Appendix of "Symbolic Cognitive Diagnosis via Hybrid Optimization for Intelligent Education Systems"

Junhao Shen, Hong Qian*, **Wei Zhang, Aimin Zhou**

Shanghai Institute of AI for Education and School of Computer Science and Technology, East China Normal University, Shanghai 200062, China

shenjh@stu.ecnu.edu.cn, {hqian, amzhou}@cs.ecnu.edu.cn, zhangwei.thu2011@gmail.com

A Cognitive Diagnosis Assessment

Cognitive Diagnosis Assessment (CDA) is a fundamental and crucial task in intelligent education systems. An illustrative example of CDA is shown in Figure I of Appendix. There are three main factors in CDA: students, exercises and knowledge attributes which are also referred to as skills (e.g., calculation). The purpose of CDA is to model the student-exercise interaction via an interaction function based on response logs, and diagnose the students' cognitive states, i.e., inferring the proficiency levels on knowledge attributes.

B Details of Genetic Programming

In the SCD Model Implementation Section of the main paper, we have introduced the genetic programming to implement symbolic regression. Herein, we will list the details in Algorithm 1 of the main paper.

B.1 Detailed Settings

We provide more detailed settings of implementation and experiments.

Function Set. In order to enhance the fitting capacity, the function set should encompass various symbols. For example, $\sin$ and $\cos$ can be involved in it in certain educational scenarios (Ouyang et al. 2023). However, to meet Assumption 1, these non-monotonic operators cannot be employed. Therefore, we define the function set as $\{+, \times, \circ, -, \tanh, \mathrm{sigmoid}\}$.

Constraints of Symbolic Tree. Recalling Eq. (1), we notice that the output of an interaction function should be a scalar. However, this requirement cannot be satisfied by every symbolic tree generated. Therefore, certain constraints need to be imposed on symbolic trees. First, the input of a symbolic tree must include $\boldsymbol{P}_{i,\bullet}$ and $\varphi_{\mathrm{rele}_j}$ to ensure that the final output is relevant to the knowledge attributes. Second, the computation for each operator should be valid (e.g., a unary operator only accepts one parameter). Finally, the output of a valid symbolic tree must be scalar. Therefore, after the creation of each symbolic tree (including the initialization and the genetic operations such as crossover, mutation and selection), it needs to be checked to ensure that it satisfies the aforementioned constraints.

Fitness. Accuracy is chosen as the metric of fitness. Similar to the way of measuring the generalization of SCDM, we assess the fitness of each individual on the training set of students' correctness. Special attention should be paid to that we do not use the test set when evaluating the fitness of each individual (i.e., symbolic tree). This is crucial during the training process to avoid the leakage problem.

B.2 Algorithmic Details

We elaborate the details in Algorithm 1 that are sketched in the main paper due to the page limitation.

$\mathbf{uniform(0, 1)}$. To simulate the random genetic operations (crossover, mutation and selection), we introduce a uniform distribution. In this paper, unless otherwise specified, the random numbers used are uniformly distributed between 0 and 1, denoted as $\mathrm{uniform}(0, 1)$.

Crossover. We use an intuitive and simple way to realize crossover: randomly select crossover point in each individual and exchange each subtree with the point as root between each individual (Poli, Langdon, and McPhee 2008). It is shown in Algorithm I of Appendix.

Mutation. The mutation involves two parts: insert and prune. The former enhances tree complexity, while the latter prevents excessive complexity. The detail implementation is shown in Algorithm II of Appendix. From line 3 to 7 in Al-

```
Algorithm I: Crossover
Input: Two individuals $f_1$, $f_2$.
Output: New individuals $f_1'$, $f_2'$.
1: if $\mathrm{height}(f_1) < 2$ or $\mathrm{height}(f_2) < 2$ then
2:     return $f_1$, $f_2$
3: else
4:     $f_1', f_2' \leftarrow f_1, f_2$
5:     randomly choose nodes $node_1$, $node_2$ from $f_1'$, $f_2'$ respectively
6:     search the subtrees $\tilde{f}_1$, $\tilde{f}_2$ beginning from $node_1$, $node_2$ respectively
7:     $\tilde{f}_1, \tilde{f}_2 \leftarrow \tilde{f}_2, \tilde{f}_1$
8:     return $f_1'$, $f_2'$
9: end if
```

*Corresponding Author: hqian@cs.ecnu.edu.cn.

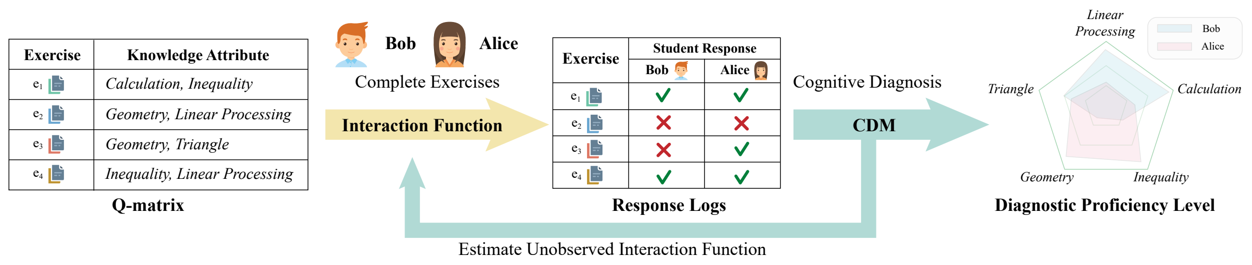

Figure I: An illustrative example of cognitive diagnosis assessment

Table I: Generalization and interpretability performance (expressed in percentage) of baselines, state-of-the-art methods and SCDM. In each column, an entry is marked in bold if its mean value is the best and underline for the runner-up. By t-test, a bold one is significantly better than others on the corresponding metrics with significant level $\alpha = 5\%$. The symbol "—" means that the value is unavailable.

	Math1				Math2				FracSub				NeurIPS2020			
	Accuracy	AUC	F1	DOA	Accuracy	AUC	F1	DOA	Accuracy	AUC	F1	DOA	Accuracy	AUC	F1	DOA
IRT	67.44	73.61	72.49	—	69.28	76.54	69.27	—	80.19	87.94	81.44	—	**69.31**	**75.28**	70.79	—
MIRT	67.49	73.67	71.83	—	69.28	76.11	68.64	—	80.39	87.77	81.69	—	66.27	71.28	69.12	—
DINA	50.00	68.20	34.43	53.61	52.68	69.78	28.55	54.31	67.82	77.24	60.81	73.89	53.75	67.08	37.52	53.85
NCDM	67.36	$\underline{74.02}$	72.43	$\underline{53.67}$	69.78	$\underline{76.85}$	$\underline{70.78}$	$\underline{55.01}$	82.55	**89.72**	83.19	$\underline{80.72}$	68.17	73.78	$\underline{71.06}$	$\underline{54.98}$
KaNCD	**68.23**	**75.28**	$\underline{73.27}$	50.40	**70.35**	**78.12**	**70.84**	48.88	$\underline{82.85}$	$\underline{89.51}$	$\underline{84.08}$	55.69	$\underline{68.77}$	$\underline{75.06}$	70.98	48.19
SCDM	$\underline{67.78}$	73.13	**74.14**	**53.75**	$\underline{69.79}$	75.17	70.15	**55.08**	**83.26**	86.80	**85.15**	**81.78**	67.25	70.48	**71.11**	**57.32**

Algorithm II: Mutation

```
Input: Individuals $f$.
Output: New individual $f'$.
 1: $f' \leftarrow f$
 2: if uniform$(0,1) < 0.5$ then
 3:     randomly choose node $node_1$ from $f'$
 4:     search the subtree $\tilde{f}_1$ beginning from $node_1$
 5:     randomly generated a subtree $\tilde{f}$
 6:     let $node_1$ and its subtree $\tilde{f}_1$ become left child of $\tilde{f}$
 7:     $\tilde{f}_1 \leftarrow \tilde{f}$
 8: else
 9:     if height$(f) < 5$ then
10:         return $f$
11:     else
12:         randomly choose node $node_1$ from $f'$
13:         search the subtree $\tilde{f}_1$ beginning from $node_1$
14:         randomly choose node $node_2$ from $\tilde{f}_1$
15:         search the subtree $\tilde{f}_2$ beginning from $node_2$
16:         $\tilde{f}_1 \leftarrow \tilde{f}_2$
17:     end if
18: end if
19: return $f'$
```

gorithm II, a new branch will be inserted at a random position in individual, and this subtree at the chosen position is used as child node of the created subtree. From line 9 to 16, a branch will be randomly chosen and replaced with one of the branch's nodes (De Rainville et al. 2012).

Selection. The selection is implemented by tournament selection (De Rainville et al. 2012), which selects the best

Algorithm III: Tournament Selection

```
Input: Population $\{f_1, f_2, \ldots, f_V\}$, selection times $T_{\text{se}}$.
Output: Selected individuals $\{f_1, f_2, \ldots, f_{T_{\text{se}}}\}$.
1: initialize selected individual list $l \leftarrow \{\}$
2: while length$(l) < T_{\text{se}}$ do
3:     obtain $f'$ from $\{f_1, f_2, \ldots, f_V\}$, where $f'$ is the
       fittest individual in the population
4:     add $f'$ to $l$
5:     remove $f'$ from population
6: end while
7: return $l$
```

individual among population T_{se} times and gives the selected individuals. Algorithms III in Appendix shows the details. In the implementation of GP, k is set as 3.

C Extra Experiment Results

C.1 Statistics of SCDM Performance

All statistics in the generalization performance and interpretability performance are shown in Table I (expressed in percentage) of Appendix. Since IRT and MIRT utilize latent vectors, DOA is unavailable, and thus we use "—" to represent it. SCDM outperforms existing models on half of metrics of each dataset and performs competitively with other CDMs on the left metrics. The interpretability metric (DOA) of SCDM is always the best in Table I.

C.2 SCDM without (w.o.) Adam

In order to implement parameter optimization without utilizing the Adam optimizer, we introduce the evolutionary

Algorithm IV: Evolutionary Strategy (ES)

Input: Response logs $R = \{s, e, r\}$, Q-matrix: $\boldsymbol{Q}$, current interaction function f_{curr}.
Parameter: generation T_{ES}, population size of ES V_{ES}, mutation rate of ES $p_{\text{es-mu}}$, learning rate lr.
Output: Proficiency levels $\boldsymbol{P}$, exercise features $\boldsymbol{\varphi}_{\text{diff}}, \varphi_{\text{disc}}$.

```
1:  Randomly initialize a population of parameters $\mathrm{Para} = \{\{\boldsymbol{P}^{(1)}, \boldsymbol{\varphi}_{\text{diff}}^{(1)}, \varphi_{\text{disc}}^{(1)}\}, \{\boldsymbol{P}^{(2)}, \boldsymbol{\varphi}_{\text{diff}}^{(2)}, \varphi_{\text{disc}}^{(2)}\}, \ldots, \{\boldsymbol{P}^{(V_{\text{ES}})}, \boldsymbol{\varphi}_{\text{diff}}^{(V_{\text{ES}})}, \varphi_{\text{disc}}^{(V_{\text{ES}})}\}\}$
2:  for $t_1 = 1, 2, \ldots, T_{\text{ES}}$ do
3:    for $i = 1, 2, \ldots, V_{\text{ES}}$ do
4:      if $\mathrm{uniform}(0, 1) < p_{\text{es-mu}}$ then
5:        $\boldsymbol{P}^{(i)} \leftarrow \boldsymbol{P}^{(i)} + lr \cdot \mathrm{normal}(0, 1)$
6:        $\boldsymbol{\varphi}_{\text{diff}}^{(i)} \leftarrow \boldsymbol{\varphi}_{\text{diff}}^{(i)} + lr \cdot \mathrm{normal}(0, 1)$
7:        $\varphi_{\text{disc}}^{(i)} \leftarrow \varphi_{\text{disc}}^{(i)} + lr \cdot \mathrm{normal}(0, 1)$
8:      end if
9:    end for
10:   $\mathrm{Para} \leftarrow \mathrm{selection}(\mathrm{Para})$
11:   Evaluate all individuals in $\mathrm{Para}$,
12: end for
13: $\boldsymbol{P}, \boldsymbol{\varphi}_{\text{diff}}, \varphi_{\text{disc}} \leftarrow \boldsymbol{P}', \boldsymbol{\varphi}'_{\text{diff}}, \varphi'_{\text{disc}}$, where $\boldsymbol{P}', \boldsymbol{\varphi}'_{\text{diff}}, \varphi'_{\text{disc}}$ are the fittest parameters in $\mathrm{Para}$
14: return $\boldsymbol{P}, \boldsymbol{\varphi}_{\text{diff}}, \varphi_{\text{disc}}$
```

strategy (ES) (Beyer and Schwefel 2002) for optimizing the object function as Eq. (1) to obtain the optimal parameters.

$$\mathcal{L}_{r \in R}(y_f, r) = -\sum_{i=1}^{|R|} (r^{(i)} \log y_f^{(i)} + (1 - r^{(i)}) \log(1 - y^{(i)})), \quad (1)$$

where $y_f^{(i)}$ is the i-th output of interaction function f and $r^{(i)}$ is the corresponding label. Algorithm IV in Appendix describes the detailed implementation.

Algorithm IV is similar to the GP module of aforementioned Algorithm 1 in the main paper. However, to better suit the ES, certain modifications have been introduced. Specifically, the learning rate is 0.1, the population size is 100, the number of generation is 50, and the mutation rate is 0.5 which is much larger than that of GP due to full coverage the search space. The fitness of each individual is the value of loss (the smaller the better). $\mathrm{normal}(0, 1)$ is a normally distributed random variable with zero mean and standard deviation 1. The selection is as same as Algorithm III in Appendix, and $k = 3$.

References

Beyer, H.-G.; and Schwefel, H.-P. 2002. Evolution Strategies–A Comprehensive Introduction. *Natural Computing*, 1: 3–52.

De Rainville, F.-M.; Fortin, F.-A.; Gardner, M.-A.; Parizeau, M.; and Gagné, C. 2012. Deap: A Python Framework for Evolutionary Algorithms. In *Proceedings of the 14th Annual Conference Companion on Genetic and Evolutionary Computation*, 85–92. philadelphia, PA.

Ouyang, F.; Wu, M.; Zheng, L.; Zhang, L.; and Jiao, P. 2023. Integration of Artificial Intelligence Performance Prediction and Learning Analytics to Improve Student Learning in Online Engineering Course. *International Journal of Educational Technology in Higher Education*, 20: 1–23.

Poli, R.; Langdon, W. B.; and McPhee, N. F. 2008. *A Field Guide to Genetic Programming*. Springer.