], PT [15]–[17], DGS [18], GFPT [19], and DTC were used to train an RBM in sequence in order to verify the proposed DTC. The performance of these algorithms is evaluated on three metrics: reconstruction error, classification accuracy, and training time.

A. BENCHMARK DATASETS AND NETWORK STRUCTURES
Four datasets were used in RBM training: MNIST, MNORB, Cifar10, and Cifar100 (all shown in Fig. 4).
MNIST, created by “re-mixing” the samples from NIST’s original datasets [31], consists of 60,000 handwritten number images. Each image is 28 × 28 pixels, and each pixel is white or black. The MNORB dataset is simplified and was obtained by binarizing the NORB dataset [32]. In the MNORB dataset, there are five categories of objects, and every image is 32 × 32 pixels. The Cifar10 [33] dataset contains 60,000 32 × 32 color images, which are divided into 10 categories. The Cifar100 [33] dataset is similar to the Cifar10 dataset but has up to 100 categories. Color images contain much more information than black-and-white images. Unlike the MNIST and MNORB datasets, each category in the Cifar10 and Cifar100 datasets contains more than one object, which increases the complexity. Therefore, Cifar datasets are high-dimensional and complex. By comparing algorithms using different datasets, the experiments can accurately reflect their performance.

The structures and parameters of the RBM are listed in Table 2. To ensure a fair comparison, the common parameters of each network for every dataset are the same for all five algorithms.

### TABLE 2. Network Structures and Parameters.

| Data set   | Structure       | Learning rate η | Batch size |
|------------|-----------------|-----------------|------------|
| MNIST      | 784×500         | 0.1             | 60         |
| MNORB      | 1024×700        | 0.03            | 50         |
| Cifar10    | 3072×2000       | 0.1             | 6          |
| Cifar100   | 3072×2000       | 0.1             | 6          |

The number of nodes in the visible and hidden layers is set separately for every dataset, fixing the structure of the RBM. For example, the RBM structure for the MNIST dataset is 784 × 500, which indicates that the visible layer has 784 nodes, whereas the hidden layer has 500 nodes. The number of nodes in the hidden layer can be chosen from a wide range and has little effect on the learning result in that range. The learning rate was introduced previously. The learning rates of these algorithms were carefully chosen to be the same value for the same parameter updating method. Brief experiments proved that the learning rate can influence the training result in limited learning iterations. A large learning rate will shut down the convergence too early, and a small learning rate will lead to slow convergence. However, the convergence tendency of these algorithms shows little difference in the same learning rate. The number of images trained in one iteration is called the batch size, which also has little influence on the training result.

### B. INDICATORS USED IN THE EXPERIMENTS

The experiments evaluated the five algorithms in terms of three metrics: reconstruction error, classification accuracy, and training time.

1) RECONSTRUCTION ERROR AND ONE-STEP RECONSTRUCTION ERROR

The reconstruction error is the mean square error between the original input data $v_0$ and the reconstruction data $v_k$.

$$e_r = (v_k - v_0)^2$$

The difference between the reconstruction and input data will decrease during training. This value is often used as an indicator in the training process because the computational complexity required to calculate this index is relatively small.

As shown in Fig. 5, the reconstruction error decreases rapidly at the beginning of the training process, the descent rate gradually decreases to zero, and then the error stabilizes at a certain value. Owing to the bias in the gradient estimation, fluctuation of the reconstruction error exists for the entire training process. The lower the reconstruction error, the better the RBM is trained. However, the difference between the original and reconstructed data does not reflect a poor training result. The reconstruction error will easily be larger for multi-step Gibbs sampling than for one-step Gibbs sampling, as shown in Fig. 6. The experimental results in Table 2 indicate that CD with more sampling steps will obtain higher accuracy and a better-trained RBM. Therefore, the reconstruction error is not a perfect indicator for training results [34]. Some dynamic strategy algorithms change the
sampling parameters or methods during the training process, after which the reconstruction error may sharply increase or decrease. However, this cannot correctly indicate the convergence of the training process. It is very difficult to produce an ideal value when training all the parameters in an RBM with a fixed learning rate and limited training iterations. The reconstruction error is an indicator between \( v_0 \) and \( v_k \), instead of an indicator for \( h \). Thus, it can only be regarded as an indicator of convergence tendency under certain conditions.

An RBM is typically used to extract features or as a basic module in deep learning. The data in the hidden layer \( h \) are regarded as feature or transferred to another module in deep learning. However, no ideal hidden layer data are available for comparison. This article proposes a new indicator called a one-step reconstruction error \( e_{os-er} \), which uses the mean square error between the original input data \( v_0 \) and the reconstruction data \( v_1 \) from \( h \). This more reasonable training indicator will help monitor the training process and improve the reconstruction error. The calculation of \( e_{os-er} \) is shown in (17).

\[
e_{os-er} = (v_1 - v_0)^2 \tag{17}
\]

This indicator can also reflect the bias between the reconstruction data by using the RBM itself and the original input instead of different sampling results of the algorithms. In other words, the one-step reconstruction error can measure the reconstruction bias of all the training algorithms using the same standard, making it more of a reference value. Moreover, the one-step reconstruction error can avoid the influence of the dynamic training strategy and better reflect the convergence tendency. However, neither the reconstruction error nor the one-step reconstruction error can reflect the training result of \( h \) directly. Therefore, a very low one-step reconstruction error does not indicate a good RBM.

2) CLASSIFICATION ACCURACY
Classification accuracy is an intuitive training indicator. RBM is often used as a feature extraction module, which means that the feature extraction performance can directly reflect the training effect of the RBM. As there is no direct method to measure the feature, a Softmax network layer is added after the RBM to complete the corresponding classification task according to the labels and data of the hidden layer \( h \). The classification accuracy can be calculated based on the classification results.

Fig. 5 and Table 3 indicate that although the reconstruction error of CD with more Gibbs sampling in one iteration is larger, the classification accuracy is higher. Compared with the reconstruction error, the classification accuracy can better reflect the effect of RBM training and aligns with the conclusions of theoretical analysis. This is because the reconstruction error and the training effect do not have a monotonic relationship. The calculation cost of the reconstruction error is relatively low and can reflect the training convergence, but it cannot indicate the training effect directly. Therefore, it can only be considered as an indicator during the training process. Conversely, the relationship between classification accuracy and the training effect is monotonic. When the classification accuracy is higher, the feature extraction and network training are more effective. Owing to high complexity, it is only calculated at the end of the training.

### TABLE 3. Classification Accuracy of CD With Different Sampling Steps for MNIST Dataset.

| Algorithm | Classification accuracy (%) |
|-----------|----------------------------|
| CD-1      | 86.28                      |
| CD-5      | 87.86                      |
| CD-10     | 88.34                      |
| CD-100    | 88.68                      |

3) TRAINING TIME
Training time refers to the time taken to complete the training, based on a stop condition. If the reconstruction error is used as the standard and the training is stopped after the reconstruction error decreases to a certain threshold, training may stop too early or too late because of the fluctuation of the reconstruction error. Using the reconstruction error as the stop condition cannot lead to a steady training effect, as there is no guarantee that the same training effect will be obtained every time. If classification accuracy is used as the stop indicator, the Softmax network layer needs to be trained after each step of training. This will greatly slow down the training process because the training time for the classifier is much longer than the time for one iteration of RBM training. Therefore, the experiments are designed to stop and compare the time spent after running the same number of iterations for each algorithm. This stop condition follows the principle of controlling variables, which can ensure the same training conditions. The comparative computational complexity of the algorithms is also analyzed.

C. EXPERIMENTAL ANALYSIS
This section presents the detailed experimental results and analysis of the five algorithms.
1) ANALYSIS OF RECONSTRUCTION ERROR AND ONE-STEP RECONSTRUCTION ERROR

As can be seen in Fig. 7, the RBM quickly converges for the relatively simple MNIST dataset. The reconstruction error of DGS has several increased intervals in the entire training period because of the dynamic sampling strategy, indicating poor convergence. In terms of the one-step reconstruction error, all algorithms converge well over the entire training process, including DGS. GFPT, with a gradient fixing sampling strategy, converges the fastest. In general, each algorithm is effective in training the MNIST dataset.

When training an RBM with the MNORB dataset, PT has the largest variance and slowest decrease in the reconstruction and one-step reconstruction errors, as shown in Fig. 8. This indicates that PT converges slowly in all the training algorithms. The sharp increase in the reconstruction error of DGS is weakened in the one-step reconstruction error, where the convergence of the algorithms can be better compared. MNORB is a more complex dataset and contains more information, so the differences between the reconstruction and one-step reconstruction errors are larger than in the MNIST dataset.

When training an RBM with much more complex datasets such as Cifar10 and Cifar100, the law is similar, as seen in Figs. 9 and 10.

In terms of the reconstruction error, PT has the largest variance and decreases quite slowly over the entire training process. DTC has several increases in the training process of Cifar10 and Cifar100, particularly in Cifar10, where the reconstruction error exceeded that of PT with a sharp rise between 7000 and 8000 iterations. This phenomenon appears because the dynamic strategy in DTC changes the standard of the reconstruction error, so it cannot reflect the convergence
tendency properly through the entire training process. With a constant standard, the one-step reconstruction error shows a different convergence tendency for DTC. GFPT stops converging in the early training process for Cifar10 and Cifar100. The one-step reconstruction errors of CD and DGS decrease stably, indicating that the convergence of CD and DGS are steady during the training process. DTC has several rapid decreases at the same iterations where the reconstruction error sharply increases. This phenomenon indicates that the dynamic strategy of training may lead to improved convergence. A similar situation occurs in PT, where the one-step reconstruction error of PT quickly decreases in all training processes.

In summary, the comparison of the one-step reconstruction error demonstrates that DTC converges well when learning simple and complex data. However, the training result cannot be evaluated based on only the reconstruction and one-step reconstruction errors; classification accuracy needs to be considered to obtain the final result.

2) ANALYSIS OF CLASSIFICATION ACCURACY
Classification accuracy is usually regarded as a relatively objective indicator of the final algorithm performance to indicate the accuracy of the RBM training results. This section compares the classification accuracy with one- and five-step Gibbs samplings shown in TABLE 4 and TABLE 5 separately. As the sampling step of DGS changes during the training process, it is compared for both conditions.

For the MNIST dataset, the gap between the classification accuracy of each algorithm is small. When utilizing the same Gibbs sampling steps, these algorithms exhibit little difference in classification accuracy. DGS using a dynamic sampling strategy adopts five-step sampling at the end of the training process, producing higher training accuracy than the other algorithms with one-step sampling. The DTC proposed in this article obtains the highest accuracy with five-step Gibbs sampling for all datasets.

For more complex datasets, the situation will be different. Compared with the MNIST dataset of two-dimensional information, the MNORB dataset contains photos of three-dimensional objects from different views, thus involving more information. It can be intuitively seen that the algorithms using five-step sampling are more accurate than one-step sampling algorithms. This result proves that the larger the number of Gibbs sampling steps, the higher the classification accuracy, which conforms to theoretical analysis. Moreover, the classification accuracy of PT is higher than that of CD, indicating that PT is relatively more accurate when learning the MNORB dataset. GFPT and DTC obtain higher classification accuracy than PT, and DTC is the highest, indicating that it has a greater advantage in dealing with the MNORB dataset.

The training results obtained for the more complex Cifar10 and Cifar100 datasets are similar to those of the MNORB dataset. However, it can be seen that PT is no longer usable for training for the most complex Cifar100 dataset, producing the lowest one-step reconstruction error. On the one hand, this result indicates that PT is not suitable in dealing with high-dimensional and complex datasets. On the other hand, GFPT and DTC, improved by PT, can train high-dimensional and complex datasets. DTC produces the highest classification accuracy for the Cifar100 dataset, demonstrating that it can avoid unnecessary inter-chain exchange in the initial stage of training and will not be as severely diverged as PT. As a result, DTC can converge better than PT in the early stage, which leads to higher training efficiency. When the algorithm is gradually stabilized, the number of temperature chains is increased to improve the ability of the algorithm to extract information from the entire model and prevent the algorithm from falling into the local optimum. At this time, the inter-chain exchange will not cause serious divergence, but will further improve the accuracy.

| TABLE 4. Classification Accuracy of the five algorithms with one-step Gibbs Sampling (including DGS). |
|---|---|---|---|---|
| CD | PT | DGS | GFPT | DTC |
| MNIST | 87.86 | 87.96 | **88.94** | 88.04 | 88.54 |
| MNORB | 68.15 | 68.59 | 73.00 | 73.21 | **78.11** |
| Cifar10 | 24.98 | 24.22 | 27.16 | 22.14 | **29.28** |
| Cifar100 | 7.66 | 0.92 | 7.72 | 7.12 | **8.16** |

| TABLE 5. Classification Accuracy of the Five Algorithms With Five-Step Gibbs Sampling (including DGS). |
|---|---|---|---|---|
| CD | PT | DGS | GFPT | DTC |
| MNIST | 88.90 | 89.40 | 88.94 | 88.02 | **89.52** |
| MNORB | 70.86 | 72.30 | 73.00 | 74.94 | **79.22** |
| Cifar10 | 25.48 | 28.82 | 27.16 | 22.54 | **29.32** |
| Cifar100 | 8.22 | 0.92 | 7.72 | 7.12 | **8.88** |
Overall, the proposed DTC significantly improves the accuracy of training for different datasets. It also obtains the highest classification accuracy, especially in high-dimensional and complex datasets, where the improvement is more evident.

3) ANALYSIS OF TRAINING TIME

The training time of the MNORB dataset is used as an example for analysis because the order of training time for each dataset is similar, as shown in Fig. 11. It can be clearly seen that CD and DGS take less time because they use a single sampling chain, which has a smaller number of sampling steps. Comparatively, PT takes a long time owing to its training strategy. Under the same number of iterations, with more temperature chains, PT needs to perform more Gibbs sampling, which inevitably increases the run time of the algorithm. In addition to Gibbs sampling, GFPT needs to perform gradient correction, which adds additional computation and increases training time. The proposed DTC is similar to CD in the early stage, which saves considerable time. Although the number of temperature chains increases in the later stage, the total time is still much lower than that of PT.

*FIGURE 11. Training time of five algorithms.*

To analyze the computational complexity of all the algorithms, several parameters need to be clarified: $out$ denotes the number of outer loops, $n$ denotes the number of mini-batches in our dataset, $M$ denotes the number of temperature chains, and $k$ denotes the number of Gibbs samplings in our algorithms. With these parameters, the asymptotic time complexity could be calculated, as summarized in Table 6.

**TABLE 6. Asymptotic Time Complexity of the Five Algorithms.**

| Algorithm | Asymptotic Time Complexity |
|-----------|-----------------------------|
| CD        | $O(out \cdot n \cdot k)$    |
| PT        | $O(out \cdot n \cdot M \cdot k)$ |
| DGS       | $O(out \cdot k)$            |
| GFPT      | $O(out \cdot n \cdot M \cdot k)$ |
| DTC       | $O(out \cdot n \cdot M \cdot k)$ |

By comparing the asymptotic time complexity, it can be seen that the computational complexities of CD and DGS are smaller than those of PT, GFPT, and DTC. For a detailed comparison of these algorithms, the comparisons of the time frequen tness of CD, PT, DGS, GFPT, and DTC, respectively, are as follows:

$$T_{(CD)} = out + 20out \cdot n + 3out \cdot n \cdot k,$$

$$T_{(PT)} = 3out \cdot n + out \cdot M + 5out \cdot n \cdot M + 3out \cdot n \cdot M,$$

$$T_{(DGS)} = out + 5out \cdot n + 3out \cdot n \cdot k,$$

$$T_{(GFPT)} = 24out \cdot n + out \cdot M + 5out \cdot n \cdot M + 3out \cdot n \cdot M \cdot k,$$

and

$$T_{(DTC)} = \frac{1}{2} out + 11 out \cdot n + \frac{7}{20} out \cdot M + \frac{3}{2} out \cdot n \cdot k + \frac{7}{4} out \cdot n \cdot M + \frac{21}{20} out \cdot n \cdot M \cdot k.$$

As the simulation program is self-designed, most of the constants in (24)–(28) are related to personal programming style and can only be compared in this article. Clearly, $T_{(DGS)} < T_{(CD)} < T_{(DTC)} < T_{(PT)} < T_{(GFPT)}$, which accords with Fig. 11.

**D. SUMMARY OF EXPERIMENTAL RESULTS**

Through the above experiments, it can be seen that DTC converges well and has the highest accuracy compared to the other algorithms, especially in high-dimensional and complex datasets.

1) The one-step reconstruction error result indicates that DTC converges well for the entire training process. It has a low one-step reconstruction error and demonstrates better convergence, especially in learning high-dimensional and complex datasets.

2) When comparing the classification accuracy, DTC outperforms other algorithms and the accuracy improvement is more obvious, especially when dealing with high-dimensional and complex datasets.

3) The run time of DTC is also shorter compared with the original PT, which provides a wider application range for DTC.

With all these advantages, DTC can produce better results when training an RBM with high-dimensional and complex data. Therefore, RBMs trained by DTC can be used as a well-trained pre-training layer for other deep learning models. This can help other deep learning models start training from a better state.

These experiments also indicate that the ability of a single-layer RBM to deal with complex problems is limited. The proposed DTC can improve convergence and accuracy when training high-dimensional and complex datasets.

**VI. CONCLUSION**

By comparing and analyzing RBM training algorithms, the conventional CD and PT algorithms were found to have
shortcomings. After analyzing the mathematical model of PT, a DTC algorithm was proposed that has the advantages of fast convergence and high accuracy. The training improvement was obvious, especially when dealing with high-dimensional and complex datasets such as MNORB, Cifar10, and Cifar100. Compared with the original PT, the accuracy of DTC has an obvious advantage. A new indicator, called the one-step reconstruction error, was also proposed to measure the training convergence of the RBM. This indicator can avoid the impact of multi-step Gibbs sampling on the reconstruction error and can be used as a more reliable training indicator.

Although the training speed of the proposed DTC increased compared to that of PT, the asymptotic time complexity was larger than that of CD. When the number of training iterations increases, the training time will be far more than that of CD. Therefore, it is important to improve the training speed in future studies. Additionally, the sampling method of DTC is Gibbs sampling, which is a biased estimate method. In our experiments, this biased estimate did not negatively influence the result; however, efficient unbiased estimation is needed in the future to improve training accuracy. Not only in images, but future works will also study the application of DTC in different databases and other data types.

REFERENCES
[1] A. G. Ivakhnenko and V. G. Lapa, Cybernetics and Forecasting Techniques. New York, NY, USA: North-Holland, 1967.
[2] G. E. Hinton, “Reducing the dimensionality of data with neural networks,” Science, vol. 313, no. 5786, pp. 504–507, Jul. 2006.
[3] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2016, pp. 770–778.
[4] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A.-R. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath, and B. Kingsbury, “Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups,” IEEE Signal Process. Mag., vol. 29, no. 6, pp. 82–97, Oct. 2012.
[5] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, “ImageNet: A large-scale hierarchical image database,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., Jun. 2009, pp. 248–255.
[6] P. Smolensky, “Information processing in dynamical systems: Foundations of harmony theory,” Dept. Comput. Sci., Colorado Univ. Boulder, Boulder, CO, USA, Tech. Rep. CU-CS-321-86, 1986.
[7] G. E. Hinton, S. Osindero, and Y.-W. Teh, “A fast learning algorithm for deep belief nets,” Neural Comput., vol. 18, no. 7, pp. 1527–1554, Jul. 2006.
[8] R. Salakhutdinov and G. Hinton, “An efficient learning procedure for deep Boltzmann machines,” Neural Comput., vol. 24, no. 8, pp. 1967–2006, Aug. 2012.
[9] G.-S. Xie, X.-B. Jin, X.-Y. Zhang, S.-F. Zhang, C. Yang, Z. Wang, and J. Pu, “From class-specific to class-mixture: Cascaded feature representations via restricted Boltzmann machine learning,” IEEE Access, vol. 6, pp. 69393–69406, 2018.
[10] W. K. Hastings, “Monte-Carlo sampling methods using Markov chains and their applications,” Biometrika, vol. 57, no. 1, p. 97, 1970.
[11] A. Fischer and C. Igel, “Training restricted Boltzmann machines: An introduction,” Pattern Recognit., vol. 47, no. 1, pp. 25–39, Jan. 2014.
[12] G. E. Hinton, “Training products of experts by minimizing contrastive divergence,” Neural Comput., vol. 14, no. 8, pp. 1771–1800, Aug. 2002.
[13] T. Tieleman, “Training restricted Boltzmann machines using approximations to the likelihood gradient,” in Proc. 25th Int. Conf. Mach. Learn. (ICML), 2008, pp. 1064–1071.
[14] T. Tieleman and G. Hinton, “Using fast weights to improve persistent contrastive divergence,” in Proc. 26th Annul. Int. Conf. Mach. Learn. (ICML), 2009, pp. 1033–1040.
[15] R. R. Salakhutdinov, “Learning in Markov random fields using tempered transitions,” in Proc. Adv. Neural Inf. Process. Syst., 2009, pp. 1598–1606.
[16] G. Desjardins, A. Courville, Y. Bengio, P. Vincent, and O. Delalleau, “Tempered Markov chain Monte Carlo for training of restricted Boltzmann machines,” in Proc. 13th Int. Conf. Artif. Intell. Statist., 2010, pp. 145–152.
[17] K. Cho, T. Raiko, and A. Ilin, “Parallel tempering is efficient for learning restricted Boltzmann machines,” in Proc. Int. Joint Conf. Neural Netw. (IJCNN), Jul. 2010, pp. 1–8.
[18] F. Li, X. G. Gao, and K. F. Wan, “Research on RBM training algorithm with dynamic Gibbs sampling,” Acta Automat. Sinica, vol. 42, no. 6, pp. 931–942, 2016.
[19] F. Li, X. Gao, and K. Wan, “Training restricted Boltzmann machine using gradient fixing based algorithm,” Chin. J. Electron., vol. 27, no. 4, pp. 694–703, Jul. 2018.
[20] M. Sambridge, “A parallel tempering algorithm for probabilistic sampling and multimodal optimization,” Geophys. J. Int., vol. 196, no. 1, pp. 357–374, Jan. 2014.
[21] Y. W. Teh and G. E. Hinton, “Rate-coded restricted Boltzmann machines for face recognition,” in Proc. Adv. Neural Inf. Process. Syst., 2001, pp. 908–914.
[22] L. Luo, Y. Wang, H. Peng, Z. Tang, S. You, and X. Huang, “Training restricted Boltzmann machine with dynamic learning rate,” in Proc. 11th Int. Conf. Comput. Sci. Educ. (ICCSSE), New York, NY, USA, Aug. 2016, pp. 103–107.
[23] J. Yin, Q. Y. Mao, D. Y. H. Liu, Y. Xu, and J. C. Lv, “Method to improve the performance of restricted Boltzmann machines,” in Advances in Neural Networks (Lecture Notes in Computer Science), vol. 10878, T. Huang, J. Lv, C. Sun, and A. V. Tuzikov, Eds. Cham, Switzerland: Springer, 2018, pp. 115–123.
[24] S. Guo, C. Zhou, B. Wang, and X. Zheng, “Training restricted Boltzmann machines with modified objective function based on limiting the free energy value,” IEEE Access, vol. 6, pp. 78542–78550, 2018.
[25] H. Manukian, Y. R. Pei, S. R. B. Bearden, and M. Di Ventra, “Mode-assisted unsupervised learning of restricted Boltzmann machines,” Commun. Phys., vol. 3, no. 1, pp. 1–8, Jun. 2020.
[26] M.-A. Côté and H. Larochelle, “An infinite restricted Boltzmann machine,” Neural Comput., vol. 28, no. 7, pp. 1265–1288, Jul. 2016.
[27] D. Chen, J. Lv, and Z. Yi, “Graph regularized restricted Boltzmann machine,” IEEE Trans. Neural Netw. Learn. Syst., vol. 29, no. 6, pp. 2651–2659, Jun. 2018.
[28] S. Pirmordi, M. Teshnehlab, N. Zarghami, and A. Sharifi, “The self-organizing restricted Boltzmann machine for deep representation with the application on classification problems,” Expert Syst. Appl., vol. 149, Jul. 2020, Art. no. 113286.
[29] H. Savitha, A. Ambikapathi, and K. Rajaraman, “Online RBM: Growing restricted Boltzmann machine on the fly for unsupervised representation,” Appl. Soft Comput., vol. 92, Jul. 2020, Art. no. 106278.
[30] P. Xie, Y. Deng, and E. Xing, “Diversifying restricted Boltzmann machine for document modeling,” in Proc. 21th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining (KDD), 2015, pp. 1315–1324.
[31] P. J. Grother, “NIST special database 19 handprinted forms and characters database,” NIST, Gaithersburg, MD, USA, Tech. Rep., 1995.
[32] Y. LeCun, F. J. Huang, and L. Bottou, “Learning methods for generic object recognition with invariance to pose and lighting,” in Proc. CVPR, vol. 2, Jun. 2004, pp. II–104.
[33] A. Krizhevsky and G. Hinton, “Learning multiple layers of features from tiny images,” Univ. Toronto, Toronto, ON, Canada, Tech. Rep., 2009.
[34] S. C. Xiao, H. J. Wu, S. Qiu, Z. D. Yang, and J. L. Qiao, “Contrastive divergence constrained by reconstruction error,” Inf. Hiding Multimedia Signal Process., vol. 8, pp. 182–190, Jan. 2017.

XINYU LI received the B.E. degree in electronic and information engineering from Northwestern Polytechnical University, Xi’an, China, in 2017, where he is currently pursuing the Ph.D. degree in control science and engineering with the School of Electronics and Information. His current research interests include deep learning and its applications in aviation systems.
XIAOGUANG GAO (Member, IEEE) received the Ph.D. degree in aircraft navigation and control systems from Northwestern Polytechnical University, Xi’an, China, in 1989. She is currently a Professor with the School of Electronics and Information, Northwestern Polytechnical University. Her research interests include Bayesian networks, modeling, and analysis of complex systems.

CHENFENG WANG received the B.S. degree in electronic and information engineering and the M.S. degree in systems engineering from Northwestern Polytechnical University, Xi’an, China, in 2017 and 2020, respectively. Her research interests include machine learning and data mining.

* * *