Intelligent LMS with an agent that learns from log data

1 author:

Maomi Ueno
The University of Electro-Communications

Original Paper

Intelligent LMS with an agent that learns from log data

Maomi Ueno*

Abstract This paper describes an agent that acquires domain knowledge related to the content from a learning history log database and automatically generates motivational messages for the learner. The unique features of this system are as follows: The agent builds a learner model automatically by applying the decision tree model. The agent predicts a learner's final status (Failed; Abandon; Successful; or Excellent) using the learner model and his/her current learning history log data. The constructed learner model becomes more exact as the amount of data accumulated in the database increases. Furthermore, the agent compares a learner's learning processes with "Excellent" status learners' learning processes stored in the database, diagnoses the learner's learning processes, and generates adaptive instructional messages for the learner. A comparison between a class of students that used the system and one that did not demonstrates the effectiveness of the system.

Keywords: e-learning, learning management system, agent, learning log data, data mining, decision tree

1. Introduction

Motivation is essential to learning and performance, particularly in e-learning environments where learners must take an active role and be self-directed in their learning[(1)]. Keller[(2)] argues that although motivation is idiosyncratic, learner motivation can also be affected by external aspects. Visser and Keller[(3)] reported that motivational messages can reduce dropout rates and subsequently, Visser et al.[(4)] attempted to improve motivation in e-learning situations using such messages. Gabrielle[(5)] applied technology-mediated instructional strategies to Gagne's events of instruction and showed how these strategies affect motivation. All these studies emphasized the effects of the mentor's motivational messages adapted to a learner's status in e-learning. However, when the number of learners is large, it becomes difficult for a mentor to individualize messages to students. The main idea of this paper is to develop a learning management system (LMS) in which an animated agent substitutes for the teacher as a virtual facilitator;[(6)] that is, an intelligent agent provides adaptive instructional messages to learners using learner models and learning history data.

As is well known, almost all traditional learner models proposed in intelligent tutoring system (ITS) research depend on domain knowledge. This means that a new program has to be developed or metadata have to be input whenever creating new content is desired. In fact, many ITSs have been developed independently for different subjects. However, this is a major problem for e-learning systems because content providers seldom have the necessary special programming skills. To solve this problem, we introduce a learner model from probabilistic approaches and propose~~s~~ a way to automatically build learner models using learning history data. The Bayesian belief network is a well-known tool for constructing a probabilistic learner model (see, for example, (7)–(14)). In particular, we have proposed a method to construct learner models automatically using test data, but found that the difficulty in building structures in Bayesian networks made it impossible to apply structures with a large number of variables[(10)].

On the other hand, the decision tree model[(15)], which is a well-known method that is equivalent to the Bayesian belief network, enables users to obtain valid results even if the number of variables in the tree increases significantly, although interpreting the meaning of a structure is more difficult than in the Bayesian belief network.

Building a meaningful learner model requires a number of variables for representing a learner's status. For these reasons, in this study we used an intelligent agent based on the decision tree model and installed it into an LMS.

The unique features of this system are summarized as follows.

1. The agent builds a learner model automatically by applying the decision tree model.
2. The agent predicts a learner's final status (1. Failed;~~,~~ 2. Abandon;~~,~~ 3. Successful;~~,~~ or 4. Excellent) using the learner model and his/her current learning history log data. The constructed learner model becomes more exact

*Graduate School of Information Systems,
The University of

as the amount of data accumulated in the database increases.

3. The agent compares a learner’s learning processes with excellent learners’ learning processes in the database, diagnoses the learner’s learning processes and generates adaptive instructional messages for the learner.

The developed LMS with the agent system was compared with one without it in actual e-learning classes for one semester. The results showed that a much lower number of students withdrew from classes when the LMS with the agent system was used. In addition, the average score of the final test was significantly higher in the case of the LMS with the agent system. Answers to questions and interviews with learners showed that the agent system enhances learners’ motivation and contributes to learners’ maintaining a constant learning pace.

2. Related studies

Various studies have been done that have applied data mining techniques to learning history data in e-learning.

Becker and Vanzin[16] tried to detect meaningful patterns of learning activities in e-learning using the association rule. Minaei-Bidgoli,et al.[17] proposed a method to predict a learner's final test score using the combination of multiple classifiers constructed from learning history data in e-learning, and they reported that a modified method using a genetic algorithm could improve the prediction performances.

Talavera and Gaudioso [18] and Hamalainen et al.[19] separately proposed a method to predict final test scores using the naive Bayes model from learning history data in e-learning.

Huang et al.[20] predicted learning efficiency as defined by test score/learning time using a support vector machine (SVM) from learning history data in e-learning.

However, these studies only tried to predict learner's performance in e-learning from learning history data, and therefore, they did not discuss how to effectively utilize the predicted data mining results to improve learners’ results. Furthermore, the employed data mining engines in these studies were not installed into an LMS to automatically analyze the learning log database.

Here, we proposed not simply a system to predict a learner's final status using a data mining technique, but an agent that acquires the domain knowledge related to the content from a learning history log database and that automatically generates adaptive instructional messages for the learners.

3. LMS “Samurai”

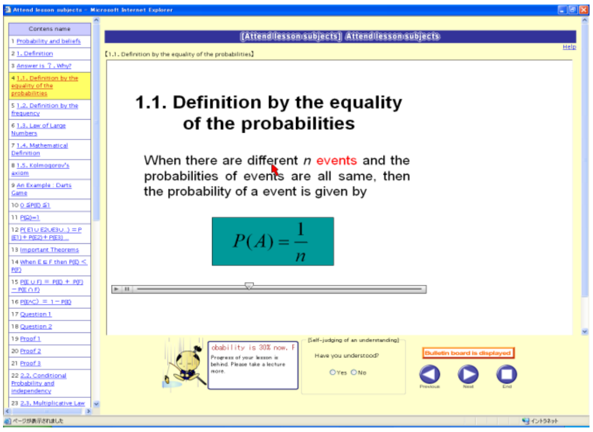


Figure 1 The LMS “Samurai”


Figure 2 Example test frame

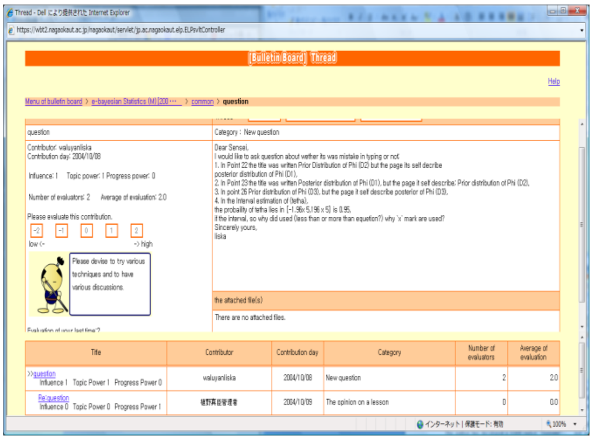


Figure 3 Example discussion board

We have developed an LMS called “Samurai”[21] that is used with many e-learning courses; 128 e-learning courses are now offered by The University of Electro-Communications, through this LMS. Samurai consists of a contents presentation system (CPS), a contents database (CD), a discussion board (DB), a learning history database (LHD), and a data mining system –(DMS).

The CPS integrates various kinds of content and presents the integrated information on a web page.

Figure 1 shows a typical e-learning content presentation by Samurai. The contents are presented by clicking on the menu button. A sound track of the teacher's narration is also presented based on the research of" Mayer and Anderson [22], and the red pointer moves automatically as the narration proceeds. This lesson corresponds to a 90-minute university lecture and includes 42 topics. Although the content in Figure 1 is textual, the system also provides illustrations, animations or computer graphics, and video clips. In this lesson, there are 11 topics presented as textual content, 11 as illustrations, 10 as animations, and 10 as video clips. The CPS also presents some test items to assess the learners' degree of comprehension as soon as the lessons have been completed (Figure 2).

The CD consists of various kinds of media, text, jpeg and mpeg files, and so on. The teacher prepares a lecture and saves the contents in the CD. Then the CPS automatically integrates the contents, and presents them to the learners.

The learners can ask questions about the contents in the DB (Figure 3). They can also submit the products of their learning for the given task (for example, a report or program source) using the DB.

The LMS monitors learners' learning processes and stores them as log data in the LHD. The stored data consist of a Contents ID, a Learner ID, the number of topics that the learner has completed, a Test Item ID, a record of data input into the DB, an Operation Order ID (which indicates what operation was done), a Date and Time ID (which indicates the date and time that an operation started), and a Time ID (which indicates the time it took to complete the operation). These data enable the LMS to recount the learner's behavior in e-learning.

4. An agent using the decision tree model for e-learning history data

4.1. Prediction of learner's final status

The main idea here is to apply a data-mining method to the huge amount of stored data and construct a learner model to predict each learner's final status: (1) Failed (Final examination score below 60); (2) Abandon (The learner withdraws before the final examination);, (3) Successful (Final examination score is more than 60 but less than 80); or (4) Excellent (Final examination mark is more than 80.) For this purpose, the well-known data-mining decision tree model[15], is employed using the following variables reflecting each learner's status each week.

1. The number of topics the learner has learned.
2. The number of times the learner accessed the e-learning system.
3. The average number of times the learner has completed each topic. (This implies the number of times the learner repeated each topic.)
4. The average learning time for each lecture, which consists of several types of contents and runs 90 minutes.
5. The average of the degree of understanding of each topic. (This is measured by the response to the question corresponding to each topic.)
6. The average learning time for each course consisting of fifteen lectures.
7. The average number of times the learner has changed the answer to questions in the e-learning.
8. The number of times the learner has posted opinions or comments on the discussion board.
9. The average learning time for each topic.

As all courses run for 15 weeks, 15 decision trees are prepared corresponding to the learners' learning history data for the 15 weeks.

The continuous variables are categorized into the number that minimizes the entropy.

We used the ID3 algorithm[22] as a learning algorithm for the decision trees because the computation cost is low and the estimators are robust.

The program source was developed using Java and installed in Samurai. The decision trees are always learned using updated learning histories. Therefore, the decision trees' structures for predicting the learner's final status are always changing. In this algorithm, all variables are always used. A decision tree learned from 1,344 learners' data is shown in Figure 4. This tree was prepared using 14 weeks of learning history data. The two values in parentheses indicate the number of cases in which the inference is correct and incorrect. For example, (408/18) indicates that the probability of the correct inference is 408/426. In this system, decision trees corresponding to the weekly learner's status are constantly being constructed.

4.2. Outline of intelligent agent system

The main purpose of the intelligent agent system is to provide optimum instructional messages to a learner using the previous automatically constructed learner model. The agent appears in the LMS as shown in Figure 5. First, this section introduces an outline of the system. The agent provides adaptive messages to the learner using the learner model. The agent also performs various actions based on the learner's current status, as shown in Figure 6. The instructional messages to a learner are generated as follows.:

1. The system predicts the target learner's future status and probability using the constructed decision tree.
2. If the predicted status is "Excellent", the agent provides messages like "Looking great!", "Keep doing your best", and "Probability of success is xx%". If the predicted status is not "Excellent", the

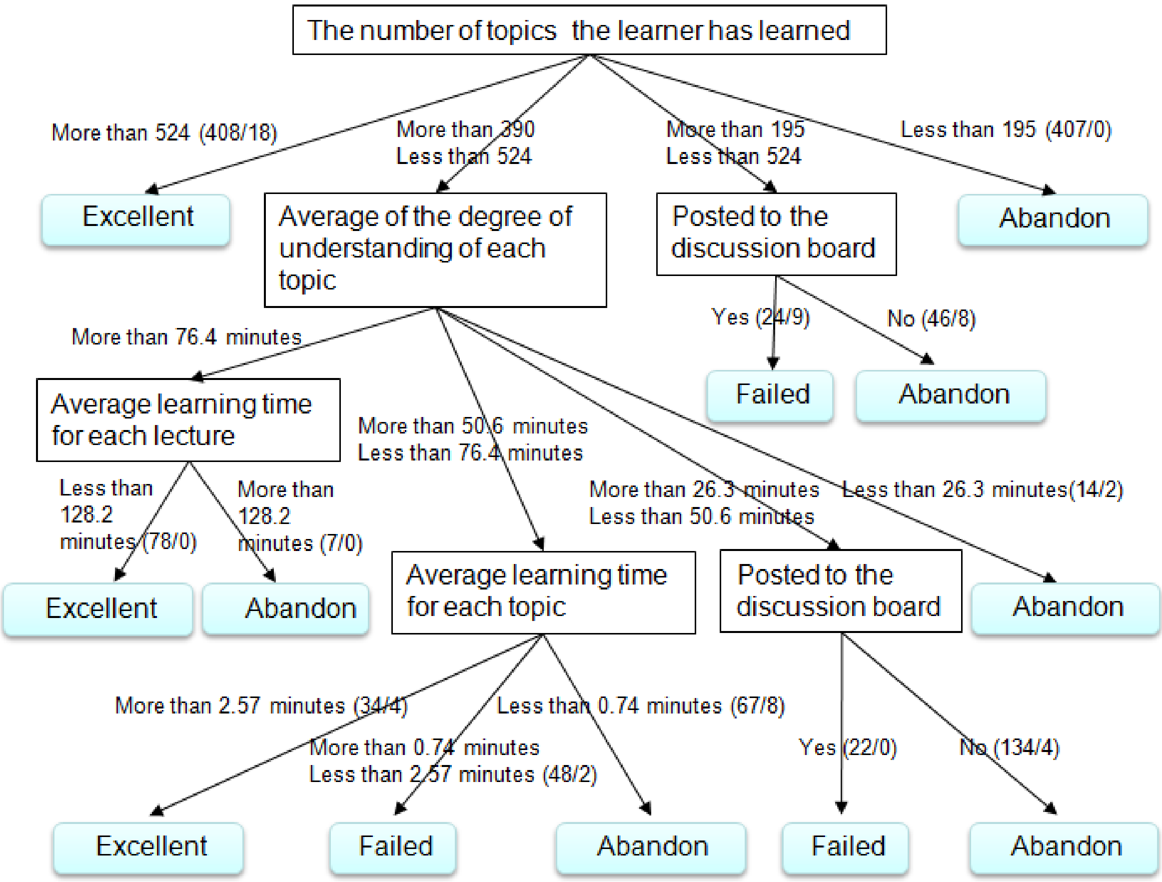


Figure 4. Example of a constructed decision tree

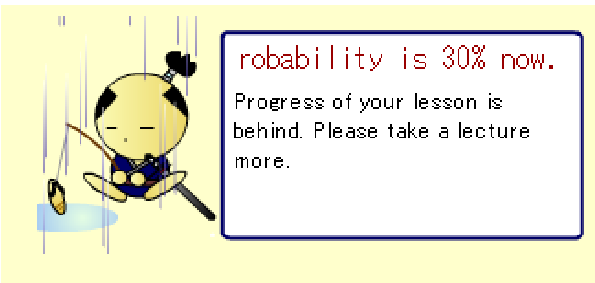


Figure 5 An intelligent agent (Note that the presented message is not misspelled. The message is moving continuously across the frame.)

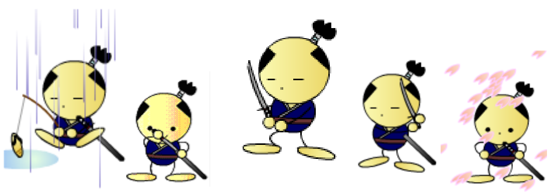


Figure 6 Various actions of the agent

system searches for the closest “Excellent” node from the current predicted status node. For example, using Figure 7, we consider a part of the decision tree in Figure 4. If the predicted status is “Failed”, the nearest “Excellent” node is the gray node in the figure. The system finds the nearest “Excellent” node and determines

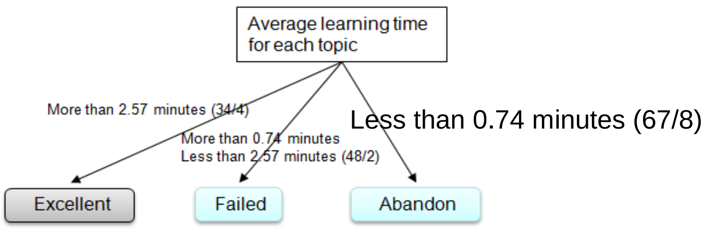


Figure 7 Part of the decision tree in Figure 4

the operations that will change the learner’s predicted status to “Excellent”. In this case, “the average learning time for each topic” is detected. The system provides messages with the predicted future status, the probability of success estimated by the decision tree, and the instructional messages according to Table 1.

4.3. Data structure

The system constructs a decision tree from learning history data and stores it in the database. The data structure of the constructed decision tree is defined using XML, as shown in Figure 8. <NAME> indicates the course subject name and variable names. <VARIABLE TYPE> has two types: "Explain", which means "explaining variables" and "Object", which means "object variable" in the decision tree

Table 1. Instructional messages corresponding to the detected variables

Variables	Instructional messages
1. The number of topics the learner has learned.	1. Progress in your lesson is behind. Please do more lectures. 2. Progress in the lesson is liable to be slow. Let's do more lectures.
2. The number of times the learner has accessed the e-learning system.	3. You have not engaged in the lesson well. Let’s access the system more often.
3. The average number of times the learner has completed each topic.	4. Don’t forget previously learned contents! Let’s review the previous contents again.
4. The average learning time for each lecture, which consists of several types of contents and runs 90 minutes.	5. It seems that you are working through the lecture too quickly. Please spend more time on each lecture.
5. The average of the degree of understanding of each topic (This is measured by the response to a question that corresponds to each topic.).	6. Were the contents of the lesson difficult? Let's do the lecture from the beginning once again. 7. When there is something you don’t understand, let's post a question on the discussion board.
6. The average learning time for each course consisting of fifteen lectures.	8. You have not engaged in the lesson well. Let’s access the system and study more slowly and carefully.
7. The average number of times the learner has changed the answer to the e-learning questions.	9. Your knowledge does not appear to be very robust. Let's do the lecture from the beginning once again.
8. The number of times the learner has posted opinions or comments on the discussion board.	10. Learning is more effective when done between learners. Let's participate in and contribute to the discussion board.
9. The average learning time for each topic.	11. Did you do the lecture correctly? Ordinarily, a lesson should take more time.

```
<Tree>
<NAME>Statistics</NAME>  ←The Subject Name
        <VARIABLE TYPE="Explain">   ← Node  setting
                <NAME>The average learning time to each topics</NAME>←Node Name
                <ATRIBUTE>More than 2.57 minutes </ ATRIBUTE >  ←Value
                < ATRIBUTE > More than 0.74 Less than 2.57 minutes </ATRIBUTE >
                <ATRIBUTE > Less than  0.74 </ATRIBUTE>
        </VARIABLE> ←    The end of Node 1 setting

        <VARIABLE TYPE="Object">  ← Learner's Final States setting
                <NAME>FINAL STATUS(</NAME>
                <OUTCOME>Failed </OUTCOME>
                <OUTCOME>Abandon </OUTCOME>
                <OUTCOME> Successful</OUTCOME>
                <OUTCOME> Excellent </OUTCOME>
        <DEFINITION>
                <FOR> The average learning time to each topics </FOR>    ← Root Node
                    <TABLE> 163<TABLE>  ← The number of happens
        </DEFINITION>

        <DEFINITION>
                <FOR> Excellent</FOR>      ←Children Node
                <GIVEN> The average learning time to each topics </GIVEN> ←Parents Node
                <ATRIBUTE>More than 2.57 minutes </ ATRIBUTE >   ←The value of the Parents Node
                <TABLE> 34 4</TABLE>                ← The number of positive results and negative results
        </DEFINITION>
        <DEFINITION>
                <FOR> Failed</FOR>      ←Children Node
                <GIVEN> The average learning time to each topics </GIVEN> ← Parents Node
                < ATRIBUTE > More than 0.74 Less than 2.57 minutes </ATRIBUTE > ←The value
                <TABLE> 48 2</TABLE>                ← The number of positive results and negative results
        </DEFINITION>
        <DEFINITION>
                <FOR> Abandon</FOR>        ←Children Node
                <GIVEN> The average learning time to each topics </GIVEN> ←Parents Node
                < ATRIBUTE > Less than 0.74 minutes </ATRIBUTE > ←The value of the Parents Node
                <TABLE> 67 8</TABLE>                ← The number of positive results and negative results
        </DEFINITION>
</Tree>
```

Figure 8. Example of the data structure of the constructed decision tree

Predict a learner's future status "*Pred_status*" from his/her learning histories data.

1. Let *DIFF* be a set of different explaining variables between the target learner and the past "Excellent" learners.
2. Let *Pred_status* be a predicted learner's future status node (1. Failed, 2. Abandon, 3. Successful, 4. Excellent).
3. Let Anc_j be the *j*_th previous node from *Pred_status* (*j*=number of the ancestor nodes; *j=1*: parent, *j=2* : grandparent,..., Root node).
4. Let $Desc_j$ be the set of descendant nodes of the Anc_j.

Input Decision Tree, a target learner's learning histories data

Output Instructional messages

```
if "Excellent" ∈ Pred_status
  then
  return complement messages.
  else
    for (j ≦ number of the ancestor nodes of Pred_status) do
      if "Excellent" node ∈ Desc_j
      then
        Search the shortest path V from the Anc_j to "Excellent" node ∈ Desc_j .
        Append the nodes set which the path V includes to the DIFF.
        break;
      end if
    end for
  return the messages corresponding to the nodes ∈ DIFF.
end if
```

Figure 9. Instructional message generation algorithm

model. <ATRIBUTE> refers to the values an explaining variable takes. <OUTCOME> means the values an object variable takes. <DEFINITION> corresponds to the node structure. For a target variable defined by <FOR>, a parent variable in the tree is expressed by <GIVEN>. <TABLE> means a table that has a number of positive instances and a number of negative instances.

4.4. Message generation algorithm

Figure 9 shows the algorithm to generate instructional messages in the proposed intelligent agent system. According to the algorithm, the system first predicts the learner's future status from his/her current learning histories data using the constructed decision tree, and if the predicted status is "Excellent" the system sends complimentary messages to the learner. Otherwise, the system searches the nearest ancestor node whose descendant node has an "Excellent" node. If the system finds the ancestor node which has an "Excellent" descendant node, the system searches for the "Excellent" node which has the shortest path from the ancestor node. If t~~here~~ ~~exists~~ several "Excellent~~-~~" nodes exist which have the shortest path length from the ancestor node, the system selects the first searched for "Excellent" node to generate instructional messages.

Next, the system selects a set of nodes that form a path from the ancestor node to the "Excellent" node and generates instructional messages corresponding to the ~~–~~set of ~~–~~node~~s~~ variables according to Table 1. If there are several instructional messages corresponding to one node variable, the system selects a message using a random number.

This algorithm is installed into "Samurai" using Java. This system can create 2,048 patterns of adaptive instructional messages to learners, such as the one shown in Figure 5; thus, it is expected to adaptively correspond to various learner statuses.

5. Comparative prediction experiments

Some previous studies have been done on predicting a learner's final test score using several machine learning methods from learning history data in e-learning. Minaei-Bidgoli, et al.[17] compared prediction performances of machine learning methods (decision tree model, Naive Bayes, and SVM) to predict a learner's final test score from learning history data in e-learning. The decision tree showed the best performance in the results. On the other hand, Talavera and Gaudioso [18] and Hamalainen et al. [19] conducted similar experiments and insisted that Naive Bayes was the best model.

Table 2. Correct prediction rates(%) obtained in the cross-validation experiment

NC	DT	SVM	Naïve Bayes
2	75.00(88.70)	80.75(89.25)	75.50(76.25)
3	80.00(84.75)	81.00(88.7)	76.00(77.25)
4	82.00(88.75)	74.00(91.5)	77.00(77.75)
5	80.25(84.75)	78.76(91.5)	76.75(77.75)

Note NC: number of categories; DT: decision tree model using the ID3 algorithm. The parenthetical values indicate the fitting rate of the training data.

Finally, Huang et al.[20] found that SVM was the most effective model. Thus, these previous studies reported different results~~,~~ which means that the predictive performance depends on the characteristics of the data (the kinds of variables, data size, domain, learners' age, and so on). Therefore, we also needed to evaluate various models with respect to data obtained from the LMS "Samurai" just as the previous studies did. We compared the decision tree model with the ID3 algorithm Naive Bayes model, and SVM. Here, we employed the most popular Naive Bayes model, the "multivariate Bernoulli model"[23] and a well known SVM that has a "polynomial kernel"[24].

First, the latest data from 800 learners were randomly sampled from the learning history database for 128 courses in the LMS "Samurai". Furthermore, learner history data from 400 out of the 800 learners were randomly sampled as training data, and the remaining 400 learners' history data were used as validation data (test data) for a cross-validation experiment. The cross-validation experiment was performed to predict learners' final status from their learning history data. Decision tree and Naive Bayes models use only categories variables as input data, but the learning history data use continuous variables data. Consequently, the continuous variables data in the learning history data were categorized to be uniformly distributed in each category. Although SVM can use the continuous variables data for input data, this experiment applied the categorized data to SVM under the same conditions as the other models. Here SVM employed the polynomial kernel as a kernel function. To categorize the input data, the range (from the minimum value of data to the maximum value of data) of each variable was divided by the number of the categories m into the category ranges. As a result~~s~~, the continuous data were transformed to category data x_{icj} (if the i-th variable's category c's range includes j-th learner's data then $x_{icj}=1$, otherwise $x_{icj}=0$),($i=1,\cdots,9$, $c=1,\cdots,m$, $j=1,\cdots,N$). The number of categories for all variables was changed from two to five in the experiment.

The results are listed in Table 2. Each value indicates the correct prediction rates of the cross-validation given the number of categories in the corresponding model. When there is a small number of categories, SVM shows the best performance. However, it is clear that SVM over fits the data when there are four or more categories. Antagonistically, although the decision tree model shows lower performance than SVM when there are a small number of categories, it shows the best performance with four or more categories.

Although Naïve Bayes shows lower correct prediction rates, the reason is that the explaining variables that are used all have a mutually strong correlation; nevertheless, the model assumes the variables are conditionally independent respectively.

From these results, the decision tree is the most suitable for the data stored in LMS "Samurai" because the proposed agent needs to use four categories as variables.

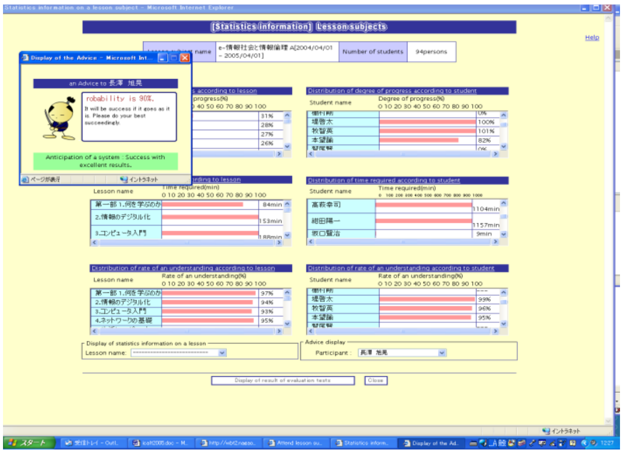


Figure 10. Feedback for a teacher

6. Feedback for teachers

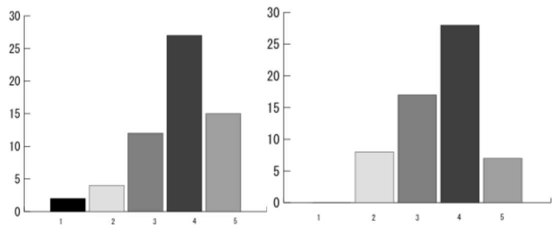
The proposed LMS can also provide feedback on all learners to a teacher, as shown in Figure 10. In this system, the feedback comprises the degree of the learning progress, the learning time, and the rate of understanding for each learner. In addition, this system also presents the current instructional messages to the teacher that the agent has sent to each learner.

7. Evaluations

The system was evaluated by comparing a class of students that used the agent system with one that did not for one semester. The decision tree for the agent system was learned using 1,344 learners' histories. The details of the two e-learning classes are summarized in Table 3. The results show that far fewer students withdrew from the class if they had used the LMS with the agent system. In addition, the final test scores, learning time data, and progress of learning data also indicate that the proposed agent system enhanced learning significantly.

Table 3. Comparison between classes with and without the system

	With agent system	Without agent system
Subject name	Information & Communication Technology	Information & Communication Technology
Students	Undergraduate students (third and fourth year)	Undergraduate students (third and fourth year)
Learning place	Each student’s home	Each student’s home
Credits	2	2
Number of students	74	92
Term	2003, April 10 - July 31	2004, April 10 - July 31
Number of students who withdrew from the course	14 (18.9%)	49 (53.2%)
Final test scores	Average: 93.26 Variance: 43.2 (n=60)	Average: 78.74 Variance: 215.24 (n=43)
P-value	1.33E-07	
Total learning time (minutes)	Average: 1045.13 Variance: 71721.8 (n=60)	Average: 801.88 Variance: 65426.9 (n=43)
P-value	1.25E-05	
Average degree of progress of lesson	Average: 0.93 Variance: 0.64 (n=60)	Average: 0.84 Variance: 2.03 (n=43)
P-value of the statistical difference test of two averages	0.00031	
Total number of contributions to discussion board	714	928



(a) (b)

Figure 11. Plotted results of Question **A** given to (a) the class with the system and (b) the one without it

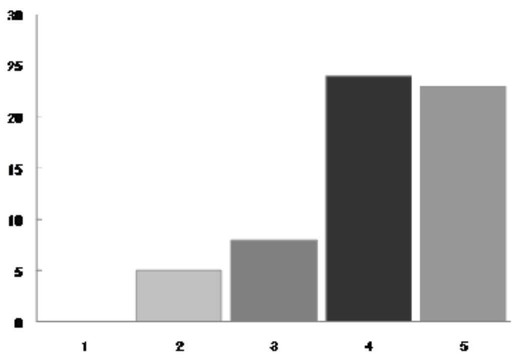


Figure 12. The results of Question **B**

The presentation of the predictive learner’s future status and the presentation of adaptive instructional messages help learners maintain the required learning pace. As a result, the learners can progress until they reach their predicted future status.

Furthermore, all learners were asked Question **A:** **“**How would you rate the system’s ability to enhance your e-learning? 1. Very poor; 2. Poor; 3. Fair; 4. Good; 5. Very good.”

The group with the agent system was asked an additional question, Question **B:** “How would you rate the adequacy of the instructional messages from the agent system? 1. Very bad; 2. Bad; 3. Fair; 4. Good; 5. Very good.”

The results for the Question **A** are shown in Figure 11. Response frequencies of answers 2 and 3, "Poor" and "Fair" were less for the class with the system than for the one without it. This indicates that the system is effective in enhancing learning and the instructional messages have a positive effect on e-learning. However, it should be noted that the response frequency of "Very poor" increased for the class with the system. If we assume that the difference between the results for the two classes are due only to the agent system use, the results mean that learners' opinion about the agent system tended to be polarized compared to the opinions of the class without it .

Figure 12 summarizes the learners' responses frequencies to Question **B**. The results show that many learners rated the agent system's messages as "Good" or "Very good" and this means that the instructional messages from the agent system are acceptable for many learners. However, it should be noticed that five learners rated it as "Bad".The learners who rated the system as "Bad" gave the following reasons :

- " The messages from the agent were distracting. I didn't concentrate on my learning due to the agent's constant actions."
- "The messages from the agent were meddling because I previously knew my-self almost all the messages content even if the agent didn't send them."

This means that the messages from the system are sometimes meddling for some autonomous learners who can learn by themselves. Therefore we think that the system needs a function whereby learners can hide the agent from the system whenever they want.

8. Conclusion

This paper proposed an LMS in which an intelligent agent provides effective adaptive messages to learners using learning history data and data mining techniques. The unique features of this system are as follows.

- The agent builds a learner model automatically by applying the decision tree model.
  The agent predicts a learner's final status (Failed; Abandon; Successful; Excellent) using the learner model and his/her current learning history data. The constructed learner model becomes more precise as the amount of data accumulated in the database increases.
- The agent compares a learner's learning processes with the "Excellent" status learners' learning processes in the database, diagnoses the learner's learning processes and generates adaptive messages to the learner.

This paper compared the developed LMS with an LMS without the agent system through actual university e-learning classes for one semester. The results showed that the number of students who withdrew from the class was significantly lower than in the case of the LMS without the agent system. In addition, the results showed that the average score on the final test was significantly higher when the agent system was used. Some questions and interviews with the learners showed that the agent system enhanced learning motivation and was instrumental in learners' maintaining the required learning pace. Thus, the results demonstrate that the agent system is very effective in maintaining learners' motivation in e-learning.

In addition, it is important to note that in practical use we should not use the automatically constructed tree structure without reviewing it. This is because some teachers are not earnest in facilitating e-learning, and the automatically constructed tree structure is not valid for e-learning. For example, some teachers give a final result of "Excellent" to all learners without deliberation, and the constructed tree's structure shows that any learner might be predicted to be "Excellent". If we consider the constructed tree structure to be invalid, we use the typical structure in Figure 4 instead of the invalid structure. This procedure is also adopted when there is no data and no structure because the course has never been provided before.

Acknowledgement

This research was supported by Grant-in-Aid for Scientific Research of Japan Society for the Promotion of Science: Scientific Research on Priority Areas (B) 19300275, Maomi Ueno, Development of Intelligent LMS with learning agents from learning histories database.

References

(1) Lee, C.Y.: Student motivation in the online learning environment, Journal of Educational Media & Library Sciences, Vol. 37, No. 4, pp.365-375 (2000).

(2) Keller, J.M.: Motivation in cyber learning environments, International Journal of Instructional of Educational Technology, Vol. 1, No. 1, pp.7-30 (1999).

(3) Visser, L. and Keller, J.M.: The clinical use of motivational messages" An inquiry into the validity of the ARCS model to motivational design, Instructional Science**,** Vol. 19, pp.467-500 (1990).

(4) Visser, L., Plomp, T., and Kuiper, L.: Development research applied to improve motivation in distance education, Proc. the National Convention of the Association for Educational Communications and Technology, Houston, TX pp.17-28 (1999)

(5) Gabrielle, D.M.: The effects of technology –mediated instructional strategies on motivation, performance, and self directed learning, Proc. of ED-Media, pp. 2569-2575 (2000).

(6) Ueno, M.: "Intelligent LMS with an agent that learns from log data." In G. Richards (Ed.), *Proc. World Conference on E-Learning in Corporate, Government, Healthcare, and Higher Education 2005*, pp.3169-3176, (2005).

(7) Ueno, M.: Environments of learning by using Internet, in Advanced Research in Computers and Communications in Education, eds. G. Cumming et al., pp.748-571, IOS Press, Amsterdam, (1991).

(8) Ueno, M., Ohnishi, H., and Shigemasu, K.: Proposal of a test theory with probabilistic network, Electronics and Communications in Japan, John Willy and Sons, Inc., vol.78, No. 5, pp.54-66, (1995).

(9) Ueno, M.: Intelligent tutoring system based on belief networks, Proc. IEEE International Workshop on Advanced Learning Technologies (IWALT2000), Palmerston, New Zealand, pp.141-142 (2000).
(10) Ueno, M: Student models construction by using information criteria, Proc. IEEE International Conference of Advanced Learning Technologies, Madison, WI, pp. 331-334 (2001).
(11) VanLehn, K. and Martin, J.: Evaluation on an assessment system based on Bayesian student modeling, International Journal of Artificial Intelligence and Education, Vol. 8, No. 2, pp.179-221 (1998).
(12) VanLehn, K., Niu, Z., Siler, S. and Gertner A.: Student modeling from conventional test data": A Bayesian approach without priors. Proc. the 4th Intelligent Tutoring Systems ITS'98 Conference. Springer-Verlag, pp.434-443 (1998)
(13) VanLehn, K. & Niu, Z.: Bayesian student modeling, user interfaces and feedback: A sensitivity analysis", International Journal of Artificial Intelligence in Education , Vol. 12, pp.421-442 (2001).
(14) Reya, J.: Student Modelling Based on Belief Networks, International Journal of Artificial Intelligence in Education, Volume 14 , Issue 1 pp.63-96 (2004).
(15) Quinlan, J.R.: Induction of decision trees, Machine Learning, Vol. 1, pp. 81-106 (1986).
(16) Becker, K. and Vanzin, M.: Discovering increasing usage patterns in web based learning environments, Proc. Int. Conf. on Utility, Usability, and Complexity of e-information Systems, pp. 57-72 (2003).
(17) Minaei-Bidgoli, B., Kashy, D.A., Kortemeyer, G. and Punch, W.F.: Predicting student performance: an application of data mining methods with an educational web-based system, . Proc. 33rd Annual Frontiers in Education (FIE 2003), Vol. 1, Boulder, Colorado**,** pp.T2A-13-18 (2003).
(18) Talavera, L. and Gaudioso, E.:Mining student data to characterize similar behavior groups in unstructured collaboration spaces, Proc. of the Workshop on Artificial Intelligence in CSCL, 16th European Conference on Artificial Intelligence, ECAI2004, pp.17-23, (2004).
(19) Hamalainen, W., Laine, T.H., and Sutinen, E. Data mining in personalizing distance education courses", in Data Mining in e-Learning, eds. Romero, C. and Ventura, S., WIT Press, Boston, pp.157-171 (2006)
(20) Huang, C.J., Chu, S.S. and Guan, C.T.: Implementation and performance evaluation of parameter improvement mechanisms for intelligent e-learning systems, Computers & Education, Vol. 49, No. 3, pp. 597-614 (2007).
(21) Ueno, M.: Learning Log Database and Data Mining system for e-Learning, Proc. International Conference on Advanced Learning Technologies, Joensuu, Finland, pp.194-201 (2002).
(22) Mayer, R.E. and Anderson, R.B.: Animations need narrations, Journal of Educational Psychology, Vol. 83, No. 4, pp. 484-490 (1991).
(23) Domingos, P. and Pazzani, M.: On the optimality of the simple Bayesian classifier under zero-one loss, Machine Learning, Vol. 29, pp.103–137 (1997).
(24) Vapnik, V.: The Nature of Statistical Learning Theory, Springer , New York, (1995)