MetaCD: A Meta Learning Framework for Cognitive Diagnosis based on Continual Learning

Jin Wu, Chanjin Zheng
Shanghai Institute of Artificial Intelligence for Education
East China Normal University
52275901018@stu.ecnu.edu.cn, chjzheng@dep.ecnu.edu.cn

Abstract

Cognitive diagnosis is an essential research topic in intelligent education, aimed at assessing the level of mastery of different skills by students. So far, many research works have used deep learning models to explore the complex interactions between students, questions, and skills. However, the performance of existing method is frequently limited by the long-tailed distribution and dynamic changes in the data. To address these challenges, we propose a meta-learning framework for cognitive diagnosis based on continual learning (MetaCD). This framework can alleviate the long-tailed problem by utilizing meta-learning to learn the optimal initialization state, enabling the model to achieve good accuracy on new tasks with only a small amount of data. In addition, we utilize a continual learning method named parameter protection mechanism to give MetaCD the ability to adapt to new skills or new tasks, in order to adapt to dynamic changes in data. MetaCD can not only improve the plasticity of our model on a single task, but also ensure the stability and generalization of the model on sequential tasks. Comprehensive experiments on five real-world datasets show that MetaCD outperforms other baselines in both accuracy and generalization.

1 Introduction

Cognitive diagnosis is a crucial educational measurement model in intelligent education, which aims to explore students' cognitive processes in problem-solving through their measurement data [1]. In general, the cognitive diagnosis system can assesses students' mastery of various skills by modeling cognitive processing. And it can provide timely feedback on students' weak skills [2].

Unfortunately, with the explosive growth of educational data, traditional cognitive diagnostic models (such as IRT [3] and DINA [4]) cannot mine the potential nonlinear interactive relationships between students and questions [5, 6], which limits models' comprehensive understanding of students' cognitive processes. In order to address these limitations, recent efforts has begun exploring cognitive diagnosis models based on deep learning, such as neural network-based cognitive diagnosis (NCD) [7], self-supervised graph neural network-based cognitive diagnosis (SCD) [8], etc. These approaches can mine the deep non-linear interaction between students-questions-skills.

However, applying existing cognitive diagnostic methods to online learning systems still faces numerous challenges: ***(a)*** some questions receive very few responses from students, while others receive many, resulting in sparse data and subsequently leading to long-tailed problems [9]. ***(b)*** student data in online learning systems typically exhibit two dynamic changes: (1) students learn new skills over time, and their mastery of previously learned skills changes; (2) new learning tasks appear in the system can lead to changes in the data. Existing methods, designed for fixed datasets [7, 10, 11, 12], struggle to adapt to these dynamic conditions. They typically exhibit significant performance degradation on new, emerging data patterns due to an inability to integrate

Submitted to 39th Conference on Neural Information Processing Systems (NeurIPS 2025) Workshop.

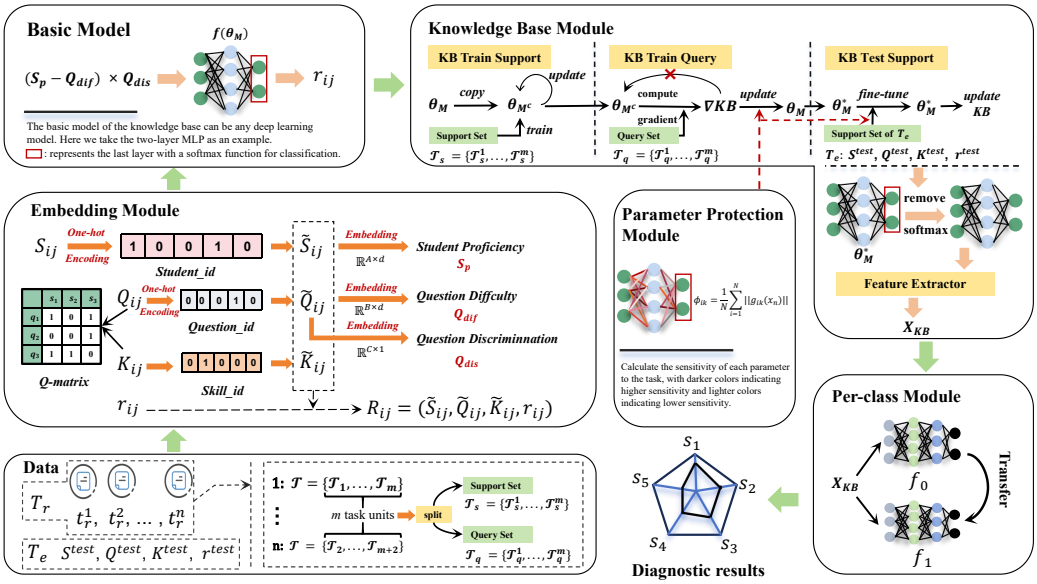

Figure 1: The overall structure of MetaCD.

new knowledge effectively. Furthermore, they are highly susceptible to catastrophic forgetting, whereby learning new information causes a rapid and severe loss of previously acquired knowledge.

In order to solve the above problems, we proposed a meta learning framework for cognitive diagnosis based on continual learning, called MetaCD. Our approach focuses more on considering the model's plasticity and stability in adapting to the long-tailed distribution, and dynamic changes in the data and new tasks. Our key contributions are summarized as follows: ***(1)***. We empower the knowledge base module through meta-learning methods to ensure that the model can alleviate the problem of long-tailed distribution through few data, achieving the reliability and robustness of MetaCD on sparse data. ***(2)***. We use parameter protection mechanism (PPM) to ensure the stability and plasticity of the model, enabling MetaCD to adapt to the dynamic changes of data and achieve its continual learning ability. ***(3)***. We use a knowledge extraction method based on Kullback-Leibler divergence to avoid fuzzy boundaries of diagnostic results, thereby improving the classification accuracy of the model. ***(4)***. We conduct comprehensive experiments on five real-world datasets to validate the performance of MetaCD, particularly on long-tailed data and task incremental learning.

2 Our Proposed MetaCD Method

2.1 Task Overview

Suppose that we are given n training task units set $T_r = \{t_r^1, t_r^2, ..., t_r^n\}$ and a testing task $T_e = \{S^{test}, Q^{test}, K^{test}, r^{test}\}$. The data of each task unit t_r^i consists of response logs of students, including student IDs S_i, question IDs Q_i, and skill IDs K_i, $i \in [1 : n]$. Each student has a corresponding score s for each question, $s \in \{0, 1\}$ (1 indicates that the student answered a question correctly, otherwise 0) and $s \in r_i$, so $T_i = \{S_i, Q_i, K_i, r_i\}$. In addition, we need to construct the $\mathcal{Q}$-matrix[1] [7] for each task unit based on the questions and their corresponding skills.

Problem Definition: Given a set of students' response data from training task units set T_r and $\mathcal{Q}$-matrix, the purpose of MetaCD is to learn a model that can predict students' proficiency in skills and their corresponding scores by effectively transferring knowledge across tasks. The ultimate goal is to obtain the best model that can accurately predict students' performance on the testing task T_e.

[1]The $\mathcal{Q}$-matrix differs from Q_i. Q_i represents the ID of the question answered by the students, while the $\mathcal{Q}$-matrix shows the binary relationship between questions and skills.

2.2 The Structure of MetaCD

Figure 1 illustrates the overall structure of MetaCD. The training process is as follows: we first initialize the network parameters θ_M and create a copy θ_{M^c}. Then, response data are passed through the embedding module to obtain vectors $(\tilde{S}_{ij}, \tilde{Q}_{ij}, \tilde{K}_{ij}, r_{ij})$. MetaCD randomly samples m task units $\mathcal{T} = \{\mathcal{T}_1, \ldots, \mathcal{T}_m\}$[2] from T_r, and constructs their corresponding $\mathcal{Q}$-matrices. Using the support sets, it updates θ_{M^c} to obtain θ'_{M^c}. Next, the query sets are used to compute gradients from θ'_{M^c}, which are then used to update θ_M, resulting in θ^*_M. This optimized θ^*_M is used to initialize MetaCD for the test task T_e, and further fine-tuned using its support set. Finally, the model M^* is obtained. Given the query set in T_e, MetaCD predicts each student's response (0 or 1).

Embedding module:Let A, B, and C denote the numbers of students, questions, and skills, with embedding dimension d set to the number of skills per task. A $\mathcal{Q}$-matrix [13] maps questions to skills. Each response is encoded as $R_{ij} = (\tilde{S}_{ij}, \tilde{Q}_{ij}, \tilde{K}_{ij}, r_{ij})$, where $\tilde{S}_{ij}$, $\tilde{Q}_{ij}$, and $\tilde{K}_{ij}$ are one-hot vectors, and r_{ij} is the score. Trainable embeddings estimate student proficiency, question difficulty, and discrimination. The knowledge base, initialized as θ_M, is updated via task-specific copies θ_{M^c}.

Knowledge base module: Figure 1 illustrates the Knowledge Base (KB) module, comprising three components: KB train support, KB train query, and KB test support. The KB module uses meta-learning to accumulate and store experiential knowledge from multiple training task units. The base model $f(\theta_M)$ can be any deep model, such as MLP [14] or CNN [15]. The KB supports both updates and retrieval. The process is detailed as follows:

1. KB Train Support: We repeatedly sample tasks $\mathcal{T}$ from T_r, and use support sets $\mathcal{T}_s$ to train the KB with parameters θ_M. Task units differ in dataset source, data distribution, etc. MetaCD learns from many $\mathcal{T}_s$ to store generalizable knowledge in KB. The specific expression is as follows:

$$\theta_{M^c} \leftarrow \theta_{M^c} - \alpha \cdot \frac{1}{n} \cdot \frac{1}{m} \sum_{i=1}^{n} \sum_{j=1}^{m} \nabla_{\theta_{M^c}} L(f(\theta_{M^c}), \mathcal{T}_s) \tag{1}$$

where n represents the number of $\mathcal{T}_s$, which corresponds to the batch_size, and L denotes the loss function. In this paper, we use the cross-entropy loss function. we set the hyper-parameter α as 0.3.

2. KB Train Query: To reduce redundancy, we refine θ_{M^c} using the query set $\mathcal{T}_q$ with a combined loss: $L_{total} = L(f(\theta_{M^c}), \mathcal{T}_q) + L_{PPM}$. The gradient from L_{total} updates θ_M, not θ_{M^c} (see Eq. (2)). The updated θ_M is treated as the optimal initialization θ^*_M. This step enhances the KB with query knowledge, improving task adaptation. We set the hyper-parameter β to 0.5.

$$\begin{aligned} \nabla KB &= \nabla_{\theta_{M^c}} L_{total} \\ \theta^*_M = \theta_M &\leftarrow \theta_M - \beta \cdot \nabla KB \end{aligned} \tag{2}$$

In KB Train Query, the gradient computed from L_{total} will no longer be used to update θ_{M^c}, but instead will be used to update θ_M (see Eq. (2)). The parameters θ_M obtained from KB train query are considered as the optimal initialization parameters θ^*_M. The process aims to optimize the parameters in the knowledge base using $\mathcal{T}_q$, allowing MetaCD to directly leverage the knowledge from the knowledge base as initial knowledge when facing new or long-tailed tasks. This ensures that the model starts with a good knowledge initial state that includes prior knowledge, leading to better performance.

3. KB Test Support: We initialize the knowledge base with θ^*_M and fine-tune it on the support set of the test task T_e. During this stage, we apply the parameter protection mechanism (see ***Parameter protection module***) to adapt KB for future tasks. After KB training, we remove the classification layer so that the knowledge base functions as a feature extractor.

Parameter protection module: The Parameter Protection Mechanism (PPM) balances stability on old tasks and adaptability to new ones by integrating into the loss during KB train query and test support. It preserves key knowledge from training tasks and reduces forgetting across sequential tasks. PPM measures parameter sensitivity, restricting updates to sensitive parameters while allowing others to change freely. The calculation is shown in Eq. (3).

[2] $\mathcal{T}_i = t^i_r$; we use $\mathcal{T}_i$ here for clarity.

$$L_{PPM} = \frac{1}{2}\sum_{i,k}\phi_{i,k}(\theta_{i,k} - \theta^*_{i,k})^2, \tag{3}$$

where $\theta^*_{i,k}$ represents the k-th parameter obtained after training in the i-th task unit, $\theta_{i,k}$ indicates the k-th initialization value in the i-th task unit. $\phi_{i,k}$ refers to the sensitivity (importance) weights of the parameter, and its calculation formula is as follows:

$$\phi_{i,k} = \frac{1}{N}\sum_{n=1}^{N}||g_{ik}(x_n)||, \tag{4}$$

where N refers to the total number of data in task unit, g_{ik} represents the gradient of the KB with respect to the x_n, and $g_{ik}(x_n) = \frac{\partial KB(x_n,\theta)}{\partial \theta_{ik}}$. Here, KB stand for the output of the knowledge base module, and θ indicates the parameters of KB. The term g_{ik} can be seen as the importance of $\theta_{i,k}$ at the data x_n. Consequently, $\phi_{i,k}$ can be seen as the importance of k-th parameter in the i-th task unit.

Per-class diagnosis module: In order to solve the problem of fuzzy boundaries, we adopt a Per-class-based diagnosis method. Since cognitive diagnosis results are binary (0 or 1), we build a separate fully connected network (FCN) for each class, called a per-class head. Each per-class head consists of 4 fully connected layers. The input to the per-class diagnosis module is the knowledge base output, and its output is computed as in $y = \arg\max_k \left(f_0\left(X_{KB}\right), f_1\left(X_{KB}\right)\right)$. Where X_{KB} represents the output of knowledge base module, f_0 and f_1 indicates the two head of result 0 and result 1, respectively. The loss function of each head is shown in Eq. (5).

$$\begin{aligned} loss = & \mathop{\mathbb{E}}_{\mathbf{X_{KB}}\sim\mathbb{P}_{\mathbf{X_{KB}}}}[-\log(\text{Sigmoid}(f_i(X_{KB})))]+ \\ & \eta\cdot\mathop{\mathbb{E}}_{\mathbf{X_{KB}}\sim\mathbb{P}_{\mathbf{X_{KB}}}}\left\|\frac{\partial f_i(X_{KB})}{\partial X_{KB}}\right\|_2^{\mu}+ \\ & \lambda\cdot\|\theta_1-\theta_0\|_2^2, \end{aligned} \tag{5}$$

where X_{KB} represents the output of knowledge base module, f_0 and f_1 indicates the two head of result 0 and result 1, respectively. The first term in Eq.(5) is called the negative logarithmic function (NLL). We can make the output value of NLL as large as possible by reducing loss. However, we cannot make it infinitely large, otherwise it will cause severe overfitting, making the results between the two heads impossible to compare. Therefore, we introduce the holistic regularization (H-reg) as the second term of Eq.(5). This allows the parameters to be as large as possible while ensuring that the parameters and output are controllable. In this work, both heads are all 4-layer FCN with ReLU activation function, so H-reg can be expressed as Eq.(6). The third term in Eq.(5) is the L2-transfer regularization, which aims to control the model complexity by penalizing large weight values. Moreover, in order to make the output values of f_1 and f_0 be in the same value space as much as possible, the parameters of f_1 is initialized with f_0.

$$\mathop{\mathbb{E}}_{\mathbf{X_{KB}}\sim\mathbb{P}_{\mathbf{X_{KB}}}}\left\|\frac{\partial f_i(X_{KB})}{\partial X_{KB}}\right\|_2^{\mu} = \mathop{\mathbb{E}}_{\mathbf{X_{KB}}\sim\mathbb{P}_{\mathbf{X_{KB}}}}\|\omega_4\cdot\omega_3\cdot\omega_2\cdot\omega_1\|_2^{\mu}, \tag{6}$$

To address the existence of fuzzy boundaries, after training is completed, we adopt a method based on Kullback-Leibler divergence [16] to reduce the shared knowledge, thereby increasing the separation distance between different diagnosis result. The calculation formula for knowledge sharing between different diagnosis results is shown in Eq. (7).

$$\Lambda^* = \text{argmin}\sum_{i=0}^{1}\kappa_i KL\left(\mathbb{P}_i\|\mathbb{P}_{0:1}\right), \tag{7}$$

3 Experimental analysis and results

Comparison Baselines: we conducted a comparative analysis with several existing cognitive diagnosis baselines using the aforementioned dataset. These baseline methods mainly include traditional

Table 1: Performance comparison of different models on student cognitive diagnostic.

Work	ASSIST2009_2010			ASSIST2017		
	ACC	RMSE	AUC	ACC	RMSE	AUC
MetaCD	**0.753**	0.425	**0.771**	**0.715**	**0.439**	**0.726**
IRT	0.654	0.472	0.681	0.658	0.464	0.668
MIRT	0.707	0.461	0.716	0.668	0.461	0.678
DINA	0.644	0.495	0.680	0.613	0.519	0.654
BCD	0.729	0.426	0.763	0.701	0.447	0.713
NCD	0.726	0.441	0.752	0.685	0.453	0.699
RCD	0.724	0.427	0.761	0.694	0.450	0.709
SCD	0.731	**0.423**	0.729	0.703	0.442	0.710

cognitive diagnostic models represented by IRT [3], MIRT [17], DINA [4] and BCD [18], and neural network-based cognitive diagnostic models represented by NCD [7], RCD [19], and SCD [8].

We use the ASSIST (ASSIST2009_2010 [7], ASSIST2012_2013 [20], ASSIST2017 [8]), CDBD_a0910, and NIPS2020 datasets (as detailed in Appendices A.1) to evaluate MetaCD on the following research questions: ***RQ1***: Can MetaCD achieve high accuracy in cognitive diagnosis compared to baseline models? ***RQ2***: Can MetaCD maintain strong generalization under long-tailed data distributions? ***RQ3***: Can MetaCD remain stable and adaptable in task-incremental learning? ***RQ4***: What is the impact of each module through ablation study? Appendices A.2 shows the detailed experimental setup.

RQ.1: Table 1 shows that MetaCD achieves the best ACC and AUC, with slightly higher RMSE than RCD, suggesting better task initialization. Considering data privacy issues, we further test MetaCD in few-shot settings. Table 2 and Table 3 (in Appendices A.4) show that as samples increase, all models improve, but MetaCD performs best, showing strong generalization and robustness under limited data.

RQ.2: As NIPS2020 is used for meta-training and is uniformly distributed, we exclude it from testing. We evaluate MetaCD on ASSIST2009_2010, ASSIST2012_2013, ASSIST2017, and CDBD_a0910. For RQ2, we group interactions by frequency (6–10 to 31–35). Figure 2 shows MetaCD consistently outperforms SCD, showing strong generalization on sparse data.

RQ.3: To answer RQ3, we evaluate MetaCD in a task-incremental setting. Trained sequentially on four datasets after NIPS2020, MetaCD with parameter protection achieves a BWT (in Appendices A.3) of -0.04 vs. -0.217 without it Table 4 (in Appendices A.4), showing reduced forgetting and better stability.

RQ.4: To evaluate the contribution of each module in MetaCD, we conduct ablation experiments on four datasets. As shown in Table 4 (in Appendices A.5), removing any module degrades performance. Notably, removing the KB module results in the largest accuracy drop (average -3.65%), highlighting its core role. Excluding the PPM module reduces ACC by 1.8% on average, showing its effectiveness in filtering redundant knowledge during KB query and support stages. The absence of the Per-class module also harms performance, confirming its utility in resolving fuzzy class boundaries.

4 Conclusion and Limitations

This article addresses the challenges posed by long-tailed problems and dynamic changes in cognitive diagnosis, offering a meta-learning framework based on continual learning as a solution. Specifically, we first set a meta-learning-based task objective to enable the neural network to mine prior knowledge from past tasks, thereby better generalizing on the long-tailed data. Then, we use a continual learning method based on parameter protection mechanism to calculate the importance level of parameters so that the model adapts well to new tasks and generalizes well to new skills. This ensures the stability and plasticity of the model. Although MetaCD outperforms existing baseline methods, computational efficiency has become an important consideration. Specifically, in PPM, the need to compute the sensitivity of each parameter adds additional computational overhead. Future work will focus on improving the computational efficiency of MetaCD.

References

[1] Y. Su, Z. Han, S. Shen, X. Yang, Z. Huang, J. Wu, H. Zhou, and Q. Liu, "Constructing a confidence-guided multigraph model for cognitive diagnosis in personalized learning," *Expert Systems with Applications*, p. 124259, 2024. . https://doi.org/10.1016/j.eswa.2024.124259.

[2] E. Akata, L. Schulz, J. Coda-Forno, S. J. Oh, M. Bethge, and E. Schulz, "Playing repeated games with large language models," *Nature Human Behaviour*, pp. 1–11, 2025.

[3] M. O. Edelen and B. B. Reeve, "Applying item response theory (irt) modeling to questionnaire development, evaluation, and refinement," *Quality of life research*, vol. 16, pp. 5–18, 2007. https://doi.org/10.1007/s11136-007-9198-0.

[4] J. De La Torre, "Dina model and parameter estimation: A didactic," *Journal of educational and behavioral statistics*, vol. 34, no. 1, pp. 115–130, 2009. https://doi.org/10.3102/1076998607309474.

[5] K. Gandhi, J.-P. Fränken, T. Gerstenberg, and N. Goodman, "Understanding social reasoning in language models with language models," *Advances in Neural Information Processing Systems*, vol. 36, pp. 13518–13529, 2023.

[6] M. H. Tessler, M. A. Bakker, D. Jarrett, H. Sheahan, M. J. Chadwick, R. Koster, G. Evans, L. Campbell-Gillingham, T. Collins, D. C. Parkes, *et al.*, "Ai can help humans find common ground in democratic deliberation," *Science*, vol. 386, no. 6719, p. eadq2852, 2024.

[7] F. Wang, Q. Liu, E. Chen, Z. Huang, Y. Chen, Y. Yin, Z. Huang, and S. Wang, "Neural cognitive diagnosis for intelligent education systems," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 34, pp. 6153–6161, 2020. https://doi.org/10.1609/aaai.v34i04.6080.

[8] S. Wang, Z. Zeng, X. Yang, and X. Zhang, "Self-supervised graph learning for long-tailed cognitive diagnosis," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 37, pp. 110–118, 2023. https://doi.org/10.1609/aaai.v37i1.25082.

[9] Y. Zhang, B. Kang, B. Hooi, S. Yan, and J. Feng, "Deep long-tailed learning: A survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2023. . https://doi.org/10.1109/TPAMI.2023.3268118.

[10] C. Frasson *et al.*, "Enhancing the effectiveness of intelligent tutoring systems using adaptation and cognitive diagnosis modeling," in *Novelties in Intelligent Digital Systems: Proceedings of the 1st International Conference (NIDS 2021), Athens, Greece, September 30-October 1, 2021*, vol. 338, p. 40, IOS Press, 2021. https://doi.org/10.3233/FAIA210073.

[11] Y. Su, Z. Cheng, J. Wu, Y. Dong, Z. Huang, L. Wu, E. Chen, S. Wang, and F. Xie, "Graph-based cognitive diagnosis for intelligent tutoring systems," *Knowledge-Based Systems*, vol. 253, p. 109547, 2022. https://doi.org/10.1016/j.knosys.2022.109547.

[12] T. Qi, M. Ren, L. Guo, X. Li, J. Li, and L. Zhang, "Icd: A new interpretable cognitive diagnosis model for intelligent tutor systems," *Expert Systems with Applications*, vol. 215, p. 119309, 2023. https://doi.org/10.1016/j.eswa.2022.119309.

[13] C. Qin, S. Dong, and X. Yu, "Exploration of polytomous-attribute q-matrix validation in cognitive diagnostic assessment," *Knowledge-Based Systems*, p. 111577, 2024. . https://doi.org/10.1016/j.knosys.2024.111577.

[14] L. A. C. Ahakonye, A. Zainudin, M. J. A. Shanto, J.-M. Lee, D.-S. Kim, and T. Jun, "A multi-mlp prediction for inventory management in manufacturing execution system," *Internet of Things*, p. 101156, 2024. . https://doi.org/10.1016/j.iot.2024.101156.

[15] A. Shah, M. Shah, A. Pandya, R. Sushra, R. Sushra, M. Mehta, K. Patel, and K. Patel, "A comprehensive study on skin cancer detection using artificial neural network (ann) and convolutional neural network (cnn)," *Clinical eHealth*, 2023. . https://doi.org/10.1016/j.ceh.2023.08.002.

[16] C. Yang, G. Weng, and Y. Chen, "Active contour model based on local kullback–leibler divergence for fast image segmentation," *Engineering Applications of Artificial Intelligence*, vol. 123, p. 106472, 2023. https://doi.org/10.1016/j.engappai.2023.106472.

[17] R. P. Chalmers, "mirt: A multidimensional item response theory package for the r environment," *Journal of statistical Software*, vol. 48, pp. 1–29, 2012. https://doi.org/10.18637/jss.v048.i06.

[18] H. Bi, E. Chen, W. He, H. Wu, W. Zhao, S. Wang, and J. Wu, "Beta-cd: A bayesian meta-learned cognitive diagnosis framework for personalized learning," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 37, pp. 5018–5026, 2023. https://doi.org/10.1609/aaai.v37i4.25629.

[19] W. Gao, Q. Liu, Z. Huang, Y. Yin, H. Bi, M.-C. Wang, J. Ma, S. Wang, and Y. Su, "Rcd: Relation map driven cognitive diagnosis for intelligent education systems," in *Proceedings of the 44th international ACM SIGIR conference on research and development in information retrieval*, pp. 501–510, 2021. https://doi.org/10.1145/3404835.3462932.

[20] S. Liu, X. Yu, H. Ma, Z. Wang, C. Qin, and X. Zhang, "Homogeneous cohort-aware group cognitive diagnosis: A multi-grained modeling perspective," in *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*, pp. 4094–4098, 2023. https://doi.org/10.1145/3583780.3615287.

[21] M. Feng, C. Huang, and K. Collins, "Promising long term effects of assistments online math homework support," in *International Conference on Artificial Intelligence in Education*, pp. 212–217, Springer, 2023. https://doi.org/10.1007/978-3-031-36336-$8_3$2.

A Technical Appendices

A.1 Dataset Description

In our experiments, we employ the following publicly available datasets: ASSIST, CDBD_a0910 and NIPS2020. ASSIST, comprising ASSIST2009_2010 [7], ASSIST2012_2013 [20], and ASSIST2017 [8], is an openly accessible dataset sourced from the ASSISTments online learning platform [21]. This dataset encompasses students' responses to mathematics questions. CDBD_a0910 stands as one of the benchmark datasets in cognitive diagnosis, offering clearly student answer logs alongside corresponding information on knowledge concepts pertaining to the each exercise [3]. NIPS2020 is supported by the NeurIPS 2020 Education Challenge competition, offering answer logs from students regarding multiple-choice math questions [20]. Type indicates the storage format of the original data.

The # of Records represents the number of response logs. The # of Student, # of Exercises and # of Knowledge concepts refer to the number of students, exercises, and knowledge concepts, respectively. Due to the presence of null records, missing values, and duplicate entries in the original datasets, preprocessing is required for all datasets before conducting the experiment. In addition, to demonstrate the performance of SCD on long-tailed data, we retain students' data with more than five interaction records. The preprocessed data is stored as files in JSON format.

A.2 Experimental Setup

In this experiment, we have set the total_tasks to 50, meaning that we have prepared 50 task units for training MetaCD. The details are outlined as follows: (1) ASSIST2009_2010, ASSIST2012_2013, and ASSIST2017 are treated as separate task units. (2) Considering that the CDBD_a0910 original data comprises three sub-datasets: train.csv, test.csv, and valid.csv, and each sub-dataset contains a specific number of records, we divide the CDBD_a0910 dataset into 3 task units. (3) Given the large volume of records in NIPS2020, we divide it into 115 task units, each containing 150,000 records.

From these, we randomly select 45 task units for model training (m = 45). In MetaCD, the selection of hyperparameters is primarily optimized using a grid search method. Specifically, the values of α, β, η, λ, and ξ are in the range of (0, 1], with a step size of 0.1 for each change. The value of μ is selected from the set [2, 3, 4]. Furthermore, we set batch_size = 5, and the number of sampling samples s = 128. This implies that during each epoch of KB train support, we will select 5 training task units and sample 128 data points for each task unit to train the model.

In addition, we use Xavier initialization to initialize the MetaCD, aiming to speed up training and improve the performance of the network; and in MetaCD, we use the Dropout algorithm (dropout rate = 0.5) to reduce the problem of overfitting in the model process and use Adam The optimization algorithm performs gradient updates to help the model quickly converge to the optimal solution.

Problem Definition: Given a set of students' response data from training task units set T_r and $\mathcal{Q}$-matrix, the purpose of MetaCD is to learn a model that can predict students' proficiency in skills and their corresponding scores by effectively transferring knowledge across tasks. The ultimate goal is to obtain the best model that can accurately predict students' performance on the testing task T_e.

A.3 Evaluation metrics

In order to better evaluate the performance of MetaCD, we used overall accuracy (ACC), root mean square error (RMSE) and area under the curve (AUC) to evaluate the performance of MetaCD from different perspectives. In addition, in order to evaluate the continual learning ability of MetaCD on task incremental learning, we introduce the backward transfer (BWT) srivastava2023lifelong to evaluate the model. The BWT is an evaluation metric utilized to assess the performance of deep learning models in task incremental learning scenarios. It primarily involves assessing the model's performance across all tasks, including both trained and untrained ones, subsequent to training it on a specific task.

$$BWT = \frac{1}{T-1} \sum_{t=1}^{T-1} M_{T,t} - M_{t,t} \tag{8}$$

[3] https://github.com/bigdata-ustc/EduData.

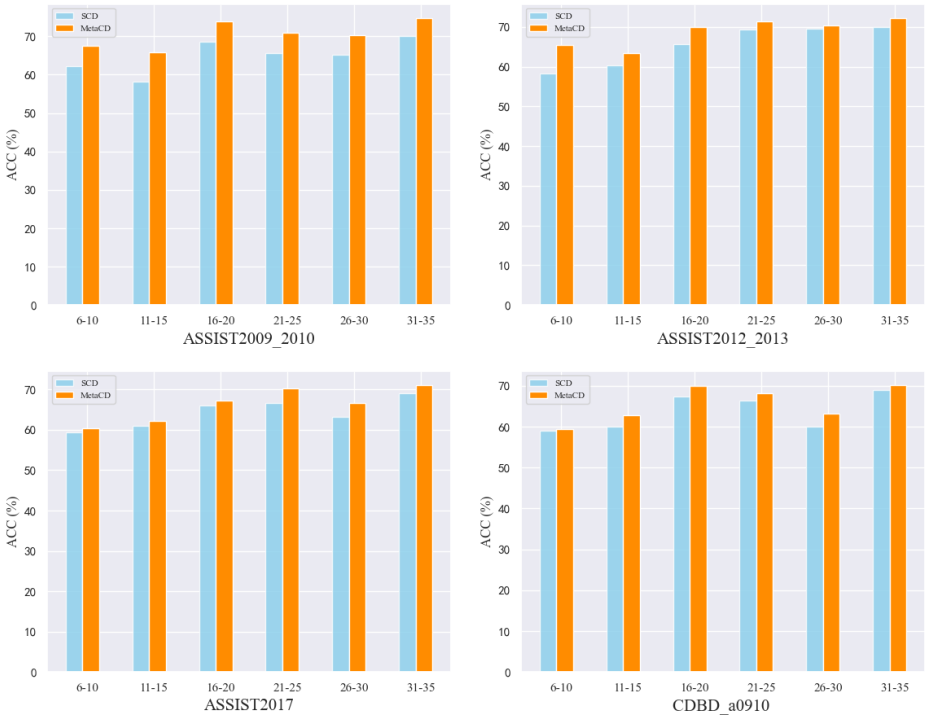

Figure 2: Performance comparison on long-tailed data. The horizontal coordinate represents the number of times that a question has been answered by different students. For example, 5-10 represents a question that has been answered by different students between 5 and 10 times.

where R $\in$ (1, T-1), $M_{R,i}$ indicates the performance of the model on the task t after the model is trained on the R^{th} task.

A.4 Detailed Results of Experiments

Table 2: Performance comparison of models based on the ASSIST 2009_2010 under different data amounts.

Amounts	5,000			100,000			150,000			190,000		
Work	ACC	RMSE	AUC	ACC	RMSE	AUC	ACC	RMSE	AUC	ACC	RMSE	AUC
MetaCD	**0.569**	**0.517**	**0.557**	**0.659**	**0.483**	**0.687**	**0.738**	**0.459**	**0.724**	**0.753**	0.425	**0.771**
IRT	0.463	0.693	0.419	0.532	0.690	0.526	0.585	0.591	0.613	0.654	0.472	0.681
MIRT	0.469	0.690	0.428	0.549	0.675	0.568	0.605	0.570	0.628	0.707	0.461	0.716
DINA	0.476	0.667	0.462	0.596	0.607	0.585	0.651	0.519	0.633	0.644	0.495	0.680
BCD	0.556	0.572	0.539	0.621	0.549	0.618	0.722	0.473	0.706	0.729	0.426	0.763
NCD	0.503	0.581	0.529	0.628	0.536	0.629	0.719	0.493	0.703	0.726	0.441	0.752
RCD	0.557	0.569	0.535	0.615	0.542	0.626	0.705	0.479	0.698	0.724	0.427	0.761
SCD	0.560	0.522	0.542	0.637	0.505	0.679	0.725	0.466	0.709	0.731	**0.423**	0.729

Table 3: Performance comparison of models based on the ASSIST2017 under different data amounts.

Amounts	5,000			10,0000			15,0000			19,0000		
Work	ACC	RMSE	AUC	ACC	RMSE	AUC	ACC	RMSE	AUC	ACC	RMSE	AUC
MetaCD	**0.553**	**0.550**	**0.549**	**0.646**	**0.499**	**0.652**	**0.703**	**0.465**	**0.715**	**0.715**	**0.439**	**0.726**
IRT	0.458	0.685	0.425	0.526	0.697	0.518	0.580	0.599	0.601	0.658	0.464	0.668
MIRT	0.483	0.679	0.417	0.535	0.683	0.547	0.576	0.586	0.618	0.668	0.461	0.678
DINA	0.459	0.665	0.453	0.553	0.639	0.571	0.609	0.558	0.607	0.613	0.519	0.654
BCD	0.540	0.565	0.528	0.635	0.510	0.645	0.690	0.513	0.692	0.701	0.447	0.713
NCD	0.523	0.572	0.528	0.609	0.551	0.603	0.689	0.537	0.673	0.685	0.453	0.699
RCD	0.537	0.570	0.530	0.626	0.532	0.638	0.685	0.519	0.682	0.694	0.450	0.709
SCD	0.541	0.562	0.531	0.628	0.515	0.642	0.692	0.489	0.703	0.703	0.442	0.710

Table 4: The performance of MetaCD on task incremental learning across sequential tasks.

	$Task_1$	$Task_2$	$Task_3$	$Task_4$
$T_1^* \mid T_1$	0.771 / 0.771	/	/	/
$T_2^* \mid T_2$	0.719 / 0.598	0.703 / 0.721	/	/
$T_3^* \mid T_3$	0.706 / 0.533	0.689 / 0.557	0.700 / 0.715	/
$T_4^* \mid T_4$	0.693 / 0.506	0.675 / 0.531	0.686 / 0.519	0.697 / 0.701

Table 5: Accuracy results of ablation study with/without KB, PPM, and Per-class on four different datasets.

Model	ASSIST2009_2010	ASSIST2012_2013	ASSIST2017	CDBD_a0910
MetaCD	0.753	0.725	0.715	0.712
w/o KB	0.719	0.685	0.687	0.668
w/o PPM	0.733	0.708	0.699	0.693
w/o Per-class	0.742	0.713	0.706	0.705