Accelerating Method of Evolutionary Ensemble Learning Based on Gaussian Random Field

Guanghua Xu, Zifeng Dai, Zhihong Liu, Chen Zhang, Bin Zhang, and Changsheng Zhang(✉)

Northeastern University, Shenyang 110819, People's Republic of China
zhangchangsheng@mail.neu.edu.cn

Abstract. In recent years, research on ensemble learning of neural networks is very popular. As a research hotspot in the field of machine learning, ensemble learning methods can effectively improve the accuracy and generalization of deep network models, but not all neural networks are suitable for participating in the construction of ensemble model. Deep network ensemble learning requires a single neural network participating in the ensemble to have a high accuracy rate, and there is a large difference between the networks. In the initial stage of deep network ensemble learning, the generation process of the candidate deep network set is first required. In this article, a multi-objective evolutionary ensemble model is improved, and an evolutionary ensemble learning acceleration method based on Gaussian random field is added before the evaluation of fitness function which can screen individuals with great potential for improvement in the evaluation of fitness function during the generation of candidate deep network sets, thereby effectively improving the quality of the solution and reduce the time spent training neural networks. This pre-screening strategy is applied to the solution of the multi-objective differential evolution algorithm, which can conveniently obtain a large number of neural network models with high accuracy and large network differences. And this strategy speeds up the solution process of multi-target algorithm.

Keywords: Gaussian random field · Ensemble learning · Differential evolution algorithm · Neural network · Deep learning

1 Introduction

Ensemble learning [1] improved the prediction accuracy of the final model by building and combining multiple basic models. Almost all machine learning algorithms or models can improve the generalization performance by introducing the idea of ensemble learning, which makes it a research hotspot in the field of machine learning. With the development of modern science and technology and the improvement of productivity, major breakthroughs have been made in computer performance, and the acquisition of massive data is no longer a difficult task. As a synonym for artificial neural networks, deep learning [2] has suddenly become the hottest research direction in the field of

H. Song and D. Jiang (Eds.): SIMUtools 2020, LNICST 369, pp. 251–262, 2021.
https://doi.org/10.1007/978-3-030-72792-5_23

machine learning, various algorithms for improving the training speed of neural networks have been proposed successively. Based on this, neural networks have achieved rapid development, and more and more deep network models [3–6] have been proposed and widely used in various fields such as face recognition, image classification, and natural language processing.

Because ensemble learning methods have significant advantages in improving generalization performance, many experts and scholars are committed to combining deep networks with ensemble learning to further improve the accuracy of neural network models in various application scenarios [7–9] and specific tasks. If the neural network is considered as an individual in ensemble learning, it is important to consider that not all neural networks are suitable for participating in the construction of the ensemble model. Deep network ensemble learning requires [10] that a single neural network participating in the ensemble has high accuracy rate, and there are large differences between networks.

In the initial stage of deep network ensemble learning, the process of generating candidate deep network sets needs to be performed first. In the candidate network generation stage, a multi-objective algorithm based on differential evolution is used to solve the problem of generating candidate deep network sets, and the targets to be optimized are the prediction accuracy of the deep network and the differences between the networks. However, since each individual in the population corresponds to a specific neural network, and the individual needs to train the corresponding neural network when evaluating the fitness function, this part will take a lot of time, so pre-screening is required. An existing pre-screening strategy is a guide model pre-screening strategy, which can accelerate the evolution rate of the population and find the optimal solution faster, but it is very expensive in terms of computational cost. This article proposes an evolutionary ensemble learning acceleration method based on Gaussian random fields, which can select individuals with great potential for evaluation during the generation of candidate deep network sets for fitness function evaluation, thereby effectively improving the quality of the solution and reducing time spent training a neural network.

The rest of this article is arranged as follows. The second part introduces an existing multi-objective deep belief networks sensitive (MODBNE) method [11]. The third part details the evolutionary ensemble learning acceleration method based on Gaussian random fields. The fourth part evaluates this method based on the existing two data sets, and its performance is analyzed. The fifth part summarizes the research results and future research plans of this article.

2 MODBNE Method

Differential Evolution algorithm [12] (Differential Evolution, DE) was originally proposed by Rainer Storn and Kenneth Price. The idea is derived from the Genetic Algorithm (GA) proposed earlier, and it is also a search and optimization strategy simulating biological evolution. The multi-objective differential evolution algorithm is often used to solve multi-objective optimization problems. The multi-objective optimization (MOP) problem [13] is a mutually exclusive relationship between the objective functions. The optimization of one goal will inevitably lead to the deterioration of other goals, which means that there is a trade-off between the various objective functions.

Zhang C et al.'s article proposes a multi-objective deep belief networks ensemble (MODBNE) method [11]. In the multi-objective evolutionary ensemble learning model, the MOEA/D algorithm [14] is integrated with traditional DBN training techniques, and multiple DBNs are evolved at the same time. Accuracy and diversity are used as two conflicting goals. The development of ensemble learning groups based on two goals effectively balances the accuracy and diversity of individuals in the ensemble learning group, and greatly optimizes the ensemble model. And the evaluation of this method on the NASA C-MAPSS aero engine data set shows that MODBNE has excellent performance.

In her article, metrics are established for accuracy and difference. In terms of accuracy metrics, the problem of maximizing accuracy is converted to the problem of minimizing error rate. The specific calculation method after converting it into an objective function is shown in formula (2.1)

$$minimize: Err_m = \frac{1}{N}\sum_{i=1}^{N}\left(p_m^i - REAL^i\right) \tag{2.1}$$

among them N represents the total number of samples in the data set,Err_m represents the m-th ($1 \le m \le M$) classification error rate (accuracy index) of each network, p_m^i represents the output result of the m-th neural network on the i-th sample, $REAL^i$ represents the true value corresponding to the i-th sample, and agreed to be used in the classification task when p_m^i is equal to $REAL^i$, the difference between the two is 0, otherwise it is 1.

In terms of difference metrics, the differences between the output of neural networks and other networks are used to characterize the differences between networks. By introducing the idea of negative correlation learning [15], it can subtly transform the problem of maximizing the difference between networks into the problem of minimizing the correlation between networks. Increasing the difference between networks is to reduce the correlation or similarity between networks. The specific calculation method after converting it into an objective function is shown in formula (2.2):

$$minimize: Div_m = \sum_{i=1}^{N}\left(p_m^i - P^i\right)\sum_{j=1,j\neq m}^{M}\left(p_j^i - P^i\right) \tag{2.2}$$

among them M represents the number of basic networks in the deep network set, N represents the total number of samples in the data set, Div_m represents the difference (correlation) value between the m-th ($1 \le m \le M$) network and other networks, P^i is the average of the prediction results of the M-th network on the i-th sample, p_j^i represents the output results of the j-th neural network on the i-th sample.

The article's way reduces the difficulty of solving effectively by decomposing the multi-objective optimization problem into N scalar sub problem. First, generate a distributed average weight vector for all sub problems, and the weight vector $\lambda_i = \left\{\lambda_1^i, \ldots \lambda_F^i\right\}$ corresponding to the i-th sub problem, and then by calculating the Euclidean distance between the weight vectors corresponding to the sub problems, we can get the closest T sub problems of each sub problem (called the neighborhood), the evolution of the multi-objective algorithm is realized through the information exchange between adjacent sub-problems.

After getting the corresponding neighborhood of each subproblem, two indexes j and k are randomly selected from the neighborhood of i-th subproblem. And then get

the corresponding individuals x_g^i, x_g^j and x_g^k, get mutant individuals $x_g^{i'}$ according to the basic mutation formula of the differential evolution algorithm and add a Gaussian random variable to each dimensional value of the mutated individual according to the probability. The specific method is shown in formula (2.3):

$$x_{gx}^{i'} = \begin{cases} x_g^i + F \cdot \left(x_g^j - x_g^k\right) + rnd_G(0, \sigma) \ if \ rnd_U(0, 1) \leq 0.5 \\ x_g^i + F \cdot \left(x_g^j - x_g^k\right) \qquad\qquad otherwise \end{cases} \tag{2.3}$$

among them $rnd_U(0, 1)$ represents the fraction obtained from uniform random sampling in range of [0, 1], scaling factor $F \in [0, 2]$, $rnd_G(0, \sigma)$ represents a Gaussian random vector with mean of 0 and standard deviation σ, and the value of σ is taken as one-twentieth of the value range of the corresponding dimensional element. In the mutation process of algorithm evolution, strict boundary control is required for each dimensional element of the mutant individual. Once the corresponding maximum or minimum boundary is exceeded, then it is mapped to a reasonable range through a specific operation.

The method mentioned in this article during the evaluation stage of the implementation of the multi-objective differential evolution algorithm requires the evaluation of fitness functions for all mutated new individuals one by one, which greatly wastes computing power and time. So this paper proposes to use an evolutionary ensemble learning acceleration method based on Gaussian random field before the fitness function evaluation. Calculate the increasable probability of all new mutant individuals by establishing a pre-screening model. The fitness function evaluation was performed on individuals with higher upgradeable probability, and Individuals with lower promotion probability are directly discarded. This pre-screening model can effectively reduce the number of fitness function evaluations and speed up the solution process of the multi-objective algorithm.

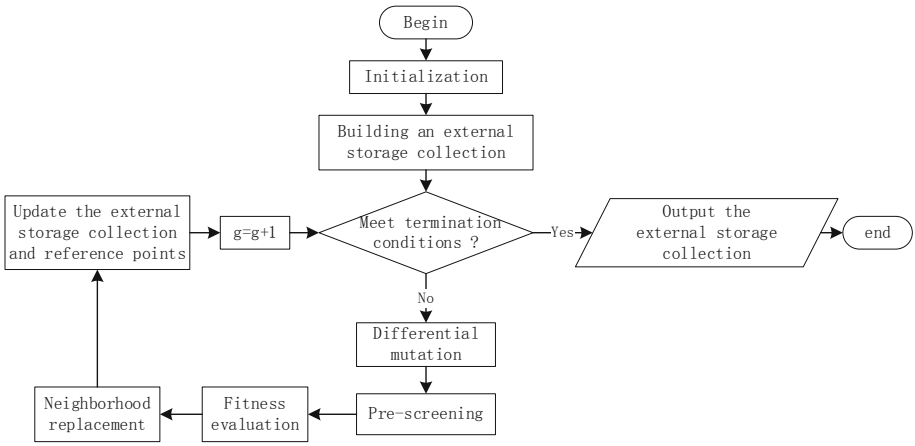

Fig. 1. The flow chart of multi-objective differential evolution algorithm based on Gaussian random field.

3 Multi-objective Differential Evolution Algorithm Based on Gaussian Random Field

As shown in Fig. 1, before the population initialization, the accuracy rate objective function is first measured and a single-objective evolutionary algorithm is used to initialize the population. The condition for stopping the iteration of this step is that the network accuracy of all individuals in a certain generation of population is greater than a certain threshold. Then all the individuals in the population are stored in the constructed external storage set, and the weight vectors and reference points are initialized to determine whether the population meets the termination conditions. If the termination conditions are not met, then the MOEA/D algorithm is used to make the difference mutation operation, and use Gaussian random field model to predict the fitness value of the mutated new individual, and use the predicted value of fitness to substitute into the sub problem corresponding to the individual to calculate the function value's *PoI* of the scalar sub problem (Possible boost value). If the value of *PoI* is greater than or equal to 0.5, it indicates that the mutant individual may be better, then it is evaluated using the true fitness function, and finally the neighborhood is replaced, and the reference point information and the external storage set are updated, while the evolution generation $g = g + 1$, then determine whether the population meets the termination conditions. If the termination conditions are met, the external storage set is directly output, and the algorithm ends.

3.1 Prediction of Fitness Value Using Gaussian Random Field Model

Michael TM et al. [16] proposed a meta-model method based on a Gaussian random field, which can effectively reduce the number of fitness function evaluations of the evolutionary algorithm in the optimization process. The basic principle is before performing the true fitness function evaluation, using the evaluated solution to build a Gaussian random field model, and then predicting the value of the function corresponding to the unknown solution. By setting a pre-screening rule, only those solutions with a large improvement space are retained, thereby achieving the purpose of reducing the true fitness function evaluation: Compared with other agent models, Gaussian random field models generally take more accurate results when making predictions because of the variance information.

The Gaussian random field model is mainly based on the following two assumptions:

(1) In terms of a time-consuming function $y = g(x), x \in R^n$, the Gaussian random field model assumes that it obeys the normal distribution of mean value μ and variance δ^2.

(2) In any $x, x' \in R^n$, $c(x, x') = exp[-d(x, x')]$ is a correlation function, represents the correlation between $g(x)$ and $g(x')$. Among them, $d(x, x') = \sum_{i=1}^{n} \theta_i |x_i - x_i'|^{p_i}$, the value of the correlation function is only related to the size of $(x - x')$. The larger $(x - x')$ is, the less correlation is and vice versa.

If we know K points $x_1, x_2, \ldots, x_K \in R^n$ and their corresponding function values are $y_1, y_2, \ldots, y_K$, Super parameter $\mu, \delta, \theta_1, \ldots, \theta_n, p_1, \ldots, p_n$ can be given by the following maximum likelihood estimate:

$$PDF = \frac{1}{(2\pi\delta^2)\sqrt{det(C)}} exp\left[-\frac{(y-\mu 1)^T C^{-1}(y-\mu 1)}{2\delta^2}\right] \tag{3.1}$$

among them C is an $K \times K$ Matrix of, and $C_{i,j} = c(x^i, x^j)$, $y = (y^1, y^2, \ldots y^K)$ as well as 1 is all K dimensional column vector. To maximize the likelihood function pdf, there must be $\mu = \frac{1^T C^{-1} y}{1^T C^{-1} 1}$, $\delta^2 = \frac{(y-1\mu)^T C^{-1}(y-1\mu)}{K}$.

The unbiased estimate of $g(x)$ is c, variance $\hat{s}^2(x) = \hat{\delta}^2\left[1 - r^T C^{-1} r + \frac{(1-1^T C^{-1} r)^2}{1^T C^{-1} 1}\right]$. Among them $r = \left(c(x, x^1), c(x, x^2), \ldots, c(x, x^K)\right)^T$, it can be considered $g(x)$ obeys $N(\hat{y}(x), \hat{s}^2(x))$.

The Gaussian random field model can effectively reduce the number of evaluations of the fitness function of the optimization algorithm in the optimization process. Although it takes a lot of time to build a Gaussian random field model, for the problem of time-consuming fitness function evaluation, the method can effectively reduce the solution time of the optimization problem.

Each time the fitness function evaluation is performed on an individual in this article, the specific neural network corresponding to the individual needs to be trained, and the training of the neural network is time consuming. In order to reduce the number of fitness function evaluations in the multi-objective algorithm, this article uses Gaussian random field model to predict fitness values for mutated new individuals.

Suppose $f_i(x)$ obeys the mean value of $\hat{y}_i(x)$, the normal distribution of variance $\hat{s}_i^2(x)$, that is $f_i(x) \sim N(\hat{y}_i(x), \hat{s}_i^2(x))$ holds for any $i = 1, 2, \ldots m$, where m represents the number of objective functions. Therefore, each independent subproblem $g^{te}(x|\lambda, z^*) = \max_{1 \le i \le m} \{\lambda_i(f_i(x) - z_i^*)\}$ has $\lambda_i(f_i(x) - z_i^*) \sim N(\lambda_i(\hat{y}_i(x) - z_i^*), [\lambda_i \hat{s}_i^2(x)])$ after decomposing the multi-objective problem based on Chebyshev decomposition method, according to the research in the article [16] we can get the normal distribution that $g^{te}(x|\lambda, z^*)$ obeys the mean value $\hat{y}^{te}$ and the variance $(\hat{s}^{te})^2$, that is, the function value of $g^{te}(x|\lambda, z^*)$ can be estimated by $\hat{y}_i(x)$ and $\hat{s}_i^2(x)$, the specific calculation method of $\hat{y}^{te}$ is shown in formula (3.2):

$$\hat{y}^{te} = \mu_1 \Phi(\alpha) + \mu_2 \Phi(-\alpha) + \tau\phi(\alpha) \tag{3.2}$$

Where $\mu_i = \lambda_i(\hat{y}_i(x) - z_i^*)$, reference point $z^* = (z_1^*, z_2^*, \ldots, z_m^*)$ for $\forall i = 1, \cdots, m$ has $z_i^* = min\{f_i(x)|x \in \Omega\}$, $\tau = \sqrt{[\lambda_1 \hat{s}_1(x)]^2 + [\lambda_2 \hat{s}_2(x)]^2}$, $\alpha = (\mu_1 - \mu_2)/\tau$, $\phi(t) = (2\pi)^{-1/2} exp(-t^2/2)$ and $\Phi(t) = \int_{-\infty}^{t} \phi(\theta) d\theta$.

$$PoI(x) = \int_{-\infty}^{f_{min}} \varphi(y) dy = \Phi\left(\frac{I(x)}{\hat{s}(x)}\right) \tag{3.3}$$

Where x represents a solution vector, $I(x) = f_{min} - \hat{y}(x)$, f_{min} represent the minimum value (unknown value) of the fitness function. This article the minimum function value obtained in the evolution process is used to replace f_{min}. $\hat{y}(x)$ as well as $\hat{s}(x)$ sees the specific definition above, $PoI(x)$ represents the possible improvement probability corresponding to the solution vector x, which is about 0.5.

First, record the individuals evaluated and their corresponding fitness function values in the set S_{eval} during the population initialization process and then each time a new mutant individual is obtained, first select $2d$ individuals from S_{eval} (d represents the dimension of the search space) during the evolution of the algorithm, and establish a Gaussian random field model, and then calculate the fitness prediction value of the mutated individual. Then use the predicted value of the fitness value to substitute into the sub-problem corresponding to the individual to calculate the function value's PoI of the sub quantum problem (Possible boost value), here PoI is defined as shown in formula (3.3), and finally predict the probability that an individual may be improved according to PoI.

3.2 Acceleration Algorithm Based on Gaussian Random Field

When the multi-objective differential evolution algorithm is executed, the final generation population obtained in advance is used as the initial population, and then the external storage set is constructed by the fitness function *outEP*, used to calculate the value of the difference objective function. After the algorithm is executed, all the individuals in the external storage set are the candidate network set. In addition, in order to prevent the individuals with lower accuracy generated during the mutation process from being excessively guided the algorithm searches in the direction of difference. Before updating the reference point, it is necessary to determine whether the accuracy of the network corresponding to the mutant individual is greater than the threshold r1. If the accuracy is greater than r1, the reference point is updated, otherwise the reference point is not updated. Where the reference point $z^* = \left(z_1^*, z_2^*, \ldots, z_m^*\right)$ has formula (3.4) for $\forall i = 1, \cdots, m$

$$z_i^* = min\{f_i(x)|x \in \Omega\} \tag{3.4}$$

When performing a neighborhood replacement operation, the i-th individual's (sub-problem)T neighboring individuals is judged, and when the formula (3.5) is satisfied, the corresponding domain individuals are replaced with mutant individuals.

$$\max_{n_f \in \{1,\ldots m\}} \lambda_{n_f}^{i} \cdot \left| z_{n_f}\left(x_g^i\right) - z_{n_f}^* \right| \leq \max_{n_f \in \{1,\ldots m\}} \lambda_{n_f}^{i_s} \cdot \left| z_{n_f}\left(x_g^{i_s}\right) - z_{n_f}^* \right| \tag{3.5}$$

Where i_s is the element of the neighborhood $B(i)$ corresponding to the i-th individual. It can be seen from the above, the accuracy of the mutant individuals is controlled here in this article, and only when the network accuracy of the mutant individuals is greater than the threshold r1 and the replacement condition shown in formula (3.5) is

satisfied, perform neighborhood replacement operation. This is to avoid the situation where individuals with higher accuracy are replaced by individuals with lower accuracy. The specific operation of the multi-objective differential evolution algorithm based on Gaussian random field is as follows:

Input M (the number of basic networks in the deep network set), T (the number of weight vectors in the neighborhood of each weight vector), G (the maximum population number), r1 (the threshold for updating outEP).

The first step is to obtain the initial population through a single-objective evolution algorithm, calculate the fitness of all individuals, and add them to outEP and S_{eval}.

The second step is to generate uniformly distributed M weight vectors $\lambda^1, \ldots, \lambda^M$. For each i in {1, …, M}, by calculating the Euclidean distance between the two vectors, find the T closest weight vectors, then add them to B(i). And initialize the reference point z^*.

In the third step, a mutant individual is generated according to formula (2.3) for each g in {1, …, G} and each i in {1, …, M}. If the elements of the mutant individuals exceed the boundary, then reset within boundary.

The fourth step is to construct a Gaussian random field model based on S_{eval} and predict $x_g^{i'}$'s fitness. Use the fitness function to predict the *PoI* of g^{te} according to formula (3.2) and formula (3.3). If the value of *PoI* is less than 0.5, it indicates that the individual is unlikely to achieve improvement, so it is abandoned to evaluate the fitness function. If the value of *PoI* is greater than or equal to 0.5, it indicates that the mutant individual may be better, and then it is evaluated using the true fitness function and added to the evaluated set S_{eval} for future use, the reference point z* is then updated, and the neighborhood is updated with formula (3.5). On this basis, if the network accuracy is greater than r1, the mutation individual is recorded and the outEP is updated. The algorithm ends.

The pseudo code of the above steps is shown in Algorithm 3.2.

Algorithm 3.2. Pseudo code of evolutionary ensemble learning acceleration method based on Gaussian random field in candidate depth network set generation.

Algorithm MOEA/D-GRF

Input

M: the number of individuals/networks in a generation
T: the number of the weight vectors in the neighborhood of each weight vector.
G: the max generations
r_1: a threshold of updating *outEP*

Output

outEP: a set of deep neural networks

Begin

1. get the initial population;
2. calculate the fitness of all individuals, and add them into *outEP* and S_{eval} ;
3. generate uniformly distributed *M* weight vectors $\lambda^1,\ldots,\lambda^M$;
4. **For** each *i* in {1,...,*M*} **do**
5. Find out *T* closest weight vectors by computing Euclidean distances between two vectors, and then add them into $B(i)$.
6. **End for**
7. initialize the reference point z^*;
8. **For** each *g* in {1,…,*G*} **do**
9. **For** each *i* in {1,…,*M*} **do**
10. generate the mutant individual $x_g^{i'}$ according to formula (2.3);
11. **If** an element of mutant individual is out of boundary **then**:
12. reset it inside the boundary;
13. **End if**
14. construct Gaussian random field model based on S_{eval} , and predict the fitness of $x_g^{i'}$;
15. compute *PoI* of g^{ie} based on the predicted fitness with formula (3.2) and formula (3.3);
16. **If** *PoI* is greater than 0.5 **then**:
17. calculate the real fitness of the mutant individual;
18. **End if**;
19. update set S_{eval} and reference point z^*;
20. update neighboring solutions with formula (3.4);
21. record the mutant individual if the network's accuracy is greater than r_1;
22. **End for**
23. update *outEP*;
24. **End for**
25. **End**

When using the algorithm proposed in this article to solve a problem of d-dimensional search space, the main steps are generating mutant individuals, establishing the high random field model, replacing individual neighborhoods, and updating reference points. Among them, the most time-consuming is to build a Gaussian random field model, whose time complexity is $O(d^3)$. Therefore, the total time complexity of the entire multi-objective evolutionary algorithm is $O(G * N * d^3)$, among them G is the evolutionary generation of the multi-objective evolution algorithm, N represents population size.

4 Experiment

In the experiments of this article, the last-generation population of the single-objective evolutionary algorithm is used as the initial population of the multi-objective optimization algorithm, which not only ensures the accuracy of the deep network corresponding to each individual in the initial population is sufficiently high, but also makes the calculation of the difference between the netwks meaningful. In order to verify the effectiveness of the multi-objective differential evolution algorithm based on the Gaussian random field in reducing running time, while the overall algorithm setting remains unchanged, five experiments were carried out to test whether the accelerated evolutionary ensemble learning method based on Gaussian random field was set up or not to eliminate the chance of the experiment and ensure the accuracy of the experimental results as much as possible. Take the average of the experimental results and round up the running time. The performance of the ensemble model in MNIST test set and fashion MNIST test set and the running time of multi-objective part are obtained under the two conditions of whether the pre- screening strategy is set or not. The specific experimental data are shown in Table 1 and Table 2.

Table 1. The influence of pre-screening strategy based on Gaussian random field on the results (MNIST).

Pre-screening strategy	Training set accuracy (%)	Verification set accuracy (%)	Test set accuracy (%)	Multi-objective algorithm running time (s)
No	99.7453	99.5840	99.4390	22432
Yes	99.7592	99.6150	99.4750	19389

With the same population size and evolutionary generation, it can be seen from the experimental data in Tables 1 and 2 that the evolutionary ensemble learning acceleration method based on Gaussian random fields can effectively reduce the running time of multi-objective algorithms under the premise of ensuring high accuracy, which is because the pre-screening strategy can reduce the algorithm's evaluate of fitness function for those individuals who have no obvious improvement possibility, thereby reducing the time to train the corresponding neural network. However, the proportion of time improvement

Table 2. The influence of pre-screening strategy based on Gaussian random field on the results (Fashion-MNIST).

Pre-screening strategy	Training set accuracy (%)	Verification set accuracy (%)	Test set accuracy (%)	Multi-objective algorithm running time (s)
No	96.7853	92.9400	92.8300	23452
Yes	96.6764	93.4800	93.1200	19947

is limited, which may be related to the time consumption of establishing a Gaussian random field. And as the number of evaluated individuals increases, the time it takes to calculate the correlation between unrated individuals and evaluated individuals also increases.

It can be seen that the ensemble model obtained by the overall algorithm of the evolutionary ensemble learning acceleration method based on the Gaussian random field is only slightly improved in the accuracy index compared to the ensemble model obtained without the pre-screening strategy. During the experiment, it was found that this situation may be related to the less evolutionary algebra of the evolutionary algorithm. The reason is that when the evolutionary algebra is less, the amount of data in the set of evaluated individuals may not be very sufficient, so when building the Gauss random field model, because the correlation between the non-evaluated solution and the known solution is weak, the fitness prediction is not particularly accurate, which leads to the limited performance improvement of the final ensemble network. In the future, more complete experiments will be used to verify this problem.

To sum up, the pre-screening strategy based on the Gaussian random field model can effectively improve the generation efficiency of candidate deep network sets and effectively shorten the running time. And at the same time, the integrated model obtained by this strategy also guarantees a high accuracy rate when verified on the MNIST and Fashion-MNIST data sets.

5 Conclusion and Future Work

This article mainly proposes an acceleration method of evolutionary ensemble learning based on Gaussian random field. This method is applied to the solution process based on the multi-objective differential evolution algorithm, which effectively reduces the number of evaluations of the adaptation function and the running time of the multi-objective algorithm, thereby making obtain the appropriate network set accurately faster. The experimental results show that the method can effectively reduce the algorithm to evaluate the fitness function of those individuals without obvious improvement. And it can conveniently obtain a large number of neural network model with high accuracy and large network differences.

In future work, consider using other methods and models to improve the pre-screening strategy, as well as using more effective methods to optimize the two metrics of accuracy and disparity of the multi-objective differential evolution algorithm.

This paper is funded by the project 202010145225 supported by National Training Program of Innovation and Entrepreneurship for Undergraduates. And this paper is supported by "the Fundamental Research Funds for the Central Universities" (project approval number: N182410001).

References

1. Dietterich, T.G.: Ensemble methods in machine learning. In: Kittler, J., Roli, F. (eds.) MCS 2000. LNCS, vol. 1857, pp. 1–15. Springer, Heidelberg (2000). https://doi.org/10.1007/3-540-45014-9_1
2. Lecun, Y., Bengio, Y., Hinton, G.: Deep learning. Nature **521**(7553), 436 (2015)
3. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: IEEE Conference on Computer Vision and Pattern Recognition. IEEE Computer Society (2016)
4. Ronneberger, O., Fischer, P., Brox, T.: U-net: convolutional networks for biomedical image segmentation. In: Navab, N., Hornegger, J., Wells, W.M., Frangi, A.F. (eds.) MICCAI 2015, Part III. LNCS, vol. 9351, pp. 234–241. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-24574-4_28
5. Girshick, R.: Fast R-CNN. Computer Science (2015)
6. Kim, J., Lee, J.K., Lee, K.M.: Accurate image super-resolution using very deep convolutional networks (2016)
7. Peimankar, A., Puthusserypady, S.: An ensemble of deep recurrent neural networks for P-wave detection in electrocardiogram. In: ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE (2019)
8. Yu, J.: A selective deep stacked denoising autoencoders ensemble with negative correlation learning for gearbox fault diagnosis. Comput. Ind. **108**, 62–72 (2019)
9. Li, J., Wu, S., Liu, C., Yu, Z., Wong, H.S.: Semi-supervised deep coupled ensemble learning with classification landmark exploration. IEEE Trans. Image Process. **29**, 538–550 (2019)
10. Jiang, W., Chen, Z., Xiang, Y., Shao, D., Zhang, J.: Ssem: a novel self-adaptive stacking ensemble model for classification. IEEE Access **7**, 120337–120349 (2019)
11. Zhang, C., Lim, P., Qin, A.K., Tan, K.C.: Multiobjective deep belief networks ensemble for remaining useful life estimation in prognostics. IEEE Trans. Neural Netw. Learn. Syst. **28**, 2306–2318 (2016)
12. Ali, I.M., Essam, D., Kasmarik, K.: A novel design of differential evolution for solving discrete traveling salesman problems. Swarm Evol. Comput. **52**, 100607 (2020)
13. Pan, L., He, C., Tian, Y., Wang, H., Zhang, X., Jin, Y.: A classification-based surrogate-assisted evolutionary algorithm for expensive many-objective optimization. IEEE Trans. Evol. Comput. **23**(1), 74–88 (2019)
14. Zhang, Q., Li, H.: MOEA/D: a multiobjective evolutionary algorithm based on decomposition. IEEE Trans. Evol. Comput. **11**(6), 712–731 (2007)
15. Liu, Y., Yao, X., Higuchi, T.: Evolutionary ensembles with negative correlation learning. IEEE Trans. Evol. Comput. **4**(4), 380–387 (2000)
16. Emmerich, M.T.M., Giannakoglou, K.C., Naujoks, B.: Single- and multiobjective evolutionary optimization assisted by gaussian random field metamodels. IEEE Trans. Evol. Comput. **10**(4), 421–439 (2006)