Science Futures
2026, Vol. 2, No. 2, pp. 118–134
https://doi.org/10.11648/j.scif.20260202.13

Research Article

Learning State Prediction and Learning Path Generation Using Hidden Markov Models

Song-Hwan Kwon[1, *], Jong-Nam Rim[1], Yun-Ho Ko[1], Yun-Sok Ham[2], Ha-Gyong Kim[1]

[1]Department of Information Science, University of Science, Pyongyang, Democratic People's Republic of Korea
[2]Institute of Information Technology, University of Science, Pyongyang, Democratic People's Republic of Korea

Abstract

The main mission of the intelligent tutoring system is to estimate and predict the students' learning status accurately and to generate a learning path that is inherent to the students. There are many learning state estimation and prediction methods, and there are also many methods to generate learning paths using the results. We propose a method to predict the learning state of a student and generate a learning path using the Hidden Markov Model, a probabilistic reasoning over time. We propose and illustrate in detail how to formalize the Hidden Markov Model using an extended knowledge graph for learning state prediction, how to collect the evidence variable values of the Hidden Markov Model in intelligent tutoring systems, how to predict the learning state using smoothing algorithms, and how to generate the stochastic learning path. The experiment was conducted to compare students using an intelligent tutoring system with those who did not. In addition, a comparison experiment of a method using Hidden Markov model with a multi-dimensional linear prediction model such as PFM and AFM was also carried out. The results showed that the average performance of students using the intelligent tutoring system was much higher than that of students without the intelligent tutoring system. In addition, the experimental results showed that the average performance of students using Hidden Markov models was higher than that of students using other models.

Keywords

HMM, Learning State Prediction & Estimation, Learning Path Generation, Probabilistic Reasoning over Time

1. Introduction

Intelligent tutoring system is an artificial intelligence system that provides a suitable learning content and learning environment based on the evaluation of individual students' knowledge level, learning ability and learning style [22]. In other words, intelligent tutoring system is an educational software that has artificial intelligence elements. These software tracks the student's learning process, implements feedback control, and gives hints to the learning process. In the literature, it is stated that the intelligent tutoring system is generally composed of three models: Learner Model, Tutor Model and Domain Model, and the intelligent tutoring system (ITS) model is formalized as follows.

ITS = (LM, TM, DM)

Here, LM is the learner model, TM is the tutor model, and DM is the domain model. The LM is a model with the ability to evaluate and analyze the state of the learner learning

*Corresponding author: ksh1974@star-co.net.kp (Song-Hwan Kwon)

Received: 21 September 2025; **Accepted:** 23 October 2025; **Published:** 20 January 2026

Copyright: © The Author(s), 2026. Published by Science Publishing Group. This is an **Open Access** article, distributed under the terms of the Creative Commons Attribution 4.0 License (http://creativecommons.org/licenses/by/4.0/), which permits unrestricted use, distribution and reproduction in any medium, provided the original work is properly cited.

through an intelligent tutoring system. The TM is a model with a learner-specific learning-guidance function based on the tutor's experience and the learners' state analysis reflected in the LM. The domain model DM is a model with the function of controlling the interaction between the learner model, the teacher model and the teaching knowledge base.

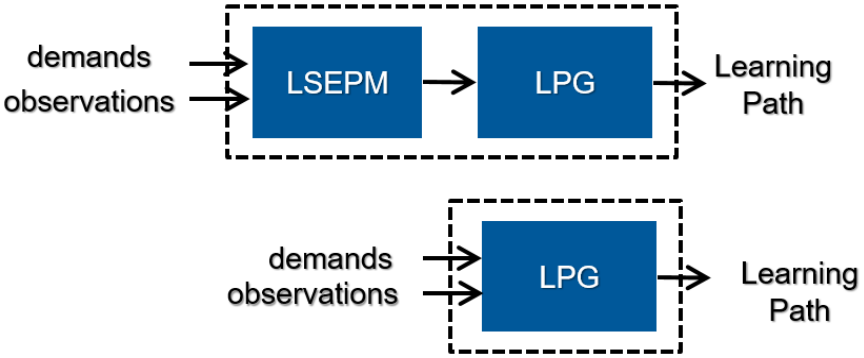

Figure 1. *Different Tutoring System Architecture.*

In MURTAZ [1], the learner model is called the student model and the instructor model is also called the teacher model.

In addition, the literature from MURTAZ and Ouyang [1, 2] uses the terms learner model, student model, learning state prediction model, learning performance prediction model, KT (Knowledge Tracing), IRT (item response theory), LFA (Learning Factor Analysis), and classification of learner's level as synonyms. We also find that the terms instructor model, teacher model, learning process generation model, learning process generator, learning path generation model, and learning path generator are also used as synonyms.

Learning state estimation and prediction models are models that summarize the past history of student learning activities, such as understanding, cognition, and response outcomes, and that, when input is given as evidence, estimate the current learning state (cognitive level) of the student and the current learning activity, and output the learning state prediction value of the next concept. Thus, a learning state prediction model is a model that can quantify the degree of student perception. In Ouyang [2], the goal of knowledge tracking is to model the student's knowledge state from the student's past learning interactions and predict the student's learning performance in the future. The remainder of our work, which is a previous work, mainly uses the term Learning State Prediction Model and Learning Path Generator.

The purpose of the intelligent tutoring system is to generate an efficient learning path for the individual student. Previous studies have proposed different intelligent instructor architectures, which include a framework that separates the learning state prediction model from the learning path generator, and a framework that integrates the learning state prediction model and the learning path generator (Figure 1).

After predicting the learning state, there are methods to generate the learning path using additional algorithms, and to simultaneously perform the learning state prediction and learning path generation in the model. Figure 1 shows the structure in which the learning state prediction model and learning path generator is exists separately, hybrid model in which the two processes proceed simultaneously in one model. We consider only the ways in which the learner's state estimation & prediction model and learning path generation are separated individually and generate learning path by taking the resulting value of the learner's state prediction model as input.

2. Related Previous Work

In Ouyang [2], the personalized learning system must track the level of learners' understanding and the concepts not understood by the learner must be clearly identified, and the task of tracking learners' knowledge and understanding over time is called knowledge tracing (KT). The knowledge-tracking method can be classified into traditional knowledge-tracking method, deep knowledge-tracking method and graph-based method.

In the Markov assumption, the current state is bounded by the past states. In the traditional knowledge-tracking model, Bayesian Knowledge Tracing (BKT), the main focus is to extract text features from queries and determine their relationships with the key concepts through knowledge tracking [3]. These models rely on order modeling, where the learner's previous questions and the interaction of answers are used to predict the probability of answering a question correctly. In the Bayesian knowledge-tracking model, only the learning state is estimated, and the recommendation engine with this value as input provides the learning path [5]. These models rely on sequential modeling, where learner's prior query and response interaction are used to predict the probability of correctly answering a query.

In Deep Knowledge Tracing (DKT), sequential models based on deep learning are used to achieve the goal of

knowledge tracking. Here, Recurrent Neural Networks (RNN), Long Short Term Memory networks (LSTM), and Gated Recurrent Units (GRU) can model the continuous data up to more previous states compared to BKT [6, 7]. The bi-directional LSTM models have also been used to understand key concepts through DKT [4].

In graph-based knowledge tracking, multiple association structures are used to find similarities between different knowledge concepts, dependencies between different knowledge concepts, and correspondences between queries and knowledge concepts [8]. Graph Neural Networks (GNNs) perform knowledge tracking using knowledge graph representations. Such models include graph-based knowledge tracking (GKT) model [9], Graph-based Interaction knowledge tracking (GIKT) [10], Bi-Graph contrast learning-based knowledge tracking [11], and Structure-based knowledge tracking (SKT) [12].

The Hidden Markov Model is defined as the relationship between the hidden states and the observed values using the transition probability set and the emission probability set. Studies have been conducted to estimate and predict the learning status of a student using a hidden Markov model to generate a learning path.

In Pardos et al. [14], the results of predicting the probability that a student will answer correctly and the probability of not answering at each learning stage using the Hidden Markov Model and random forest are analyzed. Big data (30 million training data and 1.2 million test data) were used.

In Masun et al. [15-17], the authors reported that in an Adaptive and Intelligent Web-Based Educational System (AIWBES), students used a Hidden Markov Model to predict the next concepts to choose. In this study, the prediction process was implemented in three phases: initialization phase, adjustment phase and prediction phase. These studies reported that the prediction accuracy reached 92% when the Hidden Markov Model was used for predicting student performance in AIWBES.

In Xuejing [18, 19], an artificial emotion model based on Hidden Markov Model is proposed. In this model, the teacher's mood and emotions were created probabilistically, and the students' cognitive results on facial expressions were considered as input signals of this artificial emotion model. We also used Bayesian networks to evaluate the student's learning status.

In Boyer et al. [20, 21], we propose an automatic creation method for corpus-based tutorial dialogue management models using hidden Markov models.

Although many studies have been conducted to predict student learning using Hidden Markov Models or Bayesian models and many models have been developed, we still consider the need for developing student models based on Hidden Markov Models from differences in country characteristics, region characteristics, subject characteristics, development goals and use.

3. Extended Knowledge Graph and Editor

3.1. Constructing an Extended Knowledge Graph

In our study, we define an extended knowledge graph and use it for knowledge representation. A knowledge graph is a kind of ontology, and it is a graph whose relation between nodes only reflects a forward-backward relation or a weight value or a semantic relation with a forward-backward relation. Many literatures represent nodes of knowledge graphs as vertices and arc of knowledge graphs as edges. We understand that the term node and edge represent the relationship between nodes as a two-dimensional view of the knowledge graph, and the term vertex and edge represents the relationship as a multidimensional view. In our future work, we use the terms node and vertex simultaneously to make a further description. A knowledge graph is defined as a weighted (or weightless) directed graph between knowledge concepts in terms of memory principles, in terms of the ordering of scientific content, or in terms of academic and pedagogical aspects. The knowledge concept is the central kernel of knowledge (or memory unit), and in intelligent tutoring systems, it includes slides, comment texts, corresponding video data, questions and answers to the concept. The vertex of the knowledge graph represents the knowledge concept, the edges represents the weighted order relation, i.e., the order relation with the relation strength value.

In our paper, we consider a knowledge graph whose relations between nodes have only a forward and backward order relation. In addition, many approaches have been proposed for the extension and decomposition of knowledge graphs, but our paper proposes an extended knowledge graph that is easily understandable by common teachers and applicable to their subjects.

Previous studies have defined many different types of knowledge graphs [9-12]. We define the knowledge graph as follows.

$G = (V, E)$, V is the set of vertices, E is the set of edges (relations), i.e., $V=\{v_i|\ i=1,2,...,n\}$ is the set of possible concepts. When $E=\{e_j|e_j=<v_{j1},v_{j2}>, v_{j1},v_{j2} \in V, j=1,2,\ldots,m\}$ is a set of possible directed edges constructed by concept-association relations, the graph $G = (V, E)$ is called a learning path graph. A vertex in a knowledge graph reflects the knowledge concepts present in the subject and an edge the order relationships between those knowledge concepts. The knowledge graph must be acyclic. If we have to learn concept v_1 for two concepts v_1 and v_2, then we have to learn concept v_2, we denote it by the directed edge $e=<v_1, v_2>$.

We intend to extend and refine the knowledge graph for the construction of intelligent tutoring systems. We use the knowledge graph starting from the "subject graph" and extending it to the "hyperconcept graph", "basic concept

graph", and "subconcept and help_concept graph". These graphs have an edge with a vertex, which indicates an educational or cognitive sequence. The definition of the updated knowledge graph is as follows.

3.1.1. Subject Graph

We define the subject graph as $G_{sub} = (V_{subi}, E_{subi})$. The index i is an integer index symbol, $i \in [1, N_{sub}]$, N_{sub} is the total number of vertices of the subject graph, Vsubi represents one vertex of the subject graph, and E_{subi} represents one relation. A vertex of a subject graph defines the set of the largest concept blocks in the subject and the relationship defines the sequential relationship between the vertices. Figure 2 shows an example of a subject graph.

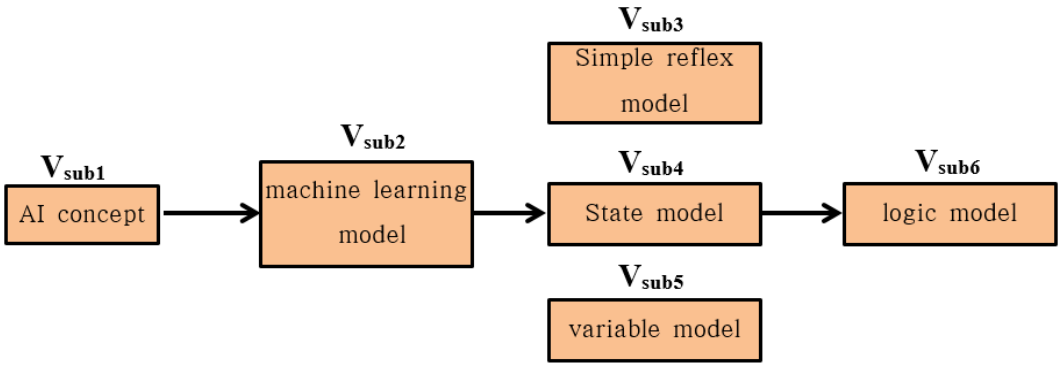

Figure 2. Subject graph example.

3.1.2. Hyper-concept Graphs

A hyperconcept graph is further constructed by decomposing the vertices of a subject graph, which is defined as follows:

$G_{sub} = (V_{subi}, E_{subi}) \approx G_{hyper} = (V_{hyperj}, E_{hyperj}) = (V_{subi}^{hyperl}, E_{subi}^{hyperl})$, $i \in [1, N_{sub}]$, $j \in [1, N_{hyper}]$, $l \in [1, N_{subi}]$ i, j, l are the total number of vertices of the hyperconcept graph, and Nsubi is the number of child vertices belonging to the ith node of the s subject graph. The symbol ≈ indicates that the meaning of the two graphs is equivalent to each other. The following figures show the hyper-concept graph.

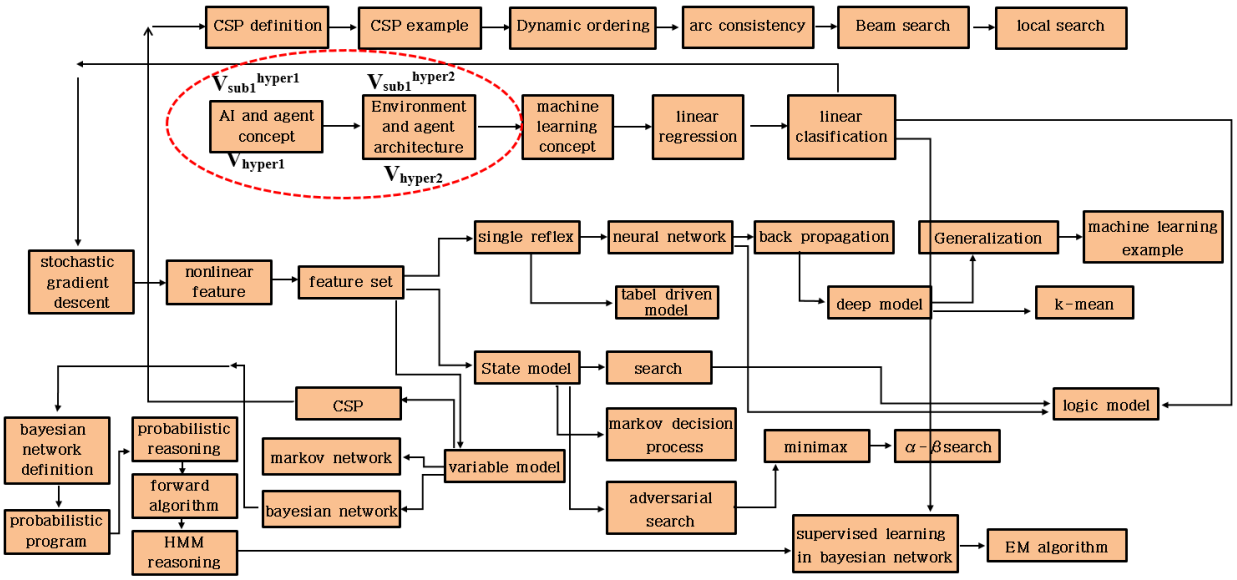

Figure 3. Hyper-concept graphs example (1).

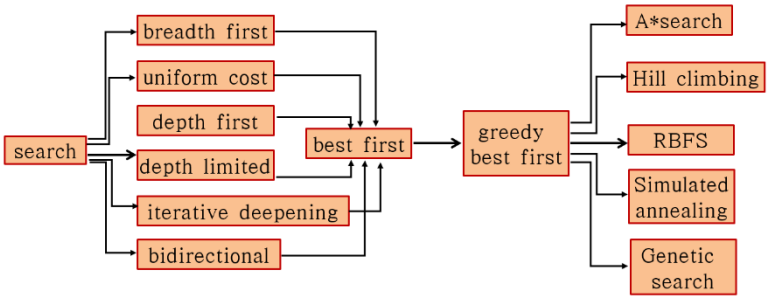

Figure 4. Hyper-concept graphs example (2).

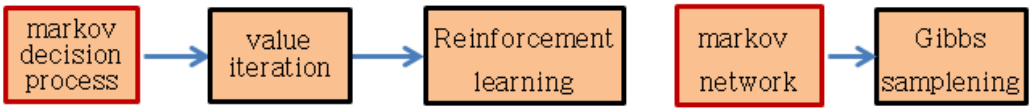

Figure 5. Hyper-concept graphs example (3).

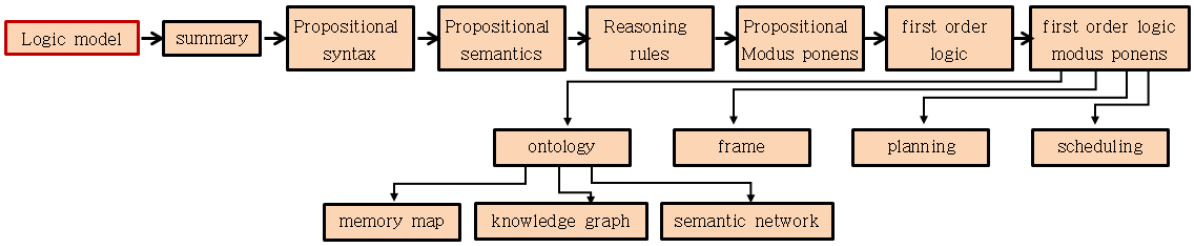

Figure 6. Hyper-concept graphs example (4).

3.1.3. Basic Concept Graphs

The basic concept graph is further decomposed into vertices of the hyperconcept graph, which is defined as follows.

$G_{sub} = (V_{subi}, E_{subi}) \approx G_{hyper} = (V_{hyperj}, E_{hyperj}) = (V_{subi}^{hyperl}, E_{subi}^{superl}) \approx G_{basic} = (V_{basick}, E_{basick}) = (V_{subi}^{superl,basicm}, E_{subi}^{superl,basicm})$, $i \in [1, N_{sub}]$, $j \in [1, N_{hyper}]$, $l \in [1, N_{subi}]$, $k \in [1, N_{basic}]$, $m \in [1, N_{subi,\ hyperl}]$

Let i, j, k, m be integers, N_{basic} be the total number of vertices of the basic concept graph, and $N_{subi,hyperl}$ are the number of children of vertices V_{subi}^{hyperl} of the hyperconcept graph. The basic teaching unit of tutoring is the basic concept. Eventually, in the intelligent tutoring system, all the instructional nodes of tutoring are composed of basic concepts. The list of contents displayed on the left-most part of the user interface of the developing intelligent tutoring system is the basic concepts in the subject, not the chapters and sections in the subject, as in the group teaching (Figure 8). Figure 7 shows an example of a basic concept graph corresponding to the nodes (V_{sub1}^{super1}) and (V_{sub1}^{super2}) (represented by a red circle) of the hyper-concept graph in Figure 3.

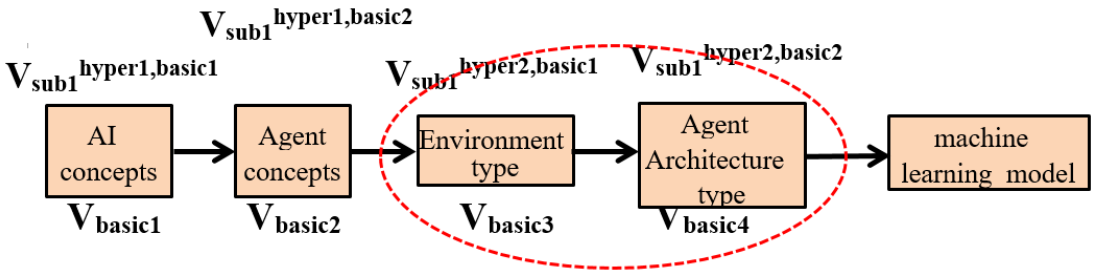

Figure 7. basic concept graph example.

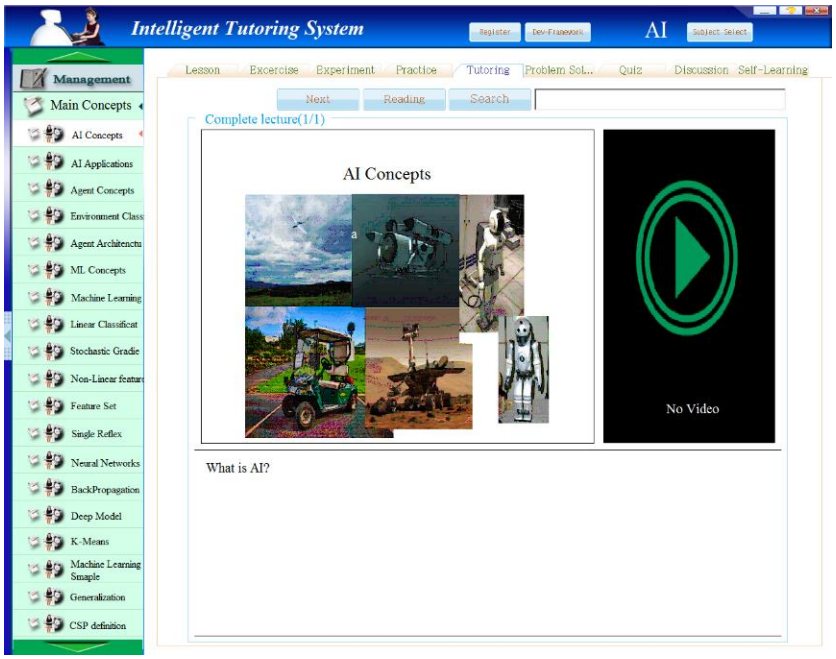

Figure 8. In tutoring system, the choice is a basic concept.

3.1.4. Subconcept and Help-concept Graphs

Figure 9 shows the sub-concepts and help-concepts between the basic concepts "Environmental Classification" and the basic concept "Active Structure Classification" in AI. This sub-concept and help-concept graph reflects the action between state and state, the result of inference of hidden Markov model, and the actual tutoring flow of intelligent teacher as tutor is implemented by this graph. The program uses this graph to generate a specific teaching flow for each student, i.e., an action that generates an appropriate teaching path for each student.

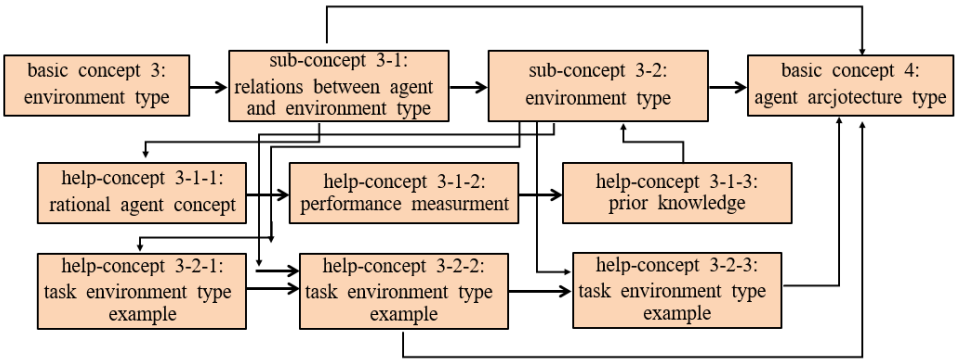

Figure 9. Subconcept and help-concept Graph Example.

3.2. Editor Development

Our team developed an editor as an intelligent tutoring system development framework. The editor's mission is to use teacher's instructional resources to develop a static instructional model or to assist in generating a dynamic and adaptive instructional model. This approach is supported by the construction of a collective pedagogical model, the construction of knowledge graphs, the construction of Hidden Markov Models, and the construction of Markov Decision Processes for tutoring.

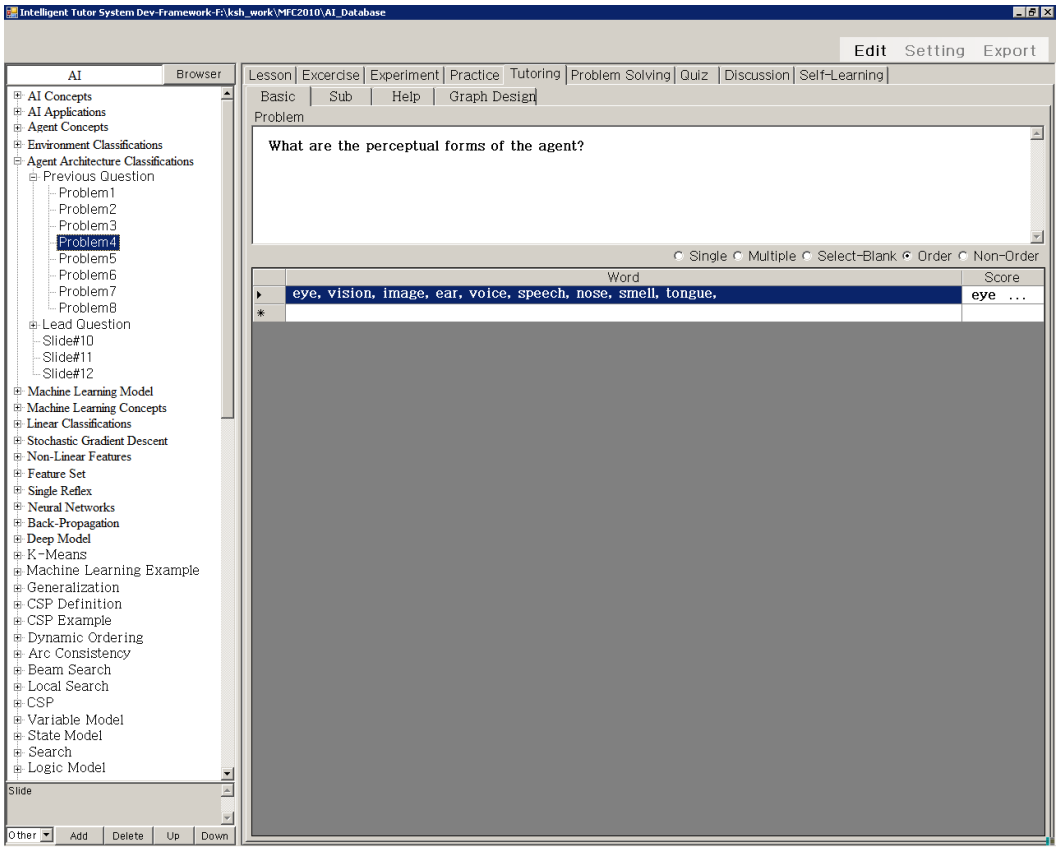

Figure 10. *Intelligent tutoring system editor.*

4. LSEPM and LPG Construction Method Using HMM

4.1. Formulation of Knowledge Graphs into Hidden Markov Models

4.1.1. Formulation of Hidden Markov Models

In the various types of knowledge graphs written by the subject teacher, the type of intelligent tutoring system using Hidden Markov Model is the basic concept graph. The Hidden Markov formalism for this basic concept graph is developed in Figure 7.

When learning state modeling is done in a probabilistic way, the learning state is represented as a random variable, which contains uncertainty, and is defined as a learning state variable. The Hidden Markov model is an extended dynamic Bayesian network, where nodes in this network consist of a sequence of learning state variables (learning state random variables). Eventually, each node of the basic concept graph must be converted into the learning state variables of the Hidden Markov model.

Learning state variables: $V_{sub_i}^{hyper_l,\ basic_m}$ (or V_{basic_k})

(This variable is a random variable that reflects the student's current learning status, i.e., objective information about cognitive performance).

The state value of the learning state variable = {excellent(ex), very good(v_g), good(g), average(aver), failure(fail)}.

The state value of this learning state variable is the value that reflects the degree of cognition as the value of the student's learning state. Since the state values of the learning state variables cannot be determined directly and accurately, we instead define the observation variables and determine them indirectly and probabilistically through the observation values of the observation variables.

Observations of Observational Variables (Answer_Marks) = {1.0(interval 1), [0.8, 1) (interval 2), [0.6, 0.8) (interval 3), [0.4, 0.6) (interval 4), [0.0, 0.4) (interval 5)}

In the future, Answer_Marks is abbreviated as A_M, and V_{basick} is also denoted as V_k. The intervals in parentheses in the observation variable indicate the values of the observation variable. Figure 11 shows the Hidden Markov model corresponding to the basic concept graph, which is indicated in bold because the learning state variables and observation variables are vectors.

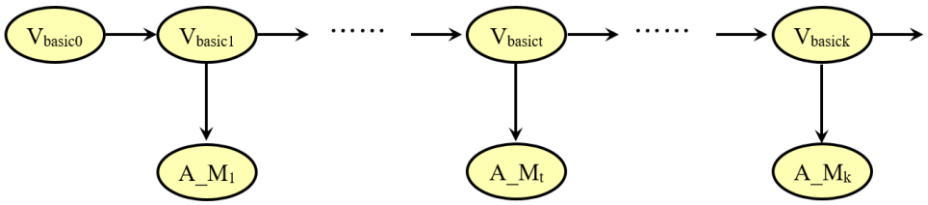

Figure 11. Hidden Markov Models Corresponding to Basic Concept Graphs (Example 1).

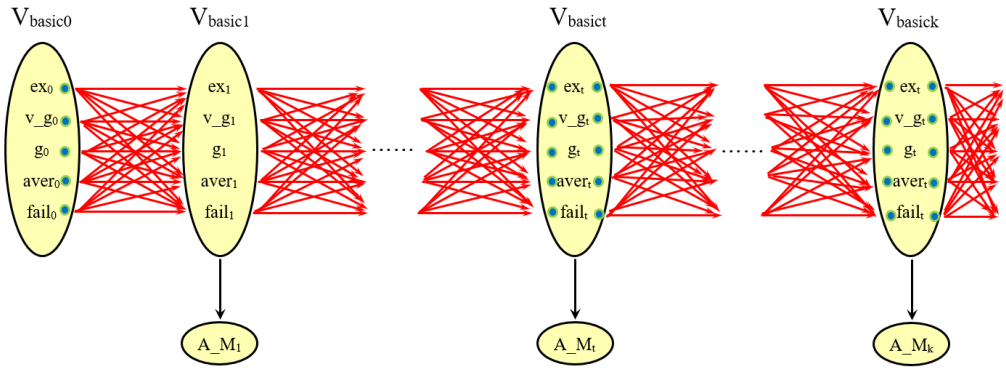

Figure 12. Hidden Markov Models Corresponding to Basic Concept Graphs (Example 2).

The transition model is shown in Table 1. Then $\sum a{:}y=1.0$. $a{:}y$ means that in any row of the transition model table, each of the values from the first value a to the last value y is listed.

Table 1. Transition model.

V_{t-1}	$P(V_t\|V_{t-1})$				
	ex	**v_g**	**g**	**aver**	**fail**
ex	0.85	0.1	0.02	0.02	0.01
v_g	0.2	0.4	0.2	0.1	0.1
g	0.3	0.2	0.3	0.15	0.05
aver	0.01	0.1	0.3	0.3	0.29
fail	0.0	0.0	0.2	0.3	0.5

The sensor model is a model that represents the probability that a certain value of the observed variable at time t will occur when the correct value of the student's learning state variable at time t-1 is already known. The sensor model is shown in Table 2. Also ∑a:y=1.0.

Table 2. Sensor model.

V_{t-1}	$P(A\_M_t\|V_{t-1})$				
	Interval 1	**interval 2**	**interval 3**	**interval 4**	**interval 5**
ex	0.7	0.1	0.1	0.1	0.0

V_{t-1}	$P(A\_M_t \mid V_{t-1})$				
	Interval 1	**interval 2**	**interval 3**	**interval 4**	**interval 5**
v_g	0.1	0.4	0.4	0.1	0.0
g	0.1	0.33	0.53	0.02	0.02
aver	0.0	0.1	0.39	0.33	0.18
fail	0.0	0.0	0.12	0.43	0.45

If the observed value at time t is judged as interval 2, then the observation matrix O_t as follows.

$$O_t = \begin{pmatrix} 0.1 & 0 & 0 & 0 & 0 \\ 0 & 0.4 & 0 & 0 & 0 \\ 0 & 0 & 0.33 & 0 & 0 \\ 0 & 0 & 0 & 0.1 & 0 \\ 0 & 0 & 0 & 0 & 0.0 \end{pmatrix}$$

4.1.2. Collection of Values of Evidence Variables

(a) Value-gathering algorithm of evidence variable

The learning state variable of the Hidden Markov model reflects the learning state of the student, which cannot be directly identified. If we can identify directly, we are not the subject of the Hidden Markov model. The state identification is confirmed by the time interval probability inference using the Hidden Markov model after indirect evidence is obtained from the answers to the students' perceptions of the concept. The algorithm for collecting the value of the evidence variable is implemented in four steps as follows.

[Algorithm 1. Value Collection Algorithm of Evidence Variables]

Step 1: Presently, elicit query presentation that enables the student to derive the basic concept of time t by himself. If the answer score for the question is 10, then move to step 3, otherwise move to step 2.

Step 2: Carry out a supplementary lecture that helps the student to derive the basic concepts by himself, and move to step 1.

Step 3: A complete lecture on the basic concept of time t (the intelligent teacher rehearses the concept comprehensively, assuming that the student has derived the basic concept of his own but is not yet fully understood).

Step 4: Give a question to understand the basic concept of time t.

(b) Method of Values Collection of Evidence Variables

We fix the proof of time t by setting the result of step 4 of Algorithm 2 as follows. In Step 4 of Algorithm 2, let us assume that the student's perception of the basic concept of time t is TD.

If the score TD = 1.0, we call the <A_M= interval 1>, if TD = [0.8, 1], then <A_M=interval 2>, if TD = [0.6, 0.8), then <A_M=interval 3>, if TD = [0.4, 0.6], then <A_M=interval 4>, if TD = [0.4], then<A_M=interval 5>.

(c) Example of collecting values of evidence variables

/*--Guided questions and supplementary lecture on basic concept 3------*/

(Basic concept 3) Relationships between agent and environment, classification of environment.

[Problem 3-1, 1, Audio] When we look at the concept definition of an agent, is there no connection between the environment and the agent?

i. connection, ii. no connection

[Answer] i. 10 ii. 0

(Supplementary lecture_slide 3-1) Slide1.png.

(Comment 3-1) There is a connection. Agents that do not receive perceptual data from the environment and do not act on the environment is little need to develop.

(Video 3-1)

[Problem 3-2, 4, Audio] What is the relationship between an agent and the environment? Try to write your ideas.

Correct: There is a perceptual relationship between an agents and an environment that uses a sensor to sense information, and then, by making appropriate reasoning, acts on his environment and affects it.

[Answer] i Perceptual ii Action

(Supplementary lecture_slide 3-2) Slide1.png.

(Comment 3.2) There is a perceptual relationship between an agent and an environment that is sensed by the use of sensors, and then acts in its environment by making appropriate reasoning.

(Video 3-2)

/*----------A complete lecture on basic concept 3----------*/

(slide 3-8) Slide1.png.

(Comment 3-8) Why should we deal with the environment in the design of an actor?

(Video 3-8)

(slide 3-9) Slide2.png.

(Comment 3-9) The table is shown in the figure.

(Video 3-9)

/*-------A question to understand the perception of Concept 3----*/

[Problem 1, 1] What is an environment? Choose the right

ones.

i. Environment? Where an agent resides. ii. Instead of the term environment, we also use the term world. iii. The environment is assumed to change by an agent.

[Answer] i 3 ii 4 iii 3

[Problem 2, 1, Audio] When we look at the concept definition of an agent, is there no connection between the environment and the agent?

i. connection ii. no connection

[Answer] i 10 ii 0

4.1.3. Limitations of the Hidden Markov Model

(a) Evaluation (correctly estimated) of the learning state sequence and prediction of the next-time learning state only

In intelligent tutoring systems, Hidden Markov models are used to evaluate current learning situations, predict future learning situations, and estimate performance in previous learning. That is, the Hidden Markov model can estimate a student's learning sequence of states through the evidence values from the past to the current time, but it cannot directly generate the best learning sequences to achieve the overall goal of learning. These learning sequences are the learning processes, and the learning flows inherent to each student that can maximize the degree of student awareness.

There are sequential decision-making methods, Markov decision-making or partially observable Markov decision-making, that are methods to calculate a sequence of actions that maximizes the learning state of each student, i.e., generate a learning flow that is specific to each student.

(b) Learning path generation principle using hidden Markov model

The Hidden Markov model can probabilistically derive estimates and predictions of current student's learning state, future state. Using this value, we have to develop a specific learning path generation algorithm separately.

4.2. Building a Tutorial Database

The tutoring database is as follows.

Table 3. tutorial database.

Record name	Data type	Main key	Explanation
Concept table			
ID	INTEGER	✓	unique identifier number
ConceptName	TEXT		concept name
ConceptType	INTEGER		concept type: basic concept-1, sub-concept-2, help-concept-3
RootConcept	INTEGER		root basic concept ID: this value is all zero, otherwise, the unique number of the root basic concept of the concept. For example, if a subconcept B is a subconcept between the basic concept A and the basic concept C, then the ID value of the basic concept A is the value of this field.
basic concept relational table			
Each row of the table represents each edge (relation) of the concept graph.			
ID	INTEGER	✓	unique identifier number
BeginVertex	INTEGER		precedence ID
EndVertex	INTEGER		ID of the posterior concept
RootConcept	INTEGER		fundamental concept of the relation
Relationship	INTEGER		What are the relationships between concepts. If either the front or rear nodes is a subconcept, then 2. if one of the two is the basic concept, and one is the subconcept, then 1. If both concepts are basic, then 0
Lecture table			
Comment: The slide are represented as a line of this table and are separated by the values of the ConceptID field. That is, to get the slide information for concept A, we get the unique identifier (ID) of the row corresponding to concept A in the concept table and return only the rows that are equal in value and ConceptID.			
ID	INTEGER	✓	unique identifier of slide

Record name	Data type	Main key	Explanation
ConceptID	INTEGER		Unique number of concepts belonging to slide is it an slide that reflects which concept.
SlideName	TEXT		The name of the picture file (including extension) in the path of "the subject name/ individual teaching/slide".
Comment	TEXT		comment
VideoName	TEXT		The name of the movie file (including extension) in the path of "the subject name/ individual teaching/slide".
ShowOrder	INTEGER		Numerical values for slide display sequences
hasTest	INTEGER		If there is a question for slide, then 1, 0, if no.
ReTest			
ID	INTEGER	✓	unique identifier
XMLData	TEXT		XML data for testing
ParentConcept	INTEGER		The ID of the parent concept, which concept is the question of understanding.
PreTest			
ID	INTEGER	✓	unique identifier
XMLData	TEXT		XML data for testing
ParentConcept	INTEGER		In the Concept Table, the ID of the concept, i.e., is it an leading question that reflects which concept
SlideName	TEXT		The name of the picture file (including extension) in the path of "the subject name/ individual teaching/slide".
Comment	TEXT		comment
VideoName	TEXT		The name of the movie file (including extension) in the path of "the subject name/ individual teaching/slide".
SlideTest (A understanding rate for a slide)			
ID	INTEGER	✓	unique identifier
XMLData	TEXT		XML data for testing
ParentSlide	INTEGER		The ID of the slide, i.e., is it an leading question that reflects which slide.

4.3. Learning State Modeling Using Smoothing Algorithm

Learning state modeling using Hidden Markov Models involves tasks such as filtering, state estimation, prediction, smoothing, generation of most likely explanation, and learning, and we set only the prediction of the learning state for student model generation for research and development purposes.

4.3.1. Learning State Prediction Method

Prediction using the Hidden Markov model may be considered as a filtering process without the presence of new evidence. The filtering process already implements a one-step prediction process and can easily derive recursive prediction computations for the state at t+k+1 from the prediction for t+k.

We first consider the filtering process and then consider the prediction method using this result. Then, the method proposed in Russell et al. [13] is applied to our learning state prediction process and an example of computation is presented.

(a) Filtering

Given the filtered result at time t, we have to calculate (estimate) the result for time t+1 from the new evidence A_M_{t+1}. This is equivalent to computing the following function f:

$P(V_{basic t+1} \mid A\_M_{1:t+1}) = f(A\_M_{t+1}, P(V_{basic t} \mid A\_M_{1:t}))$

Here $V_{basic\ t+1}$ is the learning state variable at time t+1, and $A\_M_{1:t+1}$ is the observation data at time t+1.

This recursive estimation calculation can be divided into two steps. First, we represent the distribution of the current state forward from t to t+1, and then update it using the new evidence A_M_{t+1}.

$P(V_{basic t+1} \mid A\_M_{1:t+1}) = P(V_{basic t+1} \mid A\_M_{1:t}, A\_M_{t+1})$ (separating evidence)

$= \alpha\ P(A\_M_{t+1} \mid V_{basic t+1}, A\_M_{1:t})\ P(V_{basic t+1} \mid A\_M_{1:t})$

Using Bayesian Rules Given an Evidence Value $A\_M_{1:t}$

$= \alpha\ P(A\_M_{t+1} \mid V_{basic t+1})\ P(V_{basic t+1} \mid A\_M_{1:t})$ (By the Sensor Markov assumption) (1)

In the above equation, the first term is the sensor model and the second term is the prediction. α is a normalization constant that makes the sum of probability values equal to unity. Let us further develop Eq. (1).

$$P(V_{basic t+1} \mid A\_M_{1:t+1}) =$$

$$\alpha\ P(A\_M_{t+1} \mid V_{basic t+1}) \sum_{V_{basic\ t}} P(V_{basic t+1} \mid V_{basic\ t}, A\_M_{1:t}) P(V_{basic t} \mid A\_M_{1:t})$$

$$= \alpha P(A\_M_{t+1} \mid V_{basic t+1}) \sum_{V_{basic\ t}} P(V_{basic t+1} \mid V_{basic\ t}) P(V_{basic t} \mid A\_M_{1:t}) \quad \text{(Markov assumption)} \tag{2}$$

In the above equation, the first term on the left is the sensor model, the first term on the left in the additive term is the transition model, and the last term is the recursive term.

Also, all terms are model or previous state estimates. Here, we consider the filtered estimate $P(V_{basic t} \mid A\_M_{1:t})$ as a "message" $\mathbf{f}_{1:t}$, corrected by each forward-propagating transition model in the sequence and updated by each new observation. This process is formalized as follows.

$$f\_m_{1:t+1} = \text{FORWARD}(f\_m_{1:t}, A\_M_{1:t+1})$$

$$f\_m_{1:t+1} = \alpha O_{t+1} T^{T} f\_m_{1:t} \tag{3}$$

Here, FORWARD implements the update process described in Eq. (2), which starts with f_m1:0 = $P(V_{basic\ 0})$. When all state variables are discrete, each update time is constant (i.e., dependent on t), and the required space cost is also constant. These constants, of course, depend on the size of the state space and the type of the corresponding specific time-interval model.

Let us consider only two steps of filtering in an intelligent tutoring system. Let us compute $P(V_{basic2} \mid A\_M_{1:2})$. At this time, refer to the values in Tables 2 and 3.

(b) Computational examples of filtering processes

In Figure 15, there is no evidence for the 0th learning state variable, and only the prior confidence of the observer exists.

$P(V_{baisc0})$ = <0.1, 0.3, 0.42, 0.1, 0.08>

The evidence value of the first learning state variable is 0.7, with A_M = interval 3. The state estimation from 0 to 1 is performed as follows.

$$P(V_{basic1}) = \sum_{V_{basic0}} P(V_{basic1} \mid V_{basic0}) P(V_{baisc0}) = <0.85, 0.1, 0.02, 0.02, 0.01> \times 0.1 + <0.2, 0.4, 0.2, 0.1, 0.1> \times 0.3 + <0.3, 0.2, 0.3, 0.15, 0.05> \times 0.42 + <0.01, 0.1, 0.3, 0.3, 0.29> \times 0.1 + <0.0, 0.0, 0.2, 0.3, 0.5> \times 0.08 = <0.272, 0.224, 0.234, 0.149, 0.121>$$

$P(V_{basic1} \mid A\_M_1) = \alpha\ P(A\_M_1 \mid V_{basic1}) P(V_{basic1}) = \alpha$ <0.1, 0.4, 0.53, 0.39, 0.12> <0.272, 0.224, 0.234, 0.149, 0.121> = α <0.0272, 0.0896, 0.12402, 0.05811, 0.01452> ≈ <0.0868, 0.2859, 0.3957, 0.1854, 0.0462>

The evidence corresponding to the second learning state variable is 0.5, with the A_M_2 = interval 4. The state estimation from 1 to 2 is performed as follows.

$$P(V_{basic2} \mid A\_M_1) = \sum_{V_{basic1}} \mathbf{P}(V_{basic2} \mid V_{basic1}) P(V_{basic1} \mid A\_M_1)$$

= <0.85, 0.1, 0.02, 0.02, 0.01>× 0.0868 + <0.2, 0.4, 0.2, 0.1, 0.1>×0.2859 + <0.3, 0.2, 0.3, 0.15, 0.05>×0.3957 + <0.01, 0.1, 0.3, 0.3, 0.29>×0.1854 + <0.0, 0.0, 0.2, 0.3, 0.5>×0.0462 = <0.2515, 0.2207, 0.2425, 0.1592, 0.1261>

The estimate for the current learning state is g.

$P(V_{basic2} \mid A\_M_1, A\_M_2) = \alpha\ P(A\_M_2 \mid V_{basic2}) P(V_{basic2} \mid A\_M_1) = \alpha$<0.1, 0.1, 0.02, 0.33, 0.43><0.2515, 0.2207, 0.2425, 0.1592, 0.1261> = α <0.02515, 0.02207, 0.00485, 0.05254, 0.05422> ≈ <0.1583, 0.1390, 0.0305, 0.3308, 0.3414>

When the second learning phase was followed, the student's cognitive state became worse and worse, and became aver.

(c) Prediction

Prediction is a problem that predicts what the student's cognitive state value will be when passing through the third learning state when it comes to the second learning state. The evidence is given until the second lecture. The prediction task may be considered as a filtering process without the

presence of new evidence. The filtering process already implements a one-step prediction process and can easily derive recursive prediction computations for the state at time $t+k+1$ from the prediction for $t+k$.

$$P(V_{basic\ t+k+1} | A\_M_{1:t}) = \sum_{\mathbf{V}_{basic t+k}} P(\underbrace{\mathbf{V}_{basic\ t+k+1} | \mathbf{V}_{basic\ t+k}}_{\text{transition model}}) \underbrace{P(V_{basic\ t+k} | A\_M_{1:t})}_{\text{recursive}} \quad (4)$$

In this calculation, only the transition model is included and the sensor model is not included.

4.3.2. Smoothing Algorithm

The filtering and prediction in the Hidden Markov model proposed above are implemented by a smoothing algorithm.

[algorithm 2. smoothing algorithm]

function SMOOTHING(A_M_t, *hmm*, *d*) returns a distribution over $V_{basic\ t-d}$

inputs: A_M_t, // the current evidence for time step *t*

hmm, // a Hidden Markov Model with S× S transition matrix T

d, // the length of the lag for smoothing

persistent: *t*, // the current time, initially 1

f_m, // the forward message $P(V_{basic\ t} | A\_M_{1:t})$, initially *hmm*.PRIOR

B, // the *d*-step backward transformation matrix, initially the identity matrix

A_$M_{t-d:t}$, // double-ended list of evidence from $t - d$ to t, initially empty

local variables: O_{t-d}, O_t, diagonal matrices containing the sensor model information add A_M_t to the end of A_$M_{t-d:t}$

$O_t \leftarrow$ diagonal matrix containing $P(A\_M_t | V_{basic\ t})$

if $t > d$ then

f_m$\leftarrow$ FORWARD(f, A_M_{t-d})

remove A_M_{t-d-1} from the beginning of A_$M_{t-d:t}$

$O_{t-d} \leftarrow$ diagonal matrix containing $P(A\_M_{t-d} | V_{basic\ t-d})$

$B \leftarrow O_{t-d}^{-1} T^{-1} B T O_t$

else B $\leftarrow$ BTO_t

$t \leftarrow t + 1$

if t > d + 1 then return NORMALIZE(f_m × B1) else return null

Notice that the final output NORMALIZE(f_m × B1) is just αf × b.

4.4. Learning Path Generation

4.4.1. Learning Path Generation Algorithm

[algorithm 3. Learning path generation algorithm]

1: if ($V_{basick+1}$ = ex)

2: {

3: (If the student's learning state prediction result is ex, then the derived query for 4: the k+1th concept is repeated until the score is 1.0, then a rich and complete 5: lecture on the k+1th concept is given);

6: }

7: else if($V_{basick+1}$ = v_g)

8: {

9: (Choose the shortest path among paths consisting of the sub-concepts between 10: the k-th concept and the k+1th concept);

11: }

12: else if(V_{basick} = g)

13: {

14: (choose a path among m-1 paths consisting of sub-concepts between k

15: th concept and k+1th concept using probabilistic inference algorithm);

16: }

17: else if(V_{basick} = aver)

18: {

19: (For a student with this value, we generate paths that include the subc-

20: oncept and help-concept, and then also divide the number of total sub

21: path as above and generate the learning path according to the corresp-

22: onding probability value.

23: }

24: else

25: {

26: Make a choice of the k-1th basic concept.

21: }

4.4.2. Description of Line 12 of Algorithm 3

A description of the stochastic path selection method for intervals 3. For the A_M=interval 3(0.6<= TD < 0.8), we randomly choose a path among m-1 paths consisting of the sub-concepts between the nth concept and the n+1th concept. If the distance is between 0.6 and 0.8, divide 1.0 by (n = total sub-learning path m −1) and find the ratio value that one learning path occupies in the whole interval 1. For example, if we have five sub-learning paths, then dividing 1.0 by 4 yields a value of 0.25. This means that each learning path (n = total sub-learning path m-1) has a ratio of 0.25 in the interval (actually, the interval between 0.6 and 0.8). Then, n sub-learning paths are ordered from one to the next, depending on the number of arcs present in each path, from the number of arcs to the least. The probability values of each of the four intervals are expressed as

0.6+0.25×0.2×1=0.6+0.05=0.65,

0.6+0.25×0.2×2=0.6+0.1=0.7,

0.6+0.25×0.2×3=0.6+0.15=0.75,

0.6+0.25×0.2×4=0.6+0.2=0.8.

The value of 0.25 is the ratio of one path to the path con-

sisting of n (m-1) subconcepts. This number varies with the number of paths that consist of subconcepts. The value of 0.2 is the value that makes the interval between 0.6 and 0.8 equal to 1.

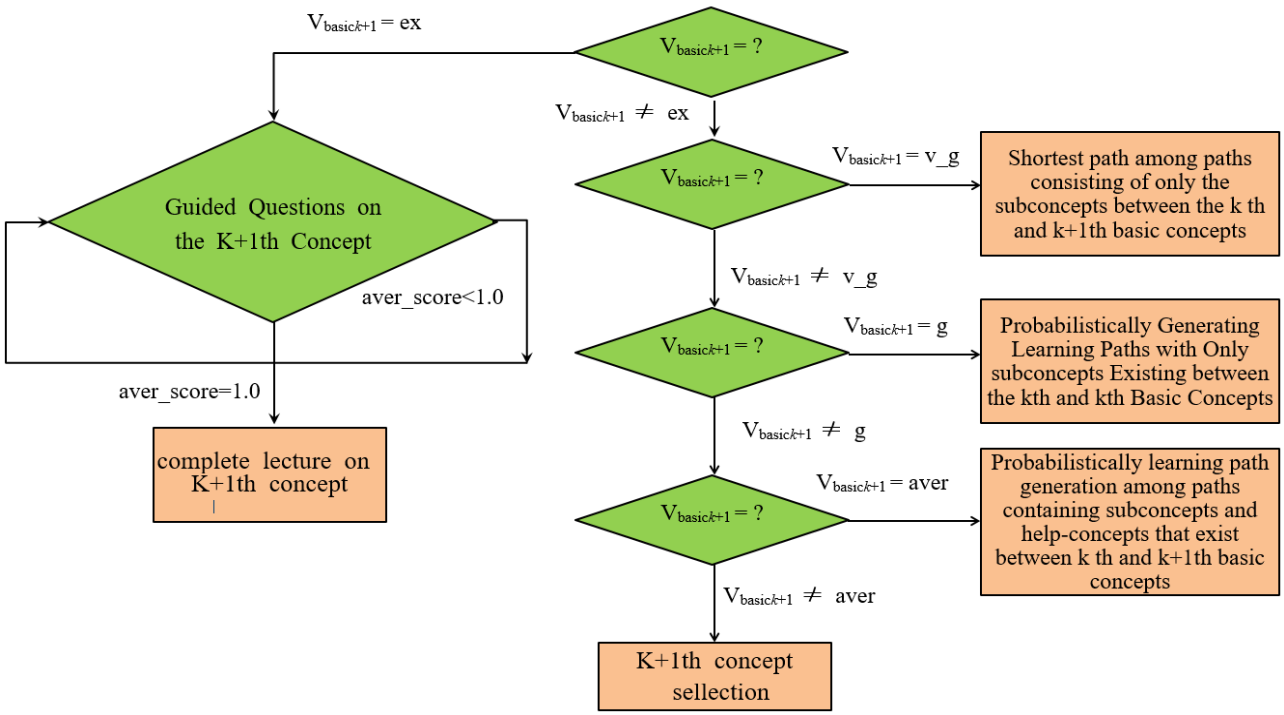

Figure 13. Learning path generation flow diagram.

5. Experiments and Evaluations on Hidden Markov Model

5.1. Analysis of Intelligent Tutoring Systems Using HMM

We evaluated how students using an intelligent tutoring system constructed using a Hidden Markov Model improved their learning performance compared to those who did not use it.

Using the intelligent tutoring system, experiments were conducted on the learning performance when students had several extra-curricular sessions. Also using the intelligent tutoring system, an experiment was conducted to evaluate the amount of work done when the students were given a task after all of the learning activities or extracurricular studies on a subject.

In the first experiment, students were divided into three groups: excellent, medium, and low, and one subject was tested for extracurricular learning. The number of students who participated in the test was 350, and the test problem averaged 20 questions in each section and 450 questions in one subject. The results are shown in the following figure. Since approximately 25% of the total amount of learning was completed for each chapter or subject, the results that students had observed for each of the clause tests embedded in the intelligent tutoring system were 75% completed. The results were positive.

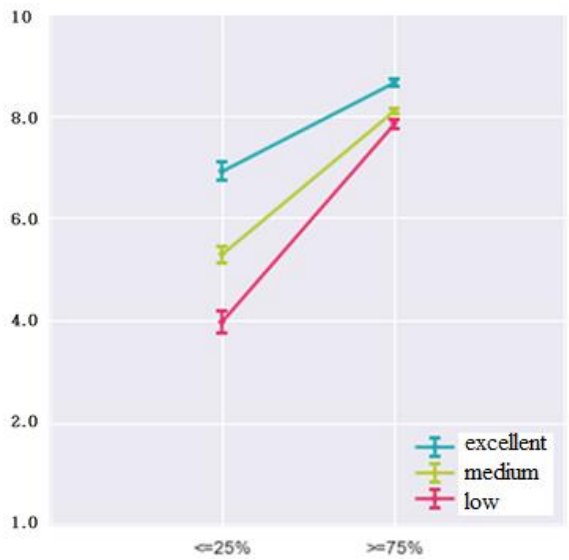

Figure 14. When students were trained for extracurricular learning using an intelligent tutoring system (Learning volume on the longitudinal axis, student score on the horizontal axis).

In the second experiment, students were divided into three groups: excellent, medium, and low, and after completing all the extra-curricular studies on one subject, the task was given and the performance was evaluated. For one subject, the average number of subjects was 20.

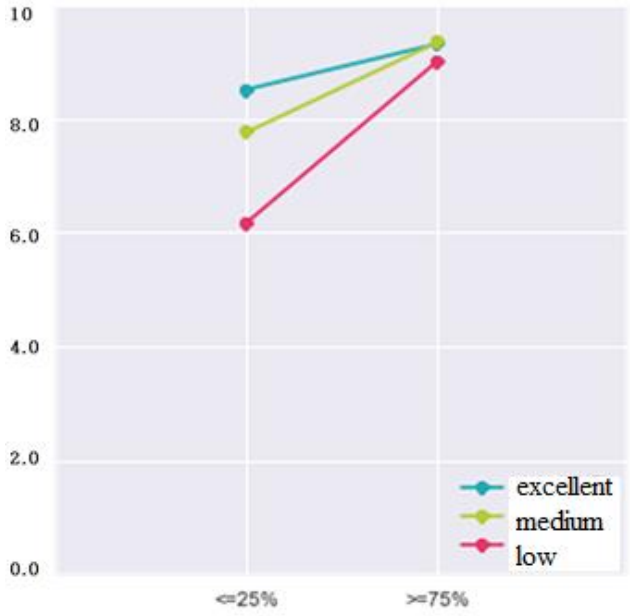

Figure 15. The percentage of work done when assigning students to a task who completed all the extra-curricular learning courses on a subject using an intelligent tutoring system (%) (the percentage done on the longitudinal axis, the amount of work done on the horizontal axis).

5.2. Comparative Evaluation Experiments with Other Models

In addition to the Hidden Markov Model, there are methods such as performance factor model (PFM), additive factor model(AFM), with partially observable Markov Decision Process. We conducted a performance evaluation experiment with the proposed intelligent tutoring system using Hidden Markov Model and intelligent tutoring system using performance factor model and additive factor model. The experiments were conducted by pre-testing and final testing and comparing the results. The test was conducted by dividing the students into high and general. The experimental results showed that the average performance of students using the Hidden Markov Model was higher than that of students using other models for intelligent tutoring systems.

Table 4. Results of the comparative evaluation of the three models.

	pre-test			final test		
	high	medium	overall	High	Medium	overall
HMM	7.5	7.2	7.1	8.4	9.2	8.8
PFM	7.4	7.0	7.4	8.6	7.4	7.9
AFM	6.8	6.0	6.7	8.9	7.6	8.2

6. Conclusion

The Hidden Markov Model is a probabilistic & stochastic inference model over time, known as a good model that can predict the learning state of students using the students' past learning history data. However, when using this Hidden Markov Model, we can generate the learning path only by using an additional learning path generation algorithm.

We proposed to use the Hidden Markov Model as a model for predicting student learning state in an intelligent tutoring system. We also proposed a learning path generation algorithm.

The experimental results showed that the learning performance of the learning state prediction method using Hidden Markov Model is higher than that using empirical multi-dimension linear regression model when using probabilistic & stochastic inference model over time such as hidden Markov model.

Abbreviations

ITS	Intelligent Tutoring System
LM	Learner Model
TM	Tutor Model
DM	Domain Model
LSEPM	Learning State Estimation and Prediction Model
LPG	Learning Path Generator
G	Graph
V	Vertex
E	Edge
Sub	Subject
Ex	Excellent
v_g	Very Good
g	Good
aver	Average
fail	Failure
A_M	Answer_Marks
KT	knowledge Tracking
BKT	Bayesian Knowledge Tracing
DKT	Deep Knowledge Tracing
RNN	Recurrent Neural Networks
LSTM	Long Short Term Memory Networks
GRU	Gated Recurrent Units
GNNs	Graph Neural Networks
GKT	Graph-based Knowledge Tracking
GIKT	Graph-based Interaction Knowledge Tracking
SKT	Structure-based Knowledge Tracking
HMM	Hidden Markov Model
AIWBES	Adaptive and Intelligent Web-Based Educational System
f_m	Forward Message

Author Contributions

Song-Hwan Kwon: Conceptualization, Methodology, Writing-original draft, Project administration

Jong-Nam Rim: Formal analysis, Writing-review & editing

Yun-Ho Ko: Investigation, Software

Yun-Sok Ham: Software, Data curation

Ha-Gyong Kim: Resources, Validation

Conflicts of Interest

The authors declare no conflicts of interest.

References

[1] MURTAZA, M., AHMED, Y., SHAMSI, J. A., SHERWANI, F., & USMAN, M. (2022). AI-Based Personalized E-Learning Systems: Issues, Challenges, and Solutions. IEEE Access, VOLUME 10, pp 81323-81326.

[2] Ouyang, Y., Zhou, Y., Zhang, H., Rong, W., and Xiong, Z. (2021). PAKT: A Position-Aware Self-attentive Approach for Knowledge Tracing. Artificial Intelligence in Education-2021, 22nd International Conference, AIED 2021 Utrecht, The Netherlands, June 14–18, Proceedings, Part II, Springer.

[3] Su, Y., Liu, Q., Liu, Q., Huang, Z., Yin, Y., Chen, E., Ding, C., Wei, S., & Hu, G. (2018). Exercise-enhanced sequential modeling for Student performance prediction. in Proc. AAAI Conf. Artif. Intell., 2018, vol. 32, no. 1, pp. 1–9.

[4] Liu, Q., Huang, Z., Yin, Y., Chen, E., Xiong, H., Su, Y., & Hu, G. (2021). EKT: Exercise-aware knowledge tracing for Student performance prediction. IEEE Trans. Knowl. Data Eng., vol. 33, no. 1, pp. 100–115, Jan.

[5] Rosen, Y., Rushkin, I., Rubin, R., Munson, L., Ang, A., Weber, G., Lopez, G., and Tingley, D. (2018). Adaptive Learning Open Source Initiative for MOOC Experimentation. © Springer International Publishing AG, part of Springer Nature 2018, C. Penstein Rosé et al. (Eds.): AIED 2018, LNAI 10948, pp. 307–311, https://doi.org/10.1007/978-3-319-93846-2_57

[6] Lipton, Z. C., Berkowitz, J., and Elkan, C. (2015). A critical review of recurrent neural networks for sequence learning. arXiv: 1506.00019.

[7] Chen, P., Lu, Y., Zheng, V. W., and Pian, Y. (2018). 'Prerequisite-driven deep knowledge tracing. in Proc. IEEE Int. Conf. Data Mining (ICDM), Nov. pp. 39–48.

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

[9] Nakagawa, H., Iwasawa, Y., and Matsuo, Y. (2019). Graph-based knowledge tracing: Modeling Student proficiency using graph neural network. in Proc. IEEE/WIC/ACM Int. Conf. Web Intell., Oct., pp. 156–163.

[10] Yang, Y., Shen, J., Qu, Y., Liu, Y., Wang, K., Zhu, Y., Zhang, W., and Yu, Y. (2020). GIKT: A graph-based interaction model for knowledge tracing. in Proc. Joint Eur. Conf. Mach. Learn. Knowl. Discovery Databases. Cham, Switzerland: Springer, pp. 299–315.

[11] Song, X., Li, J., Lei, Q., Zhao, W., Chen, Y., and Mian, A. (2022). Bi-CLKT: Bi-graph contrastive learning based knowledge tracing. arXiv: 2201.09020.

[12] Tong, S., Liu, Q., Huang, W., Hunag, Z., Chen, E., Liu, C., Ma, H., and Wang, S. (2020). Structure-based knowledge tracing: An influence propagation view. in Proc. IEEE Int. Conf. Data Mining (ICDM), Nov. pp. 541–550.

[13] Russell, S. J., & Norvig, P. (2022). Artificial Intelligence: A Modern Approach. Prentice Hall. Retrieved from http://www.amazon.com/Artificial-Intelligence-ApproachStuart-Russell/dp/0131038052

[14] Pardos, Z. A., Heffernan, N. T. (2022). Using HMMs and bagged decision trees to leverage rich features of user and skill from an intelligent tutoring system dataset. the Journal of Machine Learning Research W & CP, In Press.

[15] Masun, H., Rania, L., Rosa Maria, C., and Ghias, B. (2007). Student Modeling Using NN-HMM for EFL Course. ICTTA08, Syria, Abril.

[16] Masun, H., Rania, L., Rosa Maria, C., and Ghias, B. (2007). Building adaptive web-based educational system using Fuzzy-ART2 for EFL course. Hiroshima, Japan, November.

[17] Masun, H., L. Ghias, R. B. (2006). Adaptive web-based educational system using neural networks in EFL Course. IEEE Xplore Release 2.2, 1, pp. 622- 625.

[18] Xuejing, G. (2003). Research on modeling artificial emotion based on HMM and techniques correlated with virtual human. Univ of Science and Technology of Beijing. pp. 66-72.

[19] Xuejing, G. (2010) Design of Emotional Intelligent Tutor System Based on HMM. 2010 Sixth International Conference on Natural Computation (ICNC 2010).

[20] Boyer, K. E., Phillips, R., Wallis, M., Vouk, M., and Lester, J. (2008). Balancing cognitive and motivational scaffolding in tutorial dialogue. Proceedings of the 9th International Conference on Intelligent Tutoring Systems: 239-249.

[21] Boyer, K. E. et al, (2020). Inferring Tutorial Dialogue Structure with Hidden Markov Modeling.

[22] Chevalère, J. (2023). A sequence of learning processes in an intelligent tutoring system from topic-related appraisals to learning gains. Learning and Instruction. Volume 87, October, ScienceDirect.