LEARNING HOW STUDENTS UPDATE: GRAPH-CONDITIONED META-OPTIMIZATION OF COGNITIVE STATE FOR KNOWLEDGE TRACING

Anonymous authors
Paper under double-blind review

ABSTRACT

Knowledge tracing predicts a learner's next response from a latent knowledge state and is a core component of adaptive tutoring systems. Recent models increasingly focus on richer representations of interaction histories, but the mechanism for updating the learner state is still typically fixed by the model architecture. On concept-level benchmarks, however, more expressive encoders do not consistently outperform simple recurrent models. We revisit this design choice and argue that a learner's response provides directional evidence about mastery, so the cognitive state should evolve through a learned, preconditioned gradient step on the learner's own prediction error. GRACE represents the cognitive state in concept coordinates, differentiates the prediction loss with respect to the state, and uses a graph-conditioned meta-optimizer to transform the resulting gradient into the next state. The meta-optimizer controls per-concept retention, adaptive step size, graph-based gradient diffusion, and diagonal-plus-low-rank preconditioning from the interaction context. Experiments on ASSIST2009, Junyi, and the AAAI2023 challenge data show that GRACE achieves the strongest overall predictive performance against thirteen baselines spanning recurrent, graph-based, and attention-based knowledge tracing models. Ablation studies identify the learned update mechanism as a major source of the performance gains, while sensitivity analyses consistently favor narrow embeddings.

1 INTRODUCTION

Knowledge tracing estimates a learner's knowledge state from a sequence of interactions and uses that state to predict future responses, making it a fundamental component of adaptive tutoring systems (Corbett & Anderson, 1994; Abdelrahman et al., 2023). In the standard sequential formulation (Piech et al., 2015), a latent state summarizes past interactions, produces a prediction for the current question, and is updated once the learner's response is observed. A knowledge tracing model must therefore determine both how the learner state is represented and how that state changes after each interaction. Most recent work has focused on the former by developing increasingly expressive recurrent, memory, attention, and graph-based representations. By comparison, the transition mechanism itself is usually fixed by the chosen architecture, and it is the focus of this work.

Existing approaches differ substantially in how they encode interaction histories, but they share several limitations in how state changes are modeled. Recurrent models update a hidden state through a recurrent cell (Piech et al., 2015; Zhou et al., 2025), memory models modify learned key-value slots (Zhang et al., 2017), attention models reconstruct the current representation from previous interactions (Pandey & Karypis, 2019; Ghosh et al., 2020; Yin et al., 2023), and graph-based models incorporate concept relations into representation or propagation (Nakagawa et al., 2019; Tong et al., 2020). Forgetting-aware models further introduce temporal decay or recency mechanisms (Nagatani et al., 2019; Shen et al., 2021; Wang et al., 2021; Bai et al., 2025). In these models, the observed response is mainly treated as an input to a transition function. The model must learn how the latent state is organized, infer the direction and magnitude of the state change, and determine how evidence on one concept should affect related concepts. Increasing encoder capacity does not directly address these problems. On the concept-level benchmarks considered in this work, more elaborate attention models do not consistently outperform recurrent or graph-based alternatives (Section 5).

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

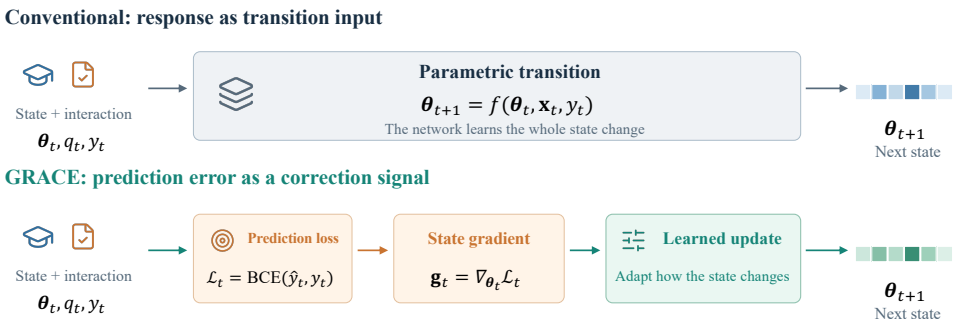

Figure 1: Conventional parametric transition (top) versus the gradient-guided update in GRACE (bottom). In a conventional transition the observed response is an input to a parametric map, so the network must learn the whole state change; in GRACE the prediction loss and its state gradient supply a correction signal, and a learned update adapts how the state changes.

A learner's response provides a supervised signal for correcting the current cognitive state. Let $\boldsymbol{\theta}_t$ denote the current state and $\mathcal{L}_t$ the prediction loss after the response at interaction t is observed. The gradient $\nabla_{\boldsymbol{\theta}_t} \mathcal{L}_t$ identifies a direction in the state space that reduces the inconsistency between the current state and the observed response. We therefore use the state gradient directly rather than requiring a transition network to infer this direction from the response alone. The remaining problem is to determine how that gradient should modify the state: how much of the previous state should be retained, how large the update should be, how the correction should propagate across related concepts, and how different coordinates should be rescaled. This separates the update direction from the update rule and connects knowledge tracing with learned optimization (Andrychowicz et al., 2016; Li & Malik, 2017), with the learner state as the optimized quantity.

We introduce GRACE, which represents the cognitive state in concept coordinates and implements the transition through a graph-conditioned meta-optimizer. The model predicts the learner's response from the current state, a structure-aware item representation, and a state-conditioned concept context. It then differentiates the prediction loss with respect to the cognitive state. The resulting gradient and the interaction context are used to determine a per concept retention gate $\boldsymbol{\rho}_t$, a step size η_t, a graph mixing coefficient β_t, and a diagonal plus low rank preconditioner parameterized by $\mathbf{d}_t$ and $\mathbf{U}_t$. The retention gate determines how much of the previous state is preserved, the step size controls the overall update magnitude, graph diffusion propagates part of the error signal to related concepts, and the preconditioner rescales and couples the resulting update. The same concept graph is used for both concept representation and gradient propagation. Figure 1 contrasts this update with a conventional parametric transition.

We evaluate GRACE on ASSIST2009, Junyi, and the AAAI2023 challenge data against thirteen baselines spanning recurrent, graph based, and attention based knowledge tracing models, and it achieves the best AUC on all three benchmarks. The ablation study shows that the preconditioner and graph-based gradient diffusion account for most of the degradation on ASSIST2009 and AAAI2023 when components are removed, while Junyi benefits more strongly from the representation modules associated with its curated prerequisite structure. The sensitivity analysis shows that widening the concept or item embeddings does not improve performance, with the narrowest settings performing best or tying for best across the datasets. These results indicate that explicitly modeling the state update can provide greater benefit than simply increasing representation capacity.

Our contributions are:

- **Explicit state transition.** Existing knowledge tracing models largely determine state evolution through the architecture of the representation model. We instead formulate cognitive state evolution as a learned optimization step on the learner's prediction loss, making the transition an explicit modeling component.
- **Gradient-guided update.** Existing transitions must infer the direction of state change from the observed response. GRACE uses the gradient of the prediction loss with respect to the cognitive state to provide this direction directly, while a context-dependent meta-optimizer learns retention and update magnitude.

- **Structured cross-concept propagation.** Existing transitions provide limited control over how evidence on one concept should affect related concepts. GRACE combines graph based gradient diffusion with diagonal plus low rank preconditioning to propagate and rescale the update across concept structure.
- **Empirical analysis of transition and representation.** Experiments on three benchmarks against thirteen baselines, together with ablation and sensitivity analyses, separate the contribution of the learned transition from that of representation capacity and show that stronger state updates can matter more than wider representations.

2 Related Work

Deep knowledge tracing models differ in how they encode history and what prior knowledge they attach to the encoder; the transition between knowledge states has remained an architectural choice.

2.1 Deep Knowledge Tracing

Bayesian knowledge tracing tracks a binary mastery latent per skill with fixed learning and slip probabilities (Corbett & Anderson, 1994), and item response theory scores learners against item difficulty statically (Rasch, 1993). DKT lets an LSTM state stand in for mastery (Piech et al., 2015), DKT+ regularizes that state for prediction consistency (Yeung & Yeung, 2018), and key-value memories write mastery into latent-concept slots through learned erase and add operations (Zhang et al., 2017; Abdelrahman & Wang, 2019). Attention models read the history directly instead: SAKT and AKT attend over past interactions (Pandey & Karypis, 2019; Ghosh et al., 2020), and later variants simplify the design, stabilize the traced state, extend the context window, or discount noisy responses (Liu et al., 2023a; Yin et al., 2023; Li et al., 2024a;b; Guo et al., 2025; Cheng et al., 2025). Every model in this family fixes the transition as a parametric map, so the network must learn both what the state means and how it should move. GRACE keeps the state in concept coordinates and learns the update rule instead, with the prediction error supplying the direction.

2.2 Structure-Aware and Forgetting-Aware Knowledge Tracing

Structure-aware models exploit concept and question-concept graphs: GKT and SKT propagate proficiency along edges (Nakagawa et al., 2019; Tong et al., 2020), GIKT, PEBG and SGKT enrich question embeddings from graph neighborhoods (Yang et al., 2021; Liu et al., 2020; Wu et al., 2022), and prerequisite relations either constrain mastery or are learned (Chen et al., 2018; Zhang et al., 2025). Cognitive diagnosis keeps proficiency in explicit concept coordinates through the Q-matrix (de la Torre, 2009; Wang et al., 2020), a representation GRACE shares for the state but not for the dynamics. DKT-Forget appends lag features (Nagatani et al., 2019), LPKT gates learning gain and forgetting by the interval since the last attempt (Shen et al., 2021; 2022), HawkesKT decays past effects with a point process (Wang et al., 2021), and FoLiBi and LefoKT bias attention toward recent interactions (Im et al., 2023; Bai et al., 2025). GRACE conditions a learned optimizer on the graph and the time gap, so the same graph diffuses the error signal across prerequisites and the same time gap sets a per-concept retention gate chosen from data.

2.3 Learned Optimizers and Meta-Learning

Andrychowicz et al. (2016) replace hand-designed gradient descent with a coordinatewise recurrent optimizer. Ravi & Larochelle (2017) cast few-shot adaptation as a learned cell whose forget and input gates act as retention and step size, while MAML learns an initialization from which a few plain gradient steps suffice (Finn et al., 2017). Later work scaled learned optimizers across tasks (Wichrowska et al., 2017; Metz et al., 2022), while preconditioners such as Adam and Shampoo precondition the gradient by fixed rules (Kingma & Ba, 2015; Gupta et al., 2018). These optimizers act on network weights, ignore task structure, and treat coordinates independently. GRACE moves the same idea inside the student model: the optimized quantity is one learner's concept-level state, the meta-optimizer reads the interaction context, and the concept graph shapes both the diffused gradient and a low-rank preconditioner coupling related concepts. No knowledge tracing model we are aware of uses a learned optimizer as its transition.

3 PROBLEM FORMULATION

Knowledge tracing can be formulated as sequential state estimation. At each interaction, the current cognitive state is used to predict the learner's response and is then revised after the response is observed. Existing deep models implement this revision through a parameterized transition determined by the model architecture. GRACE instead constrains the transition to an optimization step on the prediction loss and learns how that step should modify the cognitive state.

Notation. A learner produces a sequence $\{(q_t, y_t, \Delta_t)\}_{t=1}^{T}$, where $q_t \in \{1, \ldots, N\}$ denotes the question answered at interaction t, $y_t \in \{0, 1\}$ is the binary response, and Δ_t is the time elapsed since the previous interaction. The N questions are associated with K concepts $\mathcal{C} = \{c_1, \ldots, c_K\}$ through the Q-matrix $\mathbf{Q} \in \{0, 1\}^{N \times K}$ (de la Torre, 2009). We denote the concept mask of the current question by $\mathbf{m}_t = \mathbf{Q}_{q_t,:}$. A concept adjacency matrix $\mathbf{A} \in \mathbb{R}^{K \times K}$ defines the propagation matrix $\mathbf{S}$ by adding self-loops and row-normalizing $\mathbf{A} + \mathbf{I}$. The cognitive state $\boldsymbol{\theta}_t \in \mathbb{R}^K$ contains one latent mastery value per concept, with $\boldsymbol{\theta}_0 = \mathbf{0}$. We use $\mathcal{L}_t$ for the prediction loss at interaction t and $\mathbf{g}_t = \nabla_{\boldsymbol{\theta}_t} \mathcal{L}_t$ for its gradient with respect to the cognitive state.

Problem definition. At interaction t, the model receives the preceding history $\{(q_\tau, y_\tau, \Delta_\tau)\}_{\tau<t}$, the current question and time gap (q_t, Δ_t), and the structural information encoded by $\mathbf{Q}$ and $\mathbf{A}$. It predicts $\hat{y}_t = p(y_t = 1 \mid \boldsymbol{\theta}_t, q_t)$, the probability of a correct response. For a learner sequence, the prediction objective is the binary cross-entropy (BCE) summed over interactions,

$$\mathcal{L}_{\text{task}} = \sum_{t=1}^{T} \mathcal{L}_t, \qquad \mathcal{L}_t = -\left[y_t \log \hat{y}_t + (1 - y_t) \log(1 - \hat{y}_t)\right]. \tag{1}$$

The cognitive state is latent and receives supervision only through response prediction. Its evolution must therefore be specified by the state transition. Section 4 augments this objective with two regularization terms for the learned update.

State transition. A broad class of deep knowledge tracing models can be expressed as

$$\boldsymbol{\theta}_{t+1} = f(\boldsymbol{\theta}_t, \mathbf{x}_t, y_t), \tag{2}$$

where $\mathbf{x}_t$ encodes the interaction and f is implemented by a recurrent cell, a memory update, or another parameterized mapping (Piech et al., 2015; Zhang et al., 2017; Pandey & Karypis, 2019). The observed response enters f as an input, leaving the model to learn both how the state should change and how large that change should be.

In GRACE, the prediction loss provides the update signal explicitly. After observing y_t, we differentiate $\mathcal{L}_t$ with respect to the current state to obtain $\mathbf{g}_t$. This gradient measures the sensitivity of the prediction error to each state coordinate, and a meta-optimizer determines how it is applied at each interaction. It predicts a retention gate $\boldsymbol{\rho}_t \in (0, 1)^K$, a step size η_t, a graph-mixing coefficient $\beta_t \in (0, 1)$ that diffuses $\mathbf{g}_t$ over the concept graph, and a diagonal-plus-rank-r preconditioner $\mathbf{P}_t$; these control state retention, update magnitude, propagation across related concepts, and coordinate-wise rescaling and coupling. Rather than learning an unrestricted transition function f, GRACE learns an interaction-dependent rule for transforming the prediction gradient into the next cognitive state. Section 4 specifies the prediction model, the graph-based gradient diffusion, and the parameterization of this update.

4 METHODOLOGY

4.1 FRAMEWORK OVERVIEW

GRACE models each learner interaction as one update of the cognitive state $\boldsymbol{\theta}_t \in \mathbb{R}^K$, where the k-th coordinate represents the latent mastery of concept c_k, following the concept-coordinate representation used in cognitive diagnosis (Wang et al., 2020). As shown in Figure 2, each interaction passes through two stages. In the *response prediction* stage (Figure 2, top), the state $\boldsymbol{\theta}_t$, the question q_t, and the structural inputs $\mathbf{Q}$ and $\mathbf{S}$ are turned into structure-aware representations, which a dual-path head combines into the response probability $\hat{y}_t$. In the *meta-optimizer-driven cognitive*

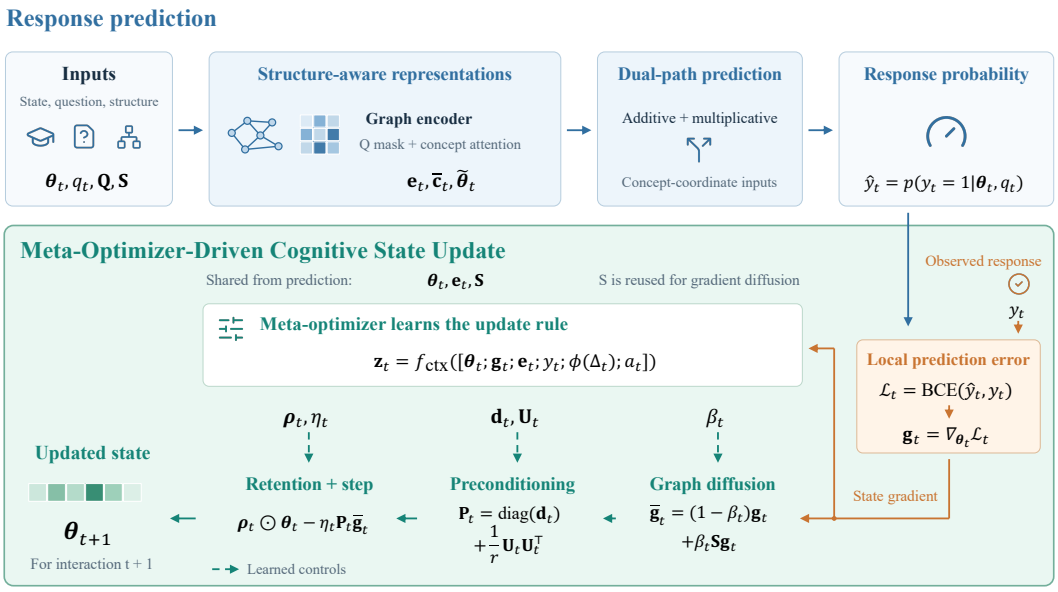

Figure 2: Overview of GRACE. Response prediction (top): the state, question, and structure are encoded into structure-aware representations by a graph encoder with Q-masking and concept attention, and a dual-path (additive and multiplicative) head over concept-coordinate inputs outputs the response probability. Meta-optimizer-driven cognitive state update (bottom): the observed response gives a local prediction error and the state gradient; a meta-optimizer reads the interaction context and outputs learned controls for graph diffusion (β_t), preconditioning ($\mathbf{d}_t, \mathbf{U}_t$), and retention and step ($\boldsymbol{\rho}_t, \eta_t$), which produce the updated state for interaction $t+1$. The state, item representation, and graph $\mathbf{S}$ are shared between the two stages.

state update stage (Figure 2, bottom), the observed response yields a local prediction error and its gradient with respect to the state, and a meta-optimizer then reads the interaction context and outputs learned controls for graph diffusion, preconditioning, retention, and step size, which transform the gradient into $\boldsymbol{\theta}_{t+1}$. The two stages share $\boldsymbol{\theta}_t$, the item representation $\mathbf{e}_t$, and the normalized concept graph $\mathbf{S}$, which first supports concept representation learning and is then reused to diffuse the state gradient.

4.2 Response Prediction

Structure-aware representations. Concept relations provide structural information that an isolated concept embedding cannot recover. We encode the concept set $\mathcal{C}$ with a graph neural network on the propagation matrix $\mathbf{S} = \text{rownorm}(\mathbf{A} + \mathbf{I})$. Starting from a learnable table $\mathbf{H}^{(0)} \in \mathbb{R}^{K \times d_c}$, the first $L-1$ layers keep width d_c and aggregate neighboring concepts through $\mathbf{S}$, and the final layer projects into the K-dimensional concept-coordinate space:

$$\mathbf{H}^{(l+1)} = \text{ReLU}\big(\mathbf{S}\mathbf{H}^{(l)}\mathbf{W}^{(l)}\big), \qquad \mathbf{C} = \mathbf{H}^{(L)} \in \mathbb{R}^{K \times K}, \tag{3}$$

where $\mathbf{W}^{(l)}$ is the weight matrix of layer l. Each row $\mathbf{c}_k$ of $\mathbf{C}$ thus lives in the same coordinate system as $\boldsymbol{\theta}_t$, so state and concept representations can be compared coordinate by coordinate. For question q_t, the Q-matrix row $\mathbf{Q}_{q_t,:}$ identifies the involved concepts; the (normalized) row aggregates their representations, a two-layer MLP maps the result to the item space, and a learnable question identifier embedding adds question-specific information:

$$\mathbf{e}_t = \text{MLP}\big(\mathbf{Q}_{q_t,:}\mathbf{C}\big) + \mathbf{e}^{\text{id}}_{q_t}. \tag{4}$$

Prediction should depend primarily on the concepts tested by the current question while retaining information from related concepts. We first mask the state with $\mathbf{m}_t = \mathbf{Q}_{q_t,:}$, $\tilde{\boldsymbol{\theta}}_t = \boldsymbol{\theta}_t \odot \mathbf{m}_t$, and let the masked state query the concept table through additive attention:

$$\gamma_{t,k} = \text{softmax}_k\left(\mathbf{w}^\top \tanh(\mathbf{W}_s\tilde{\boldsymbol{\theta}}_t + \mathbf{W}_c\mathbf{c}_k)\right), \qquad \bar{\mathbf{c}}_t = \sum_{k=1}^{K} \gamma_{t,k}\mathbf{c}_k, \tag{5}$$

where $\mathbf{W}_s$, $\mathbf{W}_c$, and $\mathbf{w}$ are learnable. The concept context $\bar{\mathbf{c}}_t$ is conditioned on the learner's current state rather than being a fixed graph summary. The triple $(\mathbf{e}_t, \bar{\mathbf{c}}_t, \tilde{\boldsymbol{\theta}}_t)$ feeds the prediction head.

Dual-path prediction. The item representation $\mathbf{e}_t$ and concept context $\bar{\mathbf{c}}_t$ are projected into $\mathbb{R}^K$, giving $\mathbf{e}'_t$ and $\bar{\mathbf{c}}'_t$ in concept coordinates. An additive path operates on their concatenation with the masked state, while a multiplicative path models concept-wise interactions; the two paths are summed and mapped to the response probability:

$$\mathbf{h}_t = \mathrm{ReLU}\left(\mathbf{W}_{\mathrm{add}}[\tilde{\boldsymbol{\theta}}_t; \mathbf{e}'_t; \bar{\mathbf{c}}'_t]\right) + \mathrm{ReLU}\left(\mathbf{W}_{\mathrm{mul}}[\tilde{\boldsymbol{\theta}}_t \odot \mathbf{e}'_t; \tilde{\boldsymbol{\theta}}_t \odot \bar{\mathbf{c}}'_t]\right), \qquad \hat{y}_t = \sigma\left(\mathbf{w}_o^\top \mathbf{h}_t + b_o\right). \tag{6}$$

Biases in the projections and hidden branches are omitted for clarity. The multiplicative path makes the concept-coordinate representation operational: the element-wise products compare the learner state with the current question and concept context in the same coordinate system.

4.3 Meta-Optimizer-Driven Cognitive State Update

The central component of GRACE is the state update. Conventional transition modules receive the response as an input and must learn both the direction and the magnitude of the state change. GRACE separates the two: the prediction loss determines the update signal, and a meta-optimizer learns how that signal is transformed.

Local prediction error and state gradient. After observing y_t, we compute the BCE loss and its gradient with respect to the state,

$$\mathcal{L}_t = -\left[y_t \log \hat{y}_t + (1 - y_t)\log(1 - \hat{y}_t)\right], \qquad \mathbf{g}_t = \nabla_{\boldsymbol{\theta}_t} \mathcal{L}_t. \tag{7}$$

Because the state enters the prediction through the question mask, $\mathbf{g}_t$ is localized to the concepts of the current question and identifies which coordinates contributed to the prediction error. What remains is to decide how far to move along this signal, how to propagate it to related concepts, how to rescale its coordinates, and how much of the previous state to keep.

Meta-optimizer. These decisions are generated from the interaction context

$$\mathbf{z}_t = f_{\mathrm{ctx}}\left([\boldsymbol{\theta}_t; \mathbf{g}_t; \mathbf{e}_t; y_t; \phi(\Delta_t); a_t]\right), \tag{8}$$

where f_{ctx} is a two-layer ReLU MLP, $\phi(\Delta_t)$ is a learned encoding of the elapsed time, and $a_t = n_t / \max(t, 1)$ is the learner's running accuracy. Five output heads produce the learned controls:

$$\beta_t = \sigma(\mathbf{w}_\beta^\top \mathbf{z}_t), \qquad \mathbf{d}_t = \mathrm{softplus}(\mathbf{W}_d \mathbf{z}_t), \qquad \mathbf{U}_t = \mathrm{clip}\left(\mathrm{reshape}(\mathbf{W}_U \mathbf{z}_t), -10, 10\right) \in \mathbb{R}^{K \times r},$$
$$\boldsymbol{\rho}_t = \sigma(\mathbf{W}_\rho \mathbf{z}_t) \in (0, 1)^K, \qquad \eta_t = \mathrm{clip}\left(\mathrm{softplus}(\mathbf{w}_\eta^\top \mathbf{z}_t), 10^{-6}, 0.1\right). \tag{9}$$

The matrices $\mathbf{W}_\rho$, $\mathbf{W}_d$, $\mathbf{W}_U$ and vectors $\mathbf{w}_\eta$, $\mathbf{w}_\beta$ are learned jointly with the context encoder; clipping only stabilizes the unrolled state recursion. The controls act on the gradient in three steps.

Graph diffusion. The mixing coefficient β_t controls how much of the state gradient is propagated through the concept graph:

$$\bar{\mathbf{g}}_t = (1 - \beta_t)\mathbf{g}_t + \beta_t \mathbf{S}\mathbf{g}_t. \tag{10}$$

Since β_t is predicted per interaction, the strength of structural propagation adapts to the current learner state and observation. The same $\mathbf{S}$ is used by the concept encoder in Eq. (3), so representation learning and state updating are defined on a common concept structure.

Preconditioning. The outputs $\mathbf{d}_t$ and $\mathbf{U}_t$ construct a diagonal-plus-low-rank preconditioner

$$\mathbf{P}_t = \mathrm{diag}(\mathbf{d}_t) + \frac{1}{r}\mathbf{U}_t \mathbf{U}_t^\top. \tag{11}$$

The diagonal term provides a concept-specific scale for the diffused gradient, and the rank-r term captures coordinated changes among concept dimensions, so a localized error can induce a structured update without a dense $K \times K$ transformation.

Retention and step. The retention gate $\boldsymbol{\rho}_t$ sets how much of each concept state is preserved; because it depends on the full context, forgetting is learned per interaction rather than imposed by a fixed temporal decay. The scalar step size η_t sets the overall update magnitude. The updated state for interaction $t+1$ is

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\rho}_t \odot \boldsymbol{\theta}_t - \eta_t \mathbf{P}_t \bar{\mathbf{g}}_t. \tag{12}$$

The four controls thus play distinct roles: retention governs persistence of the existing state, the step size governs update magnitude, graph diffusion transfers part of the error signal to structurally related concepts, and the preconditioner governs coordinate-specific scaling and coupling. All are conditioned on the current interaction rather than fixed globally, so the transition preserves an explicit gradient direction while the form of the state change adapts to the learner and the interaction.

4.4 Training Objective

The transition in Eq. (12) is applied recursively over the learner sequence, so each update is evaluated through subsequent predictions. All modules are trained jointly with

$$\mathcal{L} = \sum_{t=1}^{T} \left[\mathcal{L}_t + \lambda_1 \left\| \boldsymbol{\theta}_{t+1} - \boldsymbol{\theta}_t \right\|_2^2 + \lambda_2 \left\| \mathbf{U}_t \right\|_F^2 \right]. \tag{13}$$

The first regularizer discourages abrupt changes in the cognitive state, reflecting the assumption that mastery evolves gradually; the second bounds the low-rank component so that cross-concept coupling does not dominate the diagonal scaling. Each learner sequence is one training unit with $\boldsymbol{\theta}_0 = \mathbf{0}$ and the running response count reset at its start, and the concept encoder, item encoder, attention module, prediction head, and meta-optimizer are optimized by backpropagation through the unrolled state updates.

5 Experiments

5.1 Experimental Setup

Datasets. We evaluate GRACE on three public benchmarks (statistics in Table 2, Appendix A): ASSIST2009 (Feng et al., 2009), Junyi (Chang et al., 2015), and AAAI2023, the AAAI 2023 Global Knowledge Tracing Challenge data released by TAL Education (Peiyou) (AI4ED Workshop Organizers, 2023), whose related public release is XES3G5M (Liu et al., 2023b). ASSIST2009 and Junyi are concept-level datasets with one concept per question, whereas AAAI2023 is question-level with several concepts per question, so the state update is evaluated under both single-concept and multi-concept interactions. The concept graph $\mathbf{A}$ is built from the available concept relations and the Q-matrix $\mathbf{Q}$ from the question-to-concept annotations.

Metrics and Baselines. We report AUC and ACC for next-response prediction, following the pyKT benchmark (Liu et al., 2022), and compare GRACE with thirteen deep knowledge tracing models from three families: recurrent DKT (Piech et al., 2015) and DKT2 (Zhou et al., 2025); graph-based GKT (Nakagawa et al., 2019); and attention-based SAKT (Pandey & Karypis, 2019), AKT (Ghosh et al., 2020), simpleKT (Liu et al., 2023a), DTransformer (Yin et al., 2023), stableKT (Li et al., 2024a), extraKT (Li et al., 2024b), LefoKT (Bai et al., 2025), RobustKT (Guo et al., 2025), UKT (Cheng et al., 2025), and ACE-KT (Guo et al., 2026). All baselines use their public implementations on the same data splits and under the same protocol as GRACE.

Implementation Details. We use $d_c = d = 32$, preconditioner rank $r = 4$, and $(\lambda_1, \lambda_2) = (0.6, 0.4)$; the step size η_t is clipped to $[10^{-6}, 0.1]$ and the entries of $\mathbf{U}_t$ to $[-10, 10]$. Each learner sequence is one training unit with $\boldsymbol{\theta}_0 = \mathbf{0}$, and all components are trained jointly through the unrolled state updates (Appendix B).

5.2 Overall Performance

Table 1 and Figure 3 show that GRACE achieves the best AUC on all three benchmarks and the best ACC on Junyi and AAAI2023.More importantly, stronger history encoders do not consistently translate into better knowledge tracing performance. On the two concept-level datasets, recurrent and graph-based models remain competitive with more elaborate attention-based approaches despite

Table 1: Next-response prediction (test AUC and ACC). Best result per column in bold.

Method	ASSIST2009		Junyi		AAAI2023	
	AUC	ACC	AUC	ACC	AUC	ACC
DKT	0.7790	**0.7296**	0.8675	0.7915	0.7987	0.8209
DKT2	0.7784	0.7292	0.8752	0.8088	0.8131	0.8247
GKT	0.7731	0.7264	0.8759	0.7928	0.7618	0.8040
SAKT	0.7412	0.7014	0.8695	0.7949	0.8204	0.8315
AKT	0.7537	0.7149	0.8163	0.7442	0.7607	0.8067
simpleKT	0.7495	0.7111	0.8085	0.7368	0.7482	0.7988
DTransformer	0.7475	0.7083	0.8066	0.7369	0.7538	0.8056
stableKT	0.7555	0.7143	0.8126	0.7425	0.7547	0.8042
extraKT	0.7536	0.7124	0.7988	0.7358	0.7218	0.7941
LefoKT	0.7537	0.7154	0.8164	0.7444	0.7602	0.8077
RobustKT	0.7531	0.7136	0.8198	0.7418	0.7629	0.8091
UKT	0.7494	0.7133	0.8155	0.7424	0.7561	0.8068
ACE-KT	0.7522	0.7144	0.8118	0.7405	0.7493	0.8040
GRACE	**0.7835**	0.7222	**0.8896**	**0.8147**	**0.8451**	**0.8397**

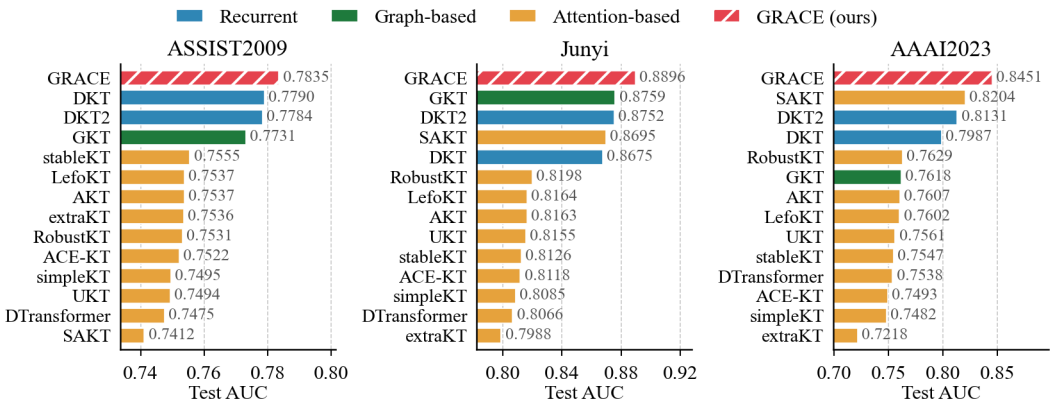

Figure 3: Test AUC by model family. GRACE is best on all three benchmarks, while more complex attention-based models do not consistently outperform recurrent or graph-based ones.

substantial differences in how they encode interaction histories. Their state transitions are still fixed by the underlying architecture. GRACE improves over these baselines by changing how the cognitive state is updated rather than by enlarging the history encoder: the prediction gradient provides an explicit update signal, and the meta-optimizer learns how to retain, propagate, and precondition this signal instead of relying on a generic parametric map to infer the entire state change. The advantage also holds on AAAI2023, where questions involve multiple concepts and attention-based models are more competitive, so the learned transition is effective across data granularities. The results support our central claim that explicitly modeling state evolution can provide greater benefit than relying solely on more complex representations.

5.3 Ablation Study

To test whether the gains of GRACE come mainly from the learned state transition or from the representation modules, we remove components cumulatively in the following order: the diagonal-plus-low-rank preconditioner, graph-diffused gradient, concept graph encoder, and concept attention. Figure 4 reports AUC, and Table 3 in Appendix C provides the complete results. On ASSIST2009 and AAAI2023, the largest degradation occurs after removing the preconditioner and the graph-diffused gradient, whereas removing the representation modules causes a smaller additional decline, so the main gains on these datasets come from how the prediction gradient is transformed into the next state rather than from representational complexity. This matches the design of GRACE: the Q-masked state localizes the gradient to the concepts of the current question, graph diffusion propa-

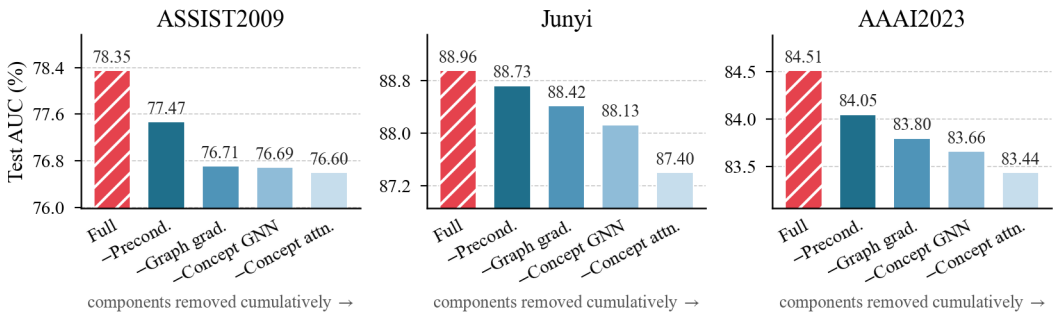

Figure 4: Cumulative ablation in test AUC. The preconditioner and graph-diffused gradient contribute most strongly on ASSIST2009 and AAAI2023, while Junyi benefits more from the representation modules supported by its prerequisite structure.

gates it to structurally related concepts, and the low-rank component of the preconditioner introduces coordinated changes across concept dimensions. So removing these components therefore weakens the structured cross-concept updates that distinguish GRACE from conventional transitions. Junyi shows a different pattern, with the concept graph encoder and concept attention contributing more strongly, consistent with its curated prerequisite structure, which already improves concept representation and context construction before the state gradient is computed. Representation learning and state updating are therefore complementary, with their relative importance depending on the structure available in the dataset; the dominant contribution of the update components on ASSIST2009 and AAAI2023 is nevertheless direct evidence that the learned transition is a major source of the gains of GRACE.

5.4 Sensitivity Analysis

We vary the concept embedding width d_c, item embedding width d, preconditioner rank r, and regularization weights (λ_1, λ_2); Figure 5 and Table 4 in Appendix D report the complete results. The clearest pattern concerns representation width: the smallest tested concept and item embeddings achieve the best or tied-best performance on all three datasets, while widening them provides no consistent benefit. Once the transition explicitly models retention, cross-concept propagation, and adaptive preconditioning, adding capacity to the representation modules contributes little and can even reduce performance. By comparison, the model is stable across the tested preconditioner ranks and regularization settings, so the performance does not rely on a narrowly selected update rank or regularization combination. Together with the ablation results, the sensitivity study points to the same conclusion: the effective capacity of GRACE comes primarily from the learned state transition rather than from wider concept or item representations.

6 Conclusion

We introduced GRACE, a knowledge tracing framework that treats cognitive-state evolution as a learned optimization process rather than a fixed architectural transition. The prediction gradient supplies an explicit correction signal, and a graph-conditioned meta-optimizer learns how that signal is retained, scaled, propagated across related concepts, and preconditioned, separating the direction of state correction from the rule that applies it. On ASSIST2009, Junyi, and AAAI2023, GRACE achieves the best AUC against thirteen recurrent, graph-based, and attention-based baselines. Ablations identify the learned update as a major source of the gains on ASSIST2009 and AAAI2023, while Junyi benefits more from representation modules supported by its curated prerequisite structure, and sensitivity results consistently favor compact representations, suggesting that improving state evolution can be more valuable than increasing representation capacity. These findings suggest that the state transition deserves the same design attention as the history encoder, and that a learned, gradient-guided update is a compact way to provide this capacity. Future work will examine whether the learned retention gates and update dynamics correspond to interpretable patterns of learning and forgetting, extend structured gradient propagation to richer multi-concept and cross-course relations, and use the resulting cognitive states for downstream adaptive learning and recommendation.

AI USE STATEMENT

Generative AI tools were used solely to improve the writing of the manuscript, including rephrasing sentences, improving clarity and readability, and maintaining consistent terminology. They were not used to generate research ideas, design the method, plan experiments, analyze results, or draw scientific conclusions. All technical content, experimental work, and interpretation of the results were carried out by the authors. The authors reviewed all AI-assisted edits and take full responsibility for the final content of the paper.

ETHICS STATEMENT

This work uses only existing public knowledge tracing benchmark datasets and does not involve new data collection or direct interaction with human participants. The study focuses on next-response prediction and methodological analysis, and does not make decisions about individual learners. Nevertheless, knowledge tracing models may be used in educational systems where prediction errors could influence the learning content or support presented to students. We therefore regard the model outputs as predictive estimates rather than direct measurements of a learner's ability or knowledge. Any real-world deployment should include appropriate validation, consideration of potential bias across learner populations, and human oversight. We are not aware of any additional conflicts of interest, privacy concerns, or legal issues arising from this work.

REPRODUCIBILITY STATEMENT

The main paper and appendices provide the information needed to reproduce our experiments. Section 5 specifies the datasets, evaluation metrics, baseline models, and experimental protocol used for the main results. Appendix A describes the datasets and preprocessing procedures, Appendix B gives the implementation and training details of GRACE, Appendix C reports the full ablation results, and Appendix D provides the complete sensitivity results. All baselines are evaluated using their public implementations under the same data splits and evaluation protocol as GRACE. Appendix E provides an anonymous link to the source code and experiment scripts used in this work, together with the instructions needed to reproduce the reported results.

REFERENCES

Ghodai Abdelrahman and Qing Wang. Knowledge tracing with sequential key-value memory networks. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 175–184, 2019.

Ghodai Abdelrahman, Qing Wang, and Bernardo Nunes. Knowledge tracing: A survey. *ACM Computing Surveys*, 55(11):1–37, 2023.

AI4ED Workshop Organizers. AAAI2023 Global Knowledge Tracing Challenge. `https://ai4ed.cc/competitions/aaai2023competition`, 2023.

Marcin Andrychowicz, Misha Denil, Sergio Gomez Colmenarejo, Matthew W. Hoffman, David Pfau, et al. Learning to learn by gradient descent by gradient descent. In *Neural Information Processing Systems*, 2016.

Youheng Bai, Xueyi Li, Zitao Liu, Yaying Huang, Teng Guo, Mingliang Hou, Feng Xia, and Weiqi Luo. Rethinking and improving student learning and forgetting processes for attention based knowledge tracing models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, 2025.

Haw-Shiuan Chang, Hwai-Jung Hsu, and Kuan-Ta Chen. Modeling exercise relationships in e-learning: A unified approach. In *Proceedings of the 8th International Conference on Educational Data Mining*, pp. 532–535, 2015.

Penghe Chen, Yu Lu, Vincent W Zheng, and Yang Pian. Prerequisite-driven deep knowledge tracing. In *2018 IEEE International Conference on Data Mining (ICDM)*, pp. 39–48, 2018.

Weihua Cheng, Hanwen Du, Chunxiao Li, Ersheng Ni, Liangdi Tan, et al. Uncertainty-aware knowledge tracing. In *AAAI Conference on Artificial Intelligence*, 2025.

Albert T Corbett and John R Anderson. Knowledge tracing: Modeling the acquisition of procedural knowledge. *User Modeling and User-Adapted Interaction*, 4(4):253–278, 1994.

Jimmy de la Torre. DINA model and parameter estimation: A didactic. *Journal of Educational and Behavioral Statistics*, 34(1):115–130, 2009.

Mingyu Feng, Neil Heffernan, and Kenneth Koedinger. Addressing the assessment challenge with an online system that tutors as it assesses. *User Modeling and User-Adapted Interaction*, 19(3): 243–266, 2009.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *International Conference on Machine Learning*, pp. 1126–1135, 2017.

Aritra Ghosh, Neil Heffernan, and Andrew S. Lan. Context-aware attentive knowledge tracing. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 2330–2339, 2020.

Teng Guo, Yu Qin, Yubin Xia, Mingliang Hou, Zitao Liu, Feng Xia, and Weiqi Luo. Enhancing knowledge tracing through decoupling cognitive pattern from error-prone data. In *Proceedings of the ACM on Web Conference 2025*, pp. 5108–5116, 2025.

Teng Guo, Yubin Xia, Jinsen Ke, Mingliang Hou, Jiaqi Zheng, and Zitao Liu. Ace-kt: Cascaded cognitive modeling for stage-wise knowledge tracing. In *Proceedings of The 29th International Conference on Artificial Intelligence and Statistics*, pp. 4537–4545, 2026.

Vineet Gupta, Tomer Koren, and Yoram Singer. Shampoo: Preconditioned stochastic tensor optimization. In *International Conference on Machine Learning*, pp. 1842–1850, 2018.

Yoonjin Im, Eunseong Choi, Heejin Kook, and Jongwuk Lee. Forgetting-aware linear bias for attentive knowledge tracing. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*, pp. 3958–3962, 2023.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015.

Ke Li and Jitendra Malik. Learning to optimize. In *International Conference on Learning Representations*, 2017.

Xueyi Li, Youheng Bai, Teng Guo, Zitao Liu, Yaying Huang, Xiangyu Zhao, Feng Xia, Weiqi Luo, and Jian Weng. Enhancing length generalization for attention based knowledge tracing models with linear biases. In *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence*, pp. 5918–5926, 2024a.

Xueyi Li, Youheng Bai, Teng Guo, Ying Zheng, Mingliang Hou, Bojun Zhan, Yaying Huang, Zitao Liu, Boyu Gao, and Weiqi Luo. Extending context window of attention based knowledge tracing models via length extrapolation. In *Proceedings of the 27th European Conference on Artificial Intelligence (ECAI)*, pp. 1479–1486, 2024b.

Yunfei Liu, Yang Yang, Xianyu Chen, Jian Shen, Haifeng Zhang, and Yong Yu. Improving knowledge tracing via pre-training question embeddings. In *Proceedings of the 29th International Joint Conference on Artificial Intelligence*, pp. 1577–1583, 2020.

Zitao Liu, Qiongqiong Liu, Jiahao Chen, Shuyan Huang, Jiliang Tang, et al. pyKT: A Python library to benchmark deep learning based knowledge tracing models. In *Neural Information Processing Systems*, 2022.

Zitao Liu, Qiongqiong Liu, Jiahao Chen, Shuyan Huang, and Weiqi Luo. simpleKT: A simple but Tough-to-Beat baseline for knowledge tracing. In *International Conference on Learning Representations*, 2023a.

Zitao Liu, Qiongqiong Liu, Teng Guo, Jiahao Chen, Shuyan Huang, Xiangyu Zhao, Jiliang Tang, Weiqi Luo, and Jian Weng. XES3G5M: A knowledge tracing benchmark dataset with auxiliary information. In *Advances in Neural Information Processing Systems*, volume 36, 2023b.

Luke Metz, James Harrison, C Daniel Freeman, Amil Merchant, Lucas Beyer, James Bradbury, Naman Agrawal, Ben Poole, Igor Mordatch, Adam Roberts, and Jascha Sohl-Dickstein. VeLO: Training versatile learned optimizers by scaling up. *arXiv preprint arXiv:2211.09760*, 2022.

Koki Nagatani, Qian Zhang, Masahiro Sato, Yong-Yi Chen, Francine Chen, and Tomoko Ohkuma. Augmenting knowledge tracing by considering forgetting behavior. In *The World Wide Web Conference*, pp. 3101–3107, 2019.

Hiromi Nakagawa, Yusuke Iwasawa, and Yutaka Matsuo. Graph-based knowledge tracing: Modeling student proficiency using graph neural network. In *IEEE/WIC/ACM International Conference on Web Intelligence*, pp. 156–163, 2019.

Shalini Pandey and George Karypis. A self-attentive model for knowledge tracing. In *Proceedings of the 12th International Conference on Educational Data Mining*, pp. 384–389, 2019.

Chris Piech, Jonathan Bassen, Jonathan Huang, Surya Ganguli, Mehran Sahami, Leonidas J Guibas, and Jascha Sohl-Dickstein. Deep knowledge tracing. In *Advances in Neural Information Processing Systems*, volume 28, 2015.

Georg Rasch. *Probabilistic models for some intelligence and attainment tests*. MESA Press, 1993.

Sachin Ravi and Hugo Larochelle. Optimization as a model for few-shot learning. In *International Conference on Learning Representations*, 2017.

Shuanghong Shen, Qi Liu, Enhong Chen, Zhenya Huang, Wei Huang, Yu Yin, Yu Su, and Shijin Wang. Learning process-consistent knowledge tracing. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pp. 1452–1460, 2021.

Shuanghong Shen, Enhong Chen, Qi Liu, Zhenya Huang, Wei Huang, Yu Yin, Yu Su, and Shijin Wang. Monitoring student progress for learning process-consistent knowledge tracing. *IEEE Transactions on Knowledge and Data Engineering*, pp. 1–15, 2022.

Shiwei Tong, Qi Liu, Wei Huang, Zhenya Huang, Enhong Chen, Chuanren Liu, Haiping Ma, and Shijin Wang. Structure-based knowledge tracing: An influence propagation view. In *2020 IEEE International Conference on Data Mining (ICDM)*, pp. 541–550, 2020.

Chenyang Wang, Weizhi Ma, Min Zhang, Chuancheng Lv, Fengyuan Wan, Huijie Lin, Taoran Tang, Yiqun Liu, and Shaoping Ma. Temporal cross-effects in knowledge tracing. In *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*, pp. 517–525, 2021.

Fei Wang, Qi Liu, Enhong Chen, Zhenya Huang, Yuying Chen, Yu Yin, Zai Huang, and Shijin Wang. Neural cognitive diagnosis for intelligent education systems. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(04):6153–6161, 2020.

Olga Wichrowska, Niru Maheswaranathan, Matthew W. Hoffman, Sergio Gomez Colmenarejo, Misha Denil, et al. Learned optimizers that scale and generalize. In *International Conference on Machine Learning*, 2017.

Zhengyang Wu, Li Huang, Qionghao Huang, Changqin Huang, and Yong Tang. SGKT: Session graph-based knowledge tracing for student performance prediction. *Expert Systems with Applications*, 206:117681, 2022.

Yang Yang, Jian Shen, Yanru Qu, Yunfei Liu, Kerong Wang, Yaoming Zhu, Weinan Zhang, and Yong Yu. GIKT: A graph-based interaction model for knowledge tracing. In *Machine Learning and Knowledge Discovery in Databases (ECML PKDD)*, pp. 299–315, 2021.

Chun-Kit Yeung and Dit-Yan Yeung. Addressing two problems in deep knowledge tracing via prediction-consistent regularization. In *Proceedings of the Fifth Annual ACM Conference on Learning at Scale*, pp. 1–10, 2018.

Yu Yin, Le Dai, Zhenya Huang, Shuanghong Shen, Fei Wang, Qi Liu, Enhong Chen, and Xin Li. Tracing knowledge instead of patterns: Stable knowledge tracing with diagnostic transformer. In *Proceedings of the ACM Web Conference 2023*, pp. 855–864, 2023.

Jiani Zhang, Xingjian Shi, Irwin King, and Dit-Yan Yeung. Dynamic key-value memory networks for knowledge tracing. In *Proceedings of the 26th International Conference on World Wide Web*, pp. 765–774, 2017.

Miao Zhang, Jiawei Wang, Kui Xiao, Shihui Wang, Yan Zhang, Hao Chen, and Zhifei Li. Learning concept prerequisite relation via global knowledge relation optimization. *Proceedings of the AAAI Conference on Artificial Intelligence*, 39(2):1638–1646, 2025.

Yiyun Zhou, Wenkang Han, and Jingyuan Chen. DKT2: Revisiting applicable and comprehensive knowledge tracing in large-scale data. In *Machine Learning and Knowledge Discovery in Databases: Research Track (ECML PKDD)*, 2025.

A Dataset Details and Preprocessing

Table 2 summarizes the three datasets. ASSIST2009 is the skill-builder subset of the ASSISTments platform, in which each question is tagged with a single concept, so the 117 questions coincide with the 117 concepts. Junyi is collected from the Junyi Academy mathematics platform and is likewise organized at the level of 715 concepts, each question mapping to one concept. AAAI2023 is the data released for the AAAI 2023 Global Knowledge Tracing Challenge by TAL Education (Peiyou); it is organized at the question level, with 7,453 questions tagged by 859 concepts and 3.25 concepts per question on average, and it has the longest learner histories of the three (a median of 100 records per learner). Learner sequences are ordered by timestamp, and the time gap Δ_t is measured between consecutive interactions of the same learner.

Table 2: Dataset statistics. Concepts/Q is the average number of concepts per question; Avg. and Med. are the mean and median records per learner; Correct is the share of correct responses.

Dataset	#Learners	#Questions	#Concepts	Concepts/Q	#Records	Avg.	Med.	Correct
ASSIST2009	3,776	117	117	1.00	181,877	48.2	33	62.76%
Junyi	42,154	715	715	1.00	2,172,984	51.5	43	52.40%
AAAI2023	33,309	7,453	859	3.25	2,970,581	89.2	100	78.79%

B Implementation Details

The concept graph encoder stacks L propagation layers of the form in Eq. (3), and its last layer has width K so that concept coordinates coincide with the coordinates of the cognitive state. The item encoder is a two-layer MLP with ReLU; the item ID embedding has the same width as its output. The concept embedding width and the item embedding width are both 32, which the sweeps in Section 5 identify as the best or tied-best setting on every dataset. The meta-optimizer context network is a two-layer MLP with ReLU whose input concatenates the state, the state gradient, the item vector, the response, a two-layer MLP encoding of the time gap, and the running accuracy. The step size is clipped to $[10^{-6}, 0.1]$ after the softplus, the entries of the low-rank factor are clipped to $[-10, 10]$, and the preconditioner rank is $r = 4$ on all datasets. The regularization weights are $(\lambda_1, \lambda_2) = (0.6, 0.4)$. Each learner sequence is one training unit, the initial state is $\boldsymbol{\theta}_0 = \mathbf{0}$, and all modules are trained end to end by backpropagation through the unrolled state updates.

C Full Ablation Results

Table 3 reports AUC and ACC for the cumulative ablation of Section 5. ACC moves within a narrow band across variants on Junyi and AAAI2023; on Junyi, removing the preconditioner raises ACC while lowering AUC, so we treat AUC as the primary metric when attributing gains to components.

Table 3: Cumulative ablation (test AUC and ACC). Each row removes one more component than the row above it.

Variant	ASSIST2009		Junyi		AAAI2023	
	AUC	ACC	AUC	ACC	AUC	ACC
GRACE (full)	0.7835	0.7222	0.8896	0.8147	0.8451	0.8397
w/o preconditioner	0.7747	0.7191	0.8873	0.8183	0.8405	0.8377
w/o graph-diffused gradient	0.7671	0.7110	0.8842	0.8154	0.8380	0.8368
w/o concept graph encoder	0.7669	0.7123	0.8813	0.8136	0.8366	0.8378
w/o concept attention	0.7660	0.7080	0.8740	0.8008	0.8344	0.8357

D Full Sensitivity Results

Figure 5 reports the AUC difference from the best configuration within each sweep of Section 5, and Table 4 lists the raw AUC and ACC behind it. The sweeps were run in stages (concept width,

then item width, then rank, then regularization weights), each stage keeping the best values chosen by the earlier stages, so absolute AUC rises slightly from stage to stage and the best row of the final stage coincides with the configuration of Appendix B.

Table 4: Full sensitivity results (test AUC and ACC) for the concept embedding width, item embedding width, preconditioner rank, and regularization weights.

Hyperparameter	Value	ASSIST2009		Junyi		AAAI2023	
		AUC	ACC	AUC	ACC	AUC	ACC
Concept width d_c	32	0.7779	0.7220	0.8891	0.8167	0.8411	0.8386
	64	0.7744	0.7145	0.8834	0.8100	0.8345	0.8355
	128	0.7642	0.7016	0.8685	0.8075	0.8410	0.8381
	256	0.7755	0.7132	0.8778	0.8066	0.8409	0.8384
	512	0.7714	0.7161	0.8732	0.8039	0.8409	0.8378
	1024	0.7764	0.7134	0.8667	0.8041	0.8411	0.8381
Item width d	32	0.7833	0.7233	0.8891	0.8167	0.8435	0.8392
	64	0.7794	0.7213	0.8839	0.8132	0.8422	0.8387
	128	0.7766	0.7183	0.8804	0.8110	0.8433	0.8391
	256	0.7776	0.7174	0.8804	0.8091	0.8430	0.8389
	512	0.7779	0.7169	0.8823	0.8133	0.8423	0.8393
	1024	0.7779	0.7220	0.8836	0.8140	0.8411	0.8386
Rank r	2	0.7768	0.7167	0.8890	0.8182	0.8436	0.8390
	4	0.7830	0.7226	0.8891	0.8167	0.8436	0.8395
	6	0.7774	0.7209	0.8849	0.8129	0.8448	0.8390
	8	0.7828	0.7226	0.8827	0.8084	0.8424	0.8380
(λ_1, λ_2)	(0.01, 0.01)	0.7830	0.7226	0.8881	0.8170	0.8448	0.8390
	(0.2, 0.8)	0.7830	0.7226	0.8883	0.8170	0.8445	0.8398
	(0.4, 0.6)	0.7796	0.7196	0.8880	0.8153	0.8437	0.8395
	(0.6, 0.4)	0.7835	0.7222	0.8896	0.8147	0.8451	0.8397
	(0.8, 0.2)	0.7790	0.7163	0.8890	0.8182	0.8446	0.8391

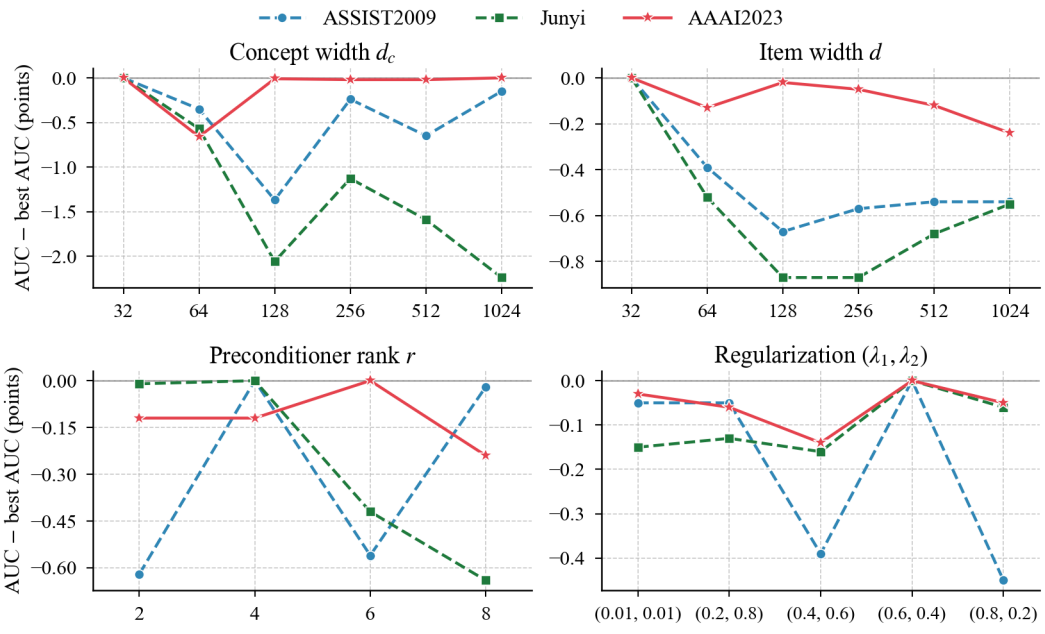

Figure 5: Sensitivity of test AUC to embedding width, preconditioner rank, and regularization. Narrow concept and item embeddings perform best or tie for best across the datasets, while rank and regularization have comparatively limited effects.

E CODE AND REPRODUCIBILITY RESOURCES

The anonymized source code and experiment scripts used in this work are available at `https://anonymous.4open.science/r/ICLR-ED2C/`. The repository contains the implementation of GRACE, data preprocessing scripts, training and evaluation scripts, experiment configurations, hyperparameter settings, and instructions for reproducing the main experimental results, ablation studies, and sensitivity analyses reported in this paper.