Prerequisite Structure Discovery in Intelligent Tutoring Systems

Louis Annabi, Sao Mai Nguyen

▶ **To cite this version:**

Louis Annabi, Sao Mai Nguyen. Prerequisite Structure Discovery in Intelligent Tutoring Systems. ICDL 2023 - IEEE International Conference on Development and Learning, Nov 2023, Macau, China. pp.176-181, 10.1109/icdl55364.2023.10364416 . hal-04401880

HAL Id: hal-04401880
https://inria.hal.science/hal-04401880

Submitted on 18 Jan 2024

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Distributed under a Creative Commons Attribution 4.0 International License

Annabi and Nguyen. Prerequisite structure discovery for an intelligent tutoring system based on intrinsic motivation. IEEE ICDL. 2023. https://doi.org/10.1109/ICDL55364.2023.10364416

Prerequisite Structure Discovery In Intelligent Tutoring Systems

Louis Annabi[1] and Sao Mai Nguyen[2]

***Abstract*—This paper addresses the importance of Knowledge Structure (KS) and Knowledge Tracing (KT) in improving the recommendation of educational content in intelligent tutoring systems. The KS represents the relations between different Knowledge Components (KCs), while KT predicts a learner's success based on her past history. The contribution of this research includes proposing a KT model that incorporates the KS as a learnable parameter, enabling the discovery of the underlying KS from learner trajectories. The quality of the uncovered KS is assessed by using it to recommend content and evaluating the recommendation algorithm with simulated students.**

***Index Terms*—Intelligent Tutoring Systems, Knowledge Tracing, Knowledge Structure Discovery**

I. Introduction

A. Context

In recent years, online education platforms have gained immense popularity, leading to a growing interest in automated methods for recommending pedagogical content. Intelligent Tutoring Systems (ITS) are computer systems that provide personalised recommendation of educational content to optimise learners' progress, for instance in the form of social robots for education [1] or massive open online courses.

An ITS is generally decomposed into four components: (i) the *domain knowledge* about the educational concepts, rules and problem solving strategies. It especially includes the list of skills or concepts that need to be mastered, called Knowledge Components (KCs), the list of KCs necessary to complete each exercise, as well as the relations between KCs, called the Knowledge Structure (KS); (ii) a *student model* that estimates the evolution of the learner's knowledge states; (iii) a *tutoring model* that recommends educational content (in our case exercises) to the learners, possibly based on the domain knowledge and on a student model; (iv) a user interface.

ITS often aim at maximising the learning progress of the student on the long run, thus relying on a model of the development of the learner's cognitive skills. Learning progress is both the natural extrinsic reward a tutor would try to maximise, and the intrinsic motivation for the learner, as theorised in developmental psychology [2] and modeled for developmental systems [3]. Typically, pedagogical activities (in our case exercises) that maximise learning progress should be neither too hard nor too easy for the learner. This idea aligns with the concept of Zone of Proximal Development (ZPD) introduced by Vygotsty [4]. The ZPD refers to the set of activities a learner is unable to do by herself but can do with some assistance. Because learning is most effective in the ZPD, several works have proposed implementations of this concept in tutoring models: [5] propose a tutoring model based on the automatic estimation of the ZPD by exploiting domain knowledge given by experts, while [6] and [7] use student models to estimate respectively student forgetting and student probability of success on given exercises.

The ZPD actually evolves as the learner acquires new knowledge. Knowledge Tracing (KT) corresponds to the task of accurately estimating the evolving knowledge states of learners on the KCs, based on their history of exercises and answers [8]. KT models are typically trained as sequence prediction models, using recorded trajectories of students. As such, they jointly perform the task of tracing student knowledge and modeling student trajectories. For this reason, we indistinctively refer to KT models and student models.

While machine learning techniques have been widely applied to KT and tutoring models, only few have focused on domain knowledge and in particular KS discovery, when most rely on an expert-given domain knowledge. Yet, knowing the KS can improve the accuracy of KT models [9], but most importantly can directly be exploited by a tutoring model in order to organize exercises into curricula [5].

B. Problem and Contribution

In this work, we depart from the common hypothesis of expert-given KS, and instead explore the use of machine learning techniques to discover prerequisite relations from recordings of learners interacting with an ITS. The KS is generally represented as a graph, where nodes are the KCs and edges are the relations between these KCs, as illustrated in Fig. 1. These relations represent for instance the prerequisite relations between KCs as in [9].

We propose a KS discovery method using only learners' performance data. Our method relies on a KT model conditioned on the KS: the KT model assumes that the learner's success on an exercise is conditional on her skill level on the exercise's KCs and prerequisite KCs. The prerequisite graph is learned together with the other parameters of the KT model, using backpropagation through time.

To exploit learners' performance data for deriving a KS, we can unfortunately not rely on publicly available datasets, where the exercise sequences are determined based on prior domain knowledge, which can be erroneous, incomplete or not available. The first reason is that the exercise sequencing

This work was supported by Inter Carnot MINES - TSN

[1] Flowers Team, U2IS, ENSTA Paris, Institut Polytechnique de Paris & Inria, Palaiseau, France louis.annabi@gmail.com

[2] Flowers Team, U2IS, ENSTA Paris, Institut Polytechnique de Paris & Inria, Palaiseau, France nguyensmai@gmail.com

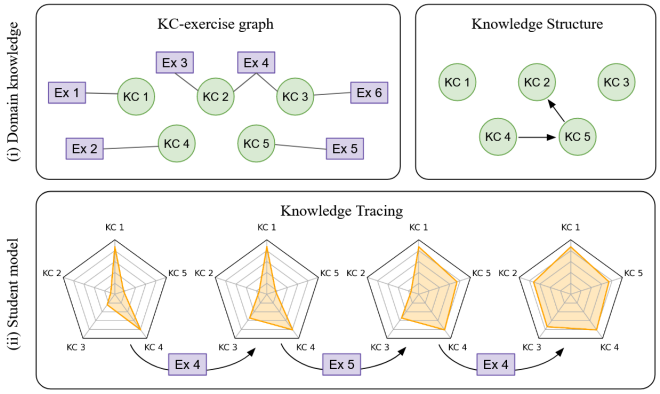

Fig. 1. Domain knowledge (top) and student models (bottom) can be exploited by tutoring models. The KC-exercise graph gives the KCs practiced on each exercise. The Knowledge Structure (KS) is represented as a directed acyclic graph where the edges represent prerequisite relations between KCs. Knowledge Tracing (KT) models predict the learner's level on each KC at each step of practice, here for a learner trajectory of length T=3.

is biased and data necessary to discover some prerequisite relations from the learners' performances might be missing in the designed curriculum. For instance, for an ITS for basic arithmetic concepts such as addition and multiplication, if exercises on multiplication are only recommended after exercises on addition, it is impossible to discover that there is a prerequisite relation between addition KC and multiplication KC. Indeed, we would need observations of attempts of exercises addressing the multiplication KC while still not proficient on the addition KC – such observations could be absent from the dataset. Second, the available datasets have generally been obtained with exercise sequences chosen by an expert or a tutoring model biased by expert knowledge, whereas we wish to address the problem even when the domain knowledge is unknown and no expert is available. Therefore, a relevant evaluation of KS discovery algorithms must use data with exercise sequences agnostic of the domain knowledge, to remove the possibility that the algorithm discovers the KS owing to this expert knowledge instilled in the exercise sequences, and to ensure that KS discovery methods capture information from user performances even in the absence of prior domain knowledge. Finally, evaluation of the discovered KS requires comparing with the ground truth KS – which is absent from publicly available datasets [10].

While building our own dataset would only ensure the validity of the results on a specific domain application, we chose in this work to use synthetic data generated by a general student model, as it also makes it possible to extensively experiment with different tutoring systems interacting with the student model. We experimentally verify in section IV that exercise sequences agnostic of domain knowledge provide richer data in order to discover KS.

We generate learner trajectories using a complex student model incorporating KS, variations of exercise difficulty and learner profiles, and learner forgetting. We evaluate our KS discovery algorithm by: (1) measuring correctness of the KS recovered from learner trajectories compared to the ground truth KS, (2) a comparative evaluation with both a KS-based tutoring model exploiting the discovered graphs, and KT-based tutoring models. We use an adaptation of the ZPDES algorithm [5] to recommend exercises based on the discovered KS.

In summary, this article presents the following contributions:

- We introduce a KT model that incorporates the KS as a learnable parameter, that we name Prerequisite Knowledge Tracing (PKT). With the given KS-exercise graph and by training PKT on learner trajectories data, we can uncover the underlying KS.
- We quantitatively evaluate our KS discovery method in comparison with other approaches from the literature.
- We qualitatively evaluate the uncovered KS as a basis for estimating and updating the ZPD [4] using a tutoring model adapted from the ZPDES algorithm [5].

II. Related Work

A. Intelligent Tutoring Systems

Intelligent Tutoring Systems (ITS) can be considered as recommendation systems for educational content. The recommendation task has been formulated in the Partially Observable Markov Decision Process (POMDP) framework [11]–[13], as a Multi-Armed Bandit (MAB) problem [5], [14]. We refer to [15] for a review of reinforcement learning based approaches for tutoring.

While domain knowledge and student models can be difficult to acquire, they have the potential to significantly improve recommendation of pedagogical contents. For instance, [16] showed that tutoring models exploiting domain knowledge could yield better learner progress than expert sequences of pedagogical contents. We can distinguish two situations, where the tutoring model (i) exploits the KS (for instance [12]) ; and (ii) exploits a given or learned student model (for instance with POMDP planning [11]). Our approach belongs to the first category, using a KS estimated by our algorithm.

B. Knowledge Tracing

KT corresponds to the task of accurately estimating the evolving knowledge states of learners on the different KCs, based on their history of exercises and answers. Bayesian Knowledge Tracing (BKT) [8] pioneered this field of research by proposing a KT model that tracks students' knowledge states using hidden Markov models. The knowledge states are represented by a vector of binary values, each value indicating whether the corresponding KC is acquired by the learner. Other methods use continuous values [17] or vector representations [18] to model the proficiency on each KC.

More recent KT research focuses on deep learning methods, pioneered by the Deep Knowledge Tracing (DKT) model [19]. DKT uses Long Short-Term Memory networks (LSTMs) [20] to model the temporal evolution of students' knowledge states encoded in the LSTM hidden states. Following the success of DKT, many deep learning architectures specialised on KT have been proposed, using key-value memories [21], graph neural networks [22], [23] or Transformers [24]. We refer to [10] for a recent and more complete review of the KT literature.

Some works have also studied ways to exploit given relations between KCs in order to improve the accuracy of KT models. The Graph Knowledge Tracing (GKT) model [22] updates the knowledge states of the KCs of the completed exercise, as well as the knowledge states for parent and children KCs in a given relation graph. The Structure-based Knowledge Tracing (SKT) model [18] conditions the knowledge state updates on two distinct graphs. An undirected graph represents similarity relations between KCs, while a Directed Acyclic Graph (DAG) represents prerequisite relations between KCs. The model presented in [9] exploits known prerequisite relations as soft constraints on the ordering of the estimated learner level on each KC, embedded in the loss function of the model. These models show that integrating knowledge on the KS can improve the accuracy of KT models. As such, improving KS discovery methods can lead to better KT.

C. Knowledge Structure with Prerequisite Structure Learning

As such, some works also propose strategies to uncover the prerequisite structure in case it is not provided by experts of the educational domain. Some methods [19], [25] build heuristics in order to extract the implicit information about the KS from trained DKT models. In [26], the authors compare different indices that can be used to induce relations between KCs based on learners data. Among the indices they experiment with, only the adjusted Kappa is asymmetric and can be used to build a DAG. [27] propose to frame KS discovery as a problem of causal structure discovery, that they solve using statistical independence tests, with simplifying assumptions.

In [22], several methods are proposed for KS discovery. A first method relies on observed statistics in the exercise sequences taken by the learners, based on the observation that the available data usually comes from online educational platforms where exercises are taken following an expert-given curriculum. If exercises are given randomly to learners, this method becomes unable to uncover the prerequisite relations. The second method proposes to learn the prerequisite relations, represented by an adjacency matrix, together with the other parameters of the neural network. However, because the graph in the GKT model [22] is not constrained to represent prerequisite relations, it is not straightforward to induce the prerequisite structure from the learned graph. Similar to GKT, our KS discovery method adopts a KT-based approach, learning the prerequisite graph together with the other parameters of a KT model. We design the KT model so that the learnable graph explicitly corresponds to prerequisite relations, which allows us to outperform other KS discovery methods.

III. Methods

Here we describe the proposed method for prerequisite structure learning, as well as the tutoring model used to evaluate the discovered KS. We provide the code for all our implementations in a git repository[1].

[1]https://github.com/sino7/KS-discovery-for-ITS

A. KS Discovery

We describe Prerequisite Knowledge Tracing (PKT), a straightforward KT model conditioned on the prerequisite relations between the KCs. The prerequisite relations are represented by an adjacency matrix M such that $M_{ij} = 1$ if the i-th KC is a prerequisite for the j-th KC. With this adjacency matrix, the model predicts the probability of success of a student s on an exercise e as:

$$p_{e,s,t} = p_g + (1 - p_s - p_g)\sigma\Big(softmin_{k \in \mathcal{P}_e}\big(\lambda_{s,k,t}\big) - \delta_e\Big) \quad (1)$$

where p_g and p_s respectively denote the *guess* and *slip* probabilities, i.e. the probability of completing (resp. failing) the exercise without the required skills (resp. while mastering the required skills). σ denotes the sigmoid function, $\lambda_{s,k,t}$ denotes skill level of learner s on the k-th KC at time t, and δ_e denotes the exercise e difficulty level. Finally, $\mathcal{P}_e$ denotes the set of KCs corresponding to the exercise e, and parent KCs in the prerequisite graph M. We can see that the estimated probability of success depends on the adjacency matrix M. Thanks to this *softmin* aggregation, if the learner skill on one of the prerequisite KCs is insufficient, the probability of completing the exercise will be low. $\lambda_{s,k,t}$ is estimated as:

$$\lambda_{s,k,t} = \mu_{s,k} + \alpha_s S_{s,k,t} + \beta_s F_{s,k,t} \quad (2)$$

where $\mu_{s,k}$ denotes the initial skill level of the learner s on the k-th KC, $S_{s,k,t}$ denotes the number of times the learner s has completed with success an exercise related to the k-th KC before time t, and $F_{s,k,t}$ denotes the number of times the learner s has failed an exercise related to the k-th KC before time t. α_s and β_s are scalar parameters of the model, learned for each individual student s, that can be interpreted as the amount of skill gained respectively by succeeding and failing an exercise.

The parameters $\{p_g, p_s, \delta_e, \mu_{s,k}, \alpha_s, \beta_s, M\}$ are optimised in order to minimise the binary cross entropy loss between the predicted probability of success $p_{e,s,t}$ and the ground truth learner response. We use L2 regularisation on the parameters α_s, β_s and $\mu_{s,k}$ to prevent overfitting. Finally, we also regularise the adjacency matrix using the L1 norm to encourage sparse prerequisite graphs.

The proposed model implements the idea that a learner does not successfully complete exercises related to a certain KC if she has not first mastered the prerequisite KCs.

B. Tutoring model

The tutoring model we present is an adaptation of the ZPDES algorithm [5], that we label ZPDES-KS. ZPDES uses as input a progression graph on exercises to structure the curricula of learners. It builds and updates a personalised pool of exercises for each learner, representing their individual ZPD. As such, this ZPD should always contain exercises on which the learner has the potential to progress, i.e. exercises that are neither too easy nor too difficult. We adapt ZPDES to use a prerequisite graph on KCs. An illustration of the ZPDES-KS algorithm on a toy example is provided in Fig. 2. The ZPD

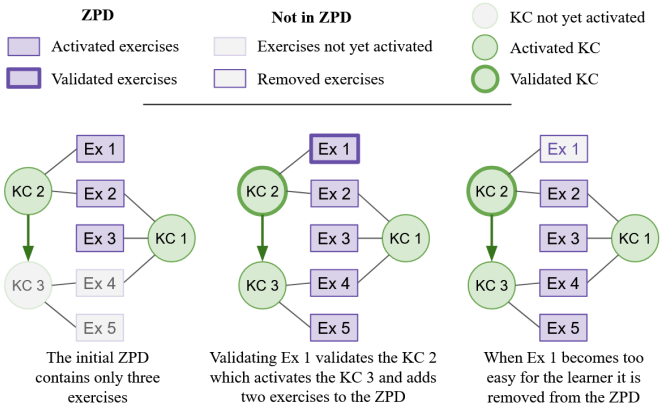

Fig. 2. Illustration of the ZPDES-KS method applied on a toy example. Learner progress can unlock new KCs and add new exercises to the ZPD, and can also deactivate exercises when success becomes too frequent.

expands with new exercises when new KCs are unlocked, and exercises are removed when they become too easy. We propose the following rules to build and update the ZPD:

1) The ZPD is initialised with exercises for which all the KCs have no prerequisite in the prerequisite graph.
2) If the empirical success on one exercise reaches a threshold value ϵ_v, we consider the exercise to be validated.
3) A KC is validated if it has at least one validated exercise.
4) A KC is activated if all its prerequisite (parent) KCs are validated.
5) An exercise is added if all its KCs are activated.
6) If the empirical success on one exercise reaches a threshold value ϵ_r, it is removed from the ZPD.

The empirical success level $\hat{S}_{s,e}$ of the learner s on the exercise e is initialised at zero, and updated whenever the exercise e is taken by the learner, using the update rule:

$$\hat{S}_{s,e} \leftarrow (1-\theta_S)\hat{S}_{s,e} + \theta_S S_{s,t} \tag{3}$$

where $S_{s,t} = 1$ if the learner successfully completes the exercise, and $S_{s,t} = 0$ otherwise; and θ_S is an update rate.

As in ZPDES, exercises are recommended to the learner based on a multi-armed bandit algorithm, using as reward function the empirical progress $\hat{P}_{s,e}$ of the learner s on the exercise e. $\hat{P}_{s,e}$ is initialised at 0, and updated with:

$$\hat{P}_{s,e} \leftarrow (1-\theta_P)\hat{P}_{s,e} + \theta_P(S_{s,t} - \hat{S}_{s,e}) \tag{4}$$

where θ_P is an update rate, and $(S_{s,t} - \hat{S}_{s,e})$ can be interpreted as a noisy measurement of learning progress. For instance, if $\hat{S}_{s,e}$ is already close to 1, then succeeding on the exercise ($S_{s,t} = 1$) should not convey a large reward.

As focusing teaching on activities that provide more learning progress can act as a motivational mechanism [28], we encourage the recommendation of exercises within the ZPD by adding a reward r_{ZPD} to those exercises. The recommendation algorithm then samples the exercise to recommend using a softmax distribution based on this reward function.

IV. Experiments

We evaluate our approach in comparison with other KS discovery methods. First, KS discovery methods are evaluated with regard to their ability to properly recover the ground truth KS. Second, all methods are evaluated on their ability to be exploited by the ZPDES-KS tutoring model.

A. Synthetic data generation

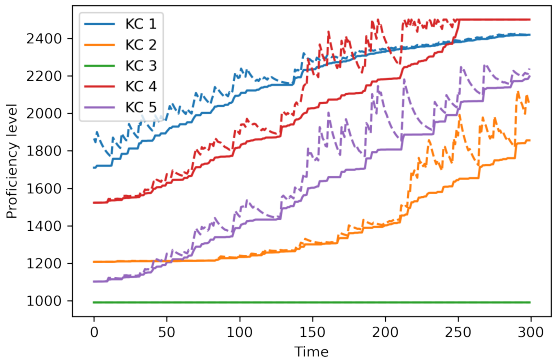

Fig. 3. Example of simulated learner trajectory. KC 3 is never practiced.

The student model used to generate synthetic data implements several mechanisms to account for prerequisite structure, variations of exercise difficulty and learner profiles, and learner forgetting. More details on our git repository[1].

Learner forgetting is implemented by representing the learner proficiency both on the long-term and short-term. While short-term proficiency increases quickly when the KC is practiced, it is also forgotten at an exponential rate when the KC is not practiced, as proposed as forgetting curve in [29]. Concurrently, progressing on the long-term proficiency requires spaced practice of the KC. This is ensured by dividing the long-term learning by a factor corresponding to the gap between short-term and long-term proficiency. If the short-term proficiency is not given time to decrease back to the long-term proficiency level, this hinders long-term learning.

The prerequisite relation graph (KS), and KC-exercise graph are both randomly sampled. The KS is sampled using the Erdős–Rényi model with probability of edge $p = 2/K$ where K is the number of KCs. We only keep the upper triangular edges to make the graph directed, and then randomly shuffle the nodes. Finally, we also clean the graph of possible shortcuts. For instance if we have $i \to j \to k$ as well as $i \to k$, the prerequisite relation $i \to k$ is useless and we remove it. The KC-exercise graph is built in order to ensure that each exercise has at least one related KC, that each KC has at least one related exercise, and that an exercise cannot be related to two KCs if there exists a path in the KS connecting the two. We generate 10 random simulators using this model, with 10 KCs and 30 exercises. All the simulators use different KS and KC-exercise graphs. We sample 400 learner profiles and generate learner trajectories of length T=300.

Fig. 3 represents an example trajectory simulated with our student model, implementing the KS represented in Fig. 1. Dashed lines represent the short-term proficiency and full lines the long-term proficiency. We observe that the learner only improves on the KC 5 once she is proficient enough on the KC 4, and on KC 2 once she is proficient enough on the KC 5. We also see that significant increases in long-term proficiency only occur once the short-term effect of previous practice vanishes.

B. Prerequisite Structure Learning

We experiment with different methods for finding the KS from the learners trajectories:

- **PKT**: Using the proposed PKT model.
- **KI** : Kappa Index as proposed in [26]
- **DKT-H** : Using the heuristic method proposed in [25] on top of a trained DKT model [19]
- **SKT-H** : Using the heuristic method proposed in [25] on top of a trained SKT model [18]
- **SKT-L** : Learning the adjacency matrix during the training of the SKT model [18]

Note that for the SKT model, we use the partial model representing only directed relations between the KCs, and not the full model that also implements undirected relations, since our primary goal is to discover the prerequisite relations.

Each method provides a matrix M where M_{ij} is the estimated strength of the prerequisite relation $i \rightarrow j$. We clean all the matrices to remove potential cycles, by ordering all the values in increasing order, and setting to 0 the values M_{ij} that participate in cycles in the graph. After this step, we find for each method a threshold value θ maximizing the average f1-score between the adjacency matrix A defined by $\left\{A_{ij} = (M_{ij} > \theta), \forall ij\right\}$ and the ground truth graph A^*.

With all 10 synthetic data generators, we sample 400 learner profiles and generate learner trajectories of length T=300, with two different scenarios. In the first scenario, exercises provided to the learners are sampled randomly. In the second scenario, exercises sequences follow a predetermined order designed using half of the prerequisite relations of the ground truth KS.

TABLE I
EVALUATION OF THE KS DISCOVERY METHODS. WE REPORT F1-SCORES FOR EACH METHOD, WITH DATA COLLECTED (I) WITH RANDOM EXERCISE SEQUENCES, AND (II) WITH PARTIAL INFORMATION OF THE KS.

	PKT	KI	DKT-H	SKT-H	SKT-L
Random seq.	**0.46**	0.27	0.19	0.22	0.37
Informed seq.	0.17	**0.23**	0.18	**0.23**	0.17

Table I reports the f1-scores for each method. We use binary classification metrics for evaluation, where the two classes correspond to the presence or absence of a directed edge in the graph. We can observe that the two methods that learn the adjacency matrix with a KT model perform better, PKT outperforming the more complex SKT model. Besides, we observe that using random exercise sequences leads to better scores. This validates the first argument given in introduction to justify our choice of not using publicly available datasets: the biased sequencing of exercises in those datasets can hinder KS discovery.

C. Exploiting prerequisite relations for recommendation

Using the ZPDES-KS algorithm, we can build tutoring models from the discovered KS. We display in Fig. 4 example results obtained with the ZPDES-KS tutoring model using the prerequisite relations found with the PKT method. Initially, only four the KCs are activated. The learners quickly progress

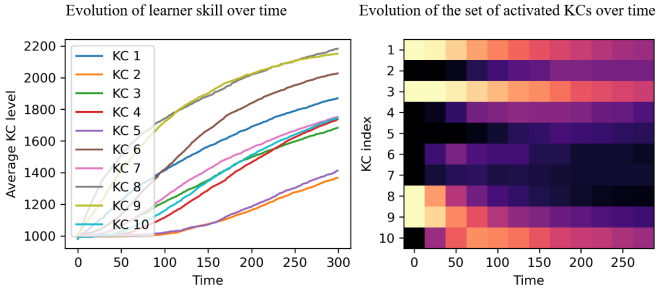

Fig. 4. Results with the ZPDES-KS algorithm on one of the datasets, with the KS discovered by the PKT method. Left: learner skills over time, averaged on 300 learners. Right: activated KCs in the ZPDES-KS algorithm over time, averaged on 300 learners (clear=activated).

on these KCs, which unlocks new KCs and adds new exercises in the ZPD. Once a threshold of empirical success level is reached, the KCs are deactivated.

We evaluate all the KS discovery methods according to their corresponding tutoring system, by measuring the average proficiency on all the KCs for 300 simulated learners practicing for T=300 iterations. The learners start with an average proficiency level of 1000 and progress at different paces according to the taken exercises. We report in table II the average proficiency level throughout practice, and the average proficiency level at the end of practice. We also provide the results obtained when using the ground truth KS (labeled GT) with ZPDES-KS, as well as with random exercise sequences.

To further validate our methods, we compare the different KS-based tutoring models with a KT-based tutoring model, that we call Model-Based Tutoring (MBT). This tutoring model aims at maximising the expected progress of the learner, estimated using a given KT model. At each time step, for each exercise, the algorithm simulates the evolution of the knowledge state of the learner, and estimates the resulting progress on each KC. For each exercise, a score is computed as the expected progress averaged on the KCs. We then sample the exercise to recommend using a softmax distribution computed on these scores. We observe that given the ground truth KS, ZPDES-KS outperforms all other approaches, despite not exploiting student models. Among the methods that automatically discover the KS, the proposed PKT approach significantly outperforms the baselines, while still remaining sub-optimal compared to the ground truth graph. Except for the SKT, the KT models do not provide better tutoring than random recommendations.

Overall, these results seem to indicate that domain knowledge in the form of prerequisite relations can be well discovered and exploited by tutoring models implementing the concept of ZPD. Interestingly, we obtain better tutoring strategies using only prerequisite relations than with trained student models, that could in principle encode more information such as exercise difficulty and student profiles.

V. CONCLUSION

We have investigated the use of machine learning techniques in order to recover KS from student trajectories, without relying on domain knowledge given by experts. The proposed

TABLE II
EVALUATION OF THE TUTORING MODELS. BEST VALUES ARE IN BOLD. BEST VALUES WITHOUT TAKING INTO ACCOUNT THE ZPDES-KS TUTORING WITH GROUND TRUTH PREREQUISITE GRAPH (GT) ARE UNDERLINED.

Tutoring model	Random	ZPDES-KS						MBT		
Graph / KT model		GT	PKT	KI	DKT-H	SKT-H	SKT-L	PKT	DKT	SKT
Average level	1414	**1629**	<u>1528</u>	1480	1402	1406	1450	1412	1411	1425
Final level	1737	**2034**	<u>1894</u>	1826	1713	1727	1788	1738	1700	1796

KS discovery method is based on a straightforward KT model conditioned on the KS, that we call PKT. The KS is learned together with the other parameters of the PKT model. The uncovered prerequisite relations can be used to deduce rules to estimate the ZPD of a learner interacting with the ITS. By recommending exercises within this ZPD, the tutoring model can maximise learning progress while keeping the learner engaged in her practice by stimulating her intrinsic motivation.

We have evaluated the accuracy of our KS discovery method, and shown that the PKT method outperforms other approaches from the literature. While the discovered graphs do not perfectly match the ground truth KS, we have also shown that tutoring models based on these graphs can perform better than tutoring models based on trained KT models.

Our results tend to show that KS-based recommendation can compete with KT-based approaches. They indicate the potential of more research on the topic of KS discovery, as this approach has been so far underexplored compared to the efforts invested in improving KT models. In future work, we would like to investigate the data efficiency of the KT models and KS discovery algorithms. Since accurate recommendation depends on the quality of the KT or KS, we may have a dilemma between exploration and exploitation. While recommending random exercises might improve the KT model or the KS estimation (exploration), the learner would benefit from optimised exercise sequencing to maximise her progress (exploitation). Besides, while the ZPDES-KS algorithm accounts for prerequisite relations, we would like to improve it in order to also account for cognitive models of forgetting, taking inspiration from [6].

REFERENCES

[1] T. Belpaeme, J. Kennedy, A. Ramachandran, B. Scassellati, and F. Tanaka, "Social robots for education: A review," *Science Robotics*, vol. 3, no. 21, 2018.

[2] E. L. Deci, *Intrinsic Motivation*. New York, US: Plenum Press, 1975.

[3] P.-Y. Oudeyer and F. Kaplan, "What is intrinsic motivation? a typology of computational approaches," *Frontiers in Neurorobotics*, vol. 1, no. 6, 2009.

[4] L. S. Vygotsky and M. Cole, *Mind in society: Development of higher psychological processes*. Harvard university press, 1978.

[5] B. Clement, D. Roy, P.-Y. Oudeyer, and M. Lopes, "Multi-Armed Bandits for Intelligent Tutoring Systems," *Journal of Educational Data Mining (JEDM)*, vol. 7, no. 2, pp. 20–48, Jun. 2015.

[6] T. Mu, S. Wang, E. Andersen, and E. Brunskill, "Combining adaptivity with progression ordering for intelligent tutoring systems," in *Proceedings of the Fifth Annual ACM Conference on Learning at Scale*, 2018, pp. 1–4.

[7] O. Vainas, O. Bar-Ilan, Y. Ben-David, R. Gilad-Bachrach, G. Lukin, M. Ronen, R. Shillo, and D. Sitton, "E-gotsky: sequencing content using the zone of proximal development," *arXiv:1904.12268*, 2019.

[8] A. T. Corbett and J. R. Anderson, "Knowledge tracing: Modeling the acquisition of procedural knowledge," *User modeling and user-adapted interaction*, vol. 4, pp. 253–278, 1994.

[9] P. Chen, Y. Lu, V. W. Zheng, and Y. Pian, "Prerequisite-driven deep knowledge tracing," in *2018 IEEE International Conference on Data Mining (ICDM)*. IEEE, 2018.

[10] G. Abdelrahman, Q. Wang, and B. Nunes, "Knowledge tracing: A survey," *ACM Computing Surveys*, vol. 55, no. 11, pp. 1–37, 2023.

[11] A. N. Rafferty, E. Brunskill, T. L. Griffiths, and P. Shafto, "Faster teaching via pomdp planning," *Cognitive science*, vol. 40, 2016.

[12] B. Clement, P.-Y. Oudeyer, and M. Lopes, "A comparison of automatic teaching strategies for heterogeneous student populations." *International Educational Data Mining Society*, 2016.

[13] S. Shen, M. S. Ausin, B. Mostafavi, and M. Chi, "Improving learning & reducing time: A constrained action-based reinforcement learning approach," in *Proceedings of the 26th conference on user modeling, adaptation and personalization*, 2018.

[14] A. Segal, Y. Ben David, J. J. Williams, K. Gal, and Y. Shalom, "Combining difficulty ranking with multi-armed bandits to sequence educational content," in *Artificial Intelligence in Education: 19th International Conference, AIED 2018, London, UK, June 27–30, 2018, Proceedings, Part II 19*. Springer, 2018, pp. 317–321.

[15] S. Doroudi, V. Aleven, and E. Brunskill, "Where's the reward?" *International Journal of Artificial Intelligence in Education*, vol. 29, no. 4, pp. 568–620, 2019.

[16] B. Clement, "Adaptive personalization of pedagogical sequences using machine learning," Ph.D. dissertation, Université de Bordeaux, 2018.

[17] F. B. Baker, *The basics of item response theory*. ERIC, 2001.

[18] S. Tong, Q. Liu, W. Huang, Z. Hunag, E. Chen, C. Liu, H. Ma, and S. Wang, "Structure-based knowledge tracing: An influence propagation view," in *2020 IEEE international conference on data mining (ICDM)*. IEEE, 2020, pp. 541–550.

[19] C. Piech, J. Bassen, J. Huang, S. Ganguli, M. Sahami, L. J. Guibas, and J. Sohl-Dickstein, "Deep knowledge tracing," *Advances in neural information processing systems*, vol. 28, 2015.

[20] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[21] J. Zhang, X. Shi, I. King, and D.-Y. Yeung, "Dynamic key-value memory networks for knowledge tracing," in *Proceedings of the 26th international conference on World Wide Web*, 2017, pp. 765–774.

[22] H. Nakagawa, Y. Iwasawa, and Y. Matsuo, "Graph-based knowledge tracing: modeling student proficiency using graph neural network," in *IEEE/WIC/ACM International Conference on Web Intelligence*, 2019.

[23] Y. Yang, J. Shen, Y. Qu, Y. Liu, K. Wang, Y. Zhu, W. Zhang, and Y. Yu, "Gikt: a graph-based interaction model for knowledge tracing," in *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2020, Ghent, Belgium, September 14–18, 2020, Proceedings, Part I*. Springer, 2021.

[24] D. Shin, Y. Shim, H. Yu, S. Lee, B. Kim, and Y. Choi, "Saint+: Integrating temporal features for ednet correctness prediction," in *LAK21: 11th International Learning Analytics and Knowledge Conference*, 2021.

[25] J. Zhang and I. King, "Topological order discovery via deep knowledge tracing," in *Neural Information Processing: 23rd International Conference, ICONIP 2016, Kyoto, Japan, October 16–21, 2016, Proceedings, Part IV 23*. Springer, 2016.

[26] W. Gan, Y. Sun, and Y. Sun, "Knowledge structure enhanced graph representation learning model for attentive knowledge tracing," *International Journal of Intelligent Systems*, vol. 37, pp. 2012–2045, 2022.

[27] R. Scheines, E. Silver, and I. M. Goldin, "Discovering prerequisite relationships among knowledge components." in *Edm*, 2014.

[28] P.-Y. Oudeyer, J. Gottlieb, and M. Lopes, "Intrinsic motivation, curiosity, and learning: Theory and applications in educational technologies," in *Progress in brain research*. Elsevier, 2016, vol. 229, pp. 257–284.

[29] P. Wozniak, E. Gorzelańczyk, and J. Murakowski, "Two components of long-term memory," *Acta neurobiologiae experimentalis*, vol. 55, pp. 301–5, 02 1995.