ANALYZING AND CLASSIFYING A RANGE OF INCORRECT ACTIONS MADE BY STUDENTS DURING AN EDUCATIONAL PROCESS USING AN INTERVAL TEMPORAL BEHAVIOR OBSERVATION

Mihail Petrov

Faculty of Mathematics and Informatics, Plovdiv University, Bialo more 5, Plovdiv 4000, Bulgaria

Abstract

Training is an incremental process involving slow and methodical achievement of small goals over a given period of time. Each step in the process involves assimilation of matter by accepting a set of errors, the correction of which leads to the improvement of knowledge and deepening the understanding of the problem area and matter as a whole.

The classical learning approach, where the facilitator is part of the learning process and has the technical ability to test the work of trained agents, allows for easy and effective classification of a set of errors to serve as the basis for changing and improving the material. E-learning suffers from this disadvantage that the process supervisor does not have this opportunity due to the lack of real-time action as well as the huge scale of these training sessions.

In this article, we will look at a mechanism for classifying a set of errors used in the UniPlayground project to classify the behavior of trained agents based on their omissions, errors, or failure to conform to the particulars of the course material. The article examines learning in the context of software development.

Key words: *eLearning, Intelligent Agents, ITL, virtual education space, behavior analysis*

1. INTRODUCTION

The main focus of modern education is the innovative approach in electronic and virtual environments. Media is a source of new opportunities that are derived from the classic classroom model. E-Learning allows access to information to an unlimited number of users in a convenient time using a convenient device that can receive any information resource in an unlimited number of times from anywhere in the world. Of course, it is not possible to implement a mirroring methodology in organizing e-learning so as to provide the same results as the classical model of teaching due to the following features: [1]

1.1 The scale of the audience

E-learning implies access to content to any number of trainee agents who are not subject to preconditions such as education, age, and interests. Freelance learning can be consumed within defined course requirements, but the outcome is usually subordinate to the audience who has consumed the knowledge provided in the curriculum. The difference between training a compact course of 30 people before having an unfamiliar audience quickly leads to the inability to analyze the difficulties of trained agents and hence the inability to adequately classify the problems arising from the complexity or completeness of the curriculum.

The impossibility of objectively building such an assessment leads to stagnation of course quality and quite easily positions it to the extreme as extremely easy or extremely complex to understand the learning material [10].

1.2 Distance of the audience

Another problem directly related to the scale of the problem arises from the fact that the trainees do not physically share the same territory with the lecturer. The facilitator of the course could not know his audience due to the specifics of this type of training. A remote connection can create a lack of communication that makes communication difficult and makes the learning process transparent, very complex. It is also important to note that there are also curricula that suggest that the learning material

is pre-recorded, and the learners only follow the notes and solve a set of pre-prepared tasks following the course material. In order to address some of these problems, it is necessary to resort to the use of the tools needed to provide the knowledge and skills in the context of the training [10].

In this article, we will look at the approach we use in the **UniPlayground** integrated system project to analyze the behavior of training agents in an e-learning process. The system is integrated and successfully used for the needs of the educational process within the **Faculty of Mathematics and Informatics of the Plovdiv University "Paisii Hilendarski"**. The platform helps teach dozens of students who choose to participate in the classroom electronically rather than attending, providing accessible and open media for further development of students' practical skills. In a broad sense, the platform can be integrated for the needs of random e-learning. We will look at the concept of classifying situations when solving practical problems and how this classification helps us to extract information to help improve the learning process of agent training in the context of an electronic system.

2. MEANS OF MEASURING THE UNDERSTANDING OF LEARNING MATERIAL

Traditionally, the learning process involves self-preparation. The learning material is accessible to the trainee agent in an arbitrary form (written resources, video lectures, live consultations). Measurement of knowledge is done through some form of testing that is traditionally in writing for the purposes of archiving and long-term storage. [2]

2.1. Tests

In the electronic environment, the easiest and fastest way to test the level of understanding the material is to test the test in which the learners are asked a set of predefined questions with a number of possible answers. This approach ensures a rapid collection of common statistics and relative accuracy in determining the relative accuracy of true responses to candidates' responses. The main problem to be avoided in this way of testing is the elimination of global accusation, where response points are obtained by the exclusion method or by incomplete knowledge. Tests have the potential to avoid this problem if they compose the individual questions well enough and systematically so that responses imply an incremental upgrade of the complexity and problem area. [3]

Testing of knowledge is a preferred approach to a large number of certification exams conducted by major technology organizations such as Oracle and Zend, but often the questions may be misleading or not complete enough to make a clear distinction between which of the possible answers, correct.

The **UniPlayground** system uses the knowledge verification test only and only at the beginning of a new course, thus creating a basis point of reference that serves as a unit of analysis of the results achieved. Tests are not the main mechanism for analyzing the knowledge of trained agents because the system aims to analyze behavior by writing programming code, and in this direction, tests are not a suitable measure.

2.2. Practical assignments

There are disciplines where traditionally theoretical knowledge is often rarely measured than skills to deal with factual problems. Such are the disciplines of the nature-mathematical nature, the engineering sciences, the informatics, the applied programming and others [4].

The practical assignment relates to the solution of a specific technical problem from the knowledge sphere, the result of which is to answer a specific question or a set of questions. Practical assignments are often related to the development of projects or a group of related tasks in the context of the discipline [9].

Tasks of this type are characterized by high performance in terms of practical understanding of the course material. They give the opportunity to further improve knowledge through learning by doing. Mistakes made in the mistakes of practical assignments often have no final character but can be used as a basis for analyzing the trainee's understanding of the material currently being studied. The practical assignments also contain the necessary theoretical knowledge that the trained agent needs to do with the

material, but they are often refracted through the purely applied aspect and cannot provide data on the depth of theoretical knowledge.

2.3. Case study

One of the most complex forms of verification of the level of material absorption is related to conducting a study within several iterations of the same curriculum taking into account the following features

- The specificity of the study material. To what extent did the subject matter correspond to the average level of the institution being conducted and to what extent the training complies with the standards and expectations of the trained audience
- Specifics of the learning group. To what extent is the audience prepared to understand the information in the course. Do all the participants in the learning process have the necessary level of language training, do they have the necessary knowledge to deal with the initial stage of the educational process, or have introductory classes.
- The specificity of the group's learning environment. This point is particularly relevant when conducting practical training. Always take into account the software product that will be used for the needs of the classes, and the pace of the classes is also dependent on its complexity.

The iterations of the same curriculum taking place within similar inputs can largely conclude on the success of the curriculum on the basis of the results achieved.

A case study is one of the most appropriate models for conducting training courses in the context of eLearning. E-learning, by concept, implies flexibility and diversity and can be targeted at a wider range of subjects with a variety of characteristics, thus making it quite easy to obtain the data needed for the success of the program, given the multitude from a variety of factors. In this way, the calibration of the course material can always follow the average score of the participants if they aim at average levels of achievement, or to fine-tune the curriculum or the curriculum by selecting agents with specific knowledge of the material being taught.

3. DEFINITION AND CLASSIFICATION OF INCORRECT ACTIONS

Incorrect states or errors can be classified into different categories depending on the subject area in which the training is conducted. For the purpose of this article, you will define a number of error types by explaining how they are positioned and accumulated over time and space, and why their classification is relevant to the analysis of the results achieved.

3.1. Incorrect result

The primary category is the error where the result is false. When the assignment implies a strict, non-interpretable response, it is very easy for both the facilitator and the information system to determine whether the response is correct or not [5]. No solution analysis is foreseeable within predictable limits, which easily classifies the assumption of error. The reasons for admitting an error in this category can be classified as follows:

- the wrong result - the problem of solving the problem is a consequence of a mistake made in a previous segment
- error in solving the problem - arbitrary to the first condition, this error may be technically inadvertently allowed, but the overall approach to reaching the correct answer is correct
- error in understanding the material - the approach to solving the problem is wrong given no understanding of the theoretical problem

It is important to address third type errors because they suggest a problem in the course material.

3.2. Syntactical errors

Syntax errors are all mistakes that can be made when the commands needed to control program processes are incorrectly written. A program component containing syntax errors cannot be executed and get an adequate result. The syntax errors are intercepted by an agent called compilation agent that takes care of the syntactic integrity of the executed program unit

Program languages define a set of rules and keywords, the violation of which results in the inability to perform properly, and hence the inability to start the program as a whole [9].

The reasons for allowing syntax errors can be reduced to the following groups:

- ignorance of the programming language and rules for the consistent design of program units
- inattention when writing the code, fast writing without checking the result

4. GRAPH CLASSIFICATION OF INCORRECT ACTIONS

In order to be able to derive sufficiently detailed information from the error, it is necessary for the person to have the following characteristics

- The type of error
- The status of the code at the time of the error
- Previous system states before committing a specific error, regardless of whether they are correct or incorrect.
- All statuses of the code from the moment the error is accepted to the next correct state
- The time at which the condition verification operation was performed.

In order to be able to easily manage the states, all input data is arranged in the form of a graph, from states containing the information described [5].

Let's look at the following task, defined for a database programming course. The students participating in the course are faced with a problem whose solution is to be defined by programming code. The person is the following source of statistical information that will be used to solve the task.

Id	fname	lname	departmen
1	Anton	Alvary	Sales
2	Nelson	Filin	Software
3	John	Mansoni	Software
...	..	..	..

Fig. 1. Sample data for example task

Let us define all possible states where the task can be defined as correct. The state that defines the correct task is a statistical sample that visualizes the following entries. We are going to represent the set of the correct data by using the same visual tool aka table of content. This representation is the same as the common representation of tables in SQL development so structurally and logically it is using the same concept for describing the valid data scenario.

Id	fname	lname	departmen
2	Nelson	Filin	Software
3	John	Mansoni	Software

Fig. 2. Successful solution to the task

The options where the task can be solved using program definitions may vary depending on the learner's approach and the type of learning material currently being studied. However, the result can be verified experimentally if the sample presented corresponds to the above scheme. Let's identify all the states where learners can make a mistake and determine why this is important for the end result [6]. Status of an incorrect solution where a syntax error has occurred. The code that solves a task is semantically correct, but syntax errors lead to a situation where the code cannot be properly activated by the system due to conflict with the correct way of writing the symbol constructs.

An error of this type demonstrates:

- misunderstanding the programming language
- inattention when writing the constructions

```
DECLARE
l_department_id VARCHAR2(100)
-- Missing; Syntactical error


BEGIN
SELECT department_id FROM w1_departments WHERE department_title = ‘Software’
FOR element IN (SELECT * FROM w1_customers WHERE department_id = l_department_id) LOOP
dbms_output.put_line(element) // Missing ; Syntactical error
END LOOP;
END
```

Fig. 3. Syntactical error classification

An incorrect decision state where an error has been committed depending on the platform on which the programs are running. The development of technology implies the improvement of the tools that are used to intelligently solve program tasks. Normally, the ease and convenience of writing a program design cannot be used in early or specific versions of the platform that is used in the workflow or even for training purposes for a number of reasons:

- Commercial requirements of the platform
- hardware requirements of training machines
- Need to achieve a medium level of platform independence for training purposes.

Such a mistake again does not have a semantic attitude towards the final result and shows the following important features in the solution of the task:

- Ignorance of the platform features of the system used for training purposes. Such a type of error can be finely classified and can again correspond to the obsolescence of the knowledge of the taught material.
- Depending on the level of learners, platform-type errors may indicate the type of material known, thus making a much finer calibration of the technology studied depending on the average set of errors of this kind.

```
DECLARE
l_department_id VARCHAR2(100);
BEGIN
SELECT department_id FROM w1_departments WHERE department_title = 'Software'
FOR element IN (SELECT * FROM w1_customers WHERE department_id = l_department_id) LOOP
dbms_output.enable(null); // Platform dependent error
dbms_output.put_line(element)
END LOOP;
END
```

Fig. 4. Syntactical error classification

It sometimes happens that the differences between platform versions are so drastic that mistakes of this kind can give a clear indication of the level of learners and give guidance to the trainer about the way he/she has to take to make it easier explanations to the current information resource of the trained agents.

A typical example of an incorrect attitude within a program task is when the answer is wrong. In this case, there are no program errors of the type described. The program was written correctly but did not meet the necessary semantic requirements described in the pre-assignment. Answers of this type indicate a misunderstanding of the assignment or experiments on the part of the learner in order to reach the correct answer. For the proper interpretation of such type of errors, it is necessary to analyse a chain of similar states of the system and to check if the last transmitted state is of a semantically wrong type [8].

Semantically improper states have a special place in programming because of the possibility of being ignored by operators. The platform does not display the wrong state of the requirements are not met. if this is the last decision, it clearly shows that the trained agent did not understand the requirements of the task. If the condition is intermediate, it can be transformed from incorrect to experimental and ignored in the final analysis of agent behaviour. Experimental states of semantic errors are easy to detect and can provide additional context information depending on the depth of the experimental graph as well as the distance from the final result as well as the frequency of the semantic errors allowed.

```
DECLARE
l_department_id VARCHAR2(100);
BEGIN
SELECT department_id FROM w1_departments WHERE department_title = 'Software'
FOR element IN (SELECT * FROM w1_customers WHERE department_id = l_department_id)
LOOP
dbms_output.enable(null);
dbms_output.put_line((element.customer_id)  // Requirements eror
END LOOP;
END
```

Fig. 5. Requirement error classification

5. LEVELING THE INCORRECT STATE SEVERITY

When we make mistakes, we learn to better understand the world around us. In the programming environment, making mistakes helps us to better recognize the tools we work with, namely programming language, code technology, and software application architecture. The big problems that are part of the classification of wrong and correct states can be classified into two large groups [7].

5.1. Sequential classification

This is classification based on the sequence of the incorrect and correct state of a problem. The solution of a problem can be separated over a number of simple steps necessary for solving a more complex task that is consisting of the whole solution. In other words, the solution generally comes down to following an algorithm in order to find the proper metrics and find the answer. The math problems are often an adequate example of a problem that can be classified sequentially.

- The number of steps necessary for finding the end result is often determined by the class of the task.
- In order to progress further, you must break the problem over to small micro-tasks generally conditioned by the previously solved task.
- The end result depends on the decisions of the previous tasks

The essential part of the sequential classification is the labeling of every single incorrect state in order to classify the percentage and the severity of every single one of them. We are going to use the example of an incorrect state that is defined in this paper.

Over the process of solving a task, the user can make the following mistakes

- Syntactical
- Requirement
- Semantical
- Platform-specific
- Operational
- Run-time

Every single one of the listed mistakes is defined as an incorrect state and it is a part of a sequential graph labeled with the type of the invalid state and the state of the task. In the sequential classification, we are interested only with the chain of invalid state and not the time between every single state

production. The process of running over multiple invalid states of the same type is labeled as a syntactical cluster and it is analyzed according to this state. There may be many reasons why agents encounter a cluster of invalid states for example:

- Not understanding the syntax of the language
- Not understanding the flow of the program
- Not understanding the language constructs

Of course, not all syntax errors can be assigned to the same category, the following groups can be sifted:

- A syntax error made inadvertently. They are usually associated with the omission of a letter when writing a program construct, the omission of a space character, or a termination symbol. The concept that can be used to analyze such a situation is called *Levenshtein distance* for measuring the distance between the keystrokes of the keyboard and the mistake that the agent made.
- Syntactic errors made by a misunderstanding of language constructs. If the agent has previous experience with another programming language, a relationship between his previous experience and the code he is writing can often be found. The use of language constructs atypical of the language under study can be readily recognized using language compilers. Such errors clearly demonstrate the need for additional guidance on tasks that support the use of language constructs typical of the current programming language.
- Syntax errors not classified in the above two categories. If a linguistic error cannot be determined, then it is the result of a much more complex error in the understanding of matter. Getting an agent into a cluster of invalid states can immediately indicate a global problem in an understanding matter or language constructs. The sequence of incorrect states produces a graph.

5.2. Interval or time-based classification

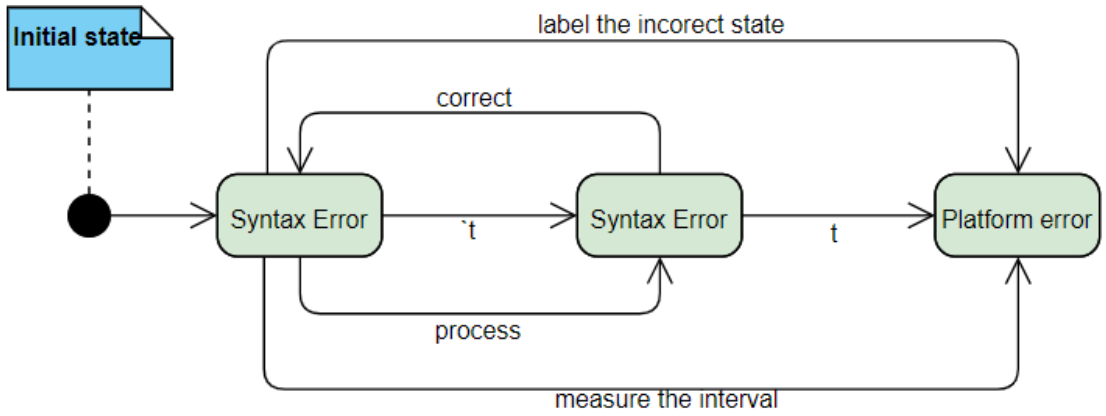

Fig. 6. Linking the incorrect states as an ordered list by leveling the severity

In the figure with a dashed line, the starting state of the column is noted and with two dashed lines the final state at which a successful or experimental state is reached in the system.

- The status column defines only syntax errors of the first type that indicate a misunderstanding of the programming language and spelling mistakes when writing commands. Within the correction graph, we can analyze the following aspects that are part of the environment surrounding the error [6].
- how many mistakes have been made in implementing the program? The single execution of the code leads to the generation of N in the number of errors, the set of {N} of all errors committed leads to the status of the column currently.

- The interval between the individual errors if there is a set of identical errors within a certain period of time. The interval can indicate whether the agent has experimented or obtained the correct syntactic construction or the state of the code over time has led to unrelated errors. In this way, it can be estimated to what extent the trained agent encounters difficulty with a single structure or a set of such.
- The state of the code in the different stages of the error assumption.

6. DISCUSSION

The experimental application testing was conducted using one base group and two experimental groups in the duration of one academic year. The base group consists of students with differential experience in computer science and a different level of knowledge and educational background. The base group experience educational process using only the guides directly influenced by the core facilitator team of the course,

The other two control groups experience the same education process but using only the application processes developed in **UniPlayground** by the same team. The resource utilization and application exercises analysis were approximately 3-time greater compare with the traditional educational process.

The time spend facilitating already developed exercises was cut in half thanks to the build in the analytics process. The single most important improvement was found in the overall record track of incorrect behavior. For every single student, the facilitator has access to the full range of mistakes done over the course of its education and also the range of improvements based on faculty guidance.

There is no doubt that in order for a correct assertion of the provided results it is necessary the corrected data spawn across a significant time span in order for the results to be critically analyzed and assessed. Using the long-term data classification process is necessary for predicting a range of behavior that is going to be observed in the control group consist of students that exhibit a specific range of behavior or have a collection of specific strengths and weaknesses necessary to achieve results over a specific course. The observed results can be generalized and used for a broad range of online education and electronic education at the university or even a company level.

7. CONCLUSION

E-learning is not just an alternative approach to providing a wide range of information resources to a large group of people using the capabilities of modern computing. It also gives us a practical approach to studying the work of learners, thus building dynamic patterns for the development of both the curriculum and the subject matter. We understand the weaknesses and strengths of trained agents and provide them with an alternative based on research and statistics.

In the present study, students from regular and elective training courses at the university have been able to look into the process of their work and analyze their mistakes by literally learning from them. In the next iterations of the study, we will expand the sample of students and integrate the **UniPlayground** platform into a broad range of related disciplines and training courses to analyze the repetition of similar mistakes and whether their burden leads to a change in the learning approach of one learning material studied within the framework of the training courses.

ACKNOWLEDGMENTS

The paper is co-funded by the Project FP19-FMI-002 "Innovative ICT for Digital Research Area in Mathematics, Informatics, and Pedagogy of Education" of the Scientific Fund of the University of Plovdiv Paisii Hilendarski, Bulgaria.

REFERENCES

1. M. Samir Abou El-Seoud, Islam A.T.F. Taj-Eddin1, Naglaa Seddiek1, Mahmoud M. El-Khouly, Ann Nosseir E-Learning and Students' Motivation: A Research Study on the Effect of E-Learning on Higher Education
2. Rakesh Agrawal and Ramakrishnan Srikant Fast algorithms for mining association rules. Proceedings of the 20th International Conference on Very Large Data Bases, VLDB, pages 487-499, Santiago, Chile, September 1994
3. Rodriguez, J. J.; Kuncheva, L. I.; Alonso, C. J. (2006). "Rotation forest: A new classifier ensemble method". IEEE Transactions on Pattern Analysis and Machine Intelligence. 28 (10): 1619 – 1630. doi:10.1109/TPAMI.2006.211
4. L.K. Hansen and P. Salamon, "Neural Network Ensembles," IEEETrans. Pattern Analysis and Machine Intelligence, vol. 12, no. 10, pp. 993-1001, Oct. 1990.
5. T.K. Ho, "The Random Subspace Method for ConstructingDecision Forests," IEEE Trans. Pattern Analysis and MachineIntelligence, vol. 20, no. 8, pp. 832-844, Aug. 1998.
6. T.K. Ho, "A Data Complexity Analysis of Comparative Advan-tages of Decision Forest Constructors," Pattern Analysis and Applications, vol. 5, pp. 102-112, 2002.
7. R.E. Schapire, "The Boosting Approach to Machine Learning: AnOverview," Proc. MSRI Workshop Nonlinear Estimation and Classi- fication, 2002.
8. R.E. Schapire, Y. Freund, P. Bartlett, and W.S. Lee, "Boosting theMargin: A New Explanation for the Effectiveness of VotingMethods," Annals of Statistics, vol. 26, no. 5, pp. 1651-1686, 1998.
9. Seifried, Jürgen; Wuttke, Eveline Student errors: how teachers diagnose them and how they respond to them
10. Ferruccio Mandorli, Herald Otto, Systematic Support of Learning from Errors and Negative Knowledge Development in MCAD Education: Empirical Analysis of Student Feedback