Student modeling in real-time during self-assessment using stream mining techniques

Zacharoula Papamitsiou, Anastasios A. Economides
Interdepartmental Progr. of Postgraduate Studies in Information Systems, University of Macedonia
Thessaloniki, GREECE
papamits@uom.edu.gr, economid@uom.gr

Abstract— **In order to personalize the assessment services, the assessment systems need to build suitable student models for heterogeneous student populations. The present study focuses on efficiently modeling students according to their time-varying behavior during web-based self-assessment, enriching the models with a notion of dynamics. The suggested approach forms and revises the student models on-the-fly, using three popular stream mining classification techniques. All methods use specific time-based features as predictors, and the students' self-assessment achievement levels as target values. The obtained results demonstrate that level of certainty, effort and time-spent on answering correctly/wrongly could contribute to pursuing fine-grained and robust student models during self-assessment.**

Keywords-assessment analytics; dynamic student modeling; response-times; stream mining; supervised classification

I. Introduction

Reconsidering the personalization of quality learning and assessment services has emerged from the expanding enrollment of large numbers of students in technology-mediated learning environments [1]. Towards this goal, creating more granular and solid student models is a prerequisite [2]. A student model summarizes multiple student's characteristics, either static (gender, ethnicity, etc.) or dynamic (cognitive skill, emotions, etc.), extracted from diverse data sources into a profile representation [3].

This study focuses on student modeling in the context of self-assessment. The need to accurately diagnose students' abilities and needs is more imperative in self-assessment conditions. The reason is that self-assessment leads students to a greater awareness, by training them to self-regulate their motivation and behavior, as well as by fostering reflection on their own progress in knowledge, skills, or processes, and finally, to understanding themselves as learners [4].

In order to support students to become better learners, we need to deliver to each individual the most appropriate self-assessment material. For this purpose, we need to develop personalized self-assessment systems that compile the students' attributes into fine-grained student models. Next, these systems will consult the student models to facilitate the assessment material selection process. Thus, the information included in the student models needs to be selected upon rigorous criteria and has to be gradually incorporated.

Towards this information "filtering" and "integration", assessment analytics could shed light to what should be included or not; an assessment analytics procedure "monitors, tracks and records data related to the context, interprets and maps the current state of these data, organizes them (e.g., filter, classify, prioritize), uses them (e.g., decide adaptations, recommend, provide feedback, guide the learner) and predicts the future state of these data" [5, p.118].

This study introduces a methodology for building and updating dynamic student models during web-based self-assessment by making practical use of assessment analytics. The core idea is to classify students "on-the-fly", using as predictors specific time-varying student's features that (a) are strongly related to the self-assessment process, and (b) are good estimators of students' knowledge, skills and abilities. In this paper, we benefit from advances in stream mining for student classification, and contribute to creating dynamic and robust student models. Thus, the research question is:

"Can we build dynamic student models during web-based self-assessment based on students' time-driven features?"

In order to address this question, we conducted a study with the LAERS self-assessment system. Five hundred and three (503) undergraduate University students enrolled in a self-assessment procedure and were classified in real-time.

The rest of the paper is organized as follows: in section II, we briefly review existing work regarding student modeling in self-assessment contexts. In section III, we briefly present the LAERS self-assessment system used in this study, as well as the basic features of the student models in LAERS. Section IV explains the methodology, the data collection process and the classification methods. Section V demonstrates the results and elaborates on our findings. Finally, section VI focuses on our conclusions and describes our future work plans.

II. Related Work

Over the past decades, student modeling has attracted increased interest as a research topic. Significant results have been demonstrated and utilized for the personalization of e-learning systems. These results originate mostly from the fields of intelligent learning environment [6] and cognitive tutors [7]. Considerable findings on student modeling are also evidenced in domains like MOOCs [8], gamified learning environments [9], and virtual learning environments [10]. Performance, goals, prior and acquired domain knowledge, as well as learning strategies, preferences and styles are among the most popular dynamic students' characteristics [11] [29]. Communication and collaboration skills, critical and analytical thinking, motivation and meta-cognitive skills on a specific domain or topic and affective states are also commonly used to complement the students' profiles [2]. In

2161-377X/17 $31.00 © 2017 IEEE
DOI 10.1109/ICALT.2017.90

[2] and [11] the authors conducted a review on key student models that have granted the success of intelligent learning environments, and a survey on student modeling, focusing on what to model, how and why, respectively.

In the context of self-assessment, the prevalent approach was to open the student models to students, visualize students' progress, and measure students' self-reflection on their own skill mastery (e.g., [12] [13]). One of the objectives was to seek for students' preferences, associate them to students' knowledge and performance, and produce graphical views of this model to enhance students' self-awareness and encourage them to participate in the modeling process [12]. In a "social" version of this approach, the researchers displayed parallel views of the students' models with the cumulative model of the entire class [13]. From another scope, estimating students' mood while undertaking the self-assessment and modeling the respective affective states [14] was also explored.

The students' attributes considered in the models included knowledge, cognitive and meta-cognitive skills [12] [13] [15] and affective states [14]. The models were constructed using overlay [12] or its combination with stereotypes and with fuzzy rules to keep the student models updated [15], or combined formula-based and neural network methods [14].

Apparently, the reported experience on student modeling in self-assessment contexts and the practical evidence on revising the student models dynamically is limited. This study introduces a mixture of time-varying features going a step further from traditional approaches (e.g., [12] [15]), and applies stream mining for revising the student models on-the-fly by considering changes in the states of the these features.

III. The LAERS Assessment Environment and the Basic Student Model Features

A. The LAERS assessment environment architecture

The Learning Analytics and Educational Recommender System (LAERS) [16] is a web-based self-assessment system consisting of (a) a testing interface, (b) a tracker that logs the students' interaction data, (c) a student modeling engine that shapes/revises the student models, (d) an adaptation engine that provides personalized feedback to the students, and (e) a database storing information about the students and the items.

The first component implements the interface that displays the self-assessment items delivered to students separately and one-by-one. The interface delivers the items to the students in predetermined order, and it allows them to temporarily save their answers, to review them, to alter their initial answer choices, and to save new answers. Students can also skip an item (because they are not sure about the answer, or because they think it is too difficult), and answer it (or not) later.

The second component records the students' interaction data during self-assessment. In log files, it tracks and aggregates students' time-spent on handling the self-assessment items, breaking it into the time-spent on correctly and time-spent on wrongly answered items. The tracker also logs how many times the students reviewed each item, how many times they changed the answers, and the respective time-spent during these interactions. The overall logged features of students' activity are listed in Table I.

TABLE I. Features from the Raw Log Files

Feature	
1. student ID	*10. the total time the student spends on viewing the items - submitting the correct answers*
2. the item the student works on	*11. the total time the student spends on viewing the items - submitting the wrong answers*
3. the timestamp the student starts viewing an item	*12. the idle time the student spends viewing each item (not saving an answer)*
4. the timestamp the student leaves an item (not saving an answer)	*13. the total idle time the student spends on re-viewing the items*
5. the timestamp the student saves an answer	*14. the student's total idle time on item*
6. the timestamp the student chooses to re-view an item	*15. how many times the student changes the answer to an item*
7. the timestamp the student saves an answer after re-viewing an item	*16. how many times the student reviews each item*
8. the answer the student saves	*17. how many times the student views the item*
9. correctness of the saved answer	

The system also calculates the score (TS) for each student according to the correctness (0/1) of the student's answer on item *i*, and to the difficulty of the item.

The student modeling engine organizes the data from the data logs for each individual's observed activity and prepares them to be loaded in the student models. The exact student characteristics included in the student models are synopsized in sub-section II.B and the methods applied for the formation of the respective student classes are discussed in section III.

B. The features of the student models in LAERS

Previous studies with the LAERS system structured a measurement model consisting of response-times (e.g., total time to answer correctly/wrongly) and latent factors (e.g. goal-expectancy, level of certainty) in order to predict students' test score (e.g., [17] [18]). The suggested measurement model was found to provide statistically significant explanation of the variance in the test score ($R^2 > .63$), and the participating variables were strongly correlated to the test sore. Thus, they can also be considered as features for student modeling purposes. Furthermore, students' effort (i.e., how engaged the students are during solving a task) is also proposed to be included in the models. Table II presents the features in the student models.

It should be noted that goal-expectancy corresponds to the students' perceptions of their desirable achievement level [19], measured via questionnaire prior to the self-assessment, and as a time-stable variable, was not admitted for student modeling.

TABLE II. List of Features in the Student Models

Variable	***Description***	***Explanation***	***Value***
TTAC	*Total time to answer correctly*	*The response-time a student aggregates on submitting correct answers*	*≥0 (msec)*
TTAW	*Total time to answer wrongly*	*The response-time a student aggregates on submitting the wrong answers*	*≥0 (msec)*
CERT	*Level of certainty*	*How certain the student wants to be – a measure of cautiousness*	*0-1*
RTE	*Response Time Effort*	*When a student exhibited solution behavior – a measure of engagement*	*0-1*

1) Total time-spent to answer the items: Total time to answer correctly (TTAC) and total time to answer wrongly (TTAW) are defined as the total time that students spend on viewing the items and submitting the correct and wrong answers respectively, and have been found to be highly correlated to the test score [17]. They are time-varying variables and indicate the respective response-time the students constantly aggregate on answering the self-assessment items. Therefore, both TTAC and TTAW have been included in the student models to represent the students' time-varying answering behavior.

2) Level of certainty: Certainty describes a person's strength of belief about the accuracy of a choice [20]. In self-assessment procedures, level of certainty (CERT) reflects how certain the students want to be before answering a question; the more certain the students want to be, the more the idle time they spend on re-viewing the items and the more the times they review the items [18]. It is a time-varying latent variable that comprises two sub-parameters: (a) the total idle time the students spend re-viewing each item, and (b) how many times the students review each item. Overall, CERT is a time-varying measure of students' cautiousness during the self-assessment.

3) Effort: According to [21, p. 158], effort is "the motivational state commonly understood to mean trying hard or being involved in a task"; it is about how much engaged the students are in answering the items. Response Time Effort (RTE) measures the proportion of items which the students try to solve instead of guessing the answers [22], according to a threshold value, discriminating solution behavior from guessing. Less engaged students will answer too quickly, before they had time to fully consider the items. In self-assessment procedures where the students usually focus on their attainment, the time-varying effort is a critical factor reflected in the student models.

C. The target classes of the student models in LAERS

The feature space of the student models in LAERS includes the time-varying variables (i.e., TTAC, TTAW, CERT, RTE). The target class is one of the two different cases for the levels of achievement that are available in LAERS, depending on the students' score; (a) two-levels, namely "pass" and "fail", and (b) three-levels, corresponding to the commonly adopted stereotypes, i.e., the "advanced", the "intermediate", and the "novice".

IV. Methodology

A. Research participants and data collection

Data were collected with the LAERS environment at a European University during a self-assessment procedure. Five hundred and three undergraduate students (231 males [45.9%] and 272 females [54.1%], aged 19-28 years old (M=20.21, SD=1.483, N=503)) answered on 45 multiple choice items, prior to their participation to the final exams. We asked two instructors to rate all items for their difficulty (easy, medium, hard). The instructors agreed on the items' difficulty, and each item contributed differently to the overall score, ranging from 0.8 points (easy) to 1.2 points (medium) and to 2 points (hard).

The participation to the procedures was optional. All participants signed an informed consent form that explained to them the procedure and was giving the right to researchers to use the data collected for research purposes. Students were aware that their answers were being tracked, but not their time-spent, because we wanted them to act spontaneously.

B. Stream mining classification methods

A critical problem in real-time applications was the continuous supply of rapidly grown data that evolve over time. This urgent situation led to the rise of the stream mining paradigm [23]. The core assumption is that training examples can be inspected a *single time only*; they arrive in a stream and next, they must be discarded to make room for subsequent examples. A widespread technique for data stream mining is the use of a sliding window to keep only a representative portion of the data. The training window size can be fixed or variable over time [28], and the data is assumed to have a small and fixed number of features and typically less than ten possible class labels, stationary or evolving [24]. In this study, we explored:

1) HoeffdingTree: an algorithm that incrementally induces a decision tree from a data stream, inspecting each example in the stream only once, assuming that the distribution generating examples does not change over time. A HoeffdingTree may choose an optimal splitting feature from a small sample, using the Hoeffding bound to decide how many examples are needed to assure that the chosen attribute using the bound is the closest to the attribute chosen when infinite examples are present into the classifier [25].

2) OzaBag: an online version of Bagging ensembles for data streams that simulates the process of bootstrap replicates to get an aggregated predictor. The probability that any individual example will be chosen for a replicate is determined by a Binomial distribution and tends to a Poisson(1) distribution, because in streams of arbitrary length, the number of examples $N \rightarrow \infty$ [26].

3) Perceptron: an example of reinforcement learning that employs a sigmoid activation function in order to optimize the squared error (having one perceptron per class value) and to minimize the number of misclassified examples [27].

C. Measures and Performance Criteria

Measuring data stream classification performance involves space (the available memory is usually fixed), learning time (i.e., processing incoming examples at the rate they arrive) and accuracy. The most popular evaluation method is the Predictive Sequential (*prequential*) error estimation [28]. The prequential error allows to monitor the evolution of the streaming classifiers and evaluates the performance of the models by testing each example and then using it for training in sequence. Furthermore, the Kappa statistic and the Temporal Kappa Statistic (K_{temp}) measure performance of streaming classifiers [28]. The Kappa statistic measures the agreement of the prediction with the true class. A value of Kappa equals to 1.0 signifies complete agreement. K_{temp} values ranges between (1, -∞) and equals 1.0 if the classifier is accurate. We implemented the stream mining classification in the MOA framework [24].

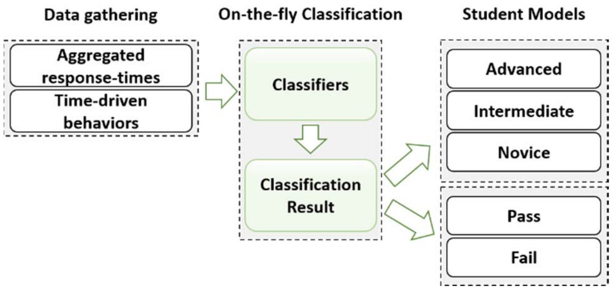

Figure 1. Illustration of the overall student modeling methodology.

Figure 1 illustrates the overall student modeling methodology followed, from the data gathering phase to shaping the complete student models.

V. Results

In this section, we present and discuss the results obtained during the experimental phase, throughout the data stream (sample size: ~2.3MB). We measured the accuracy and learning time for each of the compared classification algorithms. We implemented a fixed window approach and explored two different windows sizes (100, 500). Initially, the window is created with the first *w* labeled records, where *w* is the size of the window. After that every sample is used for classification and evaluation of the performance, and then it is used to update the window. The average classification accuracy for all methods was 77% when the predicted classes were three, and 83% when the predicted classes were two.

A. Data Stream Classification results

In this study, we explored the previously described stream mining classification methods with the same group of features. Tables III and IV summarize the classification performance results, in terms of accuracy and time, for the three methods used to develop the classification models in this study with windows sizes 100 and 500 respectively. As seen from table III, when the window size is 100 and the predicted classes are three, OzaBag is slightly more accurate (80%) than HoeffdingTree (79%) and Perceptron (70%), at a higher cost in terms of time (0.84sec vs. 0.32sec vs. 0.53sec). On the other hand, when the predicted classes are two, Perceptron achieves its highest accuracy (83%) compared to the HoeffdingTree (82%), with a small cost in time (0.61sec vs. 0.50sec), but with better learning time than that of OzaBag (1.09sec), which predicts 82% of the correctly classified examples as well. From Table IV it becomes apparent that as the window size increases, the cost in time increases as well, while accuracy appears to be stable.

TABLE III. Performance Metrics for Streaming Classifiers with w=100

Classifier	*Hoeffding Tree*		*OzaBag*		*Perceptron*	
Classes Predicted	*2*	*3*	*2*	*3*	*2*	*3*
Accuracy	*0.82*	*0.79*	*0.82*	*0.80*	*0.83*	*0.70*
Kappa	*0.64*	*0.63*	*0.63*	*0.64*	*0.64*	*0.46*
$\mathbf{K}_{temp}$	*0.65*	*0.65*	*0.64*	*0.66*	*0.65*	*0.50*
Time	*0.50s*	*0.32s*	*1.09s*	*0.84s*	*0.61s*	*0.53s*

TABLE IV. Performance Metrics for Streaming Classifiers with w=500

Classifier	*Hoeffding Tree*		*OzaBag*		*Perceptron*	
Classes Predicted	*2*	*3*	*2*	*3*	*2*	*3*
Accuracy	*0.81*	*0.77*	*0.82*	*0.79*	*0.82*	*0.69*
Kappa	*0.63*	*0.61*	*0.61*	*0.63*	*0.64*	*0.43*
$\mathbf{K}_{temp}$	*0.65*	*0.63*	*0.62*	*0.65*	*0.64*	*0.48*
Time	*1.26s*	*0.98s*	*3.14s*	*2.03s*	*1.82s*	*1.73s*

B. Discussion

As seen from the previous analysis, the results from the stream mining classification indicate that TTAC and TTAW in combination with level of certainty and effort could efficiently be used for classifying and modeling students *during* the self-assessment. Concerning the specific features in the models, students in higher achievement classes (i.e., advanced and pass) obtain the best scores and exhibit the highest response-times to answer correctly and the lowest total spent-time on wrongly answered items. These students are classified with the highest levels of certainty and the highest effort expenditure. The range of TTAC values is a bit lower for medium achievement class members (i.e., intermediates), who however, appear to spend higher total time to review the questions (which is a factor loading on the level of certainty). The major difference between these two classes is identified in the TTAW factor, which for medium classes members appears to be higher. As such, this variable could be used for discriminating the two classes. Similarly, students in lower achievement classes (i.e., novices and fail) exhibit minimum engagement with the self-assessment items in terms of time-spent, denoting low levels of cautiousness and low effort expenditure. More precisely, students in these classes aggregate the higher response-times on TTAW and the lower time-spent on TTAC. For the lower achievement students, level of certainty and effort get their lower values, as well.

However, one can observe that the classification accuracy increases as the number of the predicted classes decreases. That might happen because the classes that are combined in the revised models have a significant number of examples that are misclassified between them in the former models. Thus, reducing the number of predicted classes will decrease the misclassification errors.

VI. Conclusions and Future work

Towards delivering to students the most appropriate self-assessment items and supporting them to become better learners, a pre-requisite that needs to be addressed is dynamically shaping and revising the student models. However, only a limited number of studies provided practical evidence on updating the student models on-the-fly in self-assessment settings. Our vision was to shape and update dynamic student models, using stream mining to efficiently classify and dynamically model students during self-assessment. The major findings of this study are:

1) the suggested HoeffdingTree, OzaBag and Perceptron algorithms achieve (similar) high classification accuracy,

2) learning time of OzaBag is significantly higher than that of HoeffdingTree and Perceptron, *3) learning time of HoeffdingTree is the lower compared to the respective learning time of OzaBag and Perceptron,*

4) students assigned to the higher achievement classes score high in TTAC, CERT and RTE, and score low in TTAW,

5) students assigned to the lower achievement classes score high in TTAW, and score low in TTAC, CERT and RTE,

*6) students assigned to the medium achievement classes aggregate considerable amounts of times both in TTAC and TTA*W. *They score medium in CERT and RTE.*

Based on the above, this work contributes in the field of student modeling as follows:

1) it introduced a set of specific students' features (i.e., total time to answer correctly/wrongly, level of certainty, effort), and elaborated on their roles in the student models, and

2) it implemented a methodology for revising the student models on-the-fly by considering changes in the states of the models.

The mechanisms for tracking response-time data are cost-effective and can be easily implemented in any assessment system. The temporal factors are not contextualized in LAERS, but a similar tracker could be embedded in any adaptive learning or assessment system. Applying the suggested methodology in other assessment contexts is a challenging future work task. Moreover, other features (e.g. time-spent that corresponds to the level of difficulty of the items) could also participate in the student models to quantify the student's current level of knowledge. Enhancing the student models with these features is within our future work plans.

References

[1] J. Broadbent and W. L. Poon, "Self-regulated learning strategies & academic achievement in online higher education learning environments: A systematic review," Internet High. Educ., vol. 27, pp. 1–13, 2015.

[2] K. Chrysafiadi and M. Virvou, "Review: student modeling approaches: A literature review for the last decade," Expert Syst. Appl., vol. 40, no. 11, pp. 4715–4729, Sep. 2013.

[3] G. McCalla, The central importance of student modeling to intelligent tutoring. In E. Costa (Ed.), New Directions for Intelligent Tutoring Systems. Berlin: Springer Verlag, 1992.

[4] D. Sluijsmans, F. Dochy, and G. Moerkerke, "Creating a Learning Environment by Using Self-, Peer- and Co-Assessment," Learn. Environ. Res., vol. 1, no. 3, pp. 293–319, 1998.

[5] Z. Papamitsiou, and A.A. Economides, "An Assessment Analytics Framework (AAF) for enhancing students' progress", In, S., Caballé and R., Clarisó (eds.), Intelligent Data-Centric Systems: Formative Assessment, Learning Data Analytics and Gamification, Academic Press, Boston, (pp. 117-133), 2016.

[6] A. Bunt and C. Conati, "Probabilistic Student Modelling to Improve Exploratory Behaviour," User Model. User-adapt. Interact., vol. 13, no. 3, pp. 269–309, 2003.

[7] A. T. Corbett and J. R. Anderson, "Knowledge tracing: Modeling the acquisition of procedural knowledge," User Model. User-adapt. Interact., vol. 4, no. 4, pp. 253–278, 1994.

[8] R. F. Kizilcec, M. Pérez-Sanagustín, and J. J. Maldonado, "Self-regulated learning strategies predict learner behavior and goal attainment in Massive Open Online Courses," Comput. Educ., vol. 104, pp. 18–33, 2017.

[9] R. Paiva, I. I. Bittencourt, T. Tenório, P. Jaques, and S. Isotani, "What do students do on-line? Modeling students' interactions to improve their learning experience," Comput. Human Behav., vol. 64, pp. 769–781, 2016.

[10] J. Clemente, J. Ramirez, and A. de Antonio, "A Proposal for Student Modeling Based on Ontologies and Diagnosis Rules," Expert Syst. Appl., vol. 38, no. 7, pp. 8066–8078, 2011.

[11] M. C. Desmarais and R. S. Baker, "A Review of Recent Advances in Learner and Skill Modeling in Intelligent Learning Environments," User Model. User-adapt. Interact., vol. 22, no. 1–2, pp. 9–38, 2012.

[12] M. Antal and S. Koncz, "Student modeling for a web-based self-assessment system", Expert Syst.Appl., vol.38, no.6, pp. 6492–6497, 2011.

[13] I.-H. Hsiao, F. Bakalov, P. Brusilovsky, and B. König-Ries, "Open Social Student Modeling: Visualizing Student Models with Parallel IntrospectiveViews," in User Modeling, Adaption and Personalization: 19th International Conference, Proceedings, J. A. Konstan, R. Conejo, J. L. Marzo, and N. Oliver, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 171–182.

[14] C. N. Moridis and A. A. Economides, "Prediction of Student's Mood During an Online Test Using Formula-based and Neural Network-based Method," Comput. Educ., vol. 53, no. 3, pp. 644–652, 2009.

[15] Z. Jeremić, J. Jovanović, and D. Gašević, "Student Modeling and Assessment in Intelligent Tutoring of Software Patterns," Expert Syst. Appl., vol. 39, no. 1, pp. 210–222, 2012.

[16] Z. Papamitsiou and A. A. Economides, "Towards the alignment of computer-based assessment outcome with learning goals: The LAERS architecture," in 2013 IEEE Conference on e-Learning, e-Management and e-Services, IC3e 2013, 2013. (pp. 13-17)

[17] Z. Papamitsiou, V. Terzis, and A. A. Economides, "Temporal learning analytics for computer based testing," Proceedings of the Fourth International Conference on Learning Analytics And Knowledge, 2014, pp. 31–35.

[18] Z. Papamitsiou, and A. A. Economides, "Students' perception of performance vs. actual performance during computer-based testing: a temporal approach", In Proc. 8th International Technology, Education and Development Conference (INTED2014), 2014, pp. 401-411.

[19] V. Terzis and A. A. Economides, "The acceptance and use of computer based assessment," Comput.Educ., vol. 56, no.4, pp. 1032–1044, 2011.

[20] D. K. Peterson and G. F. Pitz, "Confidence, uncertainty, and the use of information", Journal of Experimental Psychology: Learning, Memory, & Cognition, vol. 14 no. 1, pp. 85-92. 1988

[21] M. S. Humphreys and W. Revelle, "Personality, motivation, and performance: A theory of the relationship between individual differences and information processing.," Psychological Review, vol. 91, no. 2. American Psychological Association, pp. 153–184, 1984.

[22] S.L. Wise and X. Kong, "Response Time Effort: A new measure of examinee motivation in computer-based tests", Applied Measurement in Education, vol. 18, no. 2, pp.163-183. 2005

[23] C. C. Aggarwal, Data Streams: Models and Algorithms (Advances in Database Systems). Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2006.

[24] A. Bifet, G. Holmes, R. Kirkby, and B. Pfahringer, "MOA: Massive Online Analysis," J. Mach. Learn. Res., vol. 11, pp. 1601–1604, 2010.

[25] G. Hulten, L. Spencer, and P. Domingos, "Mining Time-changing Data Streams," in Proceedings of the Seventh ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining, 2001, pp. 97–106.

[26] N. C. Oza and S. Russell, "Online Bagging and Boosting," in In Artificial Intelligence and Statistics 2001, 2001, pp. 105–112.

[27] A. Bifet, G. Holmes, B. Pfahringer, and E. Frank, "Fast Perceptron Decision Tree learning from evolving data streams," in Proceedings of the 14th Pacific-Asia Conference on Advances in Knowledge Discovery and Data Mining - Volume Part II, 2010, pp. 299–310.

[28] J. Gama, R. Sebastião, and P. P. Rodrigues, "Issues in Evaluation of Stream Learning Algorithms," in Proc. of the 15th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining, 2009, pp. 329–338.

[29] A. A. Economides, "Adaptive context-aware pervasive and ubiquitous learning". International Journal of Technology Enhanced Learning, vol. 1, no 3, pp. 169-192. 2009.