Neural Comput & Applic (2017) 28:1975–1984
DOI 10.1007/s00521-015-2170-y

ORIGINAL ARTICLE

Extreme learning machine with kernel model based on deep learning

Shifei Ding[1,2] · **Lili Guo**[1,2] · **Yanlu Hou**[1,2]

Received: 30 August 2015 / Accepted: 21 December 2015 / Published online: 12 January 2016

Abstract Extreme learning machine (ELM) proposed by Huang et al. is a learning algorithm for single-hidden layer feedforward neural networks (SLFNs). ELM has the advantage of fast learning speed and high efficiency, so it brought into public focus. Later someone developed regularized extreme learning machine (RELM) and extreme learning machine with kernel (KELM). But they are the single-hidden layer network structure, so they have deficient in feature extraction. Deep learning (DL) is a multilayer network structure, and it can extract the significant features by learning from a lower layer to a higher layer. As DL mostly uses the gradient descent method, it will spend too much time in the process of adjusting parameters. This paper proposed a novel model of convolutional extreme learning machine with kernel (CKELM) which was based on DL for solving problems—KELM is deficient in feature extraction, and DL spends too much time in the training process. In CKELM model, alternate convolutional layers and subsampling layers add to hidden layer of the original KELM so as to extract features and classify. The convolutional layer and subsampling layer do not use the gradient algorithm to adjust parameters because of some architectures yielded good performance with random weights. Finally, we took experiments on USPS and MNIST database. The accuracy of CKELM is higher than ELM, RELM and KELM, which proved the validity of the optimization model. To make the proposed approach more convincing, we compared with other ELM-based methods and other DL methods and also achieved satisfactory results.

Keywords Extreme learning machine (ELM) · Deep learning (DL) · Convolutional neural network (CNN) · CKELM

1 Introduction

Extreme learning machine (ELM) was proposed by Huang et al. [1, 2]. It has gained increasing interest from various research fields due to its high efficiency and fast learning speed. But ELM is based on empirical risk minimization which may lead to over-fitting [3]. In addition, ELM does not consider the weights error. When dataset exists outliers, the performance of ELM will be seriously affected [4]. In order to overcome these shortcomings, Deng et al. [5] proposed regularized extreme learning machine (RELM) that combined structural risk minimization theory and weighted least squares method. And then Huang et al. [6] put forward extreme learning machine with kernel (KELM) which introduced kernel function into ELM and got better results. ELM, RELM and KELM are all single-hidden layer network structure, so they cannot extract images' significant feature well. However, feature extraction is the core part of image recognition [7] and extract features which completely reflect the attribute character can improve the accuracy of the recognition system. So we are trying to find a structure which not only can extract the image's significant features but also contains the ELM's advantage of efficient and easy to implement, etc.

✉ Shifei Ding
dingsf@cumt.edu.cn

[1] School of Computer Science and Technology, China University of Mining and Technology, Xuzhou 221116, China

[2] Key Laboratory of Intelligent Information Processing, Institute of Computing Technology, Chinese Academy of Sciences, Beijing 100190, China

Deep learning (DL) is a multi-layer network structure which through establishes and imitates the human brain's hierarchical structure to extract the external input data's features [8, 9], so DL is good at extracting features. Convolutional neural network (CNN) is a typical method of DL, and it has applied in the image recognition field widely. CNN can automatically extract significant features from images [10], and it was applied in image recognition and achieved good results. CNN is regarded as one of the representatives in image recognition system. During the training of CNN, it uses BP algorithm to adjust parameters result in wasting a lot of time. In addition, CNN is particularly sensitive to learning rate. However, same architectures yielded good performance with random weights [11]. Research has shown that the effect of network is mainly determined by its structure [12]. So we need to choose the best architecture according to different databases.

For solving above problems, this paper puts forward a novel KELM method based on DL (CKELM). In CKELM, we use CNN to extract features, and during training of CNN, we use random weights so as to reduce the time of feature extraction. On the one hand, CKELM made up for the deficiency of KELM in extracting feature. On the other hand, the problem of much time in learning of DL was resolved. The structure of the other parts is as follows: Sect. 2 describes theoretical knowledge. Section 3 introduces the architecture of CKELM, its training process and advantages. Make experiments in Sect. 4, and the results of CKELM compare with ELM, RELM, KELM, ELM-based methods and DL methods. Section 5 makes conclusions and prospects.

2 Related works

2.1 ELM

ELM is a kind of algorithm with higher efficiency and easy to implement, and it contains three layers—input layer, hidden layer and output layer [13, 14]. Figure 1 shows the architecture of ELM which contains n input neurons, l hidden neurons and m output neurons.

For N different learning samples $(\mathrm{x}_i, y_i) \in R_n \times R_m$, $(i = 1, 2, \ldots, N)$, the output of hidden layer can be expressed as formula (1), and the output of neuron in output layer can be expressed as formula (2).

$$h = g(wx + b) \tag{1}$$

$$h(x_i)\beta = y_i^T, \quad i = 1, 2, \ldots, N \tag{2}$$

where $g(x)$ expresses hidden layer activation function of ELM, β is the weight between output and hidden layer, w is the weight between hidden layer and input layer, and b is the bias of hidden neuron.

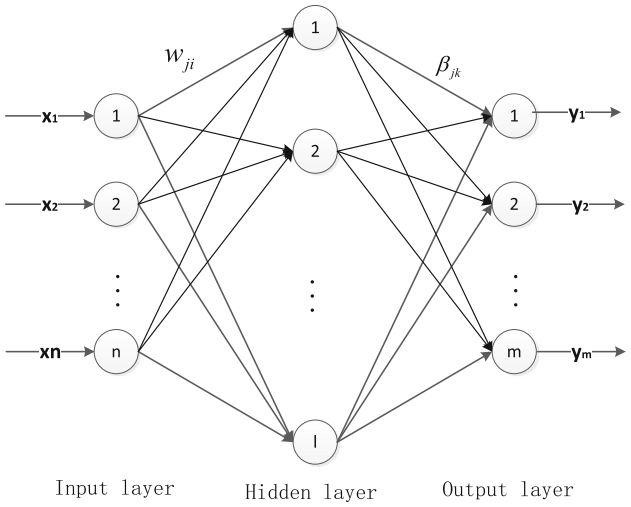

Fig. 1 Architecture of ELM

We use H to represent the connection matrix between hidden layer and output layer, Y represents the training data target matrix, so formula (2) can be abbreviated as:

$$H\beta = Y \tag{3}$$

where

$$\mathrm{H} = \begin{bmatrix} g(w_1, b_1, x_1) & g(w_2, b_2, x_1) & \cdots & g(w_l, b_l, x_1) \\ g(w_1, b_1, x_2) & g(w_2, b_2, x_2) & \cdots & g(w_l, b_l, x_2) \\ \vdots & \vdots & \ddots & \vdots \\ g(w_1, b_1, x_N) & g(w_2, b_2, x_N) & \cdots & g(w_l, b_l, x_N) \end{bmatrix}_{N \times l},$$

$$\beta = \begin{bmatrix} \beta_1^T \\ \beta_2^T \\ \vdots \\ \beta_l^T \end{bmatrix}_{l \times m}, \quad Y = \begin{bmatrix} y_1^T \\ y_2^T \\ \vdots \\ y_N^T \end{bmatrix}_{N \times m}$$

If $g(w_i, b_i, x_j)$ was infinitely differentiable, network only needs to set the number of hidden layer nodes, no need to adjust the input weights and the bias of hidden layer [15, 16]. Before training of ELM, it determines the number of input layer neurons based on the samples' feature vectors and determines the number of output layer neurons based on the samples' categories, according to the specific circumstances to determine the activation function and the number of hidden layer neurons [17].

ELM can randomly generate w and b before training, so it just make sure the number of hidden layer neurons and the activation function of the hidden layer neurons, then by least squares to calculate the β. The whole process finished in one time without iterations, so the speed was significantly faster than BP algorithm. The following steps are the main training process of ELM:

Step 1: Determine the number of hidden layer neurons, randomly set bias b of hidden layer neurons and set weights w between input layer and hidden layer;
Step 2: Choose an infinitely differentiable function as the activation function, and then calculate the hidden layer output matrix H;
Step 3: Calculate weights β between hidden layer and output layer: $\beta = H^T H$, where H^T is the generalized inverse matrix of the output matrix H.

2.2 RELM

Although ELM has fast learning and good generalization performance, it still has some shortcomings. According to statistical theory, we can know the actual risks include two components—empirical risk and structural risk [18]. But ELM only considers the empirical risk without the structural risk and therefore may result in over-fitting problem. In addition, ELM has poor robustness and controllability. So Deng et al. combined with experiential risk and structural risk put forward regularized extreme learning machine (RELM) which has better robustness.

RELM regulates the ratio of empirical risk and structural risk through the regularized parameter C, and the mathematical model of RELM can be expressed as:

$$\min L_{RELM} = \frac{1}{2}\|\beta\|^2 + \frac{C}{2}\|Y - H\beta\|^2 \tag{4}$$

By setting the gradient of formula (4) with respect to β to zero, we get formula (6).

$$\beta - CH^T(Y - H\beta) = 0 \tag{5}$$

If H has more rows than columns ($N > L$), which is usually the case where the number of training samples is more than the number of hidden neurons, the connection weight matrix β between hidden layer and output layer can be expressed as formula (6).

$$\beta = \left(\frac{I}{C} + H^T H\right)^{-1} H^T Y \tag{6}$$

where I is an identity matrix of L.

Conversely, if the number of training samples is less than the number of nodes in hidden layer ($N < L$), then H will have more columns than rows, and the connection weight matrix β between hidden layer and output layer can be expressed as formula (7).

$$\beta = H^T\left(\frac{I}{C} + HH^T\right)^{-1} Y \tag{7}$$

where I is an identity matrix of N.

2.3 KELM

In 2012, Huang et al. put forward KELM which introduced kernel function into ELM. The output of hidden layer treated as a nonlinear mapping of samples. When the mapping is unknown, it constructs the kernel function to represent HH^T:

$$\begin{aligned} HH^T = \Omega_{ELM} &= \begin{bmatrix} K(x_1, x_1) & \cdots & K(x_1, x_N) \\ \vdots & \ddots & \vdots \\ K(x_N, x_1) & \cdots & K(x_N, x_N) \end{bmatrix}, \\ h(x)H^T &= \begin{bmatrix} K(x, x_1) \\ \vdots \\ K(x, x_N) \end{bmatrix} \end{aligned} \tag{8}$$

So formula (7) can be expressed as formula (9):

$$\beta = \left(\frac{I}{C} + \Omega_{ELM}\right)^{-1} Y \tag{9}$$

The classification formula of KELM expressed as formula (10):

$$f(x) = g(h(x)H^T\beta) = g\left(\begin{bmatrix} K(x, x_1) \\ \vdots \\ K(x, x_N) \end{bmatrix}\left(\frac{I}{C} + \Omega_{ELM}\right)^{-1} Y\right) \tag{10}$$

The most frequently used kernel functions are radial basis function (RBF) and Gaussian function. In paper, Wang et al. [19] used wavelet kernel function as kernel function of KELM also achieved satisfactory results. In addition, based on hybrid kernel function (HKELM), Ding et al. [20] proposed a novel extreme learning machine. HKELM constructs a hybrid kernel function with better performance by fully combining local kernel function strong learning ability and global kernel function strong generalization ability. Compared with traditional ELM, HKELM can effectively improve the classification results, avoid local minimum and with better generalization.

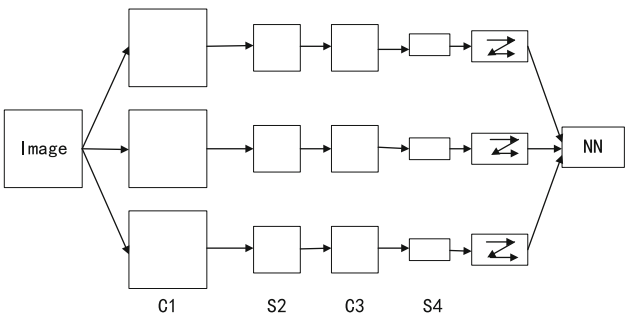

Fig. 2 Architecture of CNN

2.4 CNN

CNN is a multi-layer neural network as shown in Fig. 2. It is an important method widely used in image recognition. Each layer of CNN consists of many two-dimensional planes, and each plane consists of many single neurons [21]. Each neuron of those maps defines their receptive fields [22], and the neurons only accept the signal from local receptive fields. In general, layer C is feature extraction layer, and each input neuron of C is connected to the local receptive field in the input and extracts its local features and then determines the location relationship with other feature space by those local features. Layer S is feature mapping layer that has the advantage of position invariance. Each feature mapping is a plane, and all neurons' weights in the plane are equal, thus reducing the number of free parameters and reducing the complexity of network parameters. Each feature extraction layer (layer C) will follow a subsampling layer (layer C) which constitutes the two feature extraction, so the network has good distortion tolerance [23].

The training of CNN is supervised, and it can be divided into three steps:

Step 1: Through forward propagation to compute the output of each layer;
Step 2: By the above network output, BP algorithm is used to seek the derivative of error on the network weights;
Step 3: Through the weight updating method to update the weights.

CNN is a kind of artificial neural network, and it is good at mining data's local features. CNN adopts the weight sharing technique on one feature map resulting in reducing network model's complexity and decreasing the number of weights. These advantages make CNNs applied to many fields in pattern recognition [24, 25]. After years of research, CNNs successfully applied in face detection [26], document analysis [27], speech detection [28], license plate detection [29], etc.

3 CKELM

3.1 Network architecture and merits

KELM is a single-hidden layer neural networks that resulted in its deficiencies in feature extraction. A new model of CKELM not only preserves the advantages of KELM but also overcomes the problem of feature extraction. Figure 3 shows the architecture of CKELM that we use the most common structure of CNN as an example to introduce (two convolutional layers with filter size 5×5 and two subsampling layers with pooling size 2×2). Plainly, the architecture of CKELM is similar to KELM which consists of input layer, hidden layer and output layer. The difference between CKELM and KELM is that the hidden layer of CKELM is not a single layer but adds convolutional layers and subsampling layers. Details of the training process will be introduced in Sect. 3.2.

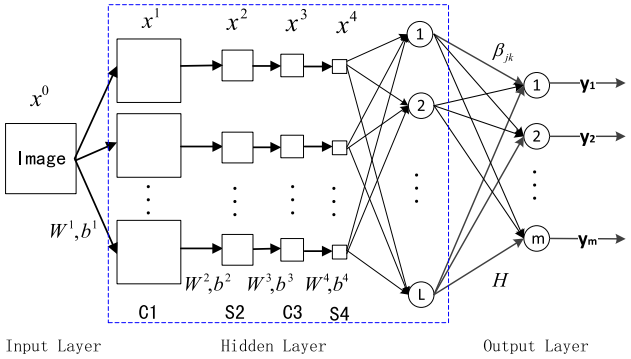

Fig. 3 Architecture of CKELM

Feedforward neural networks mostly use the gradient descent method [30], and the inherent disadvantages of gradient descent method have become hindrance in its development. (1) Training speed is slow. Because gradient descent method needs many iterations to achieve the purpose of update parameters, so the training process will waste a lot of time. (2) No matter whether it is the local or the global minima, the training process will stop. So it is easy to fall into local minima. (3) It is sensitive to the learning rate selection. The learning rate has large influence on the performance of neural network, and it must choose the suitable learning rate [31]. Smaller learning rate will slow the convergence speed and increase the training time, while greater learning rate will lead to unstable training process. CKELM is different from the situation, and it does not need to adjust the parameters. It just takes one time to get the optimal solution, so KELM has the advantages of fast learning speed.

According to the preamble we know feature extraction is the key part of image recognition [32], and it is important to extract features that reflect the character attribute fully. But KELM is a single-hidden layer neural network, so it has deficiencies on feature extraction. CKELM makes use of convolutional layers and subsampling layers to extract effective features so as to make up the deficiency of KELM in feature extraction. On the one hand, CKELM overcomes gradient algorithm's problem that tastes long time to training. On the other hand, it overcomes the deficiencies of KELM in feature extraction time. Specific experiments are taken in Sect. 4.

3.2 Training

Feature extraction mainly depends on hidden layer of CKELM which contains two convolutional layers, two subsampling layers and hidden layer of the original KELM. We use $s_{i-1} \times s_{i-1}$ to represent the image size in i layer.

The first layer is the input layer which only contains one feature map $n_0 = 1$. Each input image x^0 has a corresponding target value. The size of the input image is $s_0 \times s_0$.

The second layer C1 is convolutional layer. Feature map of the input layer was convolved by convolutional kernel and then by activation function to get feature maps with size of $s_1 \times s_1$. C1 contains n_1 feature maps, and n_1 needs to be set according to the specific situation. We use 5×5 convolutional kernel, so $s_1 = s_0 - 4$. We use sigmoid function as the activation function that puts the output value compressed to [0, 1]. The output of the C1 layer can be expressed as formula (11).

$$x_j^1 = f\left(\sum_{i \in M_j} x_i^0 * w_{ij}^1 + b_j^1\right) \tag{11}$$

where M_j represents a collection of input feature maps and w_{ij} represents connection weight between the neuron j of convolutional layer and the neuron i of the input layer. In addition, each output features map corresponds to a bias b.

The third layer S2 is subsampling layer with feature kernel size of 2×2. On subsampling layer, the number of feature maps unchanged is n_1, but the size of each output feature map is smaller to $s_2 \times s_2$: $s_2 = \lceil s_1/2 \rceil$. The output of S2 layer can be expressed as formula (12).

$$x_j^2 = f(W_j^2 down(x_j^1) + b_j^2) \tag{12}$$

where $down(\cdot)$ represents a sampling function. Each output map corresponds to a weight w and a bias b [33].

The fourth layer C3 is the convolutional layer too, and its calculation is similar to C1 layer. On C3 layer, the number of feature map is n_2, and the size of feature map is $s_3 \times s_3$: $s_3 = s_2 - 4$. The output can be expressed as:

$$x_j^3 = f\left(\sum_{i \in M_j} x_i^2 * w_{ij}^3 + b_j^3\right) \tag{13}$$

The fifth layer S4 is subsampling layer too, and its calculation is similar to S2. The number of feature map is n_2 too, but the size of feature map is $s_4 \times s_4$: $s_4 = \lceil s_3/2 \rceil$. The output of S4 can be expressed as:

$$x_j^4 = f(W_j^4 down(x_j^3) + b_j^4) \tag{14}$$

Then, the output of layer S4 is as the input x for classification learning. The output matrix H of the hidden layer is as follows:

$$H = \begin{bmatrix} h(x_1) \\ \vdots \\ h(x_N) \end{bmatrix}_{N \times L} \tag{15}$$

where L is the number of neurons in sixth layer and N is the number of samples. The hidden layer output $h(x_i)$ of each sample is regarded as a nonlinear mapping of sample x_i.

When the mapping is a unknown mapping, it constructs kernel function to replace HH^T. Gaussian function is the most commonly used kernel function, so we selected the Gaussian kernel:

$$HH^T(i,j) = K(x_i, x_j) = \exp\left\{-\|x_i - x_j\|^2 \Big/ 2\gamma^2\right\} \tag{16}$$

$$HH^T = \Omega_{ELM} = \begin{bmatrix} 0 & \cdots & \exp\left\{-\|x_1 - x_N\|^2 \Big/ 2\gamma^2\right\} \\ \vdots & \ddots & \vdots \\ \exp\left\{-\|x_N - x_1\|^2 \Big/ 2\gamma^2\right\} & \cdots & 0 \end{bmatrix} \tag{17}$$

$$h(x)H^T = \begin{bmatrix} \exp\left\{-\|x - x_1\|^2 \Big/ 2\gamma^2\right\} \\ \vdots \\ \exp\left\{-\|x - x_N\|^2 \Big/ 2\gamma^2\right\} \end{bmatrix} \tag{18}$$

where γ is kernel parameter which take experiments to determine optimal value.

The representation of weights matrix β between hidden layer and output layer is the same as formula (9), and the classification formula is the same as formula (10).

4 Experiments

In order to evaluate CKELM model, we carried out experiments in the famous USPS and MNIST database and compared with ELM, RELM and KELM. We took experiments in a desktop with a Core i7 Intel Xeon E5-16200 3.6 GHz processor and 18 GB RAM running MATLAB 2012b.

4.1 USPS database

USPS [34] is the handwritten digits recognition database of US Postal Service which contains a training set of 7291 samples and a testing set of 2007 samples. Images in the USPS have been size-normalized and centered in a fixed-size image 16×16 pixels. We selected all the training samples and testing samples to take experiments.

As is stated above, we need to select the optimal network structure for USPS database. Classification experiments were run on different architectures with varying

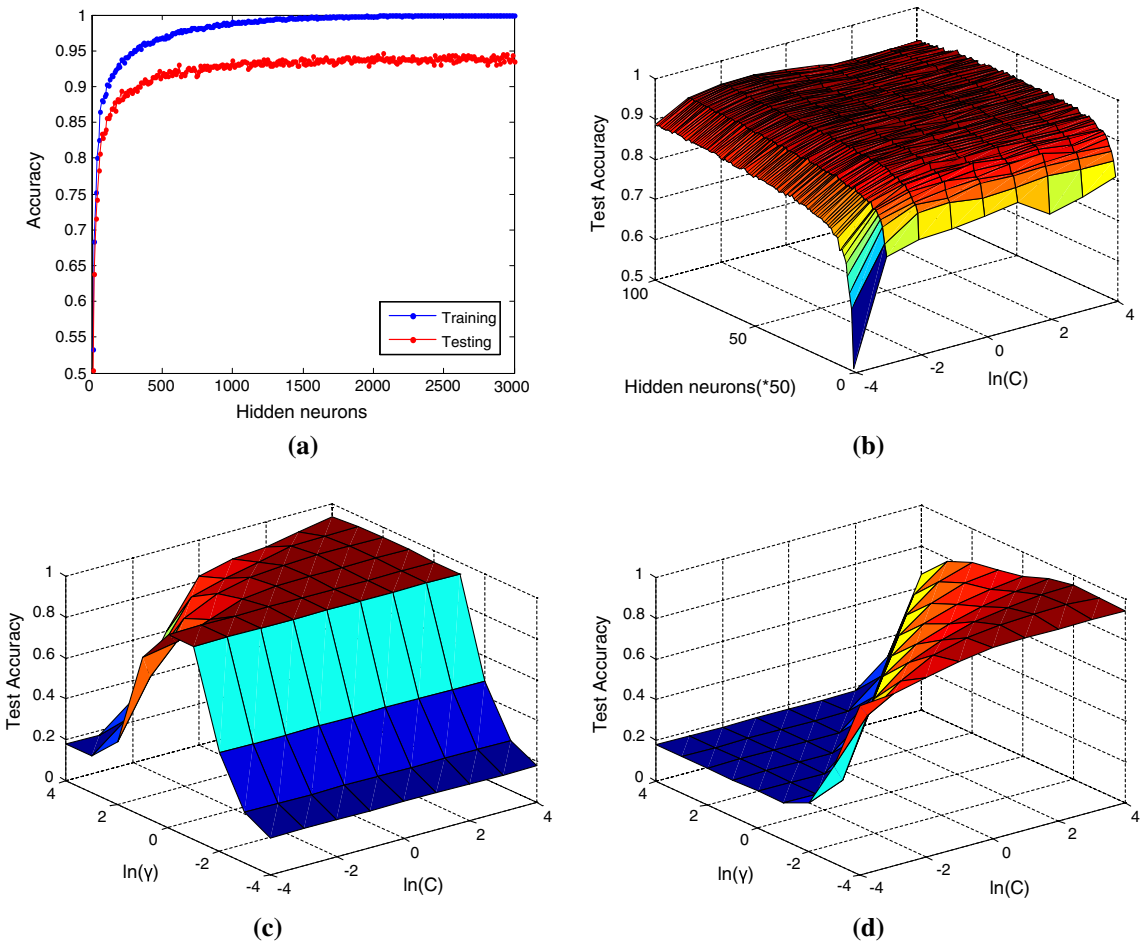

Fig. 4 ELM, RELM, KELM and CKELM for USPS database. **a** ELM for USPS database, **b** RELM for USPS database, **c** KELM for USPS database, **d** CKELM for USPS database

filter sizes 5 × 5, 8 × 8, 9 × 9 and pooling sizes 2 × 2, 3 × 3. In addition, the number of filters also affects the generalization ability, so we choose 25, 50 and 100 to take experiments. Finally, select the architecture with filter sizes 8 × 8, pooling sizes 3 × 3 and 100 filters.

During the training of ELM, we need to make sure the number of hidden layer neurons, so we took many experiments to obtain the optimal value. Figure 4a shows the relationship between hidden neurons and accuracy, and we can know the optimal number of hidden layer neurons is 1800. Figure 4b shows the relationship chart between the accuracy of RELM, regularized parameter C and hidden layer nodes. It can be seen from the figure hidden layer nodes have little influence on accuracy, but the parameter C plays the leading role. In RELM, the number of hidden layer node is 1650 and parameter C is 0.1. As shown in Fig. 4c, the regularized parameter C and the kernel parameter γ of KELM are 100. Figure 4d shows the relationship chart of CKELM, and we can know that the regularized parameter C is 10,000 and the kernel parameter γ is 0.1.

Table 1 lists the accuracies of ELM, RELM, KELM and CKELM on the USPS database. From the table, we can see the test accuracy of CKELM is 96.21 % which is higher than other methods. Shen et al. [35] presented a SVM–ELM model that obtained the accuracy of 94.32 %, which is lower than CKELM on the USPS database.

Table 2 lists the time of ELM, RELM, KELM and CKELM on the USPS database. From the table, we can see the training time of CKELM is less than ELM and RELM. Since CKELM adds convolutional layer and subsampling layer to hidden layer, the training time of CKELM is more than KELM. It also reflects the shortcomings of CKELM, but on the accuracy side the algorithm is effective.

4.2 MNIST database

MNIST [36] database of handwritten digits contains a training set of 60,000 examples and a testing set of 10,000 samples. It is a subset of a larger set available from NIST. Images in the MNIST have been size-normalized and centered in a fixed-size image 28 × 28 pixels. We

Table 1 Comparison of accuracies on USPS database

Database	Algorithms	#	Training accuracy (%)	Testing accuracy (%)
USPS	ELM	Average	99.78 (±0.03)	93.70 (±0.32)
		Best	99.85	94.57
	RELM	Average	99.96 (±0.004)	94.63 (±0.27)
		Best	99.97	95.22
	KELM	–	99.97	95.42
	CKELM	–	99.98	96.21

Table 2 Comparison of time on USPS database

Database	Algorithms	Training time (s)	Testing time (s)
USPS	ELM	29.53	0.64
	RELM	53.52	1.32
	KELM	8.87	0.92
	CKELM	14.43	2.11

randomly selected 6000 training samples and 1000 testing samples to take experiments.

Also we need to select the optimal network structure for the USPS database. Such as the filter size, pooling size and the number of filter all need to choose suitable value so as to ensure the classification performance of architecture with random weights. Through many trials, we found the optimal architecture consists of two convolutional layers and two subsampling layers with filter sizes 5×5 and pooling sizes 2×2 (such as Fig. 2), and numbers of filters are $n_1 = 10$ and $n_2 = 20$.

Figure 5a shows the relationship of ELM, RELM, KELM and CKELM between the accuracy and parameters. The number of hidden layer nodes of ELM is 2500. In RELM, the number of hidden layer nodes is 4700 and the regularized parameter C is 0.01. The parameter C and kernel parameter γ of KELM are 100. In CKELM, the regularized parameter C is 100 and kernel parameter γ is 0.01.

Table 3 lists the comparison accuracies on the MNIST database. CKELM got accuracy of 96.8 % which is significantly higher than other methods. Experimental results showed that the CKELM model is valid on the MNIST database.

Table 4 lists the comparison time on the MNIST database. The training time of CKELM is less than ELM and RELM. Since CKELM adds convolutional layer and subsampling layer to hidden layer, the training time of CKELM is more than KELM.

4.3 Comparison results

To make the proposed approach more convincing, we took results for comparison with other ELM-based methods, such as SVM–ELM [35] and PCA–ELM [37]. In addition, considering that the proposed method is based on deep learning, thus, it is supposed to compare with some deep learning methods on MNIST dataset, like deep belief network (DBN) [9] and stacked autoencoder (SAE) [38]. We took experiments on the subset of MNIST which contains a training set of 6000 examples and a test set of 1000 examples. Comparisons with other results of different methods published on the subset of MNIST dataset are listed in Table 5.

From above experiments, we can see that the accuracy of CKELM is higher than ELM, RELM and KELM. The accuracy is improved significantly compared with other ELM-based methods such as SVM–ELM and PCA–ELM, which proved the validity of the new model. The accuracy is not improved significantly compared with other deep learning methods, as we adopt the architecture with random weights, and the time is far less than SAE and DBN. Those all prove the feasibility and validity of CKELM.

5 Conclusion

This paper presents a new model of KELM called CKELM based on deep learning. CKELM uses alternate convolutional layers and subsampling layers to extract features for solving the problem—KELM is deficient in feature extraction. We conducted experiments on the USPS and MNIST database and obtained better results than ELM, RELM and KELM.

It is worth seeking for more efficient and faster methods:

(1) It is worth studying whether using other deep learning algorithms to extract features can provide better results, such as auto encoder (AE), sparse

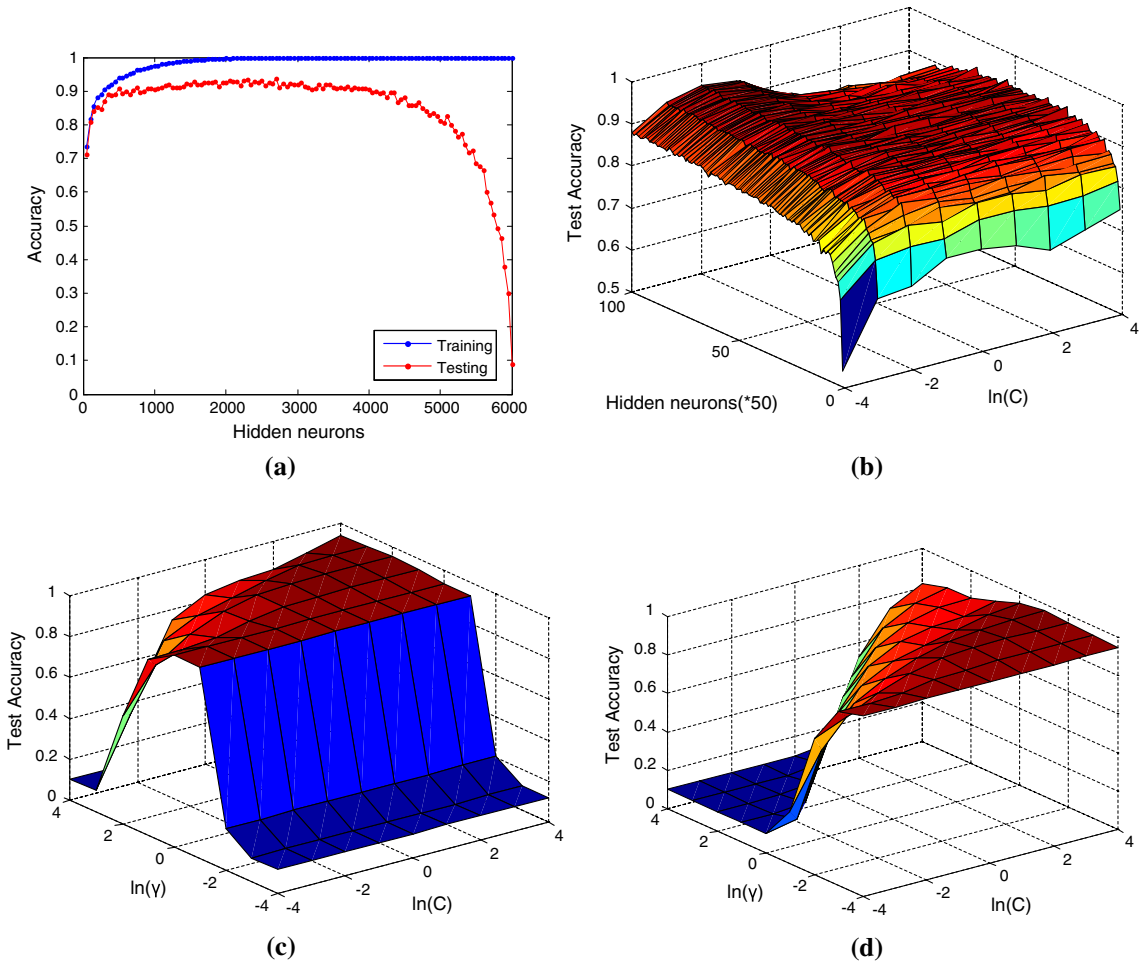

Fig. 5 ELM, RELM, KELM and CKELM for MNIST database. **a** ELM for MNIST database, **b** RELM for MNIST database, **c** KELM for MNIST database, **d** CKELM for MNIST database

Table 3 Comparison of accuracies on MNIST database

Database	Algorithms	#	Training accuracy (%)	Testing accuracy (%)
MNIST	ELM	Average	99.92 (±0.03)	92.27 (±0.52)
		Best	99.98	93.20
	RELM	Average	99.82 (±0.03)	94.50 (±0.36)
		Best	99.90	95.30
	KELM	–	100	95.80
	CKELM	–	100	96.80

Table 4 Comparison of time on MNIST database

Database	Algorithms	Training time (s)	Testing time (s)
MNIST	ELM	54.25	0.83
	RELM	43.39	1.30
	KELM	6.44	0.63
	CKELM	8.22	0.89

coding (SC), Restrict Boltzmann Machine (RBM) and deep belief nets (DBNs) that are the cumulative of RBM.

(2) Architectures of CNN for USPS database and MNIST database may not be the optimal, so choosing architecture with better efficiency also deserves further study.

Table 5 Comparisons of testing results on the subset of MNIST dataset

Algorithms	Testing accuracy rate (%)
SVM–ELM	94.46
PCA–ELM	94.92
SAE	95.89
DBN	96.47
CKELM	96.80

(3) We can try more choices on kernel function such as wavelet kernel hybrid kernel to improve the generalization ability.

Acknowledgments This work is supported by the National Natural Science Foundation of China (No. 61379101) and the National Key Basic Research Program of China (No. 2013CB329502).

References

1. Huang GB, Zhu QY, Siew CK (2004) Extreme learning machine: a new learning scheme of feedforward neural networks. Proceedings of IEEE International Joint Conference on Neural Networks 2004 2:985–990
2. Huang GB, Zhu QY, Siew CK (2006) Extreme learning machine: theory and applications. Neurocomputing 70:489–501
3. Vapnik VN (1995) The nature of statistical learning theory. Springer, New York
4. Rousseeuw PJ, Leroy A (1987) Robust regression and outlier detection. Wiley, New York
5. Deng W, Zheng Q, Chen L et al (2010) Research on extreme learning of neural networks. Chin J Comput 33(2):279–287
6. Huang GB, Zhou HM, Ding XJ, Zhang R (2012) Extreme learning machine for regression and multiclass classification. IEEE Trans Syst Man Cybern Part B Cybern 42(2):513–529
7. Ding SF, Xu XZ, Nie R (2014) Extreme learning machine and its applications. Neural Comput Appl 25(3):549–556
8. Hinton GE, Osindero S, Teh YW (2006) A fast learning algorithm for deep belief nets. Neural Comput 18(7):1527–1554
9. Hinton GE, Salakhutdinov RR (2006) Reducing the dimensionality of data with neural networks. Science 313(5786):504–507
10. LeCun Y, Bottou L, Bengio Y, Haffner P (1998) Gradient-based learning applied to document recognition. Proc IEEE 86(11):2278–2324
11. Saxe A, Koh PW, Chen Z et al. (2011) On random weights and unsupervised feature learning[C]. In: Proceedings of the 28th international conference on machine learning (ICML-11), pp 1089–1096
12. Jarrett K, Kavukcuoglu K, Ranzato M, LeCun Y (2009) What is the best multi-stage architecture for object recognition[C]. In: IEEE 12th international conference on computer vision, pp 2146–2153
13. Guo RF, Huang GB, Lin QP et al (2009) Error minimized extreme learning machine with growth of hidden nodes and incremental learning. IEEE Trans Neural Netw 20(8):1352–1357
14. Zhu QY, Qin AK, Suganthan PN et al (2005) Evolutionary extreme learning machine. Pattern Recogn 38(10):1759–1763
15. Huang GB, Liang NY, Rong HJ et al (2005) On-line sequential extreme learning machine In: The IASTED international conference on computational intelligence. Calgary
16. Huang GB, Siew CK (2005) Extreme learning machine with randomly assigned RBF Kernels. Int J Inf Technol 11(1):16–24
17. Huang GB, Lei C, Siew CK (2006) Universal approximation using incremental constructive feedforward networks with random hidden nodes. IEEE Trans Neural Netw 17(4):879–892
18. Haykin S (1999) Neural networks: a comprehensive foundation. Prentice Hall, New Jersey
19. Wang J, Guo C (2013) Wavelet kernel extreme learning machine classifier. Microelectron Comput 30(010):73–76
20. Ding S, Zhang Y, Xu Xinzheng (2013) A novel extreme learning machine based on hybrid kernel function. J Comput 8(8):2110–2117
21. Kwolek B (2005) Face detection using convolutional neural networks and Gabor filters. In: Artificial neural networks: biological inspirations–ICANN 2005. Springer Berlin, pp 551–556
22. Laserson J (2011) From neural networks to deep learning: zeroing in on the human brain. ACM Crossroads Stud Mag 18(1):29–34
23. Neubauer C (1992) Shape, position and size invariant visual pattern recognition based on principles of recognition and perception in artificial neural networks. North Holland, Amsterdam, pp 833–837
24. Vincent P, Larochelle H, Bengio Y et al. (2008) Extracting and composing robust features with denoising autoencoders. In: Proceedings of the 25th international conference on machine learning (ICML'2008). ACM Press, New York, pp 1096–1103
25. Huang FJ, LeCun Y (2006) Large-scale learning with SVM and convolutional for generic object categorization. In: Proceedings of IEEE computer society conference on computer vision and pattern recognition. IEEE Computer Society, Washington, pp 284–291
26. Tivive FHC, Bouzerdoum A (2003) A new class of convolutional neural networks (SICoNNets) and their application of face detection. Proc Int Jt Conf Neural Netw 3:2157–2162
27. Simard PY, Steinkraus D, Platt JC (2003) Best practice for convolutional neural networks applied to visual document analysis. In: Seventh international conference on document analysis and recognition, pp 985–963
28. Skittanon S, Surendran AC, Platt JC et al (2004) Convolutional networks for detection. Interspeech, Lisbon, pp 1077–1080
29. Chen Y, Han C, Wang C et al. (2006) The application of a convolution neural network on face and license plate detection. In: 18th international conference on pattern recognition, ICPR. IEEE Computer Society, Hong Kong, pp 552–555
30. Poultney C, Chopra S, Cun YL (2006) Efficient learning of sparse representations with an energy-based model. In: Advances in neural information processing systems. MIT Press, MA, pp 1137–1144
31. Platt JC, Steinkraus D, Simard PY (2003) Best practices for convolutional neural networks applied to visual document analysis. In: Proceedings of the international conference on document analysis and recognition, ICDAR. Edinburgh, pp 958–962
32. Keysers D, Deselaers T, Gollan C, Ney H (2007) Deformation models for image recognition. IEEE Trans Pattern Anal Mach Intell 29(8):1422–1435
33. Zhang Yannan, Ding Shifei, Xinzheng Xu et al (2013) An algorithm research for prediction algorithm of extreme learning machines based on rough sets. J Comput 8(5):1335–1342
34. Lu J, Lu Y, Cong G (2011) Reverse spatial and textual k nearest neighbor search. In: Proceedings of SIGMOD. ACM Press, Athens, pp 349–360

35. Shen Fengshan, Wang Liming, Zhang Junying (2014) Reduced extreme learning machine employing SVM technique. J Huazhong Univ Sci Technol (Natural Science Edition) 42(6):107–111
36. Zhang D, Ooi BC, Tung A (2010) Locating mapped resources in web 2.0. In: Proceedings of ICDE. Long Beach, pp 521–532
37. Castaño A, Fernández-Navarro F, Hervás-Martínez C (2013) PCA-ELM: a robust and pruned extreme learning machine approach based on principal component analysis. Neural Process Lett 37(3):377–392
38. Jirayucharoensak S, Pan-Ngum S, Israsena P (2014) EEG-based emotion recognition using deep learning network with principal component based covariate shift adaptation. Sci World J 2014:627892-1–627892-10