International Journal for Housing Science and Its Applications
Publish August 6, 2025. Volume 46, Issue 3 Pages 6124-6134

https://doi.org/10.70517/ijhsa463526

A neural network-based systematic assessment of programming game interaction behavior of 0-6 year olds in an educational big data environment

Yuanyuan Su[1,*] and Xianda Sun[2]
[1] Faculty of Education Sciences, Jilin Normal University, Siping, Jilin, 136000, China
[2] Assets Department, Jilin Normal University, Siping, Jilin, 136000, China
Corresponding authors: (suyuanyuan1999@126.com).

Abstract Existing methods for assessing interaction behavior in programming games for young children have problems such as insufficient accuracy and lack of systematicity. How to scientifically assess the interaction behavior of young children in programming games and accurately grasp the characteristics of teacher-student verbal interaction has become the key to improving the quality of early childhood programming education. In this paper, we constructed an evaluation index system for the interaction behavior of programming games for 0-6 years old children, and proposed an evaluation model based on the Improved Particle Swarm Algorithm Optimized Radial Basis Neural Network (IPSO-RBF). The system contains two primary indicators, teacher speech behavior and student speech behavior, subdivided into 12 secondary indicators. Methodologically, nonlinear dynamic inertia weights and dynamic learning factors are used to improve the traditional PSO algorithm and optimize the width and weight parameters of the RBF neural network. The model performance is verified by 200 sets of training samples and 50 sets of test samples, and the results show that the test correct rate of IPSO-RBF neural network reaches 96%, and the average value of MSE is 0.121, which is significantly better than the 89.7% correct rate and 0.355 MSE value of PSO-RBF. Eleven programming teaching activities for 60 young children were analyzed for three types of instructional models: didactic, demonstrative, and socially constructive, and it was found that demonstrative classrooms had the highest number of significant sequences (27), followed by socially constructive (24), and didactic had the fewest (14). This study provides an effective tool for assessing the quality of early childhood programming education and is an important reference for optimizing teaching strategies.

Index Terms Early childhood programming games, interaction behavior assessment, IPSO-RBF neural network, educational big data, teacher-student verbal behavior, teaching model

I. Introduction

The 21st century is the era of digitization and artificial intelligence, and such a highly technologically transformative era places new demands on talent [1].To maintain competitiveness in the era of digitization and artificial intelligence, 21st century talent needs to have future-oriented computational mindsets and skills. If a person lacks computational thinking skills, it will be difficult for him to be invincible in industry competition [2], [3]. If a country fails to develop computational thinking among its citizens, it will struggle to take the initiative in international competition [4].Young children in the 21st century are digital natives of the digital and computational age, who have grown up living in a computer and networked environment and are accustomed to using computers and networks to learn and interact with their peers [5]. Computational thinking is precisely the new pervasive thinking skill that has arisen in the digital and computational era, and it is a cognitive activity that flexibly uses computational tools and methods to solve problems [6], [7]. It is evident from international experience that programming education has a role in developing computational thinking in students [8]. The background of the era of digitalization and artificial intelligence provides a certain good foundation for developing children's programming education and cultivating children's computational thinking.

Programming in foreign countries started earlier, the earliest recognition is the highest is the programming software is Logo and scratch, the initial learning programming audience groups are mainly teenagers [9]. 2013, Obama proposed and carried out the programming hour activities, and then the United States pay more attention to the early childhood programming education, and vigorously promote the KIBO early childhood programming education courses [10]. The findings of Sullivan, A and Bers, M. U showed that KIBO programs allow young children to learn about areas such as technology and engineering and have a positive effect on language, creativity and social interaction [11]. Programming also hones young children's metacognitive, problem-solving and reasoning

skills [12]. Through programming games, young children implicitly learn math and science in the form of sequencing, estimating, and counting while engaging in engineering design [13]. Another study by Sullivan, A et al. supports that programming education can be realized not only in traditional activities, but also in combination with early STEM education for young children [14].

Young children are able to master basic concepts and improve their learning of mathematics in programming courses, as well as to create rich creations, try to solve problems in the learning process, and express themselves through programming [15]. Rose, S et al. explored the use of Scratchr and Lightbot programming tools by young children to develop computational thinking skills and found that children were exposed to Scratchr-like programming programs in different ways depending on their abilities [16]. Mladenović, S et al. explored the use of a game-making curriculum that could improve young children's attitudes towards learning by engaging in programming games in a more engaging and fun way, developing good motor skills and hand-eye coordination [17]. A study by Critten, V et al. found that preschool children aged 0-6 years old can develop computational thinking skills, as well as concepts such as cooperation, logical thinking, and debugging through guided programming game activities [18]. In summary, a review of the current state of research on programming education reveals that through programming education, countries aim to cultivate computational thinking, an essential basic literacy for contemporary children, which, like reading, writing, and arithmetic, serves as a cultural cornerstone to help children cope with the challenges of the digital society, and at the same time cultivate more computer science talents for the future society [19].

In recent years, programming education has gradually developed to a lower age, and programming game activities for young children aged 0-6 are widely carried out in various educational institutions. Programming games can not only cultivate young children's logical thinking ability, but also enhance their problem solving ability and creativity. Under the background of educational big data, how to effectively assess the interaction behavior of young children in programming games and accurately reflect the teaching effect has become an important issue in current educational research. Traditional assessment methods mostly rely on subjective observation and empirical judgment, lack objectivity and systematicity, and are difficult to accurately capture the complex interactive characteristics of young children's programming learning process. Teacher-student verbal interaction, as a core element of teaching activities, has a direct impact on young children's learning outcomes and cognitive development. There are significant differences in teacher-student interaction patterns under different teaching modes, which have different impacts on young children's learning engagement and knowledge construction. Therefore, it is of great significance to establish a scientific assessment system and use intelligent algorithms to improve the assessment accuracy to enhance the quality of early childhood programming education. This study adopts a systematic research idea, firstly, based on pedagogical theories and practical observations, to construct an evaluation index system containing teachers' verbal behavior and students' verbal behavior. Secondly, to address the shortcomings of traditional evaluation methods, neural network technology is introduced and the IPSO-RBF evaluation model is proposed by optimizing the network parameters through the improved particle swarm algorithm. Finally, actual teaching cases are selected to empirically analyze the teacher-student interaction behaviors under different teaching modes to verify the effectiveness and practicality of the model. Through the combination of theoretical construction, algorithm optimization and empirical testing, a complete assessment framework is formed.

II. Evaluation index system of programming game interaction behavior of 0-6 years old children

Based on the principle of constructing the index system, this study constructed an evaluation index system for the interactive learning needs of programming games among children aged 0-6 years old, as shown in Table 1. The first-level indicators are divided into teachers' speech acts and students' speech acts. In the dimension of teacher-student interaction language, the four types of codes representing teachers' feedback on students' viewpoints or performances, such as "teacher acceptance emotion", are combined into "Feedback", which centrally represents teachers' feedback on students' viewpoints or performances. Combine "asking open-ended questions" and "asking closed-ended questions" into "asking questions"; Divide "discussing with peers" into three categories: passive response, active response and feedback, in order to reflect the situation of peer discussion more comprehensively.

III. IPSO-RBF-based interactive behavior assessment model for programming games for young children

In the previous section, this paper has constructed an evaluation index system of programming game interaction behavior of 0-6-year-old children based on the teacher-student speech behaviors of 0-6-year-old children and ECE teachers. In order to further realize the evaluation of the interaction behavior of programming game for 0-6-year-old children on this basis, this chapter will propose an evaluation model of particle swarm algorithm optimized radial

basis neural network (IPSO-RBF) based on improved inertia weight factor and learning factor, which improves the accuracy of the evaluation of the interaction behavior of programming game for 0-6-year-old children.

Table 1: Children's programming game interaction behavior evaluation index

Evaluation index system	Classification	Coding	Teaching
0-6 years old children 's programming game interaction behavior evaluation index system	Teachers ' speech act	TS	Questioning
		TQ	Feedback
		TF	instruction
		TD	Host
		TC	Active atmosphere
		TA	Content
	Students ' speech act	SR	Answer
		SQ	Questioning
		SS	Explain / Share
		SF	Feedback
		SC	Host
		SH	Apply for assistance

III. A. RBF Neural Network Principles and Learning Algorithms

RBF neural network consists of input, hidden and output layers [20]. In this case, the transformation function of the neurons in the hidden layer is responsible for transforming a linearly indivisible problem into a linearly divisible one.

In the structure, n denotes the number of nodes in the input layer, that is, the number of input data x; h denotes the number of neurons in the hidden layer: m denotes the number of nodes in the output of the network; ω is the weight from the hidden layer to the output layer; b is the threshold; $\phi(\cdot)$ represents the activation function of the hidden layer, which is commonly used as the Gaussian function of Eq. (1):

$$\phi(\mu) = e^{\frac{\mu^2}{2\delta^2}} \tag{1}$$

The Gaussian function is chosen as the activation function, and the independent variable is the Euclidean distance $\|\cdot\|$ between the input samples to a center point, then the output expression of the hidden layer is:

$$\phi(\| x - c_i \|) = e^{-\frac{\|x-c_i\|^2}{2\delta^2}}, i = 1,2,...,h \tag{2}$$

where $\| x - c_h \|$ is the Euclidean distance, c_h is the data center, and δ denotes the function's expansion constant, i.e., width.

The $\sum$ denotes the linear function used by the neurons in the output layer, and the output of the network structure is:

$$y_j = \sum_{i=1}^{h} \omega_{ij} e^{-\frac{\|x-c_i\|^2}{2\delta^2}}, j = 1,2,...,m \tag{3}$$

The RBF neural network requires three parameters, the data center, the expansion constant, and the weights from the implicit layer to the output layer, and the different parameters are determined as follows.

1) The data center c_i can be used by K-means clustering center method. Firstly, the initial data center is selected, and h data are randomly chosen as data center $c_i (i = 1,2,....,h)$, and establish the corresponding data center class for each data center. Calculate the distance of the remaining data to each center class, add it to the nearest center class based on the distance, calculate the average value of the data in each class, and use it as the new data center in the respective class $c_i (i = 1,2,....,h)$.

2) The expansion constant δ can be determined by $\delta = \frac{d}{\sqrt{2h}}$, d is the maximum distance of all classes and h is the number of data centers.

3) A least squares method is used to determine the weights ω. When the input layer data is x_j, the output of the i th hidden layer neuron is:

$$h_{ji} = \varnothing_i(\| x_j - c_i \|) = e^{-\frac{h\|x_j - c_i\|^2}{c_{max}^2}} \tag{4}$$

The output matrix of the implicit layer is $\hat{H} = [h_{ji}]$ and the actual output of the output layer is:

$$\hat{y} = \hat{H} \cdot \omega \tag{5}$$

ω can be obtained by the least squares method:

$$\omega = \hat{H}^{+} y \tag{6}$$

where $\hat{H}^{+}$ is the pseudo-inverse of $\hat{H}$.

III. B. PSO algorithm implementation process

The PSO algorithm is motivated by the behavioral patterns of bird flocks, which are modeled in a simplified way using group intelligence, and the ultimately unstructured solution process is made traceable by sharing among the particles in the population, which eventually leads to the optimal solution of the whole process. Each particle is a potential solution, i.e. the position of each bird in the flock. Individual extremes ($Pbest$) and global extremes ($Gbest$) are updated according to the values of the fitness function. $Pbest$ is the position with the best fitness value among the positions experienced by the particle itself; $Gbest$ is the optimal position found by all the particles.

Setting in D -dimensional space, there are m particles X in the particle swarm X $X = (X_1, X_2, ..., X_m)^T$, each particle is a vector $X_i = (x_{i1}, x_{i2}, ..., x_{iD})$, which represents the position of the particle, and the velocity is given by $V_i = (v_{i1}, v_{i2}, ..., v_{iD})$ is denoted by $Pbest$ as $P_i = (p_{i1}, p_{i2}, ..., p_{iD})$, $Gbest$ is $P_g = (p_{g1}, p_{g2}, ..., p_{gD})$. The velocity of the particle is updated by Eq. (1) and the position of the particle is updated by Eq. (8):

$$v_{id} = v_{id} + c_1 \times rand() \times (p_{id} - x_{id}) + c_2 \times rand() \times (p_{gd} - x_{id}) \tag{7}$$

$$x_{id} = x_{id} + v_{id} \tag{8}$$

where $rand()$ is a random number between 0-1; c_1, c_2 are non-negative acceleration factors, c_1 describes the tendency of the particles to bias towards the individual extremes, and c_2 describes the tendency of the particles to bias towards the global extremes.

Equation (7) obtains the optimal result by fixed value, compared with dynamic optimization searching can obtain better optimization searching results, and Equation (9) realizes particle updating by introducing dynamic inertia factor:

$$v_{id} = \omega \times v_{id} + c_1 \times rand() \times (p_{id} - x_{id}) + c_2 \times rand() \times (p_{gd} - x_{id}) \tag{9}$$

where ω denotes the inertia factor, which varies linearly during the particle optimization process. The larger its value is, the more favorable the global search, and the smaller its value is, the more favorable the local search. In order to balance the global and local search ability, the linear decreasing weights strategy is introduced.

The ω of the linear decreasing weights strategy is obtained as follows:

$$\omega^k = \frac{(\omega_{ini} - \omega_{end})(K_{\max} - k)}{K_{\max}} + \omega_{end} \tag{10}$$

where $K_{\max}$ is the upper limit of iteration number, ω_{ini} is the initial value of inertia weight, ω_{end} is the inertia weight when $K_{\max}$ is reached, and k is the current number of iterations.

The main process of PSO algorithm for optimization is as follows:

Step1: Initialize the position and velocity of the particle.

Step2: Calculate the initial fitness value and find the initial $Pbest$ and $Gbest$.

Step3: Update the velocity and position of the particles.

Step4: Calculate the adaptation value of each particle and update the $Pbest$ and $Gbest$.

Step5: If the maximum number of iterations is reached or the adaptation threshold is satisfied, output $Gbest$; otherwise return to Step3.

III. C. Optimization of RBF neural network algorithm

III. C. 1) IPSO algorithm improvement strategy

Particle Swarm Algorithm (PSO) is a common bio-inspired optimization algorithm which finds the optimal solution through collaboration and information sharing among individuals in the population [21]. The PSO algorithm has the following velocity and position updating formulas during each iteration:

$$v_{id}^{k+1} = wv_{id}^{k} + c_1 r_1 (p_{id,pbest}^{k} - x_{id}^{k}) + c_2 r_2 (p_{d,gbest}^{k} - x_{id}^{k}) \tag{11}$$

$$x_{id}^{k+1} = x_{id}^{k} + v_{id}^{k+1} \tag{12}$$

where v_d^{k+1} denotes the current velocity of particle i in the d-dimension at the $k+1$th iteration; x_u^{k+1} denotes the current position of particle i in the d-dimension at the $k+1$th iteration, respectively; and c_1, c_2 denote the individual learning factor and social learning factor, respectively; r_1, r_2 denote 2 random numbers taking values in the interval [0, 1]; w denotes the inertia weight factor.

However, the traditional particle swarm algorithm is easy to be trapped in the local optimal solution, and this paper improves the inertia weight factor and learning factor in 2 aspects:

1) Improvement of particle inertia weight factor. The inertia weight factor is generally calculated by linearly decreasing in the traditional particle swarm algorithm, as shown in equation (5). In order to balance the search effect of the algorithm more effectively, this study adopts the nonlinear dynamic inertia weight to improve it, and the improved inertia weight formula is shown in Equation (13):

$$w = w_{\max} - (w_{\max} - w_{\min}) \times \frac{k}{k_{\max}} \tag{13}$$

$$w = \begin{cases} w_{\min} - \dfrac{(w_{\max} - w_{\min}) \times (f - f_{\min})}{(f_{avg} - f_{\min})}, f \le f_{avg} \\ w_{\max}, f > f_{avg} \end{cases} \tag{14}$$

where $k_{\max}$ is the maximum number of iterations; $w_{\max}, w_{\min}$ are the maximum inertia weight and the minimum inertia weight, which are generally taken to be 0.9 and 0.4, respectively. f is the real-time objective function value of the particles, and $f_{avg}, f_{\min}$ are the average and minimum objective values of all the particles in real time. average and minimum target values of all current particles, respectively.

2) Improvement of dynamic learning factor. The learning factors c_1 and c_2 in the traditional particle swarm algorithm are improved by the dynamic adjustment method, which expands the search range in the initial stage of evolution and improves the efficiency of the later optimization process. The improved dynamic learning factor formula is shown in equation (15):

$$\begin{cases} c_1 = 2 - 5 + 2(k / k_{\max})^2 - 2(2k / k_{\max}) \\ c_2 = 3 - c_1 \end{cases} \tag{15}$$

III. C. 2) Implementation of IPSO optimized RBF neural network

The parameter training method is a key factor affecting the performance of RBF neural networks. Intelligent optimization algorithm is usually chosen instead of the traditional training algorithm to optimize the parameters of the neural network as an optimization-seeking target. In this study, the fuzzy clustering algorithm is used to calculate the center of the RBF neural network, and the IPSO algorithm is used to optimize the width and weight parameters of the RBF neural network, and the IPSO-RBF neural network model is established [22].The convergence speed of the IPSO algorithm and whether it is able to find the optimal solution are critically important in the stem fitness function, and the fitness function for the optimization of the IPSO algorithm is chosen to be RBF in this model. Neural network mean square error. The particle fitness function formula is as follows:

$$\mathrm{MSE} = \frac{1}{n} \sum_{i=1}^{n} (Y_i - y_i)^2 \tag{16}$$

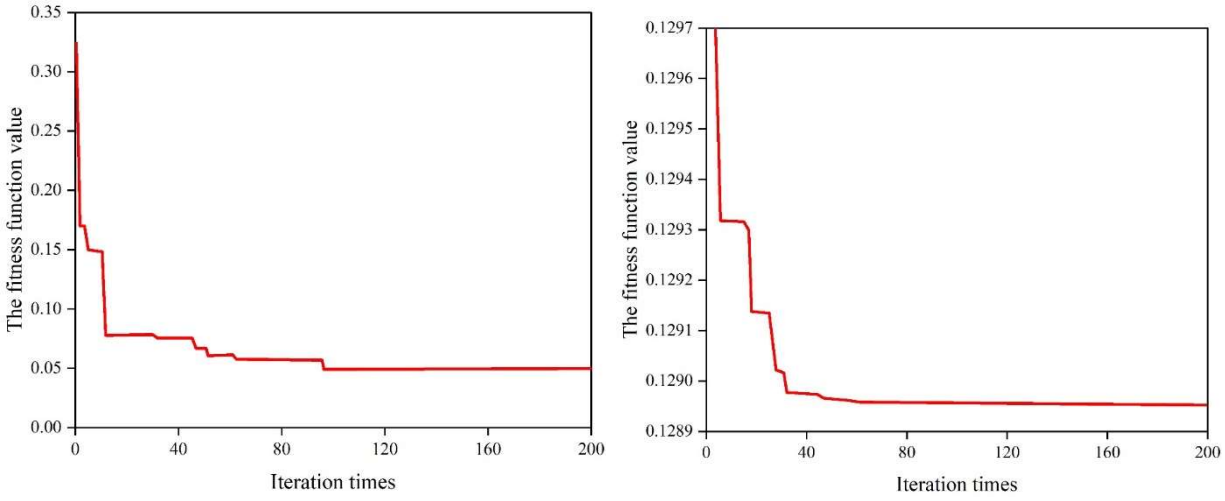
where n is the number of training samples; Y_i is the reference output; and y_i is the actual output.

IV. IPSO-RBF neural network simulation test

In this chapter, the proposed IPSO-RBF neural network-based interaction behavior assessment model for toddler programming game proposed in this paper will be simulated and tested to examine the performance of IPSO-RBF neural network.

IV. A. Analysis of training results

200 sets of programming game interaction behavior data samples of 0-6 years old toddlers were selected for training, and the convergence diagrams of PSO-RBF and IPSO-RBF neural networks obtained are shown in Fig. 1, and Figs. (a) and (b) correspond to PSO-RBF and IPSO-RBF neural networks, respectively. From the convergence of the IPSO-RBF neural network, the accuracy of the network is further improved and the convergence is slightly faster compared to the PSO-RBF neural network model.



(a)PSO-RBF (b)IPSO-RBF

Figure 1: Training convergence diagram

The prediction accuracy of the network needs to be further judged by test samples. The trained IPSO-RBF neural network is fed with 50 groups of sample data, and the test results are shown in Fig. 2. From the test result graph, only 2 groups of 50 test samples are misjudged, and the correct rate of the test reaches 96%, which matches the training results.

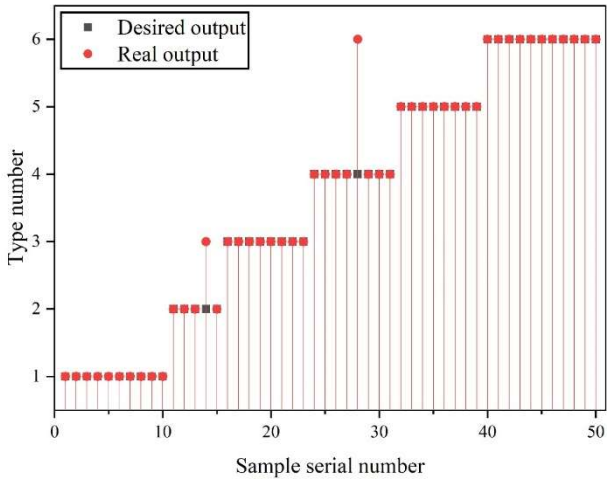


Figure 2: PSO-RBF neural network test result diagram

Some of the IPSO-RBF neural network test results are specifically shown in Table 2. Most of the samples of IPSO-RBF neural network are judged correctly, however, the 30th group of test samples are judged incorrectly, but

the main reason may be that the number of samples is not enough. On the contrary, in the case of insufficient number of samples, judging from the training results and test results, the convergence speed and optimization searching ability of IPSO-RBF neural network are still considerable, which indicates that the IPSO-RBF neural network model has better practicality and robustness.

Table 2: Test result data(part)

Sample number	Sample input (μL/L)					Expected output	Actual output
	H2	CH4	C2H6	C2H4	C2H2		
8	20.434	17.025	52.822	5.667	4.023	1	1
14	26.6	90.5	95	49.2	0.5	3	3
19	18.4	110.2	50.6	163.2	0.7	3	3
30	982	73	12	58	0	4	6
35	1667	651.8	1004.8	80.6	418.2	5	5
45	4.1	3.5	1.2	0.62	5.3	6	6

IV. B. Comparative analysis of results

In order to further verify the stability and accuracy of the IPSO-RBF neural network, the trained single RBF neural network, the standard PSO-RBF neural network and the IPSO-RBF neural network are compared. Among all the samples, 50 groups were randomly selected as test samples, and the three network models were simulated independently for 30 times, and the same test samples were input to each simulation for the three network models, and the simulation platform was MATLAB R2018b. The average MSE, the average diagnosis correctness, and the average network computing time (the time used from the start of the algorithm to the stopping of the iteration) were calculated for each network. The computational results for each of the three network models are shown in Table 3. From the test results, the RBF neural network has an average correct rate of only 80.5% and an average number of misjudged samples of about 10 groups, the PSO-RBF neural network has an average correct rate of 89.7% and an average number of misjudged samples of about 5 groups, while the IPSO-RBF neural network has an average correct rate of 95.7% and an average number of misjudged samples of 2.5 groups.

Table 3: Comparison of test results of three network models

Network model	MSE			Accurate rate (%)			Running time (s)		
	Best value	The worst value	Average value	Best value	The worst value	Average value	Best value	The worst value	Average value
RBF	0.436	0.532	0.504	90	75	80.5	1.192	1.551	1.339
PSO-RBF	0.306	0.387	0.355	95	85	89.7	1.015	1.291	1.149
IPSO-RBF	0.042	0.134	0.121	100	91	95.7	0.993	1.152	1.071

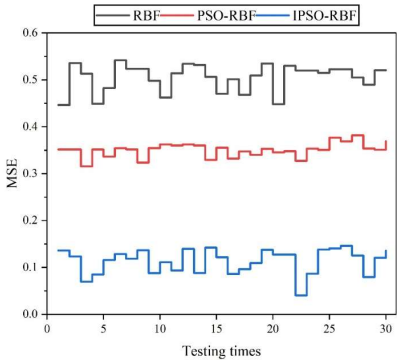


Figure 3: MSE

The changes in the MSE values of the three networks in the 30 simulation tests are specifically shown in Fig. 3. In fact, in order to judge which fault type the output of the network belongs to, it is necessary to round the output vector, i.e., the largest value in the output vector is taken as 1, and the rest is set to 0. Therefore, the same samples judged to be the correct samples may actually have different MSE values for the output of the network, and the

smaller the MSE value is, the smaller the error of the network is, and the higher the accuracy is. It can be seen that the MSE value of the IPSO-RBF neural network in this paper is always lower than that of the RBF and PSO-RBF neural networks, and the network error is smaller and has higher accuracy.

V. Systematic evaluation and analysis of interactive behaviors in programming games for 0-6 year olds

This chapter will use the IPSO-RBF-based interactive behavior assessment model of programming games for young children proposed in this paper, combined with the evaluation index system of interactive behavior of programming games for young children aged 0-6 years old constructed in this paper, to assess and analyze the interactive behavior of programming games for young children aged 0-6 years old. In this study, the programming game program for 0-6-year-old children is divided into three types of teaching modes: didactic, demonstrative and social constructive.

V. A. Subjects of the study and data sources

A pediatric programming training institution in City D was selected for this study. Sixty children in eleven classes that are receiving the extended and advanced program of programming for young children, which are concentrated in the age group of 0-6 years old, were selected for this study. In this paper, starting from May 2024, we used Saturday or Sunday time to visit the research institution for activity observation, observing one teaching activity of one class at each visit, with an activity length of 90 minutes, observing a total of 11 programming teaching activities in 11 classes, and scoring the children's deep learning process according to the self-developed Observation Scale for Young Children's Deep Learning in Programming Teaching Activities, and also combining the teachers' interviews to obtain relevant research data.

V. B. Analysis of assessment results

There are some differences in the significant sequences of teacher-student verbal behaviors occurring in the three types of classrooms: lecture-type, display-type and social construction-type. In the didactic classroom, the least significant behavioral sequences, the behavioral transitions are mostly initiated by the teacher's speech, mainly occurring between the teacher's speech and the teacher-student's speech, presenting the characteristics of teacher-led; in the demonstrative classroom, the most significant behavioral sequences occur between the teacher-student and the student-student's speech, presenting the characteristics of bidirectional interactions; in the socially constructive classroom, the behavioral transitions are mostly initiated by the student's speech, occurring mostly between the student's speech , presenting the characteristics of student-student interaction. Further results are as follows.

V. B. 1) Lecture-based classroom analysis

In the lecture-based classroom, there were 14 significant sequences involving seven teacher speech behaviors and two student speech behaviors, as shown in Figure 4. The results showed that the significant sequences of teachers' behaviors were mainly related to the teacher's lecture (TS). In order to enhance students' classroom status, teachers often enliven the atmosphere during knowledge transfer (TS→TA; TA→TS), and pay attention to and respond to students' information (TF→TS), so that the classroom forms a good two-way communication. In the classroom, the teacher will initiate the interaction by asking questions in the lecture (TS→TQ1), and specify the solution steps for students to answer the questions (TQ1→SR1; SR1→TQ2; TQ2→SR1; SR1→TF); in the general game class, the teacher is more inclined to put forward the open-ended questions to encourage the students to take the initiative to share (TQ1→TD), but the classroom silence can also occur due to the lack of timely response from the students (TD→BS). (TD→BS). In addition, the teacher also solicits students' collective opinions (TC→GS) through the functions of blackboard and polling, which elevates individual participation to collective participation and enhances the classroom activity.

V. B. 2) Demonstration-based classroom

In the demonstration-based classroom, there were 27 significant sequences involving 7 types of teacher verbal behavior and 9 types of student verbal behavior, as shown in Figure 5. The results show that the students' verbal behaviors in this model revolve around the sharing and interaction of the results, the student in charge of the presentation carries out the sharing after applying for technical authorization from the teacher (TD→SH;SH→SS2), which may be chaotic on the way due to the operations such as resource sharing errors (SH→BS;BS→SS1;BS→SS2); the student in charge of the hosting obtains the right of the hosting through the support of the teacher (TC→SC), organizing classmates' discussion around the sharing theme (SC→SQ1), and multiple students expressing their views on the issues raised by the moderator respectively (SR3→SR3). The teacher's verbal behavior mainly centered on teacher feedback (TF), where the teacher evaluated the students after they presented the results (SS1

→TF; SS2→TF) and asked questions about the solution to test whether the students really understood the meaning of the step (TF→TQ1; TQ1→SR1; SR1→TF; SR1→TQ2); the teacher of the educational theory course played an active role in promoting student-student interaction. The teacher played an active role in facilitating student-student interaction by encouraging the students to speak up during the Q&A sessions to promote full discussion of the questions (SQ1→TD) and by enlivening the atmosphere to enhance the students' motivation to participate in the discussion of open-ended questions (TA→SR3).

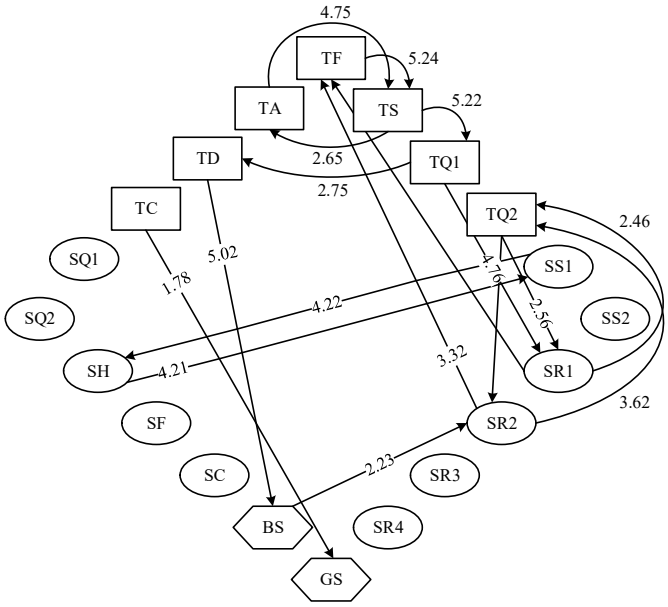


Figure 4: Lecture classroom

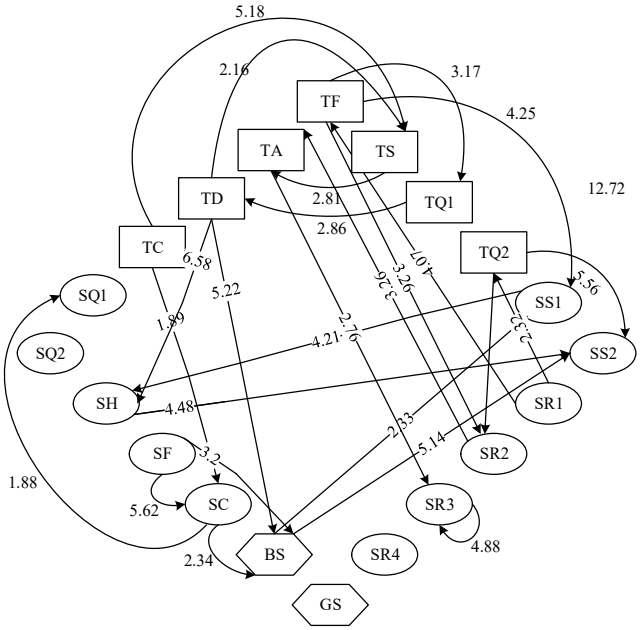


Figure 5: Show-type classroom

V. B. 3) Socially constructed classrooms

The significant sequences of verbal behaviors in the socially constructed classroom are specifically shown in Figure 6, in which there are 24 significant sequences, involving six types of teacher verbal behaviors and nine types of student verbal behaviors. The results show that the teacher's verbal behaviors are mainly centered on assisting students to discuss and explore. Before the discussion and exploration began, teachers issued questions or tasks,

organized students to discuss or collaborate in groups (TC→TD; TD→GS), and dealt with the confusion caused by operational errors when students were grouped together (TC→BS; BS→TC). At the end of the discussion and inquiry, the teacher in the small-scale educational theory class (Classroom 6) assisted the students to share the results of the inquiry in turn (TC→SS2; SS2→TD), facilitated the construction of knowledge by asking questions during the students' sharing process (TQ1→SR1), and provided comments and feedback on the students' sharing (SS2→TF); whereas, in the medium-scaled educational theory class, the interactions were more reflected in the small-group inquiry, and the students' significant The sequence of verbal behaviors reveals students' subjectivity and two-way interaction in this type of classroom. For example, the student facilitator guides the group inquiry activities (SC→GS; SC→SS1; SC→SQ1), the students in the group describe and add to the ideas of the speaking students (SS1→SF; SF→SS1), and a number of students follow up with consecutive questions (SQ2→SQ2) and discussions (SR4→SQ2; SQ2→SR4).

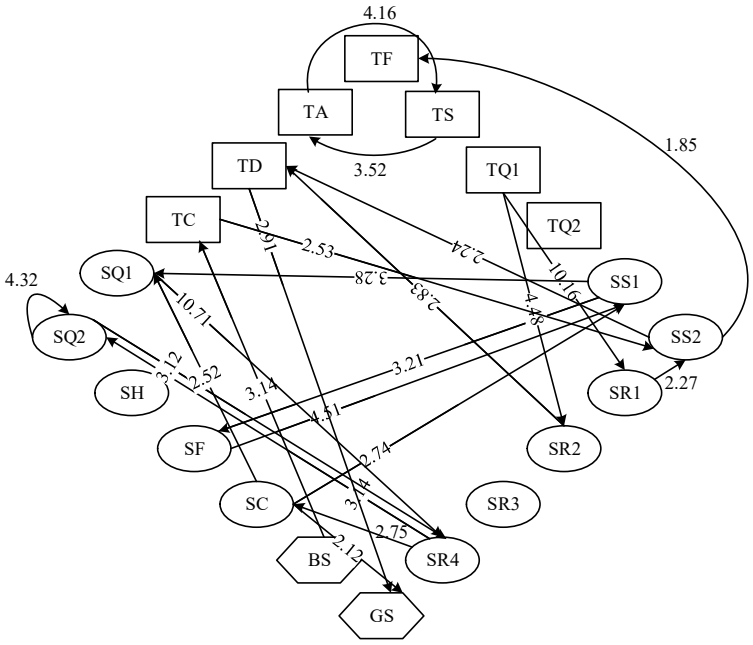


Figure 6: Socially Constructive Classroom

VI. Conclusion

The evaluation index system of programming game interaction behavior for 0-6-year-old children covers six dimensions of teacher speech behavior and six dimensions of student speech behavior, which comprehensively reflects the characteristics of teacher-student interaction. The IPSO-RBF neural network model shows excellent evaluation performance, with an average correctness rate of 95.7% and an average computation time of 1.071 seconds in 30 independent simulation tests, which is much better than that of single RBF network with 80.5% correct rate and 1.339 seconds operation time, which is a significant improvement compared to the single RBF network with 80.5% correct rate and 1.339 seconds operation time. The teacher-student interactions in the three types of teaching modes show different characteristics: the display-type classroom contains 16 types of verbal behaviors, which is the richest interaction; the social construction-type classroom contains 15 types of verbal behaviors, which is more subjective to the students; and the lecture-type classroom contains only 9 types of verbal behaviors, which is clearly dominated by the teacher. Only 2 groups of samples in the test set were misclassified, indicating that the model has good generalization ability and robustness. The assessment system and model provide scientific tools for the practice of early childhood programming education, and help teachers optimize teaching strategies and improve teaching quality. Future research can expand the sample size, further improve the index system, and enhance the adaptability of the model in different teaching situations.

Funding

This work was supported by Humanities and Social Sciences Research Project of Jilin Provincial Department of Education: Research on the Construction of Childcare Service System for 0-3 Years Old Children (JJKH20220417SK).

References

[1] Popkova, E. G., & Gulzat, K. (2019, May). Technological revolution in the 21st century: digital society vs. artificial intelligence. In Institute of Scientific Communications Conference (pp. 339-345). Cham: Springer International Publishing.

[2] Wang, L., Geng, F., Hao, X., Shi, D., Wang, T., & Li, Y. (2021). Measuring coding ability in young children: relations to computational thinking, creative thinking, and working memory. Current Psychology, 1-12.

[3] Angeli, C., & Valanides, N. (2020). Developing young children's computational thinking with educational robotics: An interaction effect between gender and scaffolding strategy. Computers in human behavior, 105, 105954.

[4] Su, J., & Yang, W. (2023). A systematic review of integrating computational thinking in early childhood education. Computers and Education Open, 4, 100122.

[5] Relkin, E., de Ruiter, L. E., & Bers, M. U. (2021). Learning to code and the acquisition of computational thinking by young children. Computers & education, 169, 104222.

[6] Angeli, C., & Giannakos, M. (2020). Computational thinking education: Issues and challenges. Computers in human behavior, 105, 106185.

[7] Cansu, F. K., & Cansu, S. K. (2019). An overview of computational thinking. International Journal of Computer Science Education in Schools, 3(1), 17-30.

[8] Sullivan, A. A., Bers, M. U., & Mihm, C. (2017). Imagining, playing, and coding with KIBO: using robotics to foster computational thinking in young children. Siu-cheung KONG The Education University of Hong Kong, Hong Kong, 110.

[9] Bers, M. U. (2018). Coding and computational thinking in early childhood: The impact of ScratchJr in Europe. European Journal of STEM Education, 3(3), 8.

[10] Lee, J., Yunus, S. A., & Lee, J. O. (2025). Investigating children's programming skills through play with robots (KIBO). Early Childhood Education Journal, 53(1), 109-117.

[11] Sullivan, A., & Bers, M. U. (2018). Dancing robots: integrating art, music, and robotics in Singapore's early childhood centers. International Journal of Technology and Design Education, 28, 325-346.

[12] Segura, R. J., del Pino, F. J., Ogáyar, C. J., & Rueda, A. J. (2020). VR‐OCKS: A virtual reality game for learning the basic concepts of programming. Computer Applications in Engineering Education, 28(1), 31-41.

[13] Papadakis, S. (2021, June). The impact of coding apps to support young children in computational thinking and computational fluency. A literature review. In Frontiers in Education (Vol. 6, p. 657895). Frontiers Media SA.

[14] Sullivan, A., Kazakoff, E. R., & Bers, M. U. (2013). The wheels on the bot go round and round: Robotics curriculum in pre-kindergarten. Journal of Information Technology Education. Innovations in Practice, 12, 203.

[15] Papadakis, S., & Kalogiannakis, M. (2019). Evaluating a course for teaching introductory programming with Scratch to pre-service kindergarten teachers. International Journal of Technology Enhanced Learning, 11(3), 231-246.

[16] Rose, S., Habgood, M. J., & Jay, T. (2017). An exploration of the role of visual programming tools in the development of young children's computational thinking. Electronic journal of e-learning, 15(4), pp297-309.

[17] Mladenović, S., Krpan, D., & Mladenović, M. (2016). Using games to help novices embrace programming: From elementary to higher education. The International journal of engineering education, 32(1), 521-531.

[18] Critten, V., Hagon, H., & Messer, D. (2022). Can pre-school children learn programming and coding through guided play activities? A case study in computational thinking. Early Childhood Education Journal, 50(6), 969-981.

[19] Wong, G. K. W., & Cheung, H. Y. (2020). Exploring children's perceptions of developing twenty-first century skills through computational thinking and programming. Interactive Learning Environments, 28(4), 438-450.

[20] Hao Ling,Tengfei Wu,Yonghui Wu,Zheng Liu,Lihua Zhang & Xiaorong Lv. (2025). Optimization of motion strategy for a micro multi-functional chassis based on RBF neural network in intercropping mode. Computers and Electronics in Agriculture,235,110316-110316.

[21] Fei Li,Hao Pan,Yilong Ji & Haibin Ouyang. (2025). Particle swarm optimization based on particle perturbation and elite preservation strategies for dynamic optimization problems. Expert Systems With Applications,279,127423-127423.

[22] Yaxuan Jiang & Yipeng Ding. (2024). A single-frequency Doppler radar target tracking method based on the IPSO algorithm. Journal of Physics: Conference Series,2849(1),012135-012135.