An Indirect Coding Neural Evolution Algorithm for Feature Selection

Suping Liu *

College of Software Engineering, Guangdong University of Science and Technology, Dongguan, Guangdong 532000, China

*Corresponding author e-mail: liusuping@gdust.edu.cn

Abstract. This paper first addresses the problem of excessively long neural network coding on high-dimensional data, and introduces the indirect coding neural evolution algorithm HyperNEAT, so that the neural network for high-dimensional data can also be coded with fewer genes, greatly reducing search space. Next, in order to solve the problem that the neuroevolution algorithm is difficult to effectively filter the features in the classification problem, this paper uses the feature-related information as the basis for the position of the input neuron. With the help of HyperNEAT, the advantage of the neuron geometric information can be effectively used, and an embedded feature is proposed. The selection method can improve the feature selection ability of the neuroevolution algorithm without increasing the time complexity.

Keywords: Neuroevolution algorithm; HyperNEAT; Feature selection.

1. Introduction

Neuroevolution (Neuroevolution) algorithm, as a training method of Artificial Neural Network (ANN), was first proposed in 1993. At the beginning, it was only used to replace the Gradient Descent method on a simple neural network [1]. Train the neural network. It encodes the neural network into genes, and finds the best combination of genes through some classic evolutionary algorithms (Evolutionary Algorithm, EA), such as genetic algorithm (GA), differential evolution (DE) [2]. Although traditional neural networks perform better on many practical problems, especially those with obvious gradient information, they usually need to artificially design the network structure, activation function and other information. Even the most experienced experts can hardly guarantee the designed network is the most effective and streamlined. The neuroevolution method can not only evolve the weights, but also evolve the network structure, activation function and even learning rules [3]. While greatly reducing the influence of human factors, it is possible to obtain a more efficient model. In addition, as a kind of evolutionary algorithm, the neuroevolution method has very good parallel characteristics. Although the speed of a model is not as fast as the traditional neural network training method, the parallel training method may actually require more time [5]. Although neuroevolution has achieved good results in the classification and clustering of low-dimensional data sets, it performs poorly on high-dimensional data sets. The main reason is that the neural network will become very complicated when facing high-dimensional data. Long genes are needed to encode the neural network, and the spatial complexity is too high; indirect encoding means that the weight and sum of the neural network are not directly
encoded. The structure is a coding method that regenerates the corresponding neural network by coding a simple function, which greatly reduces the number of genes required to code a neural network. Therefore, it provides a new idea for the coding of neural network models on high-dimensional data. Feature selection refers to the method of filtering the features of the data to exclude redundant and irrelevant features, thereby improving the performance of the model [6].

This thesis first addresses the problem of long neural network coding on high-dimensional data, and introduces the indirect coding neuroevolution algorithm HyperNEAT, so that the neural network for high-dimensional data can also be coded with fewer genes, which greatly reduces the search space. Next, in order to solve the problem that the neuroevolution algorithm is difficult to effectively filter the features in the classification problem, this paper uses the feature-related information as the basis for the position of the input neuron. With the help of HyperNEAT, the advantage of the neuron geometric information can be effectively used, and an embedded feature is proposed. The selection method can improve the feature selection ability of the neuroevolution algorithm without increasing the time complexity.

2. Theory and method

2.1. Indirect encoding

Unlike direct coding, the indirect coding method does not directly describe the weights and connections of the neural network. It encodes a functional formula, and then generates the corresponding neural network through the function, that is, establishes the mapping between genes and neural networks. In general, the number of genes required to encode a functional formula is much smaller than the number of genes required to directly encode a neural network. Therefore, by simply recording the network structure in a functional manner, and then generating the corresponding neural network, the coding of the complex neural network can be realized through shorter genes. Through indirect coding, no matter how high the data dimensionality is and how complex the network is, the length of the coding will not increase too much, which can effectively solve the problem of neuroevolutionary coding on high-dimensional data. As shown in Fig.1, CPPNs is a structure similar to a neural network [7].

![Fig. 1 A typical CPPNs](image)

In addition to the indirect coding of neural networks, the advantage of CPPNs lies in the combination of different activation functions to generate a network structure with symmetry and regularity, which is also the same as the nature of the human brain with a large number of repetitive patterns. Through symmetry and regularity, CPPNs effectively compresses the originally complex patterns, and at the same time avoids the excellent structure in the direct encoding method, the redundancy that needs to be repeatedly discovered in different parts, and improves the overall discovery. Possibility of structure.
2.2. **HyperNEAT algorithm**

Before applying indirect coding to the problem of high-dimensional data, it is obviously necessary to determine the method of converting CPPNs to the corresponding neural network. Hypercube-based NEAT (HyperNEAT) is a non-direct coding-based neuroevolution algorithm proposed by Stanley on the basis of CPPNs. It uses the CPPNs network as an indirect encoding method to establish a mapping between the geometric position of the neuron and the weight of the neural network to generate the final corresponding neural network. The general process is as follows:

First of all, in the preparation stage of the algorithm, it is necessary to determine the geometric position and activation function of the neurons in the neural network according to actual problems. For the neurons of the layer, HyperNEAT uses a fully connected method to connect. In this way, the structure of the neural network except for the weights has been determined. In order to distinguish it from the structure of CPPNs, this paper refers to the pre-designed neural network structure as a substrate.

Then, like the classic neuroevolution method, the CPPNs network is first coded. If the NEAT algorithm is used to evolve CPPNs, the coding method follows the coding method in NEAT, and the neurons and connections are represented by different genes. In HyperNEAT, the number of input and output neurons of CPPNs is usually determined by the basic network structure. Since CPPNs records the weights of pairwise neurons in the basic network, the number of inputs of CPPNs is twice the dimension of the neurons in the basic network. After determining the basic network structure, the structure of CPPNs, and the coding of genotypes, the next process performs crossover and mutation operations on individuals in the population, that is, genotypes representing CPPNs. But unlike the classical neuroevolutionary method, when evaluating individuals, one cannot simply directly evaluate the CPPNs network. Instead, it should be combined with the basic network structure, and the corresponding real network should be constructed first through CPPNs. After that, evaluate the effectiveness of the generated network in the problem as the fitness value of the individual, and participate in the selection process to decide whether to keep it. The pseudo code for constructing the real network through CPPNs and the basic network structure is shown in Table 1.

![Table 1. Algorithms for constructing corresponding networks from CPPNs](image)

3. **HyperNEAT algorithm based on feature information**

For high-dimensional data, HyperNEAT is the same as other networks. The weight adjustment and structural evolution of the network itself actually incorporate feature selection into the search space. However, because the search space of high-dimensional data is too large, this feature selection method is more effective. Difference. With the help of the geometric position of neurons to determine the weight distribution of HyperNEAT to a certain extent, this paper considers the feature information as the basis for the distribution of neurons in a certain dimension, so as to provide additional feature information for the network and guide the feature selection of the algorithm. Statistical feature correlation is an effective indicator for evaluating features. At the same time, its computational complexity is usually low, much less than the time complexity of neuroevolution itself, so it is very suitable for adding additional feature information to the evolution of the network. Therefore, this paper proposes an embedded feature selection method, which determines the location of neurons in the basic network structure in
HyperNEAT through feature-related information, thereby affecting the weight distribution of the neural network. In this way, the weight of the feature can be adjusted according to the importance of the feature or even some invalid features can be removed, and the features on the high-dimensional data can be effectively screened.

However, what kind of feature-related information is used as the basis for the location of neurons in HyperNEAT is also a question worthy of discussion. First of all, if the correlation between the two features is used as the feature information, it is obviously difficult to find the neuron position distribution that satisfies the distance relationship in a plane. At the same time, when the feature scale is large, the time complexity of calculating the correlation between the two features is also high. Therefore, this paper uses the correlation between features and their corresponding categories as the basis for designing neuron locations. Commonly used correlation measurement methods in feature selection are correlation coefficient method and chi-square test method.

Chi-square test. The chi-square test method is used to test the correlation between independent variables and dependent variables, and has also been widely cited in feature selection. The chi-square test method first counts the relevant information of the characteristics and categories, that is, the true value, and calculates the expected value in the case of mutual independence, and finally calculates the gap between the expected value and the true value. The formula is as follows [8]:

$$x^2 = \sum \frac{(A-T)^2}{T}$$  \hspace{1cm} (1)

Among them, A is the true value and T is the expected value. The greater the deviation between the actual observation value and the expected value, the greater the chi-square value, and the greater the correlation between the feature and the predicted value. In the conventional chi-square check, you should judge whether the hypothesis is reliable according to the critical value table, but because this paper only uses the chi-square value to determine the neuron position, there is no need to judge, just give this step to the subsequent neuron evolution. Chi-square test is suitable for classification problems and can well measure the relationship between variables and their categories. Therefore, this paper uses chi-square test as one of the characteristic information tested in the experiment.

Mutual information. Mutual information comes from information theory, which describes information through the knowledge of probability theory and mathematical statistics, and has been widely used in feature selection, especially filtering feature selection methods. In order to understand mutual information, this paper first introduces the concept of information entropy. Information entropy is a very important concept in information theory, which is used to measure the amount of information. If the entropy of a variable is greater, the uncertainty of the random variable it represents is greater, and the corresponding amount of information is also greater. Conversely, if the entropy of a variable is smaller, it contains less information. For random variable X={X_1, X_2,...,X_W}, the calculation formula of its information entropy H(X) is as follows [9]:

$$H(X) = \left[ -\sum_{i=1}^{W} p(x_i) \log(p(x_i)) \right]$$ \hspace{1cm} (2)

Where p(x_i) represents the probability function of x_i distribution. Mutual information is an effective indicator to measure the correlation between different variables, which describes the shared information between two random variables. For two random variables, the joint distribution is p(x, y), and the marginal distributions are p(x) and p(y). Then the mutual information I(X, Y) is the relative of the joint distribution and the marginal distribution. Entropy, namely

$$I(X, Y) = \sum_{x \in X} \sum_{y \in Y} p(x, y) \log \frac{p(x, y)}{p(x)p(y)}$$ \hspace{1cm} (3)
Mutual information can effectively measure the related information of features and categories, so mutual information is also one of the feature information selected in the experiment of this chapter.

Classifier testing. In addition to information indicators based on statistics, a simple encapsulated algorithm is used to test the classification accuracy set of each feature on the data as one of the feature information of the experiment. This method uses actual models for testing, so it is more realistic than statistical-based metrics, but its time complexity is also higher. KNN, as the most commonly used classifier for performance testing in feature selection research, has time characteristics of low complexity and good classification effect. Therefore, KNN is selected as the actual model in this paper to test each feature.

4. Experimental results and analysis

4.1. Data set selection

In order to fully verify the influence of the feature selection operation on the neuroevolution algorithm, this paper selects 8 groups of UCI classification data sets for testing, as shown in Table 2. These 8 sets of data include different number of features, number of instances, and number of categories, which can cover various situations well and avoid the contingency of the final experimental results.

| Name of data set | Number of features | Number of instances | Number of categories |
|------------------|--------------------|---------------------|---------------------|
| Wine             | 13                 | 178                 | 3                   |
| WBCD             | 30                 | 569                 | 2                   |
| Ionosphere       | 34                 | 351                 | 2                   |
| Hill Valley      | 100                | 606                 | 2                   |
| Musk Version     | 166                | 476                 | 2                   |
| Isolet5          | 617                | 1559                | 26                  |
| BioResponse      | 1777               | 3751                | 2                   |
| Gisette          | 5000               | 6000                | 2                   |

4.2. Evaluation index

The accuracy and the number of features are used to evaluate the effect of classification. The formula for calculating the accuracy rate is:

\[
\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}
\]

Among them, TP (True Positive) represents the number of positive samples correctly classified, TN (True Negative) represents the number of negative samples correctly classified, FP (False Positive) represents the number of positive samples incorrectly classified, and FN (False Negative) represents the negative Number of samples.

4.3. Experimental results and analysis

Table 3 shows the classification accuracy of the best individual on the test set and the corresponding feature number of each algorithm in 20 independent experiments. It can be seen from the experimental results that the HyperNEAT algorithm basically retains all the features before the feature information is introduced, but after the feature measurement information is introduced, the retained features are significantly reduced, and the performance of the algorithm is improved to varying degrees.
Table 3. The accuracy of the best individual classification and the number of corresponding features

| Name of data set | HyperNEAT accuracy | HyperNEAT-C accuracy | HyperNEAT-M accuracy |
|------------------|--------------------|----------------------|----------------------|
| Wine             | 0.6603             | 0.6792               | 0.7358               |
| WBCD             | 0.8464             | 0.8823               | 0.9058               |
| Ionosphere       | 0.7904             | 0.8095               | 0.8190               |
| Hill Valley      | 0.5635             | 0.5690               | 0.5911               |
| Musk Version     | 0.5774             | 0.6267               | 0.6760               |
| Isolet5          | 0.0899             | 0.0963               | 0.1049               |
| Bio Response     | 0.6880             | 0.7128               | 0.7208               |
| Gisette          | 0.8055             | 0.8205               | 0.8288               |

It can be seen from the experimental results that the HyperNEAT algorithm basically retains all the features before the feature information is introduced, but after the feature measurement information is introduced, the retained features are significantly reduced, and the performance of the algorithm is improved to varying degrees.

5. Conclusion

In the neural evolution algorithm for high-dimensional data, direct coding cannot effectively code the required complex neural network, and the introduction of indirect coding solves this problem well. At the same time, although the neuroevolution method itself can remove part of the input features through structure and weight evolution, the effect is poor. Therefore, additional feature selection can effectively improve its performance, especially on high-dimensional data sets with many redundant and irrelevant features. On the basis of HyperNEAT, relying on its advantages of effective use of spatial geometric information, using mutual information to determine the location of neurons can improve feature selection capabilities and algorithm performance without increasing time complexity.

References

[1] Risi S, Stanley K O. Indirectly encoding neural plasticity as a pattern of local rules[C]//International Conference on Simulation of Adaptive Behavior. Springer, Berlin, Heidelberg, 2010:533-543.
[2] Tan Z, Li K. Differential evolution with mixed mutation strategy based on deep reinforcement learning [J]. Applied Soft Computing, 2021: 107678.
[3] Risi S, Stanley K O. An enhanced hypercube-based encoding for evolving the placement, density, and connectivity of neurons [J]. Artificial Life, 2012, 18(4):331.
[4] Long J, Zhang S, Li C. Evolving deep echo state networks for intelligent fault diagnosis [J]. IEEE Transactions on Industrial Informatics, 2019, 16(7):4928-4937.
[5] Marill T, Green D. On the effectiveness of receptors in recognition systems [J]. IEEE transactions on Information Theory, 1963, 9(1):11-17.
[6] Maxine Tan, Michael Hartley, Bister M, et al. Automated feature selection in neuroevolution [J]. Evolutionary Intelligence, 2009, 1(4):271-292.
[7] Stanley K O, Clune J, Lehman J, et al. Designing neural networks through neuroevolution [J]. Nature Machine Intelligence, 2019, 1(1).
[8] Bharti K K, kumar Singh P. A survey on filter techniques for feature selection in text mining[C]//Proceedings of the Second International Conference on Soft Computing for
Problem Solving (SocProS 2012), December 28-30, 2012. Springer, New Delhi, 2014:1545-1559.

[9] Vergara J R, Estévez P A. A review of feature selection methods based on mutual information [J]. Neural computing and applications, 2014, 24(1):175-186.