软件学报 ISSN 1000-9825, CODEN RUXUEW
Journal of Software, 2023,34(3):1365−1395 [doi: 10.13328/j.cnki.jos.006715]

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel: +86-10-62562563

深度知识追踪模型综述和性能比较*

王 宇, 朱梦霞, 杨尚辉, 陆雪松, 周傲英

(华东师范大学 数据科学与工程学院, 上海 200062)

通信作者: 陆雪松, E-mail: xslu@dase.ecnu.edu.cn

摘 要: 知识追踪是一种重要的认知诊断方法, 往往被用于在线学习平台、智能辅导系统等信息化教学平台中. 知识追踪模型通过分析学生与课程作业的交互数据, 即时模拟学生对课程知识点的掌握水平, 模拟的结果可以用来预测学生未来的学习表现, 并帮助他们规划个性化的学习路径. 在过去 20 多年中, 知识追踪模型的构建通常基于统计学和认知科学的相关理论. 随着教育大数据的开放和应用, 基于深度神经网络的模型(以下简称“深度知识追踪模型”)以其简单的理论基础和优越的预测性能, 逐渐取代了传统模型, 成为知识追踪领域新的研究热点. 根据所使用的神经网络结构, 阐述近年来代表性深度知识追踪模型的算法细节, 并在 5 个公开数据集上对这些模型的性能进行全面比较. 最后讨论了深度知识追踪的应用案例和若干未来研究方向.

关键词: 深度知识追踪; 深度学习; 循环神经网络; 记忆网络; 自注意力网络; 应用案例; 性能比较

中图法分类号: TP18

中文引用格式: 王宇, 朱梦霞, 杨尚辉, 陆雪松, 周傲英. 深度知识追踪模型综述和性能比较. 软件学报, 2023, 34(3): 1365–1395. http://www.jos.org.cn/1000-9825/6715.htm

英文引用格式: Wang Y, Zhu MX, Yang SH, Lu XS, Zhou AY. Review and Performance Comparison of Deep Knowledge Tracing Models. Ruan Jian Xue Bao/Journal of Software, 2023, 34(3): 1365−1395 (in Chinese). http://www.jos.org.cn/1000-9825/6715.htm

Review and Performance Comparison of Deep Knowledge Tracing Models

WANG Yu, ZHU Meng-Xia, YANG Shang-Hui, LU Xue-Song, ZHOU Ao-Ying

(School of Data Science and Engineering, East China Normal University, Shanghai 200062, China)

Abstract: Knowledge tracking is an important cognitive diagnosis method, which is often used in digitalized education platforms such as online learning platforms and intelligent tutoring systems. By analyzing students' interactions with course assignments, knowledge tracing models can simulate their mastery level of knowledge concepts in courses in real time. The simulation results can be used to predict students' future learning performance and help them plan personalized learning paths. In the past 20 years, knowledge tracing models have been constructed based on theories of statistics and cognitive science. With the openness and application of educational big data, models based on deep neural networks, referred to as “deep knowledge tracing models”, have gradually replaced traditional models due to their simple theoretical foundations and superior predictive performances, and become a new research hotspot in the field of knowledge tracing. According to the neural network architectures, the algorithm details of recent representative deep models for knowledge tracing are illustrated, and a comprehensive performance evaluation of the models on five publicly available datasets is conducted. Finally, some use cases and future research directions of deep knowledge tracing are discussed.

Key words: deep knowledge tracing; deep learning; recurrent neural network; memory network; self-attention network; use case; performance comparison

及时了解学生对知识的掌握程度, 是个性化教育中的关键一环. 一旦老师能够及时掌握学生的知识状态, 他们就可以凭借经验或者借助工具预测学生在未来的学习表现, 并且帮助学生规划个性化学习路径. 学生的知识状态往往在完成作业的过程中得到体现. 随着教育信息化的逐步普及, 在线教育平台如慕课(MOOC)、智

* 基金项目: 国家自然科学基金(61977026, 62072185)
收稿时间: 2021-07-18; 修改时间: 2022-03-02, 2022-05-07; 采用时间: 2022-05-27; jos 在线出版时间: 2022-07-22

能辅导系统(intelligent tutoring system, ITS)等, 已经在教学过程中得到了广泛应用, 并且累积了大量学生解题的数据. 而学生的知识状态正蕴含在这些海量的数据中.

知识追踪(knowledge tracing, KT)是根据学生过去的解题表现完成两个任务: 1) 评估学生对各个知识点的掌握水平(即获取学生的知识状态); 2) 预测学生在之后的解题表现. 知识追踪任务一般通过构建学生模型来完成. 过去 20 年中, 知识追踪模型可大致归纳为以下 3 种: 项目反应理论模型(item response theory, IRT)[1]、贝叶斯模型(Bayesian knowledge tracing, BKT)[2]和因子分析模型(knowledge tracing based on factors analysis)[3].

IRT 模型在 1950 年代被实验性地使用在教学场景中[1]. 它建立一个简单的函数关系, 根据学生的能力等级和题目的难度来预测学生正确解答当前题目的概率. IRT 模型基于以下简单的假设: 一个学生能力更高, 那么他更有可能正确解答题目; 另一方面, 如果一个题目难度更高, 那么它被正确解答的可能性就更低. 尽管 IRT 模型简单易用, 但是它并没有考虑到学生的能力(即知识状态)会随着学习的深入而发生改变. 20 世纪 90 年代提出的 BKT 模型[2]使用了隐马尔可夫模型, 通过观察学生能否正确答题来模拟他们的知识获取过程. 最早的 BKT 模型使用 4 个变量来模拟学习过程: 初始掌握概率 $P(L_0)$、从未掌握到掌握的转移概率 $P(T)$、未掌握知识点的情况下猜对题目答案的概率 $P(G)$、已掌握知识点的情况下做错题目的概率 $P(S)$. 尽管很好地模拟了学生解题过程中知识状态的改变, BKT 模型并没有考虑学生能力的差异以及包含相同知识点的不同题目的区别, 因此在实际教学过程中较难被应用. 2000 年以来, 一系列基于因子分析的逻辑回归模型被陆续提出, 代表模型包括学习因子分析(learning factors analysis, LFA)[3]、表现因子分析(performance factors analysis, PFA)[4]以及教学因子分析(instructional factors analysis, IFA)[5]. LFA 沿用了 IRT 理论中的假设, 并且考虑了学生解答某个知识点的题目所花费的次数, 因此在模型中融入了学生能力的差异. PFA 在 LFA 的基础之上增加了两个变量, 用来表示学生对某个知识点相关题目答对和答错的次数. IFA 则在 PFA 的基础之上, 考虑了教学过程中教师对于某个知识点的讲述次数, 因此综合考虑了学生学习和教师教授的因素. 关于传统知识追踪模型的详细结构, 读者可以阅读刘恒宇等人的综述工作[6]以获得全面了解.

上述概率模型虽然具备很好的解释性, 但需要基于理论假设, 人为地去构建模型的输入特征. 而特征构建往往具有片面性和局限性, 并不能充分挖掘数据中隐藏的信息, 因此模型的预测效果一般. 例如, 这些模型忽略了知识点之间的依赖关系, 也无法模拟学习的遗忘规律. 随着教育大数据的爆发和深度神经网络的复兴, 使用深度学习技术来解决知识追踪问题, 成为过去几年引人关注的一个研究热点. 相较于传统概率模型, 基于深度神经网络的模型可以直接从海量的解题数据中学习能够代表学生知识状态的特征, 从而避免了特征构建, 充分挖掘了数据本身蕴含的信息, 并且在预测学生未来的表现时取得了突破性的性能提升(预测能力是衡量知识追踪模型性能的重要指标). Liu 等人[7]整理归纳了知识追踪模型的演进过程, 但是并未对深度模型的性能进行量化比较. 本文在总结主要深度知识追踪模型的基础上遴选出最具代表性的 8 个模型和 5 个公开数据集, 通过实验全面比较模型性能的差异.

综上所述, 本文回顾了近年来深度知识追踪领域的一系列代表性工作. 我们根据模型所采用的骨干结构, 将这些模型分成 4 个大类, 并进行详细阐述. 然后, 针对其中 8 个最具代表性的模型进行实验并比较性能. 接着, 我们给出了几个深度知识追踪模型在实际教学中的应用案例. 最后, 我们对深度知识追踪的未来研究方向做出展望.

1 深度知识追踪

2015 年, Piech 等人[8]首次提出运用深度神经网络构建知识追踪模型, 即基于循环神经网络的 DKT (deep knowledge tracing)模型. DKT 将学生过去的解题表现转化为时间序列, 输入一个循环神经网络(recurrent neural network, RNN), 输出为该学生正确解答每一道题的概率, 然后通过反向传播预测概率与真实表现(即每一道题是否正确解答)差异的损失函数来训练模型参数. 其详细结构如图 1 所示.

图 1 中, $x_t=(q_t,r_t)$为输入时间序列中 t 时刻的输入; q_t 代表 t 时刻的习题编号, r_t 代表该习题是否被正确解答, 两个变量均使用独热编码. x_t 也称为响应元组. 当题目数量巨大时, DKT 可选择用降维将 x_t 嵌入成一个低维向

量 v_t, 缩小特征空间. 随后, DKT 使用 RNN 计算每个时刻的隐层向量 h_t, 用来表示学生在该时刻的知识状态. t 时刻的知识状态 h_t 由当前时刻的输入 x_t 和 t–1 时刻的知识状态 h_{t-1} 计算得到, 其公式为:

$$h_t=\phi(v_t,h_{t-1},\omega) \tag{1}$$

其中, $h_t\in\mathbb{R}^k$, ω为权重矩阵, ϕ是 RNN 的计算单元. 获得 t 时刻的知识状态表示 h_t 之后, RNN 可以通过非线性映射计算学生正确解答下一道习题(即 t+1 时刻的习题)q_{t+1} 的概率 y_{t+1}, 其公式为:

$$y_{t+1}=\delta(q_{t+1})\sigma(\omega\cdot h_t+b) \tag{2}$$

其中, $\sigma(\cdot)$是 Sigmoid 函数, $\omega\in\mathbb{R}^{Q\times k}$ 是权重矩阵, $b\in\mathbb{R}^Q$ 是偏置, $\delta(\cdot)$代表独热编码. 最后, DKT 通过计算 y_{t+1} 和 r_{t+1} 的差异建立损失函数, 累加并反向传播整个时间序列的损失来调整模型参数. 具体实现时, DKT 的计算单元可以使用 RNN、长短期记忆模型(long short-term memory, LSTM)或者门控循环单元(gate recurrent unit, GRU)来实现, 在不同的数据上, 可能会取得不同的表现.

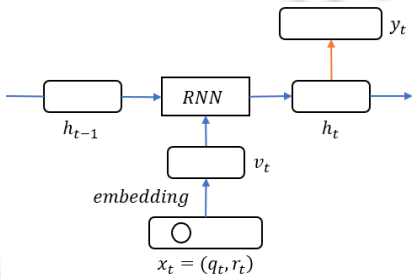


图 1 基于循环神经网络的 DKT 模型[8]

DKT 模型的提出, 是深度神经网络在知识追踪领域的首次尝试. 它可以对学生未来的解题表现做出较为准确的预测. 但 DKT 模型认为每道题仅包含一个知识点, 包含相同知识点的题目均相同, 因此模型无法捕捉涵盖相同知识点的不同题目的区别. 受到 DKT 模型的启发, 研究人员提出了一系列基于深度神经网络的模型来改进知识追踪的效果.

2 深度知识追踪模型的演化

深度知识追踪模型的演化历程和其所采用的神经网络结构息息相关, 其中有 3 种主流结构受到了研究人员的青睐: 首先, 鉴于学生解题的过程形成了时间序列数据, 在深度神经网络复兴之初, Piech 等人[8]提出了基于循环神经网络的知识追踪模型; 随后, 为了提高模型隐层状态捕捉时间序列中历史信息的能力, Zhang 等人[9]构建了基于记忆网络[10]的模型; 接着, 在自注意力机制[11]被应用于处理序列数据之后, Pandey 等人[12]提出了基于自注意力架构的模型, 以期更好地捕捉知识点之间的关联关系. 针对上述每种基础结构, 研究人员提出了各种改进方法, 提高模型的预测能力和在真实教学场景中的适用性. 这些改进方法也可以大致分为 3 类: 改进模型的基础结构、增加额外的解题特征以及模拟经典的教育理论. 除了上述 3 种主流结构以外, 一些工作也尝试使用其他神经网络构建模型, 如卷积神经网络和图神经网络. 还有部分工作研究如何针对特殊教学情形构建模型, 如构建针对不同教学机构学习数据的联邦学习和不同学科数据的迁移学习模型.

基于上述分析, 本节将现有工作分成 4 节进行叙述. 其中, 前 3 节分别介绍基于循环神经网络的模型、基于记忆网络的模型和基于自注意力机制的模型, 每一节的模型依据上述 3 种不同的改进方向进行分类和叙述; 最后一节则介绍有别于上述 3 种主流结构的工作, 由于这部分工作较少, 我们将不对它们进行细分.

2.1 基于循环神经网络的模型

2.1.1 改进模型的基础结构

为了提高 DKT 模型捕捉序列数据中长期依赖的能力, Sha 等人[13]利用双层堆叠 LSTM (stacked LSTM)模拟学生的解题序列, 如图 2 所示. Stacked LSTM 将两个 LSTM 网络纵向堆叠起来, 并使用残差网络将原始输入

与首层 LSTM 的输出合并以防止网络退化. 在输出端, 他们认为隐层的学生知识状态与预测正确答题概率的输出之间存在更加复杂的关系, 因此, 他们尝试使用均值法、最小值法、和增加全连接层等方法对隐层知识状态进行处理, 然后用于预测输出.

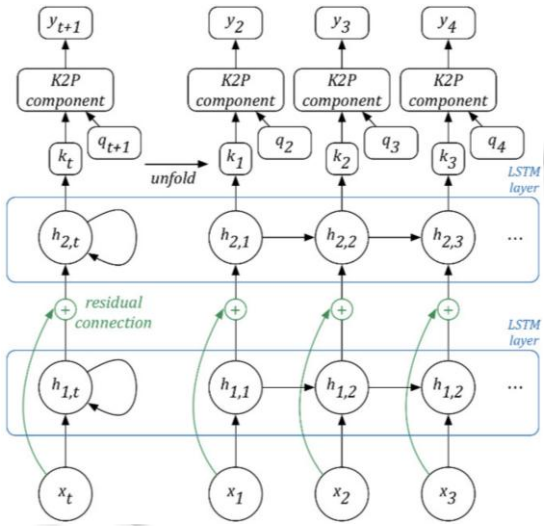
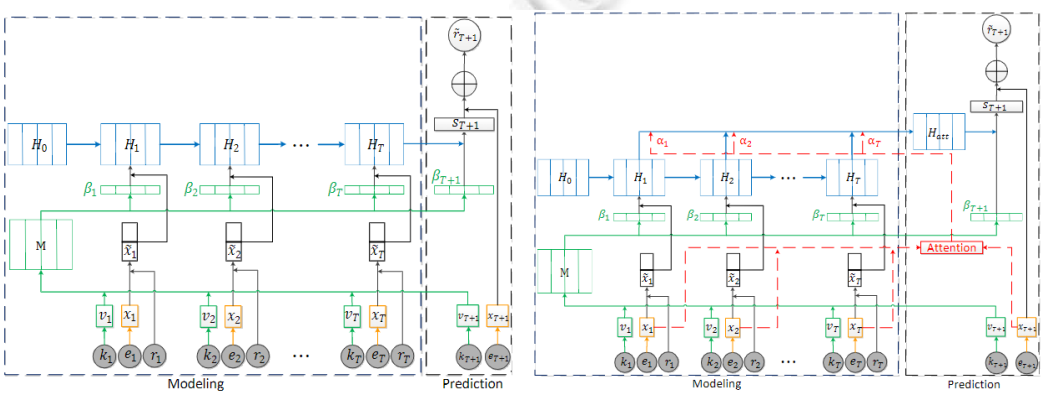


图 2 堆叠 LSTM 的主要结构[13]

另一方面, DKT 将学生的学习状态表示为一个隐藏向量, 其特征表达能力略显不足. 为了增加对单个知识点掌握状态的表示能力, Liu 等人[14]提出了 EKT (exercise-aware knowledge tracing)模型. 该模型引入一个隐层状态矩阵 H_T 以代替 RNN 的隐层状态向量 h_T, H_T 中的每个列向量即代表学生在 T 时刻对一个知识点的掌握程度. 为了更新 H_T, 模型需要知道 T 时刻习题对各个知识点的影响程度. 为此, 模型又引入一个静态查询矩阵 M, 它的每一个列向量代表一个知识点, 然后通过计算 M 中每个列向量与输入习题向量的余弦相似度, 得到输入习题对各个知识点的影响权重. 最后, EKT 将每个时刻的输入与每一个知识点的权重相乘, 通过 LSTM 计算单元更新相应的知识点掌握程度向量. 在输出端, 他们实现了马尔可夫模型和注意力机制两种结构, 模型结构如图 3 所示. 左图的输出端基于马尔可夫模型, 即 T+1 时刻隐层状态 H_{T+1} 仅依赖于前一时刻的隐层状态 H_T; 右图使用了注意力机制, 利用已经解答的习题嵌入与当前正在解答的习题嵌入的相似性作为权重, 将过去的隐层状态加权求和, 得到 T 时刻的注意力隐层状态 H_{att}, 作为预测 T+1 时刻解答正确概率的隐层输入.



(a) EKTM with Markov property (b) EKTA with Attention mechanism

图 3 EKT 模型的两种详细结构[14]

此外, DKT 模型在输出端没有显示地模拟学生知识状态和下一道习题知识点的关联关系, 因此预测能力

受到一定的限制. 鉴于此, Lee 等人[15]提出了知识查询网络 KQN (knowledge query network), 用来直观地描述知识状态和习题的交互关系, 如图 4 所示. Lee 认为, 若将学生当前的知识状态 v_t 和概念 k_i 看作二维向量, 则两者的内积可以直观地被看作该学生在对于当前概念 k_i 的理解程度. 因此, KQN 把当前知识状态的隐层向量与下一习题所对应的知识点的向量做点积, 将得到的结果作为一个逻辑函数的输入, 预测学生正确解答下一习题的概率.

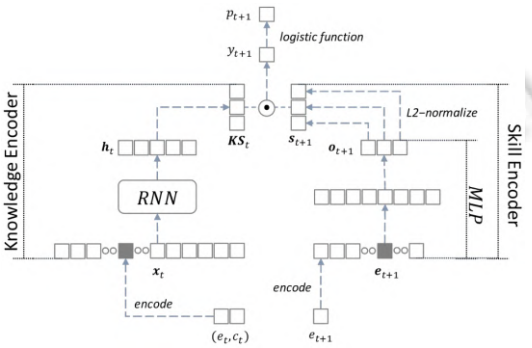


图 4 KQN 模型的结构[15]

2.1.2 增加额外的解题特征

DKT 模型仅仅采用了习题知识点和学生解答作为输入特征, 对于学生解题过程的特征利用不够充分. 事实上, 学生在解题过程中还会形成在同一道习题尝试的次数、解答当前习题所花费的时间、当前习题的内容信息等多种异质特征. 因此, 研究人员提出了各种方法在模型中引入这些特征.

Yang 等人[16]提出了基于决策树的 DKT 模型(tree-based DKT), 采用决策树对原始数据进行预训练, 自动选取各种异质特征输入到 DKT 模型. 他们采用了随机森林和梯度提升决策树训练分类器, 预测学生能否正确解题. 然后将决策树预测的结果向量和 DKT 的缺省输入向量拼接, 输入到 RNN 结构中.

Zhang 等人[17]提出使用自编码器来学习异质特征, 使得模型输入能够结合更丰富的特征. 在输入端, 多个特征进行编码和拼接, 再经过自编码器降维, 得到的特征向量输入至模型. 这样, 隐层状态将包含更加丰富的学生答题历史信息.

Su 等人[18]提出了结合习题内容的 EERNN 模型(exercise-enhanced recurrent neural network). 在 DKT 的基础之上, EERNN 将习题文本内容作为额外的特征输入到 RNN 中. 其详细结构如图 5 所示. 在模型的输入端, EERNN 首先将习题 q_T 中的单词(或中文的词组)嵌入成低维向量, 然后使用双向 LSTM 获得前向和后向两个隐层表示, 并拼接成为每个单词的隐层向量 v_t. 接着, 利用基于习题中 M 个单词的最大池化运算, 得到 q_T 的全局嵌入 $x_T=\max(v_1,v_2,\ldots,v_M)$. 最后根据学生的解题表现 r_T, 用全零向量扩展 x_T, 得到最终的输入向量 $\tilde{x}_T$. 其公式为:

$$\tilde{x}_T=\begin{cases}[x_T \oplus a_T], \text{ if } r_T=1\\ [a_T \oplus x_T], \text{ if } r_T=0\end{cases} \tag{3}$$

其中, a_T 为维度和 x_T 相同的全零向量; $\oplus$为向量拼接操作; r_T 为 1 代表解答正确, 为 0 代表解答错误. 随后, 根据输入的习题序列, 模型使用 LSTM 或者 GRU 来学习和更新隐层状态 h_T. 在输出端, EERNN 采用了和 EKT 相似的两种方式获取用于预测的隐层状态, 即基于马尔科夫性质和注意力机制的方法.

Wang 等人[19]提出了 DKTS 模型(DKT with side information), 将题目之间基于知识点相似性的相关关系作为辅助信息. 他们将这种相关关系构建成图, 如果两个习题之间的知识点相似, 则用一条边将它们连接起来, 然后利用图嵌入算法, 可以生成包含题目相关关系的习题特征向量, 输入到 DKT 模型. 在输出端, 利用相关关系图可以计算一个正则项加入损失函数中, 更好地控制正确解答具有相关关系的题目的预测概率.

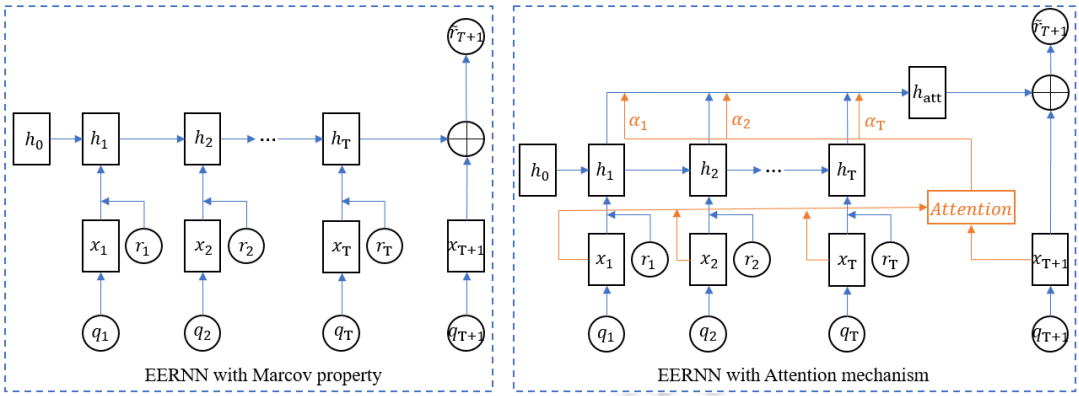


图 5 EERNN 模型的两种详细结构[18]

Sonkar 等人[20]提出了 qDKT 模型(question-level deep knowledge tracing), 基本沿袭了 DKT 的架构. 不同的是: qDKT 的输入粒度精确到题目, 而不是知识点, 并且评估了学生对了每一道题的掌握情况.

Liu 等人[21]认为, 在模型中加入习题特征和学生行为特征有助于提高预测学生答题对错的准确率, 并提出把 RNN 网络和注意力机制相结合. 他们把所有可用特征合并及降维后输入到模型中, 经过 RNN 结构, 输出学生的知识状态, 再通过注意力机制, 从输入特征中提取关键信息, 对过去的知识状态赋予不同权重并加权求和, 得到经多特征以及注意力机制加强后的知识状态, 最后预测学生能否正确解题.

Tong 等人[22]将题目文本信息加入 RNN 模型的输入中. 他们用 BERT (bidirectional encoder representations from transformers)生成题目文本的特征向量, 并使用 TextCNN (text convolutional neural networks)模型从中获取题目的知识分布特征以及难度特征; 计算题目文本向量的相似性, 使用层次聚类方法提取题目的语义特征. 然后将所提取的特征以及学生响应输入到 RNN 结构中, 预测学生答题结果.

Zhang 等人[23]提出了相似的模型架构, 但在输入特征上有所不同. 他们使用异构信息网络从所有学生的答题记录中获取题目难度以及区分度, 得到题目的嵌入表示, 再输入到 RNN 与注意力相结合的模型中.

Yang 等人[24]指出, 由于受到数据稀疏性以及多知识点题目的限制, 现有的知识追踪模型没有使用题目和知识点之间的关联. 因此, 他们提出了 GIKT (graph-based interaction model for knowledge tracing)模型.在 LSTM 的基础上, 用基于图卷积网络(graph convolutional networks, GCN)的方法, 融合题目与知识点之间的关联信息.

2.1.3 模拟经典的教育理论

另一个针对基础模型的改进方法, 是在模型中融入能够模拟教育理论的结构或者特征, 使所得到的模型更加适用于真实的教学场景.

Chen 等人[25]认为, 知识点之间的先决条件关系, 也应该作为一个重要的特征加入 DKT 模型中. 假设知识点 1 为知识点 2 的先决条件, 那么掌握知识点 1, 才更有正确解答关于知识点 2 的习题. 因此在模型的输入端, 他们增加了一个先决条件关系矩阵, 将知识点之间的先决条件作为一种约束条件添加到模型输入中.

Minn 等人[26]认为, 学生个人能力的差异对知识状态的改变有巨大影响, 因此在模型中, 应该将学生学习能力作为一个重要因素加以考虑. 为此, 他们提出了一种基于动态学生分类的深度知识追踪模型 DKT-DSC (deep knowledge tracing and dynamic student classification). DKT-DSC 根据学生过去的解题表现, 使用动态聚类将学生每隔一段时间划入不同的类别, 每个类别中的学生被认为具有相似的学习能力. 然后, 模型将聚类结果(即学生能力)作为额外的输入特征与 DKT 相结合. 其详细结构如图 6 所示, 左边部分为动态聚类的示意图. 每隔一段时间, DKT-DSC 会根据统计学生过去对于每个知识点的答题表现, 作为对学生学习能力的评估. 评估的方法如公式(4)–(7)所示.

$$Correct(k_j)_{1:z}=\sum_{t=1}^{z}\frac{(k_{jt}==1)}{|N_{jt}|} \tag{4}$$

$$Incorrect(k_j)_{1:z}=\sum_{t=1}^{z}\frac{(k_{jt}==0)}{|N_{jt}|} \tag{5}$$

$$R(k_j)_{1:z}=Correct(k_j)_{1:z}-Incorrect(k_j)_{1:z} \tag{6}$$

$$d^i_{1:z}=(R(k_1)_{1:z},R(k_2)_{1:z},...,R(k_n)_{1:z}) \tag{7}$$

其中, $Correct(k_j)_{1:z}$ 和 $Incorrect(k_j)_{1:z}$ 表示学生在过去 z 个时间段中答对或答错知识点 k_j 的比率; $|N_{jt}|$是到 t 时间段为止, 知识点 k_j 出现过的总次数. $R(k_j)_{1:z}$ 用答对和答错知识点 k_j 的比率差值表示学生对该知识点的学习能力. 最后, 向量 $d^i_{1:z}$ 代表学生在过去对于所有知识点的学习能力. 利用代表学生学习能力的向量, DKT-DSC 在每个时间段用 K-means 对学生进行聚类, 得到学生的学习能力类别向量 c_t, 然后与原始 DKT 的输入 x_t 进行拼接, 作为 DKT-DSC 模型的输入, 其结构示意图如图 6 右半部分所示.

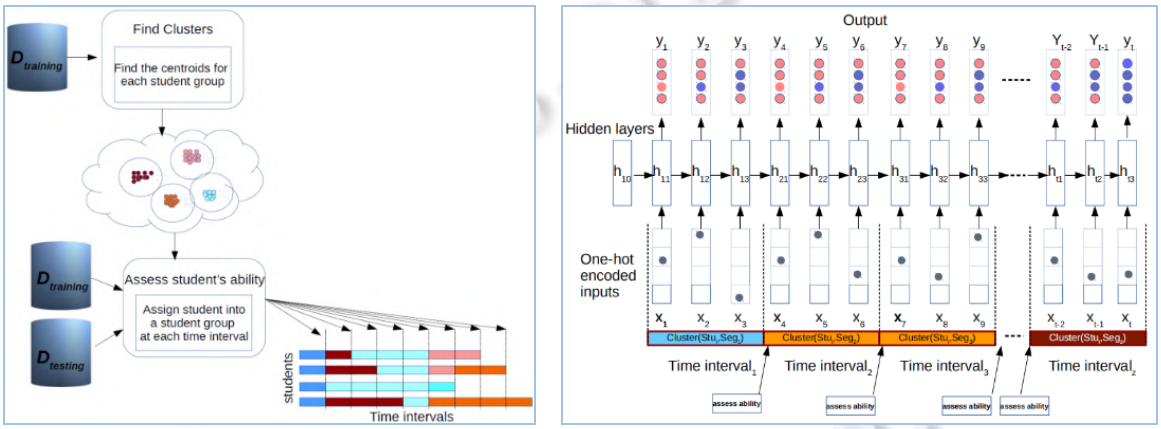


图 6 在每个时间步对学生进行聚类[26]

Nagatani 等人[27]认为应该模拟学生在解题过程中的遗忘规律. 学生的遗忘行为主要受两个因素的影响: 1) 和上一次解答相同知识点题目间隔的时间; 2) 过去针对同一知识点的解题次数. 这一假设基于 19 世纪一项关于记忆力的实验[28], 结果表明, 对特定事物的记忆随着时间推移呈指数下降, 而反复提起特定事物能够帮助避免遗忘. 因此, Nagatani 等人考虑了 3 个影响遗忘行为的特征: 重复时间间隔、序列时间间隔以及过去解题次数, 提出了模型 DKT-F (deep knowledge tracing with forgetting). 重复时间间隔指在同一个知识点上, 前后两次解题的时间差. 序列时间间隔指学生在解题过程中, 每两题之间的时间差. 过去解题次数指学生过去在同一个知识点上的答题次数. 模型的详细结构如图 7 所示.

其基本结构复用了 DKT 中的 RNN, 区别是在输入端, 上述 3 个特征的(独热)向量拼接成特征向量 c_t, 并与响应元组 x_t 的嵌入表示 v_t 融合获得最终的 t 时刻输入向量 v^c_t. 其融合公式为:

$$v^c_t=[v_t\odot Cc_t;c_t] \tag{8}$$

其中, C 是可训练的转换矩阵, $\odot$表示逐元素相乘. v^c_t 输入 RNN 计算单元后, 获得的隐层向量 h_t 与下一时刻的特征向量 c_{t+1} 进行整合, 再经过一层全连接层和 Sigmoid 激活层, 预测学生正确回答下一题的概率. 实验结果表明, 增加了遗忘行为的 DKT-F 模型, 比 DKT 模型具有更好的预测性能.

类似地, Huang 等人[29]提出了基于学生学习和遗忘规律的模型 KPT (knowledge proficiency tracing). 他们认为, 学生在练习过程中, 时间是一个非常重要的考量指标, 很久不练习的学生会对知识点产生遗忘, 反之会不断提升解题的能力. 与此同时, 文中提到, 学生在做题过程中对一道已经做错并且掌握的题目, 很少会反复尝试, 这就导致学生数据比较稀疏. 因此, 他们又通过专家人工标注的方式引入了 Q 矩阵, 代表每个习题与知

识点之间是否相关. 训练时, 模型不仅预测学生在 T+1 时对待测习题 q 的结果, 还要预测在该时刻对每个知识点的熟练程度.

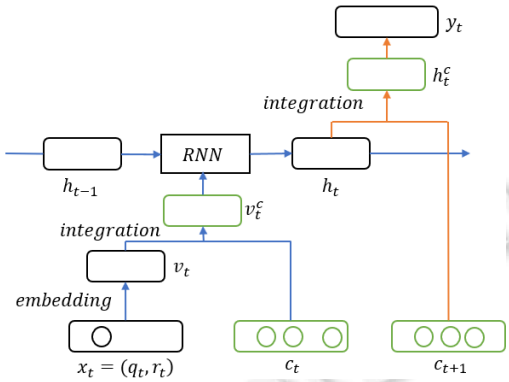


图 7 考虑遗忘行为的模型[27]

Long 等人[30]认为, 传统的 DKT 模型并没有考虑到每个学生对知识的理解以及对题目的感知能力的不同, 因此他们提出了 IEKT (individual estimation knowledge tracing)模型. IEKT 在原始的 DKT 模型上新增了认知能力评估(CE)和知识敏感度评估(KASE)两个模块. 在模型做出预测之前, 可以通过 CE 获取学生对题目的认知能力, 使得预测结果更加具有针对性. 同时, 在更新学生的知识状态时, 模型也应该通过 KASE 模块获取学生的实际接收能力, 更加个性化地更新学生的知识水平. IEKT 提出了两个矩阵 M, S, 分别代表不同等级的认知能力以及知识理解能力. 在模型预测时, Long 将当前时刻的题目表征 v_t 和上一时刻 LSTM 的隐向量 h_{t-1} 输入到 CE 模块中, 得到学生目前的状态表征:

$$h_v=v_t\oplus h_{t-1} \tag{9}$$

然后, 根据 h_v 从认知矩阵 M 中采样出符合条件的等级表征 m_t, 并与 h_v 一起做出最终的预测:

$$y=ReLU(W_1\cdot[m_t\oplus h_v]+b) \tag{10}$$

$$\tilde{r}_t=\sigma(W_2\cdot y+b_1) \tag{11}$$

与此同时, 在更新 LSTM 之前, LEKT 通过 KASE 模块, 结合学生的实际能力做出恰当的调整. 考虑到学生对于当前的题目理解可能源于正确答案的启发或者自己的解答, KASE 模块构造了如下两个输入:

$$v_p=\begin{cases}h_v\oplus\mathbf{0}, & \tilde{r}_t\geqslant 0.5\\ \mathbf{0}\oplus h_v, & \tilde{r}_t<0.5\end{cases} \tag{12}$$

$$v_g=\begin{cases}h_v\oplus\mathbf{0}, & r_t=1\\ \mathbf{0}\oplus h_v, & r_t=0\end{cases} \tag{13}$$

其中, $\mathbf{0}$ 表示维度与 h_v 相同的全零向量. v_p, v_g 分别对应独立完成以及参考答案的情况, 最终的输入 $v_m=v_p\oplus v_g$. 然后使用同 CE 相似的方式, 根据 v_m 从理解能力矩阵 S 中采样出当前该学生的理解能力表征向量 s_t, 与原始的输入 v_t 结合, 对 LSTM 做出更新. IEKT 将学生的认知和理解能力分为不同的等级, 并且在预测和更新知识状态之前都提前模拟的学生当前的能力水平, 这使得模型更加个性化且具有可解释性.

基于循环神经网络的知识追踪模型结构较为简单, 训练开销较低, 能够较为准确地预测学生在未来的答题表现. 但受限于简单的一维隐层表示, 模型无法非常充分地反映学生对于各个知识点的掌握情况. 因此, 学者们提出了一系列基于动态记忆网络的模型来解决这个问题.

2.2 基于记忆网络的模型

记忆网络[10]是受到了计算机体系结构中外存的启发, 用来提高机器学习模型从时间序列中捕获长期依赖的能力. 类似于循环神经网络, 记忆网络也采用循环结构, 区别在于: LSTM 将历史信息存储在单个隐层向量中, 而记忆网络将历史信息存储在一个记忆矩阵中, 因此它能够捕获更大容量的历史信息. 记忆网络根据外

部输入, 通过读写记忆矩阵来更新矩阵存储的内容. Zhang 等人[10]构建了将传统记忆网络应用于知识追踪任务的模型 MANN (memory-augmented neural network), 其详细结构如图 8 所示.

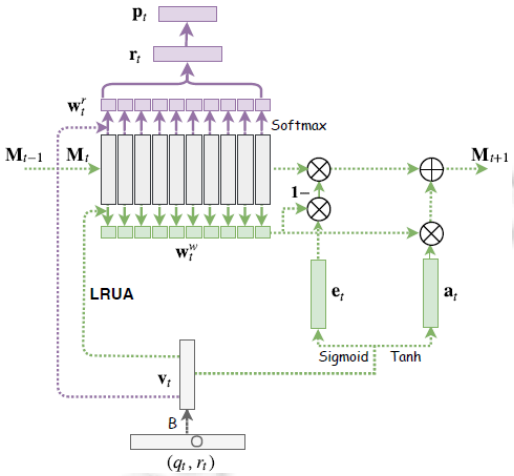


图 8 用于知识追踪的 MANN 模型结构[10]

模型中, $\boldsymbol{M}_t$ 为记忆矩阵, 代替循环神经网络中的隐层向量 h_t 存储 t 时刻学生的知识状态. t 时刻的输入 $x_t=(q_t,r_t)$被嵌入成向量 $\boldsymbol{v}_t$, 然后与 $\boldsymbol{M}_t$ 中的每个向量一起计算矩阵的读权重向量 $\boldsymbol{w}_t^r$ (通常用内积或余弦相似度)和写权重向量 $\boldsymbol{w}_t^w$ (通常使用最近最少使用法则), 用来读取和更新矩阵的内容. 其隐含的意义是: 若学生对于相同的题目给出了相同的答案, 则更新之前使用过的记忆矩阵向量; 反之, 若学生做了一道新的题目或者对于做过的题目给出了不同的答案, 则更新最近最少使用的记忆矩阵向量. 在输出端, 模型使用 $\boldsymbol{w}_t^r$ 对记忆矩阵加权求和, 得到的向量 $\boldsymbol{r}_t$ 用来计算学生正确解答 t 时刻习题的概率. 从 $\boldsymbol{M}_t$ 更新到 $\boldsymbol{M}_{t+1}$ 分为清除记忆和增加记忆两个步骤. 模型先用 $\boldsymbol{w}_t^w$ 和清除向量 $\boldsymbol{e}_t$ 相乘, 得到 $\boldsymbol{M}_t$ 中每个向量应该清除的比例, 从而清除记忆矩阵中过时的信息; 然后将 $\boldsymbol{w}_t^w$ 和增加向量 $\boldsymbol{a}_t$ 相乘, 得到 $\boldsymbol{M}_t$ 中每个向量应该增加的记忆. 其背后的含义是: 根据每个时刻题目的知识点和学生答题表现, 学生可能对某些知识点的掌握变差, 同时也有可能对某些知识点的掌握变好.

2.2.1 改进模型的基础结构

MANN 利用单个静态记忆矩阵存储信息, 读写操作均在同一个存储矩阵内进行. Zhang 等人[9]认为, 模型输入和预测并不是同一类型的数据, 同一存储矩阵不能同时表示习题信息与学生知识水平. 因此, 如果采用单个存储矩阵并不能很好地解决知识追踪任务. 他们提出了动态键值存储网络 DKVMN (dynamic key-value memory network)模型, 包含一个静态的键矩阵与动态的值矩阵. 键矩阵用于存储所有知识点的关联信息, 值矩阵用于存储学生对于不同知识点的掌握程度, 两者均在模型的训练中不断更新. 模型详细结构如图 9 所示.

模型随机初始化两个 N(N 为可调参数)列的矩阵, 键矩阵 $\boldsymbol{M}^k$ 与值矩阵 $\boldsymbol{M}^v$. 读取部分, t 时刻的习题编号 q_t 被嵌入成低维向量 $\boldsymbol{k}_t$, 通过计算 $\boldsymbol{k}_t$ 与 $\boldsymbol{M}^k$ 中每一个列向量的余弦相似度, 得到写权重向量 $\boldsymbol{w}_t$, 用于更新表示学生知识点掌握情况的值矩阵. 在预测部分, DKVMN 舍弃了读向量 $\boldsymbol{w}^r$, 将写向量 $\boldsymbol{w}_t$ 与值矩阵 $\boldsymbol{M}^v$ 加权求和得到对学生知识状态的评估向量 $\boldsymbol{r}_t$:

$$\boldsymbol{r}_t=\sum_{i=1}^{N} w_t(i)\boldsymbol{M}_t^v(i) \tag{14}$$

考虑到习题内容的不同, DKVMN 把读取到的知识状态评估 $\boldsymbol{r}_t$ 和输入习题的嵌入向量 $\boldsymbol{k}_t$ 进行拼接, 经过一层全连接网络, 预测该学生在下一时刻的正确答题概率 p_t. 矩阵的更新, DKVMN 采用了与 MANN 一样的方法. DKVMN 使用了两个记忆矩阵分别存储知识点信息与学生的知识状态信息, 解决了 MANN 中单个矩阵无法同时表达习题信息与学生知识掌握水平的问题.

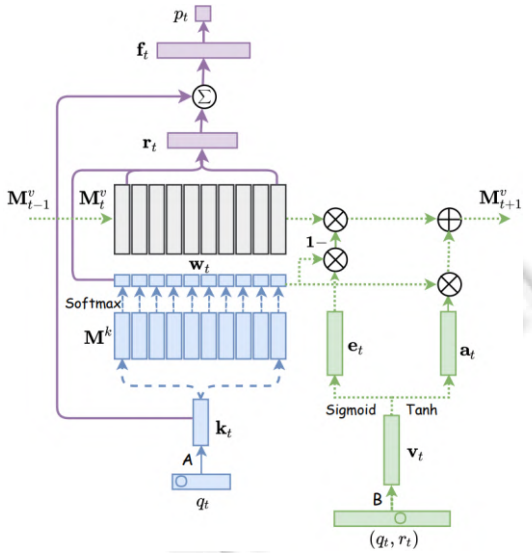


图 9 DKVMN 模型架构[9]

Abdelrahman 等人[31]认为，对当前题目解题结果的预测与历史做题记录中不包含相关知识点的题目没有关联，因此提出了顺序健值存储网络模型 SKVMN (sequential key-value memory network). 该模型在 DKVMN 的模型基础上加以改动，仅参考历史做题记录中与当前题目知识点相关的题目，来预测学生在当前题目上的解题表现.

如图 10 所示, SKVMN 在读取部分与原始模型一致，然而模型全连接部分并没有直接输出预测结果，而是将当前时刻的隐层向量 f_t 作为时序序列输入改造的 LSTM 网络中(hop-LSTM).

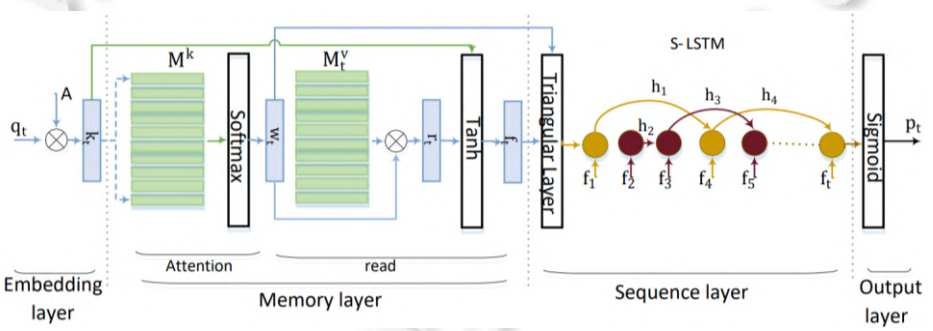


图 10 SKVMN 模型架构[31]

与传统 LSTM 不同，Hop-LSTM 序列层各单元按照依赖关系进行连接，而依赖关系由写权重向量 $\boldsymbol{w}_t$ 来决定. 由于 $\boldsymbol{w}_t$ 表示题目与不同知识点之间的相关性，因此模型通过 $\boldsymbol{w}_t$ 判断不同题目之间知识分布的相似性. 这一步通过三角隶属函数(triangular membership function)[31]来完成，计算方法如公式(15)所示.

$$\mu(x)=\max\left(\min\left(\frac{x-a}{b-a},\frac{c-x}{c-b}\right),0\right) \tag{15}$$

其中, x 是权重向量中每一个位置上的值, a, b, c 是超参数. 通过该函数，可以将权重向量中每一个位置的值分别分为 3 个等级，用 0(低), 1(中), 2(高)来表示知识点与题目相关性的强度，获得的新的特征向量，即为输入的题目与各个知识点之间相关性的表示. 如果当前 t 时刻的题目的特征向量与 t–λ时刻的题目特征向量相同，且 t–λ时刻与 t 时刻没有其他与它们特征向量相同的题，则认为这两道题目涉及的知识点是相同的，两者在序列上存在依赖关系，用 $q_{t-\lambda}$←q_t 表示.

模型只将相关题目的 LSTM 单元循环连接, 反向传播时只更新序列相关的 LSTM 单元的参数, 不相关的 LSTM 单元参数则不更新, 其余参数如嵌入矩阵、权重矩阵和偏置向量等, 在每一轮反向传播中均迭代更新. 模型最后的预测结果通过 Sigmoid 函数激活. 在更新表示学生知识水平状态的值矩阵过程中, DKVMN 没有考虑当前学生的知识水平信息, 因此在 SKVMN 中, 将当前的知识状态用读取过程的总结向量来表示, 和当前学生的实际答题结果一起作为输入来更新值矩阵, 如图 11 所示. SKVMN 模型把 DKVMN 模型与 LSTM 模型相结合, 综合了循环建模能力和存储能力的优势. 通过对 LSTM 加以修改, 提出了 Hop-LSTM, 每次对相关性高的序列进行建模, 减少序列长度, 提高预测效率. 并且在更新值矩阵过程中加入当前的知识状态信息, 以平滑对知识点掌握程度的预测.

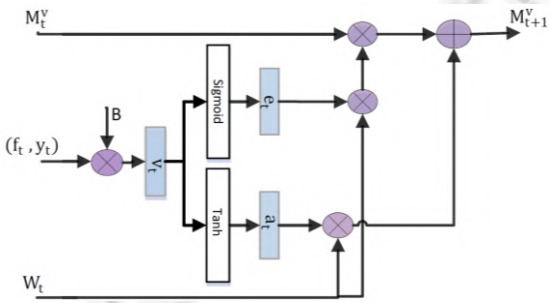


图 11 值矩阵写入过程[31]

2.2.2 增加额外的解题特征

在以往的模型中, 学生对提示的使用信息往往没有被使用到或者使用提示的被判定为解答错误. Chaudhry 等人[32]认为, 提示信息的使用能够很好地反映学生的能力, 即提示使用越多的人可能能力越差; 相反, 能力强的学生往往很少使用提示. 基于此, 他们在 DKVMN 的基础上进行了改进, 引入学生对提示的使用作为特征, 将单任务模型变为多任务协同预测. 模型结构如图 12 所示.

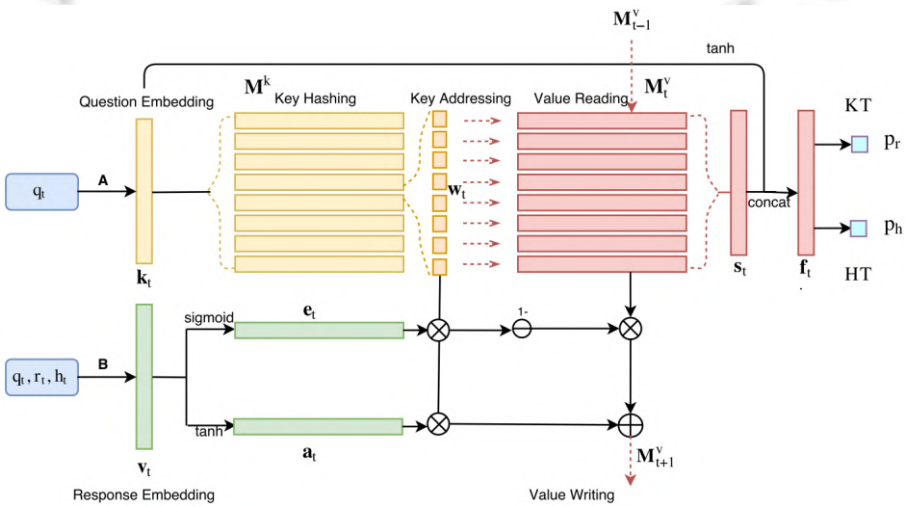


图 12 加入提示请求的多任务 DKVMN 模型[32]

他们保留了 DKVMN 的网络结构, 在更新值矩阵部分增加了一维向量 h_t, 用于表示是否使用提示. 模型输出两个预测结果: 正确率 p_r 和使用提示的概率 p_h. 两项任务参数共享, 协同训练, 提升了预测的准确率. 在学习过程中, 如果学习者陷入某个特定的难题, 那么许多学习平台都会以提示的形式提供学习帮助. 因此, 预测何时需要提供提示选项对于提示的合理使用至关重要. 何时进行提示和对学习者知识状态的评估有密切关系, 多任务 DKVMN 将提示预测任务与知识追踪任务一起建模, 一定程度上改善了模型效果.

Sun 等人[33]认为, 更多的输入特征能够提升模型预测的准确性. 因此, 他们引入回归树(classification and regression tree, CART)来对数据集中所有特征进行筛选, 并将最终的分类结果与习题编号 q_t 拼接作为输入向量, 再使用DKVMN进行预测. CART 决策树对特征的预抽取能够更好地表示学生做题习惯, 提升模型准确率.

受到 DKT-DSC 的启发, Minn 等人[34]将该想法迁移到 DKVMN 中, 提出了 DKVMN-DSC (DKVMN and dynamic student classification)模型. 在预测的同时, 按照做题表现和题目难度对学生的能力进行分类, 将学生能力作为额外特征输入, 使得最终预测精度有所提升.

为了使用更多的外部信息(如提示使用情况、题目尝试次数等等), Sun 等人[35]在上述工作的基础上又增加了 XgBoost (extreme gradient boosting)模块, 用来筛选模型的输入. 他们将学生的做题情况、提示使用情况以及尝试次数等等信息输入到 XgBoost 中, 通过预测能否作对该题来对大量的输入特征做筛选和提取, 然后使用类似于 DKVMN-DSC 的想法, 通过对不同时间段学生的学习能力进行分类来提升模型的准确率.

2.2.3 模拟经典的教育理论

Ha 等人[36]认为, 传统的 DKVMN 模型写入过程仅考虑了当前时刻响应, 并没有考虑学生之前的知识状态, 并且高估了在值矩阵更新过程中产生的遗忘. 他们引入了自适应知识增长和负影响惩罚项, 分别改善了原始 DKVMN 模型在学习和遗忘过程中存在的问题, 更好地模拟真实的学习和遗忘的过程.

Yeung 等人[37]认为, 原始的 IRT 考虑了题目难度和学生的能力水平, 模型更具有可解释性. 因此, 他们在 DKVMN 的基础上进行了改进, 在预测部分并没有直接预测答题正确概率, 而是使用两个仿射变换分别表示学生的能力和题目难度, 再使用 IRT 的方法对答题结果进行预测.

Gan 等人[38]同样使用 DKVMN 模型与 IRT 模型的结合. 模型考虑截至当前时刻的知识状态以及前 H 题的学习历史对当前做题表现的影响, 得到当前总体知识状态; 结合当前知识点特征以及其余相关知识点特征得到当前习题的总体嵌入; 通过学生学习历史得到当前习题的难度向量. 将以上 3 个向量分别进行放射变换, 得到 IRT 的学生能力、习题区分度、习题难度这 3 个参数, 以对学生答题结果进行预测.

为了显式地学习不同知识概念之间的关系, Ghodai 等人[39]提出了 DGMN (deep graph memory networks)模型. Ghodai 认为, 我们可以通过 DKVMN 中的知识矩阵 M^k 和能力矩阵 M_t^v 构建一个潜在的概念图(latent concept graph, LCG), $G=(V,E,\omega)$. 其中, V 表示知识点的数量, 对应知识点矩阵 M^k 的行; ω 表示自定义的相似度函数(如余弦相似度). 他们认为, 可以通过学生当前对不同知识点的掌握程度来反推它们之间的联系. 因此, 他们将不同知识点之间的边做如下定义:

若 $\omega(M_t^v(i),M_t^v(j))\geqslant\mu$ (μ为常数), 则知识点 c_i 和 c_j 之间存在边; 否则不存在. 其中, $M_t^v(i)$, $M_t^v(j)$ 分别对应矩阵 M_t^v 中的第 i 列和第 j 列, 代表该学生对第 i 和 j 个知识点的掌握情况.

同时, Ghodai 使用 GCN 算法得到节点的表征 $H(i)$, $i\in\{1,2,\ldots,N\}$. 最后对节点进行加权求和, 得到 LCG 的最终输出向量 z_t, 计算方式如下:

$$z_t=\mathrm{Tanh}\left(\boldsymbol{W}\left(\sum_1^N w_t(i)H(i)\right)^T+b\right) \tag{16}$$

其中, $\boldsymbol{W}\in\mathbb{R}^{N\times d_k}$ 为参数, $w_t(i)$, $i\in\{1,2,\ldots,N\}$为 DKVMN 中的写权重向量.

最后, 将 DKVMN 得到的评估向量 r_t 与 z_t 拼接, 进行最后的预测:

$$p_t=\delta(q_t)\sigma(W[z_t,r_t]+b) \tag{17}$$

2.3 基于自注意力机制的模型

2.3.1 改进模型的基础结构

随着机器翻译的发展, 以 Transformer 为基础的自注意力模型得到了广泛应用. DKT 和 DKVMN 都将隐藏单元视为学生的知识状态, 但在历史数据稀疏、学生只与少量知识点交互的情况下, 上述方法的性能并不理想. 为了解决这些问题, Pandey 等人[12]提出了一个基于自注意力的知识追踪框架 SAKT (self-attentive model for knowledge tracing). 它能从学生过去与习题的交互中识别特定知识的状态, 并根据这些知识状态进行预测.

其模型结构如图 13 所示.

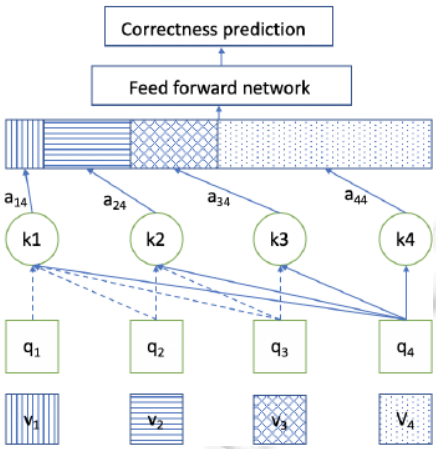


图 13 自注意力知识追踪模型[12]

首先, 在嵌入层将历史交互行为 $X=(x_1,x_2,\ldots,x_t)$ $(x_i=(q_i,r_i))$编码成 $S=(s_1,s_2,\ldots,s_t)$; 然后, 通过训练得到嵌入向量矩阵 $M\in\mathbb{R}^{2E\times d}$, 其中, E 是总习题数量. 类似地, 他们还将习题序列 $Q=(q_1,q_2,\ldots,q_t)$编码为矩阵 $\boldsymbol{E}\in\mathbb{R}^{E\times d}$. 此外, 为了获得位置信息, 他们使用与 Transformer 相同的位置编码方式: $P\in\mathbb{R}^{n\times d}$ (其中, d 是嵌入维数), 并将其与 M 相结合. 输入矩阵变换为:

$$\hat{M}=\begin{bmatrix}M_{s1}+P_1\\M_{s2}+P_1\\\cdots\\M_{s_n}+P_n\end{bmatrix},\ \hat{E}=\begin{bmatrix}E_{s1}\\E_{s2}\\\cdots\\E_{s_n}\end{bmatrix} \tag{18}$$

该模型将 $\hat{M}$ 和 $\hat{E}$ 分别输入到自注意层, 作为编码器和解码器的输入. 再利用多头缩放点积自注意力机制(multi-head scaled dot product self-attention), 学习当前习题与历史交互行为的关系:

$$Q=\hat{E}W^Q,\ K=\hat{M}W^K,\ V=\hat{M}W^V \tag{19}$$

$$Attention(Q,K,V)=softmax\left(\frac{QK^T}{\sqrt{d}}\right)V \tag{20}$$

$$Multihead(\hat{M},\hat{E})=Concat(head_1,head_2,...,head_h)W^O \tag{21}$$

其中, $W^Q,W^K,Q^V\in\mathbb{R}^{d\times d}$ 分别是查询、键和值投影矩阵. 为了共同处理来自不同表征子空间的信息, 他们使用不同的线形变换将 Q, K, V 投射到 h 个维度相同的子空间. 为了将非线性特征纳入模型并考虑不同潜在维度之间的相互作用, 他们使用了一个前馈网络(FFN), 最后, 通过线性投影预测学生的表现, 输出学生正确解答习题的概率.

Choi 等人[40]提出了 SAINT (separated self-attentive neural knowledge tracing)模型, 引入了 Transformer 的编码器-解码器结构, 如图 14 所示.

编码器的输入为学生到当前时刻的做题记录 $Q=\{E_1,E_2,\ldots,E_t\}$, 解码器的输入包含序列的开始标识以及学生到前一时刻为止的响应记录 $A=\{R_1,R_2,\ldots,R_t\}$, 两者均用注意力机制, 捕捉习题和学生响应中的隐含信息. 用编码器的输出作为键和值, 用过去的响应经过注意力机制的输出作为查询, 再经过一次注意力网络, 预测学生答题对错. 所有注意力网络中都添加了 mask 层, 用于屏蔽未来的信息, 避免对当前时刻的预测结果产生影响, 且每个注意力网络共享相同的位置编码.

SAINT 编(解)码器的内部实现与 SAKT 基本相同, 但是 SAINT 解码器使用了 2 种注意力机制(自注意力和上下文注意力), 最终, 解码器的输出经过一个 Sigmoid 激活的线性层得到最终的预测结果. SAINT 基本迁移

了 Transformer 的架构, 多次叠加的注意力使得模型有更好的表现.

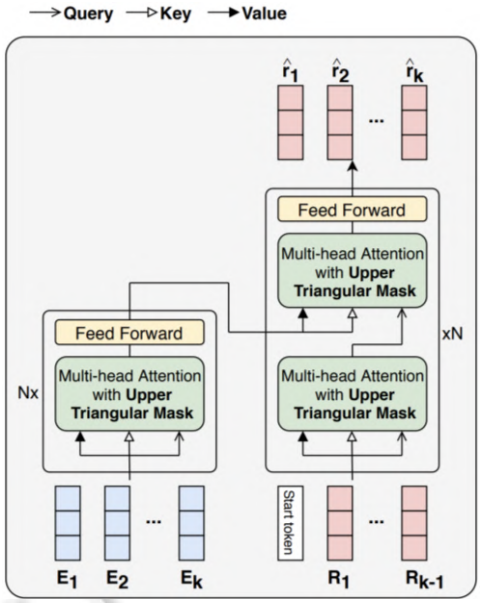


图 14 分离式自注意力知识追踪模型[40]

2.3.2 增加额外的学习特征

Bhatt 等人[41]提出用学生在每个知识点上截至当前时刻的平均正确率作为预测标签来训练网络. 他们使用了基于注意力的网络以及一个线性层进行预测.

Zhang 等人[42]在 Choi 等人[40]工作的基础上, 对学生的上课记录使用一个单独的注意力网络, 在解码器的输出后增加一层 GRU, 最后将 GRU 的输出、上课记录注意力网络的输出以及其他增加的学生学习特征拼接, 再经过一个线性层, 预测学生答题结果.

Oya 等人[43]认为, SAKT 的查询、键、值不包含过去的学习信息, 因此他们在 SAKT 输入端增加了一层 LSTM, 用于捕捉包含过去学习信息的查询、键、值. 由于 Transformer 对于序列长度有限制, 对于超出长度限制的序列, 学生的整体学习信息无法被完整捕捉到. 针对这一点, 他们在模型中增加了手工挑选的特征, 比如学生在所有学习历史中做题正确率等. 此外, 他们还在模型中增加了特殊索引, 以避免产生数据泄露问题.

2.3.3 模拟经典的教育理论

Ghosh 等人[44]提出使用情境感知的注意力知识追踪模型 AKT (attentive knowledge tracing)来解决 KT 任务, 充分使用学生的历史答题记录来对当前时刻的学习表现进行预测, 同时增加模型的可解释性. AKT 使用一系列的注意力网络, 将当前所做的习题和之前做过的所有习题建立联系, 捕捉历史答题记录对当前习题表现的影响, 还增加了单调注意力机制, 模拟学生在学习过程中的遗忘过程, 通过指数衰减, 降低较远的历史答题记录的权重值. AKT 主要包含 4 个部分: 习题编码器、知识编码器、知识获取模块和预测模块. AKT 模型架构如图 15 所示, 图中省略了单调注意力机制以及部分子层.

将原始的习题嵌入向量$\{\boldsymbol{x}_1,\ldots,\boldsymbol{x}_t\}$输入习题编码器, 并通过使用单调注意力机制, 输出情境感知的习题嵌入向量序列$\{\hat{\boldsymbol{x}}_1,...,\hat{\boldsymbol{x}}_t\}$. 情境感知的习题嵌入向量依赖于过去的所有做题记录以及当前的习题, 即:

$$\hat{\boldsymbol{x}}_t = fenc_1(\boldsymbol{x}_1,...,\boldsymbol{x}_t) \tag{22}$$

相似地, 知识编码器的输入是原始的习题-响应嵌入向量$\{\boldsymbol{y}_1,\ldots,\boldsymbol{y}_{t-1}\}$, 输出情境感知习题-响应嵌入向量序列$\{\hat{\boldsymbol{y}}_1,...,\hat{\boldsymbol{y}}_{t-1}\}$, 其中, $\hat{\boldsymbol{y}}_{t-1} = fenc_2(\boldsymbol{y}_1,...,\boldsymbol{y}_{t-1})$. Ghosh 等人认为, 学生对当前题目如何做出响应取决于学习者本身, 即取决于学生各自的学习历史, 因此使用情境感知嵌入向量来代替原始的嵌入向量. 两个编码器以及知

识获取模块各自有查询、键和值投影层，由于知识点不相关的习题以及越远的历史学习记录对当前的学习表现影响越小，模型中添加了乘法指数衰减项:

$$\alpha_{t,\tau}=\frac{\exp(s_{t,\tau})}{\sum_{\tau'}\exp(s_{t,\tau'})} \tag{23}$$

$$s_{t,\tau}=\frac{\exp(-\theta\cdot d(t,\tau))\cdot \boldsymbol{q}_t^{\boldsymbol{T}}\boldsymbol{k}_\tau}{\sqrt{D_k}} \tag{24}$$

其中，θ>0 是一个可学习的衰减参数，$d(t,\tau)$是 t 与 τ时刻之间的时间距离度量，$\boldsymbol{q}$ 与 $\boldsymbol{k}$ 分别是查询和键向量，D_k 是键向量的维度. 为了捕捉学生各自的做题顺序特征，以及加强过去学习历史中与当前习题相关题目的影响，Ghosh 等人提出，在两个编码器中加入带指数衰减机制的情境感知的时间距离度量方式:

$$d(t,\tau)=|t-\tau|\cdot\sum_{t'=\tau+1}^{t}\gamma_{t,t'} \tag{25}$$

$$\gamma_{t,t'}=\frac{\exp\left(\frac{\boldsymbol{q}_t^{\boldsymbol{T}}\boldsymbol{k}_{t'}}{\sqrt{D_k}}\right)}{\sum_{1\leqslant\tau'\leqslant t}\exp\left(\frac{\boldsymbol{q}_t^{\boldsymbol{T}}\boldsymbol{k}_{t'}}{\sqrt{D_k}}\right)},\ \forall t'\leqslant t \tag{26}$$

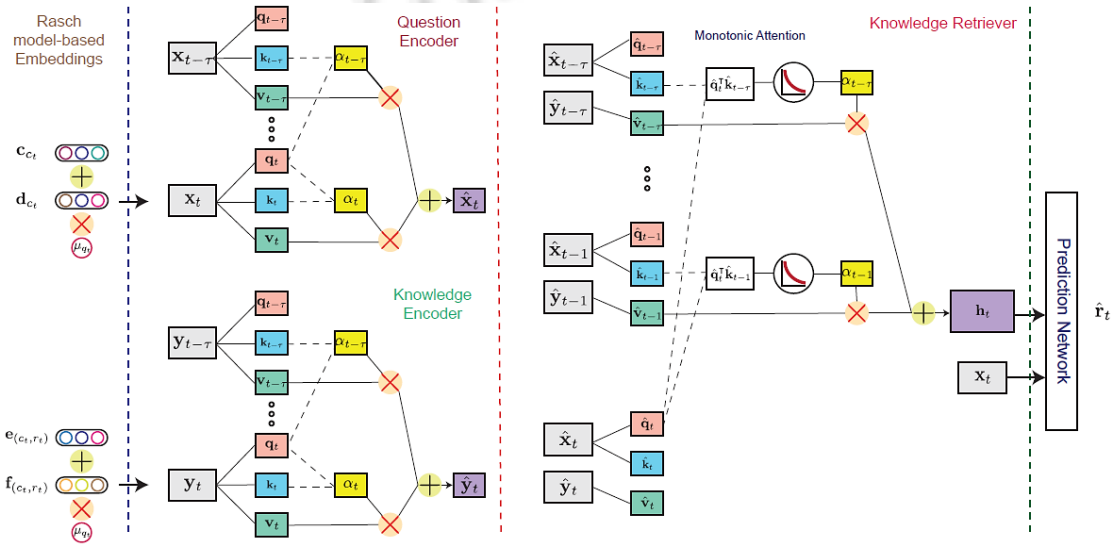


图 15 情境感知的注意力知识追踪模型[44]

他们还在注意力网络中使用多头注意力机制，不同的头拥有不同的衰减率，从不同时间尺度上总结学生的学习表现. 除此之外，每个编码器和知识获取模块中还加入了层归一化、dropout 层、全连接层以及残差网络. 预测模块的输入为知识获取模块的输出以及当前习题的嵌入向量，经过一个全连接层以及 Sigmoid 层，预测学生在当前习题上做对的概率. 在传统 KT 任务中，一个习题只对应一个知识点，在输入时，相同知识点的题目被认为是相同的，没有对覆盖相同知识点的题目做区分. Ghosh 等人提出，和同一个知识点相关的不同题目之间的区别不应被忽视. 因此，他们将 AKT 模型和 IRT 模型相结合，提出了 AKT-R 模型，不加 IRT 部分的模型为 AKT-NR，两者的主要区别在于图 15 左侧，两个编码器的输入部分为:

$$\begin{aligned}\boldsymbol{x}_t&=\boldsymbol{c}_{c_t}+\mu_{q_t}\cdot\boldsymbol{d}_{c_t}\\ \boldsymbol{y}_t&=\boldsymbol{e}_{(c_t,r_t)}+\mu_{q_t}\cdot\boldsymbol{f}_{(c_t,r_t)}\end{aligned} \tag{27}$$

其中，$\boldsymbol{c}_{c_t}$ 是习题对应的知识点的嵌入向量；$\boldsymbol{d}_{c_t}$ 向量总结了和当前知识点相关的所有习题的区分度；μ_{q_t} 是一

个标量，难度系数，表示当前习题和它所相关知识点之间的区别大小．相似地，$\boldsymbol{e}_{(c_t,r_t)}$ 和 $\boldsymbol{f}_{(c_t,r_t)}$ 分别是知识点-响应嵌入向量以及区分度向量．

基于 Rasch 模型的嵌入向量，使得对相同知识点不同习题的区分程度的捕捉以及对过量参数的避免取得了较好的权衡．

Pandey[45]等人认为，学生的每一步交互过程都对后续的学习有影响，这种影响与两种特征有关：1) 历史记录中各个习题之间的关联；2) 学生的遗忘行为．他们通过增加关系感知的自注意力层来包含学生学习过程中的上下文信息，并提出了 RKT (relation-aware self-attention model for knowledge tracing)模型，模型结构如图 16 所示．

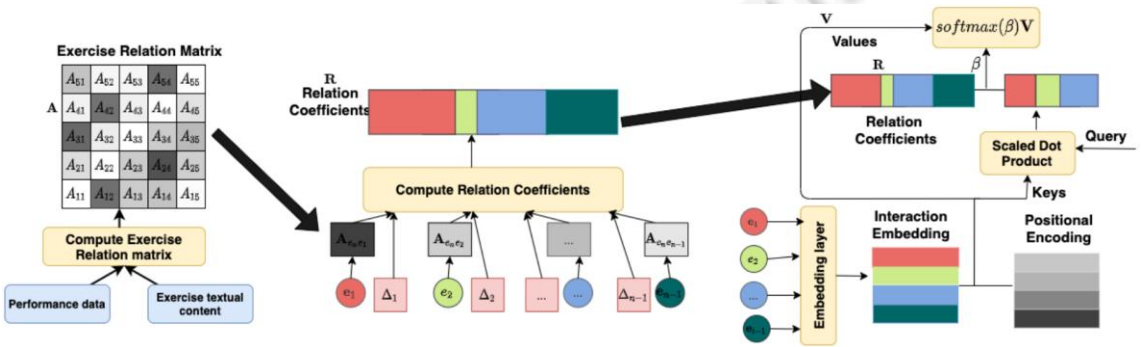


图 16　关系感知的自注意力知识追踪模型[45]

在原始 DKT 模型中，模型输入是知识点，具有相同知识点的习题被认为是相同的题目．在 RKT 中，题目与知识点不等价，模型按照题目来区分．RKT 通过学生过去的学习表现以及题目的文本内容两方面来挖掘习题之间的关联性．RKT 的输入为学生当前时刻前的交互序列 $X=\{x_1,x_2,\ldots,x_{n-1}\}$，其中，$x_i=(e_i,r_i,t_i)$，$e_i\in\{1,\ldots,E\}$是尝试过的题目，$r_i\in\{0,1\}$是学生正确与否，$t_i\in\mathbb{R}^+$是第 i 次交互发生的时间．对于每道题，模型通过：

$$\boldsymbol{E}_i=\frac{1}{|s_i|}\sum_{w\in s_i}\frac{a}{a+p(w)}f(w) \tag{28}$$

得到题目第 i 题的题目文本嵌入向量 $\boldsymbol{E}_i\in\mathbb{R}^d$，其中，$a$ 是可训练的参数，s_i 表示第 i 题的题目文本，$p(w)$是单词 w 出现的概率，$f(w)$通过 word2vec (word-to-vector)实现．对于习题 j 与习题 i 题目文本之间的相似度，通过余弦相似度进行计算，并表示为 $sim_{i,j}$．根据学生过去的答题表现，习题 j 与 i 之间的关联度通过 Phi 相关系数计算得到：

$$\phi_{i,j}=\frac{n_{11}n_{00}-n_{01}n_{10}}{\sqrt{n_{1*}n_{0*}n_{*1}n_{*0}}} \tag{29}$$

其中，n 指的是学生过去在习题 i/j 上做对/做错的次数，具体见表 1．对所有做过的题目计算习题关系矩阵 $\boldsymbol{A}$，习题 j 与 i 之间的关联度总体计算为：

$$\boldsymbol{A}_{i,j}=\begin{cases}\phi_{i,j}+sim_{i,j}, & \text{if } sim_{i,j}+\phi_{i,j}>\theta\\ 0, & \text{otherwise}\end{cases} \tag{30}$$

其中，θ是用于控制矩阵 $\boldsymbol{A}$ 稀疏程度的系数．则对当前的习题 e_n，关联度为习题关系矩阵的第 e_n 行，记为：

$$\boldsymbol{R}^E=[\boldsymbol{A}_{e_n,e_1},\boldsymbol{A}_{e_n,e_2},\ldots,\boldsymbol{A}_{e_n,e_{n-1}}] \tag{31}$$

表 1　习题 i 和 j 的关联表

		题目 i		总和
		回答正确	回答错误	
题目 j	回答正确	n_{00}	n_{01}	n_{0*}
	回答错误	n_{10}	n_{11}	n_{1*}
总和		n_{*0}	n_{*1}	n

对于遗忘行为, 模型将其表示为 $\boldsymbol{R}^T=[\exp(-\Delta_1/S_u),\exp(-\Delta_2/S_u),\ldots,\exp(-\Delta_{n-1}/S_u)]$, $\Delta_i=t_n-t_i$ 表示下一时刻的交互与第 i 时刻交互的时间差, S_u 是一个可训练参数, 用于表示学生 u 的记忆力. 因此, 关系系数可计算为:

$$\boldsymbol{R}=softmax(\boldsymbol{R}^E+\boldsymbol{R}^T) \tag{32}$$

对于模型原始输入的嵌入表示为 $\hat{\boldsymbol{X}}=[\hat{\boldsymbol{x}}_1,\hat{\boldsymbol{x}}_2,...,\hat{\boldsymbol{x}}_n]$, 它结合了交互嵌入矩阵 $\boldsymbol{E}$ 和位置编码嵌入矩阵 $\boldsymbol{P}$, 其中, $\hat{\boldsymbol{x}}_j=[\boldsymbol{E}_{e_j}\oplus\boldsymbol{r}_j]+\boldsymbol{P}_j$, $\boldsymbol{r}_j=\{r_j,r_j,\ldots,r_j\}\in\mathbb{R}^d$, r_j 是学生在习题 j 上回答正确与否; $\boldsymbol{P}_j$ 是习题的位置编码, 表示学生的做题顺序. 自注意力层根据以上输入以及关系系数计算权重:

$$\alpha_j=\frac{\exp(e_j)}{\sum_{k=1}^{n-1}\exp(e_k)},\ e_j=\frac{\boldsymbol{E}_{e_n}\boldsymbol{W}^Q(\hat{\boldsymbol{x}}_j\boldsymbol{W}^K)^T}{\sqrt{d}} \tag{33}$$

其中, $\boldsymbol{W}^Q,\boldsymbol{W}^K\in\mathbb{R}^{d\times d}$ 是查询和键的投影矩阵. 将注意力权重和相关系数相结合: $\beta_j=\lambda\alpha_j+(1-\lambda)\boldsymbol{R}_j$, λ 是一个可调参数. 自注意力层的输出为 $\boldsymbol{o}=\sum_{j=1}^{n-1}\beta_j\hat{\boldsymbol{x}}_j\boldsymbol{W}^V$, $\boldsymbol{W}^V\in\mathbb{R}^{d\times d}$ 是值投影矩阵. 自注意力层的输出再经过 FFN (feedforward network)、ReLU 激活层、全连接层以及 Sigmoid 层, 预测学生下一题能否回答正确. RKT 用学生过去表现和题目文本相似性, 捕捉题目之间的关联, 并考虑学生的遗忘行为, 提升模型效果.

为了进一步增强模型的可解释性和泛化性, Zhou 等人[46]提出了 LANA (leveled attentive knowledge tracing) 模型. 与 SAINT 不同, 他们认为, 应该确保模型有区分不同学生的能力, 当学生的做题序列足够长, 并且数据足够多时, 模型应该具有: 1) 对不同学生做题序列的辨别能力; 2) 对同一学生不同时期做题序列的辨别能力. 因此, 在 Transformer 的基础上, 他们增加了两个模块: SRFE (student-related features extractor)和 Pivot Module. SRFE 的架构如图 17 所示, 分为两个部分: Memory-SRFE 和 Performance-SRFE.

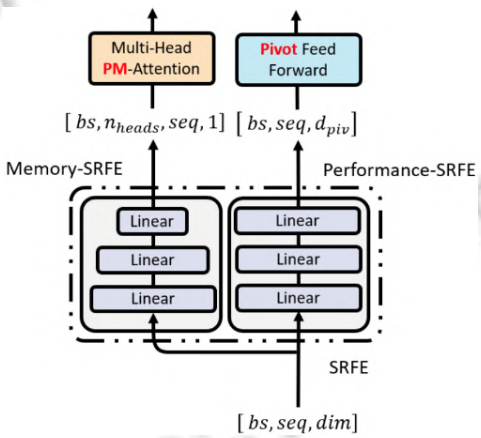


图 17 LANA-SRFE 架构图[46]

将编码器得到的表征序列经过自注意力机制蒸馏之后, 分别使用两个 SRFE 模块(多层感知机), 得到两个向量 $S_{mem}\in\mathbb{R}^{n_{heads}\times seq\times 1}$ 和 $S_{per}\in\mathbb{R}^{seq\times p_{div}}$, 其中, n_{heads}, seq, p_{div} 分别表示解码器中多头注意力机制的分区大小、序列长度以及学生的额外能力大小(如逻辑思考能力、计算能力等等). Pivot Module 分为两个部分, 分别对应解码器的注意力机制和 FFN 模块. 相对于原始的基于内积的注意力机制, 他们增加了遗忘机制来模拟真实的学生记忆情况. 对于题目 q_j 和 q_k, 他们的注意力权重为:

$$\alpha_{j,k,m}=\frac{\exp(-(\theta+m)\cdot dis(j,k))\cdot sim(j,k)}{\sum_{k'}sim(j,k')} \tag{34}$$

其中, m 由 S_{mem} 得到, 代表该学生的记忆能力; 而 θ 为可学习的参数, 代表平均的记忆能力; $dis(\cdot)$ 表示学生解答 q_j 和 q_k 直接的时间差. 为了增强模型的可解释性, 他们让模型在知道学生能力 S_{per} 的情况下, 考虑预测的结果, 因此引入了 *PivotLinear* 模块, 其中,

$$y = PivotLinear(p,x) = W \cdot x + b = (W_1^p \cdot p + b_1^p) \cdot x + (W_2^p \cdot p + b_2^p) \tag{35}$$

这使得模型强制将特征 p 纳入考虑范围, 因此, LANA 使用了基于上述思想的 PC-FFN:

$$PC\text{-}FFN(x,S_{per}) = x + PivotLinear(PivotLinear(x,S_{per}),S_{per}) \tag{36}$$

最后, 题目的预测序列由解码器输出. LANA 在 Transformer 的基础上增加了对学生个性化能力的预测, 并将其融入解码器中, 使得模型对于学生个性化的区分度更加明显.

2.4 基于其他神经网络的模型

Shen 等人[47]认为, 大多数 KT 模型没有对学生进行个性化建模. 他们认为, 学生在做题时的先验知识(即已掌握的知识)以及学习率是因人而异的. 为了解决这个问题, 他们提出了一个基于卷积神经网络的 CKT 模型(convolutional knowledge tracing), 对学生的个性化特征进行建模. 模型结构如图 18 所示.

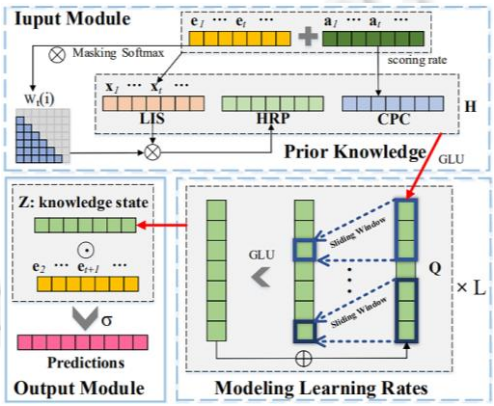


图 18 卷积知识追踪模型[47]

模型由 3 个模块组成: 输入模块、学习率模块、输出模块. 输入模块包含 3 个部分: 学习交互序列(***LIS***)、历史相关表现(***HRP***)、知识点正确率(***CPC***). $\boldsymbol{LIS} \in \mathbb{R}^{N\times 2K}$ 由各时刻历史交互行为向量 $x_t \in \mathbb{R}^{2K}$ 组成:

$$\boldsymbol{x}_t = \begin{cases} [\boldsymbol{e}_t \oplus \boldsymbol{a}_t], \text{ if } a_t = 1 \\ [\boldsymbol{a}_t \oplus \boldsymbol{e}_t], \text{ if } a_t = 0 \end{cases} \tag{37}$$

$\boldsymbol{e}_t$ 是 K 维的习题嵌入向量; $\boldsymbol{a}_t$ 是维度和 $\boldsymbol{e}_t$ 相同的全零向量; ⊕为两个向量的拼接操作; a_t 表示学生在当前题目上是否回答正确, 正确为 1, 错误为 0. Shen 等人认为, 学生的先验知识隐含在历史做题记录中, 学生在相同的题目上可能得到相同的分数, 并且得分率可以看作是学生总体知识掌握水平的一个反馈. 因此, 模型综合了学生的历史学习记录, 从 ***HRP*** 和 ***CPC*** 两个方面来捕捉不同学生的先验知识. $\boldsymbol{HRP} \in \mathbb{R}^{N\times 2K}$ 关注于衡量和当前题目相关的历史做题表现, 两道习题之间的相关系数通过将两道题的嵌入向量做点乘得到, 并使用 masking 操作将后面尚未做过的习题排除在外:

$$\begin{cases} r_t(i) = Masking(\boldsymbol{e}_i \cdot \boldsymbol{e}_t), \ i \in (t,N) \\ w_t(i) = Softmax(r_t(i)), \ i \in (1,N) \end{cases} \tag{38}$$

$$\boldsymbol{HRP}_t(t) = \sum_{i=1}^{t-1} w_t(i)\boldsymbol{x}_i \tag{39}$$

N 表示当前时刻; $w_t(i)$是通过第 i 时刻的题目和当前题目的相关系数计算得到的权重, 根据权重, 对历史交互行为向量进行加权求和, 得到 ***HRP*** 向量. $\boldsymbol{CPC} \in \mathbb{R}^{N\times M}$ 是对学生在各个知识点上掌握程度的总结, 对学生截至目前做过的题目, 分别计算正确率:

$$\boldsymbol{CPC}_t(m) = \frac{\sum_{i=0}^{t-1} a_i^m == 1}{count(e^m)} \tag{40}$$

其中, $m\in(1,M)$是指知识点 m, e^m 是和知识点 m 相关的习题, $count(e^m)$是习题 e^m 被回答过的次数, $\sum_{i=0}^{t-1}a_i^m==1$ 是习题 e^m 被正确回答的次数. 将 ***LIS***, ***HRP*** 和 ***CPC*** 这 3 个向量拼接, 并经过一个门控线性单元, 得到学习率模块的输入矩阵 $\boldsymbol{H}$:

$$\begin{cases}\boldsymbol{H}=\boldsymbol{LIS}\oplus\boldsymbol{HRP}\oplus\boldsymbol{CPC}\\ \boldsymbol{Q}=(\boldsymbol{H}\boldsymbol{W}_1+\boldsymbol{b}_1)\otimes\sigma(\boldsymbol{H}\boldsymbol{W}_2+\boldsymbol{b}_2)\end{cases}\tag{41}$$

其中, $\boldsymbol{W}_1$, $\boldsymbol{W}_2$, $\boldsymbol{b}_1$, $\boldsymbol{b}_2$ 是需要学习的参数, σ是 Sigmoid 函数, ⊗是逐元素相乘操作. 随后, 学习率模块通过堆叠 L 层一维卷积神经网络, 形成层次卷积网络, 从矩阵 $\boldsymbol{Q}$ 中提取学生的学习率特征. 在卷积过程中加入滑动窗口, 以避免包含尚未做过的题目信息. 低层的卷积网络用于捕捉较近一段时间内的学习率, 高层的卷积网络捕捉长时间内的学习率. 最后, 通过将学生知识状态和下一习题的嵌入向量做点乘来对学生能否做对下一道题进行预测.

Nakagawa 等人[48]使用图神经网络(graph neural networks, GNN)来解决知识追踪问题. 他们将习题抽象为一个无向图 $G=(V,E,\boldsymbol{A})$, 节点 $V\in\{v_1,\ldots,v_N\}$对应习题包括的全部 N 个知识点, 边 $E\subseteq V\times V$ 表示知识点之间的依赖关系, 邻接矩阵 $\boldsymbol{A}\in\mathbb{R}^{N\times N}$ 表示这些依赖关系的大小程度. 类似地, GNN 也使用隐层向量 $h^t=\{h_{i\in V}^t\}$ 表示学生对所有知识点的掌握情况. 当学生解答了与知识点 i 相关的习题后, 知识点本身对应的知识状态 h_i^t 和相邻的节点对应的知识状态 $h_{j\in\mathcal{N}_i}^t$ 均会被更新. 基于以上假设, Nakagawa 等人提出了基于图神经网络的知识追踪模型 GKT (graph-based knowledge tracing), 模型结构如图 19 所示.

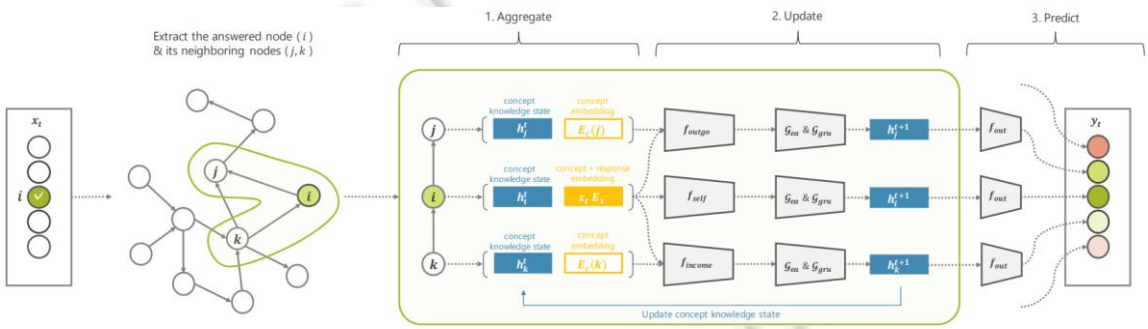


图 19 基于图神经网络的知识追踪模型[48]

从图中可以看出, GKT 模型的更新分为 3 个部分: 聚合、更新和预测. 首先, 将 t 时刻的输入 $x_t=(q_t,r_t)$与其相关知识状态 h_k^t 通过函数累加得到新的状态 h_k^{rt}. 根据更新节点是否为相关知识点 i, 对节点 i 及相邻节点做如下更新:

$$h_k^{rt}=\begin{cases}[h_k^t,x_t\boldsymbol{E}_x], & k=i\\ [h_k^t,\boldsymbol{E}_c(k)], & k\neq i\end{cases}\tag{42}$$

其中, $\boldsymbol{E}_x\in\mathbb{R}^{2N\times e}$ 为一个嵌入向量, 嵌入向量的维度为 e, 它将学生的交互行为编码; $\boldsymbol{E}_c\in\mathbb{R}^{N\times e}$ 表示知识点索引的嵌入矩阵, 其中, $\boldsymbol{E}_c(k)$表示矩阵 $\boldsymbol{E}_c$ 的第 k 行. 他们引用了 MANN 的思想, 使用多层感知机和写入擦除技术来进一步提取隐层状态, 其公式如下:

$$m_k^{t+1}=\begin{cases}f_{self}(h_k^{rt}), & k=i\\ f_{neighbor}(h_i^{rt},h_k^{rt}), & k\neq i\end{cases}\tag{43}$$

$$\tilde{m}_k^{t+1}=\mathcal{G}_{ea}(m_k^{t+1})\tag{44}$$

$$\tilde{m}_k^{t+1}=\mathcal{G}_{gru}(m_k^{t+1})\tag{45}$$

其中, f_{self} 为多层感知机, $\mathcal{G}_{ea}$ 为 MANN 中的擦写方法, $\mathcal{G}_{gru}$ 表示 GRU 的门控单元, $f_{neighbor}$ 是基于图结构向邻节点传播信息的自定义函数. 计算 $f_{neighbor}$ 的方式有两种: 基于数理统计的方法和基于学习的方法. 前者将邻接

矩阵 $\boldsymbol{A}$ 应用到统计学的方法上，并提出如下函数:

$$f_{neighbor}(h_i^{rt},h_j^{rt})=\boldsymbol{A}_{i,j}f_{outgo}([h_i^{rt},h_j^{rt}])+\boldsymbol{A}_{j,i}f_{income}([h_i^{rt},h_j^{rt}]) \tag{46}$$

其中，f_{outgo} 和 f_{income} 为多层感知机. $\boldsymbol{A}_{i,j}$ 根据图定义的不同稍有区别，共有 3 种类型的图: 密度图、概率转移矩阵和 DKT 图计算. 基于学习的方法也有 3 种，分别是参数邻接矩阵(PAM)、多头注意力机制(MHA)和变分自编码器(VAE).

DKT 在某些领域表现很好(如数学选择题预测等)，但是在某些领域表现并不好. 为了训练一个能够从源领域到目标领域自适应的模型, Cheng 等人[49]提出了 AdaptKT (domain adaption for knowledge tracing)模型，以解决以下两个问题: (1) 源域数据和目标域数据存在差距; (2) 源域与目标域本身分布不一致. 为了最小化目标域和源域的数据差异，找到与目标域相似的数据样本, AdaptKT 使用了基于 LSTM 的自编码器，对元数据 X_S 和目标数据 X_T 同时编码，并使用一个选择器 u_S 来自动筛选与目标域相近的数据. 损失函数如下:

$$L(\pi_e,\pi_d,u_S)=\frac{1}{n_S}\sum_{i=1}^{n_S}u_S^iR(\tilde{x}_S^i,x_S^i)+\frac{1}{n_T}\sum_{i=1}^{n_T}u_T^iR(\tilde{x}_T^i,x_T^i)+\lambda(u_S) \tag{47}$$

其中，π_e，π_d 分别为编码器和解码器的参数; $R(\tilde{x},x)$ 为自编码器的重构损失; $u_S^i\in\{0,1\}$ 为指示函数，表示源数据与目标数据是否相似.

AdaptKT 采用交替训练的方式，如:

- 先固定 u_S，若 $R(\tilde{x}_S^i,x_S^i)\leqslant\beta$ (β为常数)，则将 u_S^i 置为 1; 否则为 0.
- 然后固定其他参数，使用 u_S 重新筛选数据并开始下一轮训练. 他们认为，这样可以筛选出源域和目标域相似的数据. 为了缩小两个域之间的分布差异，AdaptKT 在 LSTM 的输出之后增加了 Adaptation Layer (线性层)，并且将筛选出的源数据与目标数据同时输入到模型中，并最小化它们之间的差异，如图 20 所示.
- 最后，为了最终在目标域得到预测结果，AdaptKT 保留了包括 Adaptation Layer 在内的所有参数且保持不变，并在 Adaptation Layer 之后加入了线性层做最后的预测.

AdaptKT 解决了 DKT 无法在不同领域迁移的问题，是 KT 问题在自适应学习上的首次尝试.

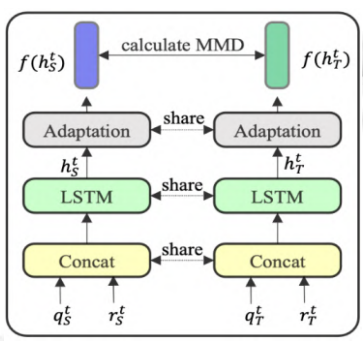


图 20 AdaptKT 步骤 2 的结构[49]

2.5 深度知识追踪模型小结

表 2 对前述模型的演化路线和所采用的结构进行了总结，其中所采用的结构包括循环神经网络(RNN)、图神经网络(graph)、记忆网络(memory)、自注意力网络(SA)以及卷积神经网络(CNN). 某些模型可能采用了一种结构作为主体结构，并且使用其他结构进行辅助. 例如, GIKT 使用了 RNN 作为主体结构，并使用图神经网络提取输入特征.

表 2 深度知识追踪模型演化路线和结构

类别	模型名称	演化路线	使用结构				
			RNN	Graph	Memory	Attention	CNN
基于循环神经网络	DKT[8]		✓	–	–	–	–
	Stacked LSTM[13]	改进模型的基础结构	✓	–	–	–	–
	EKT[14]		✓	–	–	✓	–
	KQN[15]		✓	–	–	–	–
	DKT-DT[16]	增加额外的解题特征	✓	–	–	–	–
	DKT-Ext[17]		✓	–	–	–	–
	EERNN[18]		✓	–	–	✓	–
	DKTS[19]		✓	✓	–	–	–
	qDKT[20]		✓	–	–	–	–
	MFA-DKT[21]		✓	–	–	✓	
	EHFKT[22]		✓	–	–		✓
	AKTHE[23]		✓	–	–	✓	–
	GIKT[24]		✓	✓	–	–	–
	PDKT-C[25]	模拟教育理论	✓	–	–	–	–
	DKT-DSC[26]		✓	–	–	–	–
	DKT-F[27]		✓	–	–	–	–
	KPT[29]		✓	–	–	–	–
	IEKT[30]		✓	–	–	–	–
基于记忆网络	DKVMN[9]	改进模型的基础结构	–	–	✓	–	–
	SKVMN[31]		✓	–	✓	–	–
	CoLearn[32]	增加额外的解题特征	–	–	✓	–	–
	DKVMN-DT[33]		–	–	✓	–	–
	DKVMN-DSC[34]		–	–	✓	–	–
	DKVMN-LA[35]		–	–	✓	–	–
	DTKT[36]	模拟教育理论	✓	–	✓	–	–
	Deep-IRT[37]		–	–	✓	–	–
	KIKT[38]		–	–	✓	–	–
	DGMN[39]		–	✓	–	–	–
基于自注意力机制	SAKT[12]	改进模型的基础结构	–	–	–	✓	–
	SAINT[40]		–	–	–	✓	–
	Av_Att[41]	增加额外的解题特征	✓	–	–	✓	–
	MUSE[42]		✓	–	–	✓	–
	LSTM-SAKT[43]		✓	–	–	✓	–
	AKT[44]	模拟教育理论	–	–	–	✓	–
	RKT[45]		✓	–	–	✓	–
	LANA[46]		–	–	–	✓	–
基于其他神经网络	CKT[47]	使用卷积神经网络对学生个性化建模	–	–	–	–	✓
	GKT[48]	使图网络学习知识点之间的依赖关系	–	✓	–	–	–
	AdaptKT[49]	使用自适应学习将 KT 迁移到新的领域	✓	–	–	–	–

3 深度知识追踪模型预测性能比较

上述工作讨论了将知识追踪应用于教育场景需要考虑的实际问题, 并设计了相应的模型结构加以解决. 不过, 知识追踪的核心任务始终是评价学生的知识水平, 而这种评价是否精准, 是通过模型预测学生未来答题表现的准确度来反映的. 因此在本节中, 我们从上述模型中选取最具代表性的模型, 并且使用知识追踪领域的公开数据集衡量模型预测学生答题表现的性能. 我们分析实验结果, 并对上述模型在教学场景中的使用作出相关推荐.

3.1 数据集

在本文的对比实验中, 我们选取了 5 个到目前为止在知识追踪领域最通用的公开数据集, 分别为 ASSISTments2009, ASSISTments2015, ASSISTments2017, Statics2011 以及 Synthetic-5. 表 3 展示了这 5 个数据集的统计信息.

- ASSISTments2009 (https://sites.google.com/site/assistmentsdata/home/assistment-2009-2010-data/skill-

builder-data-2009-2010)是从 ASSISTments 教育平台收集到的 2009–2010 学年学生的学习数据, 该数据集中包含 110 个不同的习题(即知识点, 根据惯例, 包含相同知识点的习题被认为是同一道习题), 共有 4 151 名学生的 325 637 条做题记录.

- ASSISTments2015 (https://sites.google.com/site/assistmentsdata/home/2015-assistments-skill-builder-data)是 ASSISTments2009 的更新版本, 是收集于 2015 年的数据, 其中包含了 100 个不同的习题, 共有 19 840 名学生的 683 801 条做题记录. 在 5 个数据集中, 这个数据集涵盖的学生数量最多, 但是平均每个学生的做题数量是最少的.
- ASSISTments2017 (https://sites.google.com/view/assistmentsdatamining)是 ASSISTments 教育平台上 2017 年收集的数据, 包含 102 个不同的习题, 共有 1 709 名学生的 942 816 条做题记录.
- Statics2011 (https://pslcdatashop.web.cmu.edu/DatasetInfo?datasetId=507)收集自 2011 年秋季卡内基梅隆大学的一门统计学课程, 包含 1 223 个不同的习题, 共有 333 名学生的 189 297 条做题记录. 在 5 个数据集中, 平均每个学生的做题数量最多.
- Synthetic-5 (https://github.com/chrispiech/DeepKnowledgeTracing/tree/master/data/synthetic)是 Piech 等人[8]生成的模拟数据集, 包含 20 个不同的习题, 模拟 4 000 名学生的 200 000 条做题记录, 每个学生的答题顺序完全相同.

表 3 5 个公开数据集的统计信息

数据集	题目数量(知识点数量)	学生数量	学生做题总数	学生平均做题数
ASSISTments2009	110	4 151	325 637	78
ASSISTments2015	100	19 840	683 801	34
ASSISTments2017	102	1 709	942 816	552
Statics2011	1 223	333	189 297	568
Synthetic-5	50	4 000	200 000	50

在这些数据集中, Statics2011 和 ASSISTments2017 的学生数量少, 但每个学生的做题序列长度较长. 因此, 按照相关工作中广泛使用的方法[8,9], 我们对过长的序列进行了折叠操作. 当一个序列的长度超过 200 时, 我们会截断该序列, 并用截断后多出的数据形成一个新的做题序列. 因此, 所有序列的长度都小于或等于 200.

3.2 实验模型

我们从上述每个类别的模型中, 选择最具代表性的一两个模型进行实验评估. 我们采用两个简单、有效的标准选择模型: 第一, 提出该模型的论文已经被发表在各自领域的顶级会议或期刊上, 从而保证所提出模型的先进性; 第二, 该模型在基准数据集上取得了较好的预测效果. 通过对这些模型进行全面性能评估, 有助于我们了解目前深度知识追踪的最先进水平(the state of the art). 根据这个筛选标准, 我们最终选取了以下 8 个模型: DKT (NIPS 2015)、DKT-F (WWW 2019)、EKT (TKDE 2019)、DKVMN (WWW 2017)、SKVMN (SIGIR 2019)、SAKT (EDM 2019)、AKT (KDD 2020)以及 CKT (SIGIR 2020). DKT、DKT-F、EKT 使用循环神经网络作为主体结构, DKVMN 和 SKVMN 使用记忆网络, SAKT 和 AKT 使用注意力网络, CKT 使用卷积神经网络. 这 8 个模型涵盖了前述 4 种模型类别, 其中, EKT 和 AKT 分别有两种具体实现[14,44]. 对于 EKT, 我们选取了在原文实验中表现更好的、使用注意力机制聚合隐层状态的模型结构. 由于公开数据集中没有题目文本信息, EKT 的原作者也并未开源所使用的数据集, 因此我们采用两种方式来表示文本信息的特征: 第 1 种是使用一个固定的随机向量来表示包含相同知识点的题目文本特征, 相应模型命名为 EKT-R; 第 2 种是使用知识点的向量来代替题目文本特征, 相应模型命名为 EKT-C. 对于 AKT, 原文中实现了 AKT-R 和 AKT-NR 两种模型: 前者在模型输入端需要同时包含知识点 ID 和题号 ID 的信息, 但是部分数据集中没有题号 ID 信息, 并且其他模型均未使用题号 ID. 为了公平起见, 我们采用 AKT-NR 模型进行比较.

对于 DKT (https://github.com/chrispiech/DeepKnowledgeTracing), EKT (https://github.com/bigdata-ustc/ekt), DKVMN (https://github.com/jennyzhang0215/DKVMN), AKT (https://github.com/arghosh/AKT)和 CKT (https://github.com/shshen-closer/Convolutional-Knowledge-Tracings), 我们使用了原作者在 GitHub 中开源的代码. 我

们从原作者处获得了 SAKT 的代码. SKVMN 和 DKT-F 的原作者未开源代码, 我们复现了这两个模型. 其中, SAKT 和 SKVMN 和原文结果有所不同, 原因如下: SAKT 作者提供的代码在计算 AUC 时, 错误地将用于序列补全的伪数据也包括在内, 大大提高了模型精度; 而 SKVMN 在构建 Hop-LSTM 结构时, 需要人为设置判断习题知识点相似度的阈值, 模型性能会受到设置的影响. 所有模型的代码均可从此仓库下载: https://gitee.com/daphnezmx/dkt-models.

3.3 模型训练及测试结果

对于每个数据集, 我们用 80%的随机数据训练模型, 并用剩余 20%的数据作为测试集. 训练数据集又被随机划分成 5 份, 用于 5 折验证. 我们汇报每个模型在测试集上的平均表现. 我们使用的评估指标是 ROC 曲线下面积, 简称 AUC, 取值在 0–1 之间, AUC 越大, 表示模型表现越好. 表 4 展示了模型在 5 个数据集上的预测精度结果, 在每个数据集上表现最好的模型结果已加粗显示, 我们使用百分制来表示模型的 AUC 结果.

表 4 8 个深度知识追踪模型的 AUC 结果(%)

数据集	DKT	DKT-F	EKT-R	EKT-C	DKVMN	SKVMN	SAKT	AKT-NR	CKT
ASSISTments2009	81.19	81.88	76.46	76.45	80.02	67.39	76.59	81.84	**82.13**
ASSISTments2015	71.95	72.96	70.65	70.35	72.33	67.01	73.27	73.43	**73.45**
ASSISTments2017	64.47	**73.48**	60.25	61.56	68.53	56.95	64.85	72.06	72.16
Statics2011	79.00	**82.76**	75.65	77.73	80.42	78.41	81.43	82.74	82.38
Synthetic-5	81.03	82.26	78.50	75.20	82.60	75.46	82.53	**83.39**	82.85

首先, 我们看到, CKT 和 DKT-F 分别在两个真实的数据集上表现最好, 而 AKT-NR 在合成数据集上表现最好. 进一步观察 CKT 和 DKT-F 的实验结果, 我们可以发现, 这两个模型在 5 个数据集上的总体表现是最好的. 值得注意的是, CKT 和 DKT-F 分别针对个性化的学习历史和学习遗忘规律构建了结构化特征, 这表明, 尽管深度神经网络可以自动从海量数据中搜寻特征, 但恰当的特征工程依然能够提升深度模型的预测性能; 另一方面, AKT-NR 使用自注意力机制来自动学习历史交互信息和模拟知识遗忘规律, 取得了接近 CKT 和 DKT-F 的总体表现. 其在 Synthetic-5 上取得的优越表现, 可能是因为所有学生都具有相同的做题序列, 因此通过自注意力机制来捕获历史情境特征较为容易. 综合这些结果, 在实际运用知识追踪模型时, 可以考虑构建与学习规律相关的特征或者捕获这些特征的模型结构来提升追踪效果.

其次, 在各种改进模型层出不穷的情况下, 最早提出的 DKT 模型仍然表现出较好的性能, 总体表现稍逊于上述 3 个模型. 其中, DKT 在 ASSISTments2009 和 Synthetic-5 数据集上预测精度相对较高, 而在其余 3 个数据集上表现较差. 这可能是由于 ASSISTments2015 数据集序列长度太短, 而 ASSISTments2017 和 Statics2011 数据集的序列长度太长导致. 这表明, 合适的序列长度对使用简单的循环神经网络的 DKT 至关重要. 而旨在改进 DKT 记忆单元的 DKVMN 模型, 在 4 个数据集上的表现超越了 DKT, 说明通过增加参数, 使用记忆矩阵来表示学生知识点的掌握状态, 能够提升知识追踪的效果. 而在 DKVMN 模型上改进得到的 SKVMN 模型, 尽管解释性较好, 但是在各个数据集上预测精度相比 DKVMN 和 DKT 均明显下降. 这可能是由于三角隶属函数并没有准确捕捉到各题目之间知识点分布的相似性, Hop-LSTM 模块在预测做题表现时起到了反作用. 综合这些结果, 我们认为, 在寻求知识追踪效率的前提下, 可以使用无需特征工程、结构较为简单的 DKT 和 DKVMN 等模型.

第三, 由于缺失题目文本内容, EKT 模型的预测精度相较原文明显下降. 因此我们猜测, EKT 模型对于文本内容信息具有强烈依赖性. 这也从侧面证明了 EKT 原模型的合理性, 即在拥有题目文本的前提下, 融合文本信息能够提升模型的预测效果.

最后, 基于自注意力网络的 SAKT 模型总体表现优于 DKT 模型. 可见, 使用自注意力机制能够很好地捕捉当前习题知识点与学生历史做题序列中各个时刻表现的关联性, 从而提高预测当前习题表现的准确性. 另一方面, SAKT 的总体性能不如 AKT. 这是由于 AKT 对于题目和知识点本身进行了情境感知学习, 提高了题目和知识点编码的精确性. 这说明, 如何构建自注意力网络捕捉做题历史的关联性, 仍然值得重点考虑.

总体而言, 我们认为, 提升对于学生未来学习表现的预测精度, 关键在于充分挖掘以及强化历史学习过

程中与当前习题相关的题目的影响, 依据学生在过去相关题目中的表现来更准确地推断学生在某个知识点上的掌握程度. 同时, 考虑学生对于知识的遗忘等学习规律, 也有助于提升知识追踪效果.

4 深度知识追踪应用案例分析

深度知识追踪模型目前应用最多的场景为获取学生的知识水平, 经过学者们对技术的探索以及对模型的完善, 当前提出的一些模型已能够处理含有多个知识点的题目, 且能够对每个知识点的掌握程度进行建模, 可解释性得到增强. 随着深度知识追踪技术的不断发展, 其应用场景也越来越多, 学者们通过借鉴深度知识追踪模型的想法, 还延伸出了在职业预测、学生编程、第二语言学习等场景下的深度知识追踪应用. 需要指出的是, 当前在实际教学中运用最多的是最早提出的 DKT 模型, 充分说明了这个模型的简洁性和通用性.

Yeung 等人[50]将 DKT 模型应用到学生职业选择的预测中, 预测学生毕业后的第一份工作是否属于 STEM 领域. 他们从 ASSISTments 学习平台(https://new.assistments.org/)收集学生在中学时期的历史答题数据, 对数据集进行分析, 并找出可能影响学生第一份工作是否属于 STEM/non-STEM 领域的重要因素. 数据集提供了包含提示请求、答题对错和题目等信息的历史答题数据以及学生资料. 数据集中包含 467 名带 STEM/non-STEM 标签的学生, 其中 117 名属于 STEM 类, 其余 350 名属于 non-STEM 类. Yeung 等人认为, 学生全面的知识状态比数据集中提供的平均知识状态更有利于预测的准确性. 因此, 他们使用 DKT 模型, 根据历史答题数据获得当前知识状态, 与学生资料中的平均知识状态相结合, 作为职业预测模型的输入. 除了职业预测外, 对学生知识状态的详细分析表明, 与 non-STEM 类学生相比, STEM 类学生通常在数学方面表现出更高的掌握水平.

Swamy 等人[51]将 DKT 模型应用到编程学情分析中, 从学生提交的代码中追踪学生当前的代码进度. 他们使用了 UC Berkeley 的数据科学入门课程《Data 8》的学生代码数据, 该课程介绍编程基础知识、统计推断和预测技术, 每学期吸引 1 000 名学生. 在作业中, 老师提供问题描述和初始代码, 学生使用 Jupyter Notebook 完成代码. 受到 DKT 模型的启发, 模型输入是题目编号、学生代码以及尝试答题的次数, 输出是对于相同知识点的所有题目剩余的答题次数. 通过观察预测结果, 每个题目预测剩余答题次数较多的学生, 可能在练习中遇到了困难, 此时, 老师可以介入来帮助学生解决问题. 类似地, Wang 等人[52]将 DKT 模型应用到学生编程能力的预测中. 他们使用的数据集来自 Code.org 的《Hour of Code》课程, 包含 20 个编程入门习题. 针对每个习题, 学生可以多次运行代码并产生快照, 因此可以利用代码快照来追踪学生的编程进度. 他们提出了两种知识追踪任务: 第 1 种是利用学生所有的代码快照预测学生能否编出下一个习题的代码; 第 2 种是利用学生在当前习题中的代码快照, 预测学生是否完成当前的编程习题. 模型使用了 LSTM 结构, 输入为每一个代码快照的嵌入表示, 输出为预测学生能否顺利完成当前或下一个编程习题. 通过预测学生接下来的编程表现, 老师们可以及时辅导编程有困难的学生.

深度知识追踪模型也被用于第二语言教学中[53], 根据学生在一门新语言学习过程中的答题记录, 预测学生在未来可能发生的错误. 模型使用的数据集是在线语言学习应用 Duolingo 的部分用户前 30 天的学习数据, 习题类别有翻译、给词造句以及听写这 3 种. 判题时, 将学生的作答结果与参考答案进行逐个单词的比较: 若与答案相同, 则认为正确; 否则错误. 模型使用两部分 LSTM 结构: 第 1 部分是双向 LSTM, 输入为由学生答题信息、用户 ID、语言类别等特征做嵌入后拼接成的向量, 将学生答案中的单词与参考答案进行逐个单词的对比, 预测学生在每个单词上是否会犯错; 第 2 部分, LSTM 利用了第 1 部分中的隐藏向量与输出信息, 追踪学生的学习历史. 通过将深度知识追踪应用于第二语言学习, 老师们能够对学生学习新语言的掌握情况获得更多的了解.

DKT 模型也可以用来作为推荐系统, 指导大学生选课. 大学课程数量众多, 合理选课一直是一个难题, 学生往往是在课程重要性、课程难度以及预期的成绩之间进行权衡. 为每个学生规划个性化的选课路径是非常困难的, 因为每个学生的目标和兴趣不尽相同. Jiang 等人[54]借鉴 DKT 模型的结构, 构建了基于 LSTM 的推荐系统, 为大学生选课提供建议. 实验使用了 UC Berkeley 从 2008 年秋季学期到 2017 年春季学期的课程学习

数据, 总共包括来自 197 个学科的 9 714 门基础课程, 每一门课程都有至少 20 位学生. 类似于 DKT 模型的答题正确率预测功能, Jiang 等人使用 LSTM 模型, 在每个时间步上输入学生历史选修记录中后一学期的选修课程以及前一学期的课程成绩, 来预测学生所选的后一学期的课程中的成绩. 由于学习知识由浅入深, 有先后顺序, 相关课程的选修在实际中也是有先后顺序的. Jiang 等人所开发的推荐系统, 可根据学生输入的长远的目标课程以及期望达到的成绩来推荐下一学期可选择的课程, 以供学生参考.

深度知识追踪模型在实际应用中还可以产生有趣的"副产品". Zhang 等人[55]利用 DKT 模型预测的正确答题概率来反推学生掌握知识点的先后顺序, 然后根据这一顺序挖掘知识点之间的拓扑次序关系. 实验同样使用了 ASSISTments 数据集, 在预处理时, 把每一道题目用对应的知识点 ID 标识, 每一道题对应一个知识点. 知识点拓扑次序的生成包含两个步骤.

- 首先, 利用 DKT 模型, 根据学生的答题历史信息来预测答对下一题的概率. 由于每个题目用知识点标识, 因此预测的概率可视为相应知识点的掌握程度.
- 然后, 用启发式方法定义发掘知识点拓扑次序的规则, 当正确答对下一题的概率大于设定的阈值时, 认为该学生已经掌握了相应的知识点. 如果当前知识点已被掌握, 则输出中概率最高的知识点可被视为当前知识点的前提条件; 如果当前知识点尚未被掌握, 则输出中概率最低的知识点可视为当前知识点的前提条件. 最后, 再用拓扑排序算法等生成拓扑图.

5 未来的研究方向

当前, 深度学习和知识追踪任务已经有了深度的结合, 主要的研究进展在于提出了各种神经网络结构来模拟学习者的学习规律, 并在预测学生未来的答题表现上取得了前所未有的准确性. 尽管如此, 知识追踪相关研究的最终目的是将所研发的模型用于现实的教学场景. 为了达到这个目的, 我们认为深度知识追踪研究尚有以下几个问题需要解决.

5.1 深度知识追踪模型的可解释性研究

早期的浅层知识追踪模型虽然预测精度不及深度模型, 但是具有较强的可解释性. 例如在 BKT[2]中, 模型使用了 4 个具有实际意义的参数来模拟学生获取知识的过程, 结合观测数据, 可以很直观地解释模型如何计算得到知识掌握程度的预测结果. 再比如基于因子分析的模型[3,5], 也可以利用回归分析得到的系数来解释模型的预测结果. 然而, 深度知识追踪模型由于使用了多层非线性结构, 无法直接使用模型的参数来解释预测结果; 另一方面, 可解释性对于知识追踪模型在实际教学中的应用又至关重要, 老师和学生在信任模型给出的知识掌握度预测结果之前, 都希望了解相应的结果是如何计算得到的. 因此, 深度知识追踪模型的可解释性研究, 是模型能否真正在实际教学中落地的关键.

深度模型的可解释性研究路线可以大致分为全局解释和局部解释两种: 前者分析模型中各个层次从输入数据中学到的具体特征, 或者哪些输入模式导致模型输出人们关注的结果[56,57]; 而后者分析对于某个给定的输入数据, 其哪些特征导致了模型给出的预测结果[58,59]. 在运用知识追踪模型时, 老师更加关注的是, 某个特定学生在和习题的交互过程中, 哪些时刻的做题特征使得模型给出了相应的知识掌握度预测. 因此我们认为, 局部解释是更加适合深度知识追踪模型的研究方向, 这其中一个近期最具代表性的领域是"输入显著性方法"[60]. 这类方法的核心目的是针对模型的每一个预测结果, 给出相应输入特征的显著性, 即每个输入特征对预测结果的贡献度. 常用的方法包括基于梯度的方法[59,61]、基于反向传播的方法[62,63]和基于遮挡的方法[64,65]. 使用这些方法, 在模型做出相关预测后, 老师可以从学生和习题的历史交互中, 根据各个时刻特征的贡献度, 找出导致预测结果的原因. 另一方面, 由于输入显著性方法将模型视为一个整体, 仅研究输入特征和输出结果之间的关联, 因此适用于各种模型结构, 对于深度知识追踪任务的模型具有普适性.

5.2 特定场景下的深度知识追踪模型研究

现有的深度知识追踪研究延续了知识追踪问题的经典定义, 聚焦教育的普遍场景, 因此对模型的输入输

出信息进行了简化和一般性假设, 例如每个题目只考察一个知识点、学生和题目的交互结果只有对和错两类、每个特定的题目只考虑学生第 1 次解答的结果等等. 然而, 这些假设在真实教学场景下往往不符合学生和习题交互的实际情况, 如果直接运用普遍场景的知识追踪模型, 可能会导致对知识点掌握度预测精度的下降. 例如在编程教育[52,66]中, 一道习题可能会考察多个编程技能的综合运用, 学生提交的代码也并非非对即错, 学生通常需要反复尝试才能提交正确的代码. 此外, 学生的知识状态直接蕴含在学生所提交的代码中, 因此需要将学生代码特征作为模型的额外输入. 其他学科如第二语言教育[67,68]、STEM 类教育[50], 也都有各自独有的学科特征. 因此, 在将深度知识追踪模型运用于特定场景时, 必须考虑结合该场景下学生和习题交互过程的特殊性.

特定场景下的深度知识追踪模型研究, 可以重点从两方面展开.

- 一是研究使用更适合场景特征的输入输出表征. 例如在第二语言教育中, 学生往往通过解答大量选择题来加深对语法的理解, 而选择题的每个选项通常会捕捉特定的知识状态, 因此学生答题结果的表征需要使用更加复杂的方法, 代替简单的 0/1 向量表示. 再比如, 高等教育中通常会考察学生综合运用知识的能力, 因此每个习题一般会包含多个知识点, 这时也不再适合用独热向量来表示习题, 可以考虑采用预训练等方式获取习题的表征, 作为模型输入.
- 另一方面是研究特定场景下特有特征的表征方法. 例如在编程教育中, 学生提交的代码蕴含了学生知识状态的丰富信息, 如何表征代码[69,70], 并将其融入模型中[71], 是一个很值得研究的问题. 再比如数学、化学等学科, 学生提交的表达式、几何图形、结构式等等, 反映了学生对于所考查知识点的掌握程度, 因此也应该作为知识追踪模型的重要输入特征, 并研究其合适的表征方法. 当传统的线下教育转变为线上教育时, 还可以结合在线平台收集的细粒度学生行为数据, 加强知识追踪模型的模拟效果, 例如鼠标轨迹[72]、手势、姿态[73,74]等多模态数据.

5.3 基于深度知识追踪的个性化导学研究

知识追踪的教学应用目标不仅仅是评估学生对于知识或者技能的掌握程度, 更重要的意义在于根据评估的结果推荐学习内容, 规划学习路径, 进而实现个性化的导学[2,8]. 然而当前深度知识追踪模型的研究往往始于对学生知识状态的模拟, 终于知识点本身以及知识点之间关联的可视化, 而基于知识追踪进行个性化导学的研究则屈指可数, 主要仍然依赖教师经验为学生制定个性化的学习方案. 这种方式显然是不可扩展的, 尤其不适用于如今快速发展的在线教育及其庞大的学生规模. 当前, 深度知识追踪模型已经能够较为精准地评估学生对于知识点的掌握程度, 因此, 如何基于对知识点掌握程度的评估, 实现自适应的个性化导学, 将会是一个可行而有趣的研究问题.

个性化导学的本质是针对每个学生的学习路径规划, 其核心是个性化的学习内容推荐, 因此, 基于深度知识追踪的个性化导学至少可以从两种技术路线展开.

- 第 1 种路线是先利用深度知识追踪模型评估学生的知识点掌握程度, 然后使用基于协同过滤的方法[75,76]推荐学习内容. 例如, 在得到某一时刻学生 S 的知识状态(知识点掌握程度或隐层表示)之后, 可以寻找具有相似知识状态、并且最终达到考核要求的一组学生, 然后根据知识状态的相似度对这些学生进行排序, 并把这些学生的后续学习内容推荐给学生 S.
- 第 2 种技术路线是采用强化学习算法[77,78]进行学习内容的推荐. 强化学习算法的训练目标通常是最大化长期、全局的收益; 而个性化导学的最终目标是让每位学生根据自身的特点和节奏, 最大限度掌握所有知识点. 两者在最大化长期收益的目标上不谋而合, 因此, 强化学习适合用于学习内容的推荐. 例如, 最基本的习题推荐强化学习环境可以这样构建: 学生和习题的交互历史可以作为状态, 并且随着学生的交互过程不断更新; 利用学生和习题的交互历史可以训练神经网络, 用于根据学生状态推荐习题; 最后利用预先训练的深度知识追踪模型, 可以预测每次交互之后学生知识掌握程度的变化, 并设计相应的推荐奖励, 强化学习模型的训练目标就是最大化习题推荐的长期奖励, 即知识掌握程度的增长. 值得注意的是, 基于强化学习的技术路线在具有强交互的学习环境下效果更好, 因此

更加适合与在线教育中的知识追踪模型相结合, 在规模化教学中实现个性化导学.

6 结 论

本文介绍了使用深度学习来解决知识追踪任务的研究进展, 并比较了 8 个代表性的深度知识追踪模型的性能. 基于循环神经网络的模型结构简单, 训练开销较小, 并且能够有效地预测学生在未来的答题表现, 其缺点是无法充分表示学生对于特定知识点的掌握状态. 基于动态记忆网络的模型使用了额外的记忆模块弥补了这一缺点, 兼顾了模型预测性能和对知识状态的表示. 基于自注意力机制的模型, 通过充分挖掘当前学习行为与历史学习行为的关联性, 增强了相关性强的特征的影响, 进一步提升了模型预测性能. 而基于其他结构的模型, 如卷积神经网络, 也提供了构建深度知识追踪模型的新思路, 并取得了良好的效果.

我们认为, 深度知识追踪模型的广泛应用, 不仅依赖于模型性能的提升, 还依赖于模型和应用场景的有机结合, 即根据课程的实际教学过程, 修改模型细节, 使其能够适应课程所产生的学生数据. 深度知识追踪模型与教学应用场景的有机结合, 将会是教育学研究中一个非常有趣的问题. 随着深度知识追踪模型的逐步应用, 教学辅助系统能够完成实时追踪学生学习状态、帮助学生制定个性化学习计划和自动推荐学习材料与习题等任务, 从而使得教学过程变得更加“智适应”.

References:

[1] Wolins L, Wright BD, Rasch G. Probabilistic models for some intelligence and attainment tests. Journal of the American Statistical Association, 1982, 77(377): 220.

[2] Corbett AT, Anderson JR. Knowledge tracing: Modeling the acquisition of procedural knowledge. User Modelling and User-adapted Interaction, 1995, 4(4): 253−278.

[3] Cen H, Koedinger K, Junker B. Learning factors analysis—A general method for cognitive model evaluation and improvement. In: Ikeda M, Ashley KD, Chan TW, eds. Proc. of the Intelligent Tutoring Systems, Vol.4053. Berlin, Heidelberg: Springer, 2006. 164−175.

[4] Pavlik PI, Cen H, Koedinger KR. Performance factors analysis—A new alternative to knowledge tracing. In: Proc. of the Conf. on Artificial Intelligence in Education: Building Learning Systems that Care: From Knowledge Representation to Affective Modelling. 2009. 531−538.

[5] Chi M, Koedinger K, Gordon G, Jordan P, VanLehn K. Instructional factors analysis: A cognitive model for multiple instructional interventions. In: Proc. of the 12th Int’l Conf. on Educational Data Mining. 2011. 61−70.

[6] Liu HY, Zhang TC, Wu PW, Yu G. A review of knowledge tracking. Journal of East China Normal University (Natural Science), 2019, 2019(5): 1−15 (in Chinese with English abstract). [doi: 10.3969/j.issn.1000-5641.2019.05.001]

[7] Liu Q, Shen S, Huang Z, Chen E, Zheng Y. A survey of knowledge tracing. arXiv:2105.15106, 2021.

[8] Piech C, Spencer J, Huang J, Ganguli S, Sahami M, Guibas L, Sohl-Dickstein J. Deep knowledge tracing. In: Proc. of the Advances in Neural Information Processing Systems. Montreal, 2015. 505−513.

[9] Zhang J, Shi X, King I, Yeung DY. Dynamic key-value memory networks for knowledge tracing. In: Proc. of the 26th Int’l Conf. on World Wide Web. 2017. 765−774.

[10] Weston J, Chopra S, Bordes A. Memory networks. arXiv:1410.3916, 2015.

[11] Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, Kaiser Ł, Polosukhin I. Attention is all you need. In: Proc. of the Advances in Neural Information Processing Systems. Curran Associates, Inc., 2017. 5998−6008.

[12] Pandey S, Karypis G. A self-attentive model for knowledge tracing. In: Proc. of the 12th Int’l Conf. on Educational Data Mining. 2019. 384−389.

[13] Sha L, Hong P. Neural knowledge tracing. In: Proc. of the Int’l Conf. on Brain Function Assessment in Learning. Cham: Springer, 2017. 108−117.

[14] Liu Q, Huang Z, Yin Y, Chen E, Xiong H, Su Y, Hu G. EKT: Exercise-aware knowledge tracing for student performance prediction. IEEE Trans. on Knowledge and Data Engineering, 2019, 33(1): 100−115.

[15] Lee J, Yeung DY. Knowledge query network for knowledge tracing: How knowledge interacts with skills. In: Proc. of the 9th Int'l Conf. on Learning Analytics & Knowledge. Tempe: ACM, 2019. 491−500.

[16] Yang H, Cheung LP. Implicit heterogeneous features embedding in deep knowledge tracing. Cognitive Computation, 2018, 10(1): 3−14.

[17] Zhang L, Xiong X, Zhao S, Botelho A, Heffernan NT. Incorporating rich features into deep knowledge tracing. In: Proc. of the 4th (2017) ACM Conf. on Learning @ Scale. Cambridge, Massachusetts: ACM, 2017. 169−172.

[18] Su Y, Liu Q, Liu Q, Huang Z, Yin Y, Chen E, Ding C, Wei S, Hu G. Exercise-enhanced sequential modeling for student performance prediction. In: Proc. of the AAAI Conf. on Artificial Intelligence. 2018. 2435−2443.

[19] Wang Z, Feng X, Tang J, Huang GY, Liu Z. Deep knowledge tracing with side information. In: Proc. of the Int'l Conf. on Artificial Intelligence in Education. Cham: Springer, 2019. 303−308.

[20] Sonkar S, Waters AE, Lan AS, Grimaldi PJ, Baraniuk RG. QDKT: Question-centric deep knowledge tracing. In: Proc. of the Int'l Conf. on Artificial Intelligence in Education. Cham: Springer, 2021. 433−437.

[21] Liu D, Zhang Y, Zhang J, Li Q, Zhang C, Yin Y. Multiple features fusion attention mechanism enhanced deep knowledge tracing for student performance prediction. IEEE Access, 2020, 8: 194894−194903.

[22] Tong H, Zhou Y, Wang Z. Exercise hierarchical feature enhanced knowledge tracing. In: Proc. of the Int'l Conf. on Artificial Intelligence in Education. Cham: Springer, 2020. 324−328.

[23] Zhang N, Du Y, Deng K, Li L, Shen J, Sun G. Attention-based knowledge tracing with heterogeneous information network embedding. In: Li G, Shen HT, Yuan Y, Wang X, Liu H, Zhao X, eds. Proc. of the Knowledge Science, Engineering and Management, Vol.12274. Cham: Springer, 2020. 95−103.

[24] Yang Y, Shen J, Qu Y, Liu Y, Wang K, Zhu Y, Zhang W, Yu Y. GIKT: A graph-based interaction model for knowledge tracing. In: Proc. of the Joint European Conf. on Machine Learning and Knowledge Discovery in Databases. Cham: Springer, 2020. 299−315.

[25] Chen P, Lu Y, Zheng VW, Pian Y. Prerequisite-driven deep knowledge tracing. In: Proc. of the IEEE Int'l Conf. on Data Mining (ICDM). Singapore: IEEE, 2018. 39−48.

[26] Minn S, Yu Y, Desmarais MC, Zhu F, Vie JJ. Deep knowledge tracing and dynamic student classification for knowledge tracing. In: Proc. of the IEEE Int'l Conf. on Data Mining (ICDM). Qingdao: IEEE, 2018. 1182−1187.

[27] Nagatani K, Zhang Q, Sato M, Chen YY, Chen F, Ohkuma T. Augmenting knowledge tracing by considering forgetting behavior. In: Proc. of the World Wide Web Conf. San Francisco: ACM, 2019. 3101−3107.

[28] Ebbinghaus H. Memory: A contribution to experimental psychology. Annals of Neurosciences, 2013, 20(4): 155−156.

[29] Huang Z, Liu Q, Chen Y, Wu L, Xiao K, Chen E, Ma H, Hu G. Learning or forgetting? A dynamic approach for tracking the knowledge proficiency of students. ACM Trans. on Information Systems (TOIS), 2020, 38(2): 1−33.

[30] Long T, Liu Y, Shen J, Zhang W, Yu Y. Tracing knowledge state with individual cognition and acquisition estimation. In: Proc. of the 44th Int'l ACM SIGIR Conf. on Research and Development in Information Retrieval. Virtual Event: ACM, 2021. 173−182.

[31] Abdelrahman G, Wang Q. Knowledge tracing with sequential key-value memory networks. In: Proc. of the 42nd Int'l ACM SIGIR Conf. on Research and Development in Information Retrieval. 2019. 175−184.

[32] Chaudhry R, Singh H, Dogga P, Saini SK. Modeling Hint-taking Behavior and Knowledge State of Students with Multi-task Learning. Int'l Educational Data Mining Society, 2018.

[33] Sun X, Zhao X, Ma Y, Yuan X, He F, Feng J. Muti-behavior features based knowledge tracking using decision tree improved DKVMN. In: Proc. of the ACM Turing Celebration Conf. Chengdu: ACM, 2019. 1−6.

[34] Minn S, Desmarais MC, Zhu F, Xiao J, Wang J. Dynamic student classiffication on memory networks for knowledge tracing. In: Proc. of the Pacific-Asia Conf. on Knowledge Discovery and Data Mining. Cham: Springer, 2019. 163−174.

[35] Sun X, Zhao X, Li B, Ma Y, Sutcliffe R, Feng J. Dynamic key-value memory networks with rich features for knowledge tracing. IEEE Trans. on Cybernetics, 2022, 52(8): 8239−8245.

[36] Ha H, Hwang U, Hong Y, Jang J, Yoon S. Deep trustworthy knowledge tracing. arXiv:1805.10768, 2018.

[37] Yeung CK. Deep-IRT: Make deep learning based knowledge tracing explainable using item response theory. arXiv:1904.11738, 2019.

[38] Gan W, Sun Y, Sun Y. Knowledge interaction enhanced knowledge tracing for learner performance prediction. In: Proc. of the 7th Int'l Conf. on Behavioural and Social Computing (BESC). Bournemouth: IEEE, 2020. 1−6.

[39] Abdelrahman G, Wang Q. Deep graph memory networks for forgetting-robust knowledge tracing. arXiv:2108.08105, 2021.

[40] Choi Y, Lee Y, Cho J, Baek J, Kim B, Cha Y, Shin D, Bae C, Heo J. Towards an appropriate query, key, and value computation for knowledge tracing. In: Proc. of the 11th Int'l Learning Analytics and Knowledge Conf. (LAK 2021). 2021. 490−496.

[41] Bhatt S, Zhao J, Thille C, Zimmaro D, Gattani N. A novel approach for knowledge state representation and prediction. In: Proc. of the 7th ACM Conf. on Learning @ Scale. Virtual Event: ACM, 2020. 353−356.

[42] Zhang C, Jiang Y, Zhang W, Gu C. MUSE: Multi-scale temporal features evolution for knowledge tracing. arXiv:2102.00228, 2021.

[43] Oya T, Morishima S. LSTM-SAKT: LSTM-encoded SAKT-like transformer for knowledge tracing. arXiv:2102.00845, 2021.

[44] Ghosh A, Heffernan N, Lan AS. Context-aware attentive knowledge tracing. In: Proc. of the 26th ACM SIGKDD Int'l Conf. on Knowledge Discovery & Data Mining. 2020. 2330−2339.

[45] Pandey S, Srivastava J. RKT: Relation-aware self-attention for knowledge tracing. In: Proc. of the 29th ACM Int'l Conf. on Information & Knowledge Management. 2020. 1205−1214.

[46] Zhou Y, Li X, Cao Y, Zhao X, Ye Q, Lv J. LANA: Towards Personalized Deep Knowledge Tracing through Distinguishable Interactive Sequences. Int'l Educational Data Mining Society, 2021.

[47] Shen S, Liu Q, Chen E, Wu H, Huang Z, Zhao W, Su Y, Ma H, Wang S. Convolutional knowledge tracing: modeling individualization in student learning process. In: Proc. of the 43rd Int'l ACM SIGIR Conf. on Research and Development in Information Retrieval. Virtual Event: ACM, 2020. 1857−1860.

[48] Nakagawa H, Iwasawa Y, Matsuo Y. Graph-based knowledge tracing: modeling student proficiency using graph neural network. In: Proc. of the IEEE/WIC/ACM Int'l Conf. on Web Intelligence. Thessaloniki: ACM, 2019. 156−163.

[49] Cheng S, Liu Q, Chen E, Zhang K, Huang Z, Yin Y, Huang X, Su Y. AdaptKT: A domain adaptable method for knowledge tracing. In: Proc. of the 15th ACM Int'l Conf. on Web Search and Data Mining. Virtual Event: ACM, 2022. 123−131.

[50] Yeung CK, Yeung DY. Incorporating features learned by an enhanced deep knowledge tracing model for STEM/non-STEM job prediction. Int'l Journal of Artificial Intelligence in Education, 2019, 29(3): 317−341.

[51] Swamy V, Guo A, Lau S, Wu W, Wu M, Pardos Z, Culler D. Deep knowledge tracing for free-form student code progression. In: Penstein RoséC, Martínez -Maldonado R, Hoppe HU, Luckin R, Mavrikis M, Porayska-Pomsta K, McLaren B, Du Boulay B, eds. Proc. of the Artificial Intelligence in Education, Vol.10948. Cham: Springer, 2018. 348−352.

[52] Wang L, Sy A, Liu L, Piech C. Deep knowledge tracing on programming exercises. In: Proc. of the 4th (2017) ACM Conf. on Learning @ Scale. Cambridge Massachusetts: ACM, 2017. 201−204.

[53] Kaneko M, Kajiwara T, Komachi M. TMU system for SLAM-2018. In: Proc. of the 13th Workshop on Innovative Use of NLP for Building Educational Applications. New Orleans: Association for Computational Linguistics, 2018. 365−369.

[54] Jiang W, Pardos ZA, Wei Q. Goal-based course recommendation. In: Proc. of the 9th Int'l Conf. on Learning Analytics & Knowledge. Tempe: ACM, 2019. 36−45.

[55] Zhang J, King I. Topological order discovery via deep knowledge tracing. In: Hirose A, Ozawa S, Doya K, Ikeda K, Lee M, Liu D, eds. Proc. of the Neural Information Processing, Vol.9950. Cham: Springer, 2016. 112−119.

[56] Zeiler MD, Fergus R. Visualizing and understanding convolutional networks. In: Proc. of the European Conf. on Computer Vision. Cham: Springer, 2014. 818−833.

[57] Nguyen A, Dosovitskiy A, Yosinski J, Brox T, Clune J. Synthesizing the preferred inputs for neurons in neural networks via deep generator networks. In: Proc. of the Advances in Neural Information Processing Systems, Vol.29. 2016. 3387−3395.

[58] Baehrens D, Schroeter T, Harmeling S, Kawanabe M, Hansen K. How to explain individual classification decisions. The Journal of Machine Learning Research, 2010, 11(6): 1803−1831.

[59] Sundararajan M, Taly A, Yan Q. Axiomatic attribution for deep networks. In: Proc. of the Int'l Conf. on Machine Learning. PMLR, 2017. 3319−3328.

[60] Bastings J, Filippova K. The elephant in the interpretability room: Why use attention as explanation when we have saliency methods? In: Proc. of the 3rd BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP. 2020. 149−155.

[61] Li J, Chen X, Hovy E, Jurafsky D. Visualizing and understanding neural models in NLP. In: Proc. of the NAACL-HLT. 2016. 681−691.

[62] Bach S, Binder A, Montavon G, Klauschen F, Müller KR, Samek W. On pixel -wise explanations for non-linear classifier decisions by layer-wise relevance propagation. PLOS ONE, 2015, 10(7): Article No.0130140.

[63] Arras L, Montavon G, Müller KR, Samek W. Explaining recurrent neural network predictions in sentiment analysis. In: Proc. of the 8th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis. 2017. 159−168.

[64] Li J, Monroe W, Jurafsky D. Understanding neural networks through representation erasure. arXiv:1612.08220, 2016.

[65] DeYoung J, Jain S, Rajani NF, Lehman E, Xiong C, Socher R, Wallace BC. ERASER: A benchmark to evaluate rationalized NLP models. In: Proc. of the 58th Annual Meeting of the Association for Computational Linguistics. 2020. 4443−4458.

[66] Jiang B, Wu S, Yin C, Zhang H. Knowledge tracing within single programming practice using problem-solving process data. IEEE Trans. on Learning Technologies, 2020, 13(4): 822−832.

[67] Renduchintala A, Koehn P, Eisner J. Knowledge tracing in sequential learning of inflected vocabulary. In: Proc. of the 21st Conf. on Computational Natural Language Learning (CoNLL 2017). Vancouver: Association for Computational Linguistics, 2017. 238−247.

[68] Zylich B, Lan A. Linguistic skill modeling for second language acquisition. In: Proc. of the LAK21: the 11th Int'l Learning Analytics and Knowledge Conf. Irvine: ACM, 2021. 141−150.

[69] Allamanis M, Brockschmidt M, Khademi M. Learning to represent programs with graphs. arXiv:1711.00740, 2017.

[70] Zhang J, Wang X, Zhang H, Sun H, Wang K, Liu X. A novel neural source code representation based on abstract syntax tree. In: Proc. of the IEEE/ACM 41st Int'l Conf. on Software Engineering (ICSE). Montreal: IEEE, 2019. 783−794.

[71] Piech C, Huang J, Nguyen A, Phulsuksombati M, Sahami M, Guibas L. Learning program embeddings to propagate feedback on student code. In: Proc. of the Int'l Conf. on Machine Learning. PMLR, 2015. 1093−1102.

[72] Wei H, Li H, Xia M, Wang Y, Qu H. Predicting student performance in interactive online question pools using mouse interaction features. In: Proc. of the 10th Int'l Conf. on Learning Analytics & Knowledge. 2020. 645−654.

[73] Chopade P, Khan SM, Edwards D, Von Davier A. Machine learning for efficient assessment and prediction of human performance in collaborative learning environments. In: Proc. of the IEEE Int'l Symp. on Technologies for Homeland Security (HST). Woburn: IEEE, 2018. 1−6.

[74] Olsen JK, Sharma K, Rummel N, Aleven V. Temporal analysis of multimodal data to predict collaborative learning outcomes. British Journal of Educational Technology, 2020, 51(5): 1527−1547.

[75] He X, Liao L, Zhang H, Nie L, Hu X, Chua TS. Neural collaborative filtering. In: Proc. of the 26th Int'l Conf. on World Wide Web. 2017. 173−182.

[76] Zhao J, Bhatt S, Thille C, Zimmaro D, Gattani N. Interpretable personalized knowledge tracing and next learning activity recommendation. In: Proc. of the 7th ACM Conf. on Learning @ Scale. Virtual Event: ACM, 2020. 325−328.

[77] Huang Z, Liu Q, Zhai C, Yin Y, Chen E, Gao W, Hu G. Exploring multi-objective exercise recommendations in online education systems. In: Proc. of the 28th ACM Int'l Conf. on Information and Knowledge Management. Beijing: ACM, 2019. 1261−1270.

[78] Bassen J, Balaji B, Schaarschmidt M, Thille C, Painter J, Zimmaro D, Games A, Fast E, Mitchell JC. Reinforcement learning for the adaptive scheduling of educational activities. In: Proc. of the CHI Conf. on Human Factors in Computing Systems. Honolulu: ACM, 2020. 1−12.

附中文参考文献:

[6] 刘恒宇, 张天成, 武培文, 于戈. 知识追踪综述. 华东师范大学学报(自然科学版), 2019, 2019(5): 1−5. [doi: 10.3969/j.issn.1000-5641.2019.05.001]

王宇(1997－), 男, 博士生, 主要研究领域为数据驱动的计算教育学.

朱梦霞(1996－), 女, 硕士, 主要研究领域为数据驱动的计算教育学.

杨尚辉(1996－), 男, 硕士, 主要研究领域为图像处理, 计算机视觉, 数据驱动的计算教育学.

陆雪松(1985－), 男, 博士, 副研究员, CCF 专业会员, 主要研究领域为数据驱动的计算教育学.

周傲英(1965－), 男, 博士, 教授, 博士生导师, CCF 会士, 主要研究领域为数据库, 数据管理, 数字化转型, 金融科技, 计算教育, 教育科技和物流科技等数据驱动的应用.