

Scott Crossley
Elvira Popescu (Eds.)

LNCS 13284

Intelligent Tutoring Systems

18th International Conference, ITS 2022
Bucharest, Romania, June 29 – July 1, 2022
Proceedings

 Springer

MOREMEDIA 

Founding Editors

Gerhard Goos

Karlsruhe Institute of Technology, Karlsruhe, Germany

Juris Hartmanis

Cornell University, Ithaca, NY, USA


Editorial Board Members

Elisa Bertino

Purdue University, West Lafayette, IN, USA

Wen Gao

Peking University, Beijing, China

Bernhard Steffen 

TU Dortmund University, Dortmund, Germany

Moti Yung 

Columbia University, New York, NY, USA

More information about this series at <https://link.springer.com/bookseries/558>


Scott Crossley · Elvira Popescu (Eds.)

Intelligent Tutoring Systems

18th International Conference, ITS 2022
Bucharest, Romania, June 29 – July 1, 2022
Proceedings

Editors

Scott Crossley 
Georgia State University
Atlanta, GA, USA

Elvira Popescu 
University of Craiova
Craiova, Romania

ISSN 0302-9743

ISSN 1611-3349 (electronic)

Lecture Notes in Computer Science

ISBN 978-3-031-09679-2

ISBN 978-3-031-09680-8 (eBook)

<https://doi.org/10.1007/978-3-031-09680-8>

© The Editor(s) (if applicable) and The Author(s), under exclusive license
to Springer Nature Switzerland AG 2022, corrected publication 2022

This work is subject to copyright. All rights are reserved by the Publisher, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilms or in any other physical way, and transmission or information storage and retrieval, electronic adaptation, computer software, or by similar or dissimilar methodology now known or hereafter developed.

The use of general descriptive names, registered names, trademarks, service marks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.

The publisher, the authors, and the editors are safe to assume that the advice and information in this book are believed to be true and accurate at the date of publication. Neither the publisher nor the authors or the editors give a warranty, expressed or implied, with respect to the material contained herein or for any errors or omissions that may have been made. The publisher remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

This Springer imprint is published by the registered company Springer Nature Switzerland AG
The registered company address is: Gewerbestrasse 11, 6330 Cham, Switzerland

Preface

The 18th International Conference on Intelligent Tutoring Systems (ITS 2022) was held in Bucharest, Romania, from June 29 to July 1, 2022 in a hybrid format that allowed participants to attend online as needed considering the continuing COVID-19 pandemic. The Hosting Institution of ITS 2022 was the University Politehnica of Bucharest.

Adhering to the mission of ITS, the title of ITS 2022 was “New Challenges for ITS During and After COVID”. Its objective was to present academic and research achievements in computer and cognitive Sciences, artificial intelligence, and, due to its recent emergence, specifically, deep learning in tutoring and education. The aim of ITS 2022 was to promote and improve learning technology systems, by combining novel and advanced technology with complex and nuanced research approaches. It offered a forum for exploring emerging and noteworthy progress in the field of artificial intelligence in education.

The call for scientific papers focused on a broad number of topics of interest in the area of ITS and beyond including the following:

- Intelligent Tutoring
- Learning Environments for Underrepresented Communities
- Artificial Intelligence in Education
- Human in the Loop, Understanding Human Learning on the Web in a Virtual (Digital) World
- Machine Behaviour (MB), Explainable AI, Bias in AI in Learning Environments
- Emotions, Modeling of Motivation, Metacognition and Affect Aspects of Learning, Affective Computing and ITS
- Extended Reality (XR), Virtual Reality (VR), Augmented Reality (AR), and Mixed Reality (MR) in Learning Technologies
- Informal Learning Environments, Learning as a Side Effect of Interactions
- Collaborative and Group Learning, Communities of Practice and Social Networks
- Analytics and Deep Learning in Learning Systems, Educational Datamining, Educational Exploitation of Data Mining and Machine Learning Techniques
- Sentiment Analysis in Learning Environments
- Data Visualization in Learning Environments
- Privacy, Security, and Ethics in Learning Environments
- Gamification, Educational Games, Simulation-based Learning and Serious Games
- Brain-computer Interface Applications in Intelligent Tutoring Systems
- Dialogue and Discourse During Learning Interactions
- Ubiquitous, Mobile, and Cloud Learning Environments
- Virtual Pedagogical Agents and Learning Companions
- Multi-agent and Service-oriented Architectures for Learning and Tutoring Environments
- Single and Groupwise Action Modeling in Learning Environments
- Ontological Modeling, Semantic Web Technologies, and Standards for Learning

- Empirical Studies of Learning with Technologies
- Instructional Design Principles or Design Patterns for Educational Environments
- Authoring Tools and Development Methodologies for Advanced Learning Technologies
- Domain-specific Learning Technologies, e.g. Language, Mathematics, Reading, Science, Medicine, Military, and Industry
- Non-conventional Interactions Between Artificial Intelligence and Human Learning
- Personalized and Adaptive Learning Environments
- Adaptive Support for Learning, Models of Learners, Diagnosis and Feedback
- Recommender Systems for Learning
- Causal Modeling and Constraints-based Modeling in Intelligent Tutoring

The call for papers sought papers that presented significant new research findings in the use of advanced computing technology and interdisciplinary research to allow, promote, and enhance human learning. Full papers allowed for discussion of more mature and finalized research results, while short papers allowed discussions around brief novel findings. There was also a Poster Track, which included an excellent network for researchers to discuss research prototypes and work in progress with conference attendees.

The international Program Committee consisted of 65 leading members of the intelligent tutoring systems community (16 senior and 49 regular).

Scientific papers were reviewed by three to four reviewers through a double-blind process. Only 28% of submitted papers were published as full papers, 26% were published as short papers, and 16% were published as posters.

The full papers outlined important new developments and theory, the short papers explored new ideas and advances, and the posters discussed research in progress, all based on the ITS philosophy.

The main topics under which the accepted papers fall, on which basis we also structured this book, are as follows:

- Tools and methods for learning sciences and practices
- Algorithms for prediction, recommendation, and classification in learning systems
- Tutoring and learning systems: new approaches, frameworks, and theories

The quality of a conference is reflected by the work of its participants as well as their ability to push the boundaries, and the rigor with which they encourage the rest of the research field to move forward. The papers of ITS 2022 stretched the limits of intelligent tutoring, much as they had over the previous years. Virtual reality, reverse image searches, sequence models, cognitive maps, recommendation systems, and natural language processing were among the fields where authors had documented remarkable work. The ITS 2022 program was also reinforced by the successful organization of Intelligent Tutor Demonstrations by Mihai Dascalu and Philippe Dessus.

The successful preparation and implementation of the ITS 2022 conference was secured by the original work of all the authors, the devoted contribution of the various conference chairs, the members of the Program Committee, the Steering Committee Chair, Claude Frasson, and in particular the General Conference Chair,

Stefan Trausan-Matu. The organization, coordination, and online operation of the conference was achieved by the organizers, the Local Organization Chair, Mihai Dascalu, and the Organization Committee Chair, Kitty Panourgia. We would like to acknowledge the Politehnica University of Bucharest, where the conference was held. Last but not least, we would like to acknowledge the Institute of Intelligent Systems (IIS) for helping organize the conference.

Like previous conferences, the emphasis of ITS 2022 was to introduce new and established scholars to one another, continue to develop and innovate ideas, develop theoretical and business interests, and broaden areas and subgenres related to intelligent tutoring systems. We hope you enjoy reading the papers, building on the research reported, and continuing to develop theories and applications in intelligent tutoring systems.

June 2022

Scott Crossley
Elvira Popescu

Noboru Matsuda	North Carolina State University, USA
Riichiro Mizoguchi	Japan Advanced Institute of Science and Technology, Japan
Roger Nkambou	Université du Québec à Montréal, France
Elvira Popescu	University of Craiova, Romania
Flippo Sciarrone	Roma Tre University, Italy
Stefan Trausan-Matu	Politehnica University of Bucharest, Romania
Christos Troussas	University of West Attica, Greece
Julita Vassileva	University of Saskatchewan, Canada

Program Committee

Marie-Helene Abel	Université de Technologie de Compiègne, France
Galia Angelova	Bulgarian Academy of Sciences, Bulgaria
Yacine Atif	Skövde University, Sweden
Renu Balyan	State University of New York at Old Westbury, USA
Kaushal Kumar Bhagat	Indian Institute of Technology, India
Maria Bielikova	Kempelen Institute of Intelligent Technologies, Slovakia
Emmanuel Blanchard	IDÚ Interactive Inc., Canada
Jesus G. Boticario	National University of Distance Education, Spain
Tharrenos Bratitsis	University of Western Macedonia, Greece
Chih-Yueh Chou	Yuan Ze University, Taiwan
Evandro Costa	Federal University of Alagoas, Brazil
Mihai Dascalu	University Politehnica of Bucharest, Romania
Diego Dermeval	Federal University of Alagoas, Brazil
Reva Freedman	Northern Illinois University, USA
Ella Haig	University of Portsmouth, UK
Yusuke Hayashi	Hiroshima University, Japan
John Hollander	University of Memphis, USA
Srecko Joksimovic	University of South Australia, Australia
Charalampos Karagiannidis	University of Thessaly, Greece
Mizue Kayama	Shinshu University, Japan
Ioannis Kazanidis	Eastern Macedonia and Thrace Institute of Technology, Greece
Aleksandra Klasnja Milicevic	University of Novi Sad, Serbia
Milos Kravcik	German Research Center for Artificial Intelligence, Germany
Blair Lehman	Educational Testing Service, USA
Frederick Li	Durham University, UK
Carla Limongelli	Roma Tre University, Italy
Fuhua Lin	Athabasca University, Canada

Chao-Lin Liu	National Chengchi University, Taiwan
Mirko Marras	University of Cagliari, Italy
Anna Mavroudi	University of Oslo, Norway
Caitlin Mills	University of New Hampshire, USA
Wolfgang Mueller	University of Education Weingarten, Germany
Elaine H. T. Oliveira	Universidade Federal do Amazonas, Brazil
Andrew Olney	University of Memphis, USA
Kuo-Liang Ou	National Tsing Hua University, Taiwan
Sasha Poquet	University of South Australia, Australia
Valery Psyche	Université TÉLUQ, Canada
Ricardo Queirós	Institute for Systems and Computer Engineering, Technology and Science, Portugal
Traian Rebedea	University Politehnica of Bucharest, Romania
Olga C. Santos	National Distance Education University, Spain
Lei Shi	Durham University, UK
Sergey Sosnovsky	Utrecht University, The Netherlands
Kaoru Sumi	Future University Hakodate, Japan
Thepchai Supnithi	National Electronics and Computer Technology Center, Thailand
Marco Temperini	Sapienza University of Rome, Italy
Radu Vasiu	Politehnica University of Timisoara, Romania
Riina Vuorikari	Institute for Prospective Technological Studies, Spain
Dunwei Wen	Athabasca University, Canada

Steering Committee Chair

Claude Frasson	University of Montreal, Canada
----------------	--------------------------------

Steering Committee

Stefano A. Cerri	LIRMM, University of Montpellier and CNRS, France
Maiga Chang	Athabasca University, Canada
Amruth Kumar	Ramapo College of New Jersey, USA
Yugo Hayashi	Ritsumeikan University, Japan
Isabel Fernandez-Castro	University of the Basque Country, Spain
Gilles Gauthier	University of Quebec at Montreal, Canada
Guy Gouarderes	University of Pau, France
Alan Lesgold	University of Pittsburgh, USA
James Lester	North Carolina State University, USA
Alessandro Micarelli	Roma Tre University, Italy
Roger Nkambou	Université du Québec à Montréal, Canada

Giorgos Papadourakis	Hellenic Mediterranean University, Greece
Elliot Soloway	University of Michigan, USA
John Stamper	Carnegie Mellon University, USA
Daniel Suthers	University of Hawai, USA
Christos Troussas	University of West Attica, Greece
Stefan Trausan-Matu	University Politehnica of Bucharest, Romania
Beverly Woolf	University of Massachusetts, USA

Organizing Committee Chair

Kitty Panourgia	Neoanalysis, Greece
-----------------	---------------------

Organizing Committee

Stefano Esposito	Neoanalysis, Greece
Isaak Tselepis	Neoanalysis, Greece
Rasa Tucinskaite	Neoanalysis, Greece
Elisavet Vasileiou	Neoanalysis, Greece

Contents

Tools and Methods for Learning Sciences and Practices

Comparative Evaluation of EduClust and Its Transfer to a Virtual Reality Environment	3
<i>Johannes Fuchs and Matthias Kraus</i>	
Function Execution Log Based Judgment System for Arduino Learning Practice	17
<i>Kangbok Seo and Woojin Lee</i>	
A Hybrid Approach for Mitigating Learners' Rogue Review Behavior in Peer Assessment	24
<i>Gabriel Badea and Elvira Popescu</i>	
DeepCode: An Annotated Set of Instructional Code Examples to Foster Deep Code Comprehension and Learning	36
<i>Vasile Rus, Peter Brusilovsky, Lasang Jimba Tamang, Kamil Akhuseyinoglu, and Scott Fleming</i>	
Cross-Cutting Support of Making and Explaining Decisions in Intelligent Tutoring Systems Using Cognitive Maps of Knowledge Diagnosis	51
<i>Viktor Uglev, Oleg Sychev, and Tatiana Gavrilova</i>	
Covering Possible Reasoning Errors for Intelligent Tutoring Systems: Order of Expression Evaluation Case	65
<i>Yaroslav Kamennov, Oleg Sychev, and Yulia Orlova</i>	
Handshape Recognition in an Educational Game for Finger Alphabet Practicing	75
<i>Tomasz Kapuscinski</i>	
Comparing Alternative Approaches to Debriefing in a Tool to Support Peer-Led Simulation-Based Training	88
<i>Sandra Katz, Patricia Albacete, John Gallagher, Pamela Jordan, Thomas Platt, Scott Silliman, and Tiffany Yang</i>	
Ontological Reference Model for Piloting Procedures	95
<i>Marc-Antoine Courtemanche, Ange Tato, and Roger Nkambou</i>	

Towards Adaptive Coaching in Piloting Tasks: Learning Pilots' Behavioral Profiles from Flight Data	105
<i>Ange Tato, Roger Nkambou, and Gabrielle Joyce Nana Tato</i>	
LORD: A Moodle Plug-in Helps to Find the Relations Among Learning Objects	115
<i>Rita Kuo, Radomir Wasowski, Ted Krahn, and Maiga Chang</i>	
Deep Knowledge Tracing on Skills with Small Datasets	123
<i>Ange Tato and Roger Nkambou</i>	
Algorithms for Prediction, Recommendation and Classification in Learning Systems	
A Learning Analytics Approach to Build Learner Profiles Within the Educational Game OMEGA+	139
<i>Deepak Chandrasekaran, Maiga Chang, and Sabine Graf</i>	
Design and Evaluation of a Competency-Based Recommendation Process	148
<i>Louis Sablayrolles, Marie Lefevre, Nathalie Guin, and Julien Broisin</i>	
A Classification Approach to Recognize On-Task Student's Behavior for Context Aware Recommendations	161
<i>Lisa Roux, Thierry Nodenot, Patrick Etcheverry, Pantxika Dagorret, Christophe Marquesuzaa, and Philippe Lopisteguy</i>	
Automated Classification of Argumentative Components in Students' Essays	171
<i>Qian Wan, Scott Crossley, and Yu Tian</i>	
Evaluating the Effect of Imperfect Data in Voice Emotion Recognition	183
<i>Mahsa Aghajani, Hamdi Ben Abdessalem, and Claude Frasson</i>	
Application of 3D Human Pose Estimation for Behavioral Reproduction	190
<i>Kodjine Dare, Hamdi Ben Abdessalem, and Claude Frasson</i>	
Evaluation Test Generation Model Using Degrees of Difficulty and Keywords	197
<i>Doru Anastasiu Popescu and Mariana Madalina Nastase</i>	
Double-Layer Controller for Detecting Learners' Erroneous Knowledge in Database Programming	204
<i>Christos Troussas, Akrivi Krouska, and Cleo Sgouropoulou</i>	

Identifying Metacognitive Processes Using Trace Data in an Open-Ended Problem-Solving Learning Environment 213
Rumana Pathan, Daevesh Singh, Sahana Murthy, and Ramkumar Rajendran

Intervention Prediction in MOOCs Based on Learners’ Comments: A Temporal Multi-input Approach Using Deep Learning and Transformer Models 227
Laila Alrajhi, Ahmed Alamri, and Alexandra I. Cristea

Not Another Hardcoded Solution to the Student Dropout Prediction Problem: A Novel Approach Using Genetic Algorithms for Feature Selection 238
Yixin Cheng, Bernardo Pereira Nunes, and Rubén Manrique

Tutoring and Learning Systems: New Approaches, Frameworks, and Theories

Equitable Access to Intelligent Tutoring Systems Through Paper-Digital Integration 255
Nirmal Patel, Mithilesh Thakkar, Bansri Rabadiya, Darshan Patel, Shrey Malvi, Aditya Sharma, and Derek Lomas

“See the Image in Different Contexts”: Using Reverse Image Search to Support the Identification of Fake News in Instagram-Like Social Media 264
Farbod Aprin, Irene-Angelica Chounta, and H. Ulrich Hoppe

A Theory Based Adaptive Pedagogical Agent in a Reading App for Primary Students - A User Study 276
Anna Riedmann, Philipp Schaper, Benedikt Jakob, and Birgit Lugin

Intelligent Tutor for Designing Function Interface in a Programming Language 293
Dmitrii Litovkin, Anton Anikin, Kirill Kulyukin, and Oleg Sychev

Effects of Guidance on Learning About Ill-defined Problems 303
Sungeun An, Emily Weigel, and Ashok K. Goel

MEMORABLE: A Multi-playEr custoMisable seriOus Game fRamework for cyBer-security LEarning 313
Jingyun Wang, Ryan Hodgson, and Alexandra I. Cristea

Improving Program Matching to Automatically Repair Introductory Programs 323
Maheen Riaz Contractor and Carlos R. Rivero

Gamification, User-Centered Design and Learning Objectives as the Basis for a Minigame-Based Cardiovascular Anatomy ITS	336
<i>Reva Freedman, Virginia Naples, Ian Sullivan, Lucas Edwards, and Dean LaBarbera</i>	
Investigating Clues for Estimating Near-Future Collaborative Work Execution State Based on Learners' Behavioural Data During Collaborative Learning	343
<i>Yoshimasa Ohmoto, Shigen Shimojo, Junya Morita, and Yugo Hayashi</i>	
Selfit v2 – Challenges Encountered in Building a Psychomotor Intelligent Tutoring System	350
<i>Laurentiu-Marian Neagu, Eric Rigaud, Vincent Guarnieri, Mihai Dascalu, and Sébastien Travadel</i>	
Integrating Speech Technology into the iSTART-Early Intelligent Tutoring System	362
<i>Renu Balyan, Tracy Arner, Tong Li, Ellen Orcutt, Reese Butterfuss, Panayiota Kendeou, and Danielle McNamara</i>	
iSTART-Early: Interactive Strategy Training for Early Readers	371
<i>Panayiota Kendeou, Ellen Orcutt, Tracy Arner, Tong Li, Renu Balyan, Reese Butterfuss, Micah Watanabe, and Danielle McNamara</i>	
Correction to: Intelligent Tutoring Systems	C1
<i>Scott Crossley and Elvira Popescu</i>	
Author Index	381

Tools and Methods for Learning Sciences and Practices



Comparative Evaluation of EduClust and Its Transfer to a Virtual Reality Environment

Johannes Fuchs^(✉)  and Matthias Kraus 

University of Konstanz, Konstanz, Germany
{fuchs,kraus}@dbvis.inf.uni-konstanz.de

Abstract. Previous research successfully demonstrated the applicability and usefulness of *EduClust*, a web platform for understanding and learning clustering algorithms. This paper presents a quantitative evaluation of the educational framework, comparing the online teaching platform to traditional teaching material. Results of the conducted experiment in a real computer science course at the University of Konstanz demonstrate its merit concerning users' confidence and perceived usefulness. Additionally, we dare a first try to exploit the potential social, perceptual, and motivational benefits of immersive media. We present *VRClust*, an initial prototype as the transfer of the web-based application to a virtual reality environment.

Keywords: Data visualization · Clustering · Education · Evaluation · Virtual reality

1 Introduction

In university teaching, digital education concepts have already been established alongside traditional front-of-class lectures. Students can simply join online meetings and profit from easy access to digital content like slides, videos, and tight integration of online teaching applications.

Algorithm visualizations (AV) [24] provide a unique opportunity to enhance traditional teaching material and to share material online. Currently, visualizations of complex algorithms, as well as the visibility of AVs to students, are limited. Basic algorithms like sorting or searching are covered well. However, when it comes to more complex approaches, like clustering or classification tasks, current AVs fall short [22].

A counterexample is the online application *EduClust*¹ [9]. The software uses simple visualizations, animations, and linked pseudo code to explain complex algorithmic behavior and, therefore, supports lecturers and students in teaching and learning complex clustering algorithms. Whereas the usefulness of *EduClust*

¹ <https://educlust.dbvis.de/>.

as a supporting teaching tool for lecturers is fairly evident, the benefits to students are harder to quantify upfront. Although students reported positively about the software, we chose to conduct a comparative user study to determine the usefulness of *EduClust* as a learning tool for students.

Independent of the results of the user experiment, we wanted to increase the engagement of students further by transferring the online application to a virtual reality (VR) setting. Since studies positively report about the benefits of VR in teaching environments [1, 18], we wanted to profit from this new technology. As a result, *VRClust* was implemented, which transfers the general workflow of *EduClust* to a VR environment and extends its functionality with additional features like cluster quality measures and the additional clustering algorithm DENCLUE [12].

In this paper, we contribute a quantitative experiment to investigate the usefulness of *EduClust* in comparison to traditional teaching material. We evaluated *EduClust* with 48 students attending a data mining lecture. Students positively evaluated their own use of the tool and observed its use by the lecturer. They also became more confident about their understanding of the included clustering algorithms. As a second contribution, we transferred *EduClust* to a VR setting and showcase the benefits of this new environment.

2 Background

Much research has been conducted in the area of interactive visualizations for education. On the one hand, visualization systems have been introduced to support the analysis of learners' behavior [5] or to manage lecture material [3, 4, 20, 25]. On the other hand, topic-specific learning applications like algorithmic visualizations for decision trees, general statistics [26], sorting [24], or applications to improve visualization literacy [7] were implemented.

In this section, we do not want to go into details about such pedagogical tools but briefly introduce *EduClust* with its main features to foster a better understanding of the application area of this online learning and teaching software. Furthermore, we would like to shed more light on the use of virtual reality in education to motivate our *EduClust* extension called *VRClust*.

EduClust - an Online Teaching Application: The design of *EduClust* was motivated by research about interactive visualizations of algorithmic behavior in education [10, 14] and visualization techniques for clustering algorithms [2, 11]. The application's main component is a scatterplot in the center, which is used to display data points as small circles and the algorithmic behavior using an animation. Another major component is the pseudo-code, which dynamically adjusts to the selected algorithm and is directly linked to the animation. The current algorithmic step is, therefore, always highlighted in the pseudo-code. Other views allow the user to adjust input parameters of the selected algorithm or to control the animation speed or replay single steps. Depending on the selected algorithm, additional views are added to the application, e.g., a dendrogram for hierarchical clustering [8] (Fig. 1).

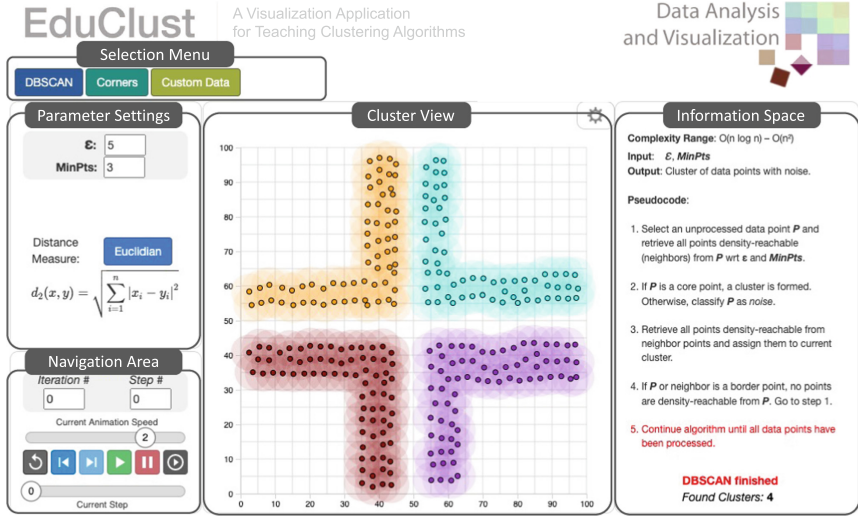


Fig. 1. EduClust: overview of the online application. Different components offer the possibility to investigate clustering algorithms in detail. In this example, the DBSCAN algorithm was used to cluster the data points. Colored areas indicate the four different cluster structures.

When incorporating the online application in lectures, teachers do not need to prepare slides anymore that show how clustering algorithms perform with specific input parameters on a certain example data set. *EduClust* allows teachers to deviate from slides and interactively show as many combinations of parameters or datasets as deemed useful. At the same time, it grants lecturers high flexibility to switch between different example cases and settings and the possibility to readily react to student questions that might require material and examples previously unanticipated.

Although the integration of educational software in the teaching routine is expected to be beneficial, there is no scientific proof about the usefulness of incorporating *EduClust* in a data mining lecture and its effect on students' performance.

Virtual Reality in Education: Introducing VR applications in the education sector has had a positive effect on stimulating interactivity [19] and motivation [18]. Famous examples are virtual field trips like Google Expedition [23], which increase the learning experiences with enhanced presence [6], or VR applications for training engineers, which help to increase students' performance [1]. These positive effects can also be transferred to VR classrooms with virtual avatar teachers [21].

3 Evaluation

The benefits of *EduClust* as a teaching tool for lecturers listed above are fairly evident, and we decided not to evaluate them. The benefits to students, on the other hand, are harder to quantify upfront, and we chose to conduct a comparative user study to determine the usefulness of *EduClust* as a learning tool.

3.1 Experiment Design

The purpose of the experiment was to compare how well students could answer questions about different clustering algorithms and to elicit subjective opinions on the usefulness of our tool. Therefore, we decided to apply a procedure commonly used in education research: a pre-test/post-test design (also known as "before and after" design) [17]. We evaluated students over a three-week period. In each week, students first attended a lecture on a specific set of clustering algorithms (partition-based, linkage-based, and density-based) and two days later took part in the evaluation session as part of their regularly scheduled tutorial class.

In the first two weeks, all students attended lectures using traditional slides. They then took part in one of two conditions: during their tutorial class, they studied clustering algorithms either using *EduClust* or using a traditional learning method consisting of lecture slides and scientific papers (control group). We compared their performance with a pre- and post-quiz, interrupted by a free study period. As research suggests, we included a control group (i.e., students learning with traditional teaching material) to correct for confounding variables like history, maturation, test-, or regression to the mean effects (RTM) [17].

In the third week, we used *EduClust* during a class lecture itself to augment traditional teaching material. The lecturer removed all examples from his slides and instead showed the algorithmic behavior of DBSCAN and OPTICS with different data sets and input parameters. In the following tutorial evaluation session, we asked students about their subjective experience with the tool as part of teaching and, additionally, about their user experience with the software in one of the previous study sessions.

Participants: We recruited 48 participants (21 female, 25 male, 2 NA) from the local student population with a median age of 24 (range 20 – 32). All participants attended the course "Data Mining: Basic Concepts" at the University of Konstanz and had no prior knowledge about clustering techniques. Participants did not receive any monetary compensation. Students who participated in the tutorial evaluation sessions for Week 1 and 2 were not required to attend the previous lecture, while we only elicited feedback in Week 3 from students who attended the lecture and had seen *EduClust* used as a teaching aid.

Setting and Procedure: Each study week began with a lecture on clustering algorithms taught by one author of this paper. During this lecture, the teacher showed slides with and without animations and made drawings on the blackboard whenever necessary. Experimental sessions followed two days after each

lecture. For each session, we split participants into two independent groups, and each group attended a different tutorial session. The two groups could not communicate during the tutorial sessions.

Students were asked to fill out several questionnaires before commencing their studies (with or without *EduClust*). In the first week, students began by filling out a questionnaire on demographic information. In both the first and second week, they also reported their confidence about the topic taught in the previous class lecture. After collecting this information, all participants filled out a quiz on the lecture topic (pre-test) to assess their current understanding of the topic. Students were told that the quiz contained questions similar to those of the final exam in order to motivate them to answer the questions as best as they could. The quiz included theoretic questions about clustering algorithms to check whether students remembered and understood fundamental characteristics of the respective algorithm, e.g., naming the input parameters of ISODATA. Additionally, participants had to apply their knowledge and evaluate whether a clustering algorithm could correctly separate a given set of clusters.

The pre-test was followed by a free study period during which students could revisit the lecture content in light of the questions they had previously answered. Students in the *EduClust* condition used their own laptops to run *EduClust* from a website we made accessible for the duration of the tutorial. Students in the other condition were given electronic copies of the lecture slides as well as links to the most relevant papers on the topic. Participants were told to use the study time (15 min) to prepare for a similar test (post-test) to the pre-test.

The free study period was followed by another quiz (post-test). Students did not have access to *EduClust* or teaching material during quizzes. The post-test contained the same questions as the pre-test. In general, it is best for the difficulty of pre- and post-tests to be similar, but as this is not always easy to determine, questions are often the same quiz [17]. Our wording and pre-test quiz primed students to study the specific questions. Nevertheless, we note that this priming was true for students in both conditions. After the post-test, we collected the final confidence of participants and qualitative information in an exit questionnaire. In the second week, we kept the groups and the procedure identical to Week 1. However, we switched the *preparation method* between groups. Thus, at the end of the evaluation, each student participant worked once with *EduClust* and with lecture slides and research papers. This change helped us to avoid learning effects and to make sure that all participants were exposed to both *preparation methods* minimizing the influence of individual differences between subjects. Thus, overall, we collected demographic information, confidence scores, pre- and post-test results, as well as qualitative feedback during Weeks 1 and 2. In the third week, both groups used *EduClust* and filled out a final questionnaire about their user experience with the software. Additionally, participants rated the last lecture in which the lecturer included *EduClust* to augment his teaching material.

3.2 Hypotheses

Based on pilot studies and our experience as lecturers, we derived the following hypotheses for our evaluation:

- H1:** *Independent of the preparation method, participants' accuracy will improve between pre- and post-tests.* Performance will be better in the post-test since participants get time to prepare for the questions.
- H2:** *Participants score better when preparing the post-test with EduClust compared to students using traditional teaching material.* We expected this result as *EduClust* is more flexible in creating examples compared to traditional teaching material or research papers. Additionally, students can control the algorithmic behavior and replay a procedure until they understand how it works.
- H3:** *Participants value the use of EduClust in the lecture.* Students positively experience the benefits of *EduClust* listed in Sect. 2 and reflect this experience in their ratings.

3.3 Results

Here, we report on significant results ($p < .05$) from our quantitative analysis and refer to the qualitative feedback in Sect. 3.4.

In the pre- and post-test, we recorded participants' *accuracy* as their number of correct answers in each test (see Fig. 2), as well as their *gain* score, i.e., their difference in accuracy between pre- and post-test. Before the pre-test and after the post-test, participants rated their *confidence* in understanding the respective clustering content and the *usefulness* of the provided material in a separate questionnaire. Given the non-normal distribution, we analyzed the results using a Mann-Whitney U-test between groups and a paired Wilcoxon test within groups. We analyzed the weeks independently since the clustering content for each session changed during the study. The condition of interest is the *preparation method* (i.e., lecture material or *EduClust*).

First Week. In the first week, participants had to work on questions about partition-based clustering.

Pre- and Post- Study Results Per Preparation Method: There was no significant effect of *preparation method* on *accuracy* for both the pre- (lecture material: 64.3%; *EduClust*: 57.1%) and post-test (lecture material: 78.6%; *EduClust*: 71.4%). The 95% confidence interval of the differences between preparation method for pre-test was [-21.4%;7.1%] and identical for the difference between preparation methods for the post-test. There was also no statistical significant effect of *preparation method* on *gain scores* (lecture material: 13.4%; *EduClust*: 10.2%). We, thus, have no evidence to conclude that any of the two groups performed more accurately when starting or at the end of the session.

When comparing the post-test questionnaires, there was a significant effect of *preparation method* on perceived *usefulness* (the mean ranks of *EduClust* and

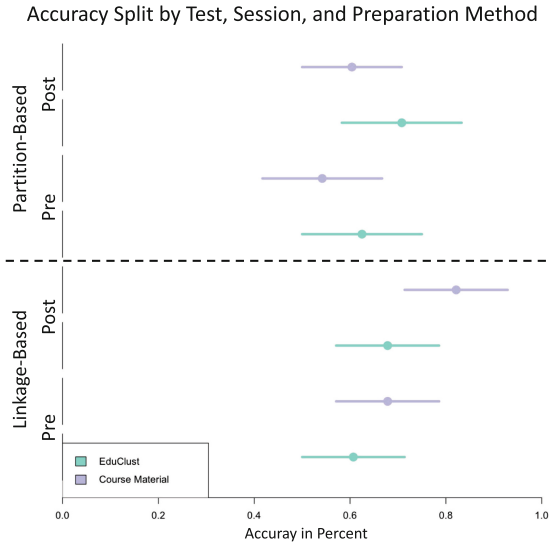


Fig. 2. Accuracy: Participants’ performance, split according to session (i.e., clustering topic), *preparation method* (color), and test condition (i.e., pre-test and post-test). Error bars indicate the 95% confidence interval.

Table 1. Percentages of participants who found the *preparation method* for the different clustering methods useful.

Partition-based clustering

Preparation method	Strongly disagree	Disagree	Neutral	Agree	Strongly agree
EduClust	0	0	4.8%	71.4%	23.8%
Lecture Mat	10%	10%	40%	35%	5%

Linkage-based Clustering

EduClust	0	0	11.8%	35.3%	52.9%
Lecture Mat	0	25%	35%	40%	0

lecture material were 27.1 and 14.6, respectively; $U = 338$, $Z = 3.66$, $p < .001$, $r = 0.57$). Participants were more satisfied with *EduClust* compared to traditional material (Table 1).

Improvement Per Preparation Method: For the *accuracy*, post-test results (lecture material: 78.5%; *EduClust*: 71.4%) were statistically significantly higher (lecture material: $p < .005$; *EduClust*: $p < .001$) than pre-test results (lecture material: 64.3%; *EduClust*: 57.1%). The 95% confidence interval of the mean difference between pre- and post-tests (i.e., improvement) lay between [14.3%:25%] for the lecture material and [7.1%:17.9%] for *EduClust* (Fig. 3).

The medians of participants’ confidence before and after the experiment were 3 and 3, respectively. A Wilcoxon Signed-rank test shows that there is a sig-

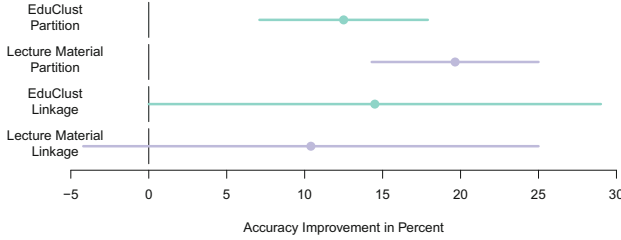


Fig. 3. Accuracy improvement between pre- and post-tests, by session and *preparation method*. Error bars indicate the 95% confidence interval.

nificant improvement of confidence ($W = 40$, $Z = -2.7$, $p < .01$, $r = 0.42$). Participants gained significantly more *confidence* in the understanding of the respective clustering algorithm when working with *EduClust* ($p < .01$). There was no significant effect for the lecture material (see Fig. 4).

Second Week. In the second week, material covered linkage-based clustering. **Pre- and Post- Study Results Per Preparation Method:** There was no significant effect of *preparation method* on *accuracy* for both the pre- (lecture material: 54.2%; *EduClust*: 58.3%) and post-test (lecture material: 58.3%; *EduClust*: 75%). The 95% confidence interval of the differences between preparation methods for pre-test was [-8.3%:25%] and [0%:25%] for the difference between preparations for the post-test. There was also no statistical significant effect of *preparation method* on *gain scores* (lecture material: 8.3%; *EduClust*: 13.2%).

When comparing the post-tests, there was a significant effect of *preparation method* on *usefulness* (the mean ranks of *EduClust* and lecture material were 26.24 and 12.85, respectively; $U = 293$, $Z = 3.92$, $p < .001$, $r = 0.64$). Participants were more satisfied with *EduClust* compared to traditional material (Table 1).

Improvement Per Preparation Method: Post-test results (lecture material: 58.3%; *EduClust*: 75%) were not statistically significantly higher than pre-test results (lecture material: 54.2%; *EduClust*: 58.3%). The 95% confidence interval of the mean difference between pre- and post-tests (improvement) lay between [-4.2%:25%] for the lecture material and [0%:29%] for *EduClust* (Fig. 3). There was no significant effect of *confidence* for each *preparation method* (see Fig. 4).

Third Week. In Week 3, we did not deploy tests. Instead, we collected qualitative feedback on the use of the tool and its usefulness during the lecture.

All 28 participants who attended the respective lecture agreed (50%) or strongly agreed (50%) that integrating *EduClust* in the lecture helped them to understand the respective clustering algorithm. Moreover, 53% agreed or strongly agreed (47%) that they would use the tool to prepare for the exam. When checking our server logs between the last experimental session and the

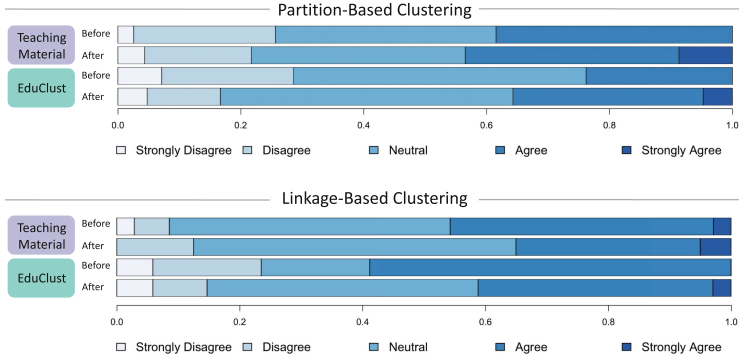


Fig. 4. Likert scale results to the question “I am confident I understand the procedure of <name of the algorithm>.” before and after the experiment.

final exam, we identified 13 different IP accesses on average per day. Compared to previous access rates (2 different IPs per day before the experiment) we experience an increase of 550% with a peak of 44 different IP addresses (48 participants) on a single day. Thus, our logs indicate students did indeed use the material to prepare for the exam. Overall, 22 participants agreed or strongly agreed (77%) that *EduClust* helped them to better understand the clustering behavior. Five students (18%) were neutral in their decision and one student (4%) disagreed with the statement.

3.4 Discussion

In the following, we discuss the results of our quantitative analysis in more detail in combination with the results of the qualitative feedback.

In the first week, participants’ accuracy improved significantly from the pre- to the post-test independent of the *preparation method*. Students working with *EduClust* also showed a significant increase in confidence about their knowledge of the clustering content. Although the mean accuracy increased in the second week for both *preparation methods* as well, the effect was not significant. Also, we no longer found a significant increase in the confidence of the participants. A possible explanation is that students were missing features to improve their understanding of linkage-based clusterings. In the free-text feedback section of the exit questionnaire, students reported that they were missing a way to compare different dendrograms (i. e., single-linkage vs. average-linkage) and a representation of the distance matrix to better understand the joining steps of the clusters. Surprisingly, these requests for additional features did not seem to affect their final evaluation in Week 3. The majority (85.2%) of participants reported that they agreed or strongly agreed with the statement that the implementation of the respective algorithms helped them to better understand the corresponding clustering behavior. In summary, given the quantitative results from the first and second session, we can only partially confirm H1.

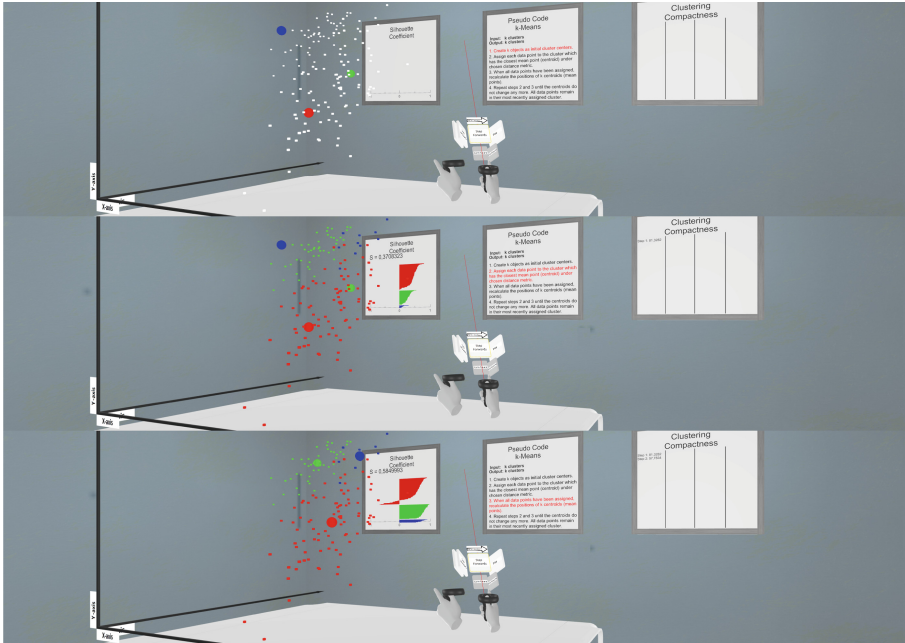


Fig. 5. K-Means clustering performed in VRClust. Exemplary, the iterative improvement of the cluster quality is shown by depicting the steps 1, 2, and 3 (top to bottom). Cluster centroids are depicted as larger spheres and color indicates cluster affiliation. The three windows in the background show the silhouette coefficient, the dynamic pseudo code and the overall cluster quality (left to right).

When comparing the two *preparation methods* based on *accuracy* and *gain scores*, we did not find a significant difference. Therefore, H2 cannot be confirmed. However, in the *EduClust* condition, participants' *confidence* increased significantly for partition-based clustering, which was not true for the traditional teaching material, thus it seems that *EduClust* was a worthy addition to the classroom. Not only did the participants value the *usefulness* of *EduClust*, they all stated that they would use the software to prepare for the final exam, a statement that seems to be supported by the increase in server accesses during the preparation phase for the exam. Thus, their statement reflects true intentions rather than a possible effect of desirability bias.

Integrating *EduClust* in a clustering lecture was well received by students, confirming H3. Although just 28/48 students attended both the lecture and the tutorial of Week 3, all (strongly) agreed that *EduClust* helped them to understand the material covered in class. However, we found that especially for ISO-DATA and OPTICS more work is needed to improve the implementations.

Given the qualitative feedback, we identified additional promising areas of improvement. Students expressed their desire for an indicator of the current clustering quality as a subjective measure that can be tracked over time. Also,

for linkage-based clustering, students asked for visual assistance when comparing two or more dendrograms at the same time.

As an extension to *EduClust*, we created a prototype that explores the immersive design space by transferring the concept of *EduClust* to Virtual Reality (VR). Our novel prototype constitutes an increment to the web framework by supporting new algorithms exploiting the benefits of actual 3D like DENCLUE [12]. Additionally, further useful information and visualizations like clustering quality measures or comparison views as desired by our participants can be easily added in the VR environment due to the extended design space. Besides these advantages, our new VR education platform also strives to exploit potential benefits provided in immersive environments: for instance, enabled social aspects in collaborative scenarios, eased spatial understanding in native three-dimensional visualizations, or increased levels of engagement in vivid virtual reality environments for classroom scenarios.

4 VRClust

With *VRClust*, we present the initial prototype of an immersive teaching platform for clustering algorithms as a transfer of *EduClust* to virtual reality (see Fig. 6, right). Our main motivation for this step is threefold: (i) improving the understandability of clustering algorithms in three-dimensional space due to improved depth-perception and stereoscopic vision, (ii) enhancing the engagement of students to try out different clustering algorithms, and (iii) facilitating the social aspect of teaching experiences in terms of a multi-user VR scenario that comes short in screen-based scenarios.

In the immersive framework, we recreated the available options and the workflow from *EduClust*, but optimized the interface for the new environment. Hence, a menu is attached to the right controller, allowing the user to select different data sets, visualizations, and clustering algorithms. All data sets contain data items with at least three dimensions. Initially, the selected data is presented as a simple 3D scatterplot but can be modified to contain various glyphs (e.g., pie and star glyphs) using the visualization sub-menu. Due to the results of an earlier experiment [16], we decided to position our visualization on a digital table to provide an overview of the data being analyzed.

Once an algorithm is selected, a second menu is shown to parameterize the algorithm, step through it, or animate the whole clustering procedure. Figure 5 depicts three steps of a K-Means clustering performed on exemplary data. Similar to the original 2D framework *EduClust*, the visualization is enriched with cluster centroids, data points are stained to convey their cluster dependency, and the current step is highlighted in the Pseudocode on the wall. Additionally, two quality measures are added to this prototype. On the left, the silhouette coefficient [13] is displayed, and the respective distances are visualized. On the rightmost board, all previous steps are listed up together with their corresponding clustering compactness score. This allows the user to monitor the development of the clustering quality over time, a feature desired by the participants of our previous user experiment.



Fig. 6. Transferring *EduClust* to VR. Left: the DENCLUE algorithm is explained in *VRClust* with the aid of a 3D heatmap visualization that represents the density distribution of datapoints in the investigated dataset. Right: in *VRClust*, the interface for selecting clustering algorithms, data, and visualizations is attached to the VR controller.

Furthermore, *VRClust* exceeds the functionality of *EduClust* by incorporating additional clustering algorithms like the DENCLUE [12] algorithm (see Fig. 6, left). When selecting a hierarchical clustering algorithm like DENCLUE, the corresponding density distribution is depicted as a three-dimensional heatmap visualization through which a cutting plane can be shifted, resembling the parameter φ that defines the threshold for density and influences the number of clusters. The three dimensional design space allows for a simultaneous presentation of the density distributions together with the raw data points, which should strengthen the understanding of the algorithmic behavior. Our previous work has shown the potential of multiple 3D heatmaps for comparative analysis [15]. Various found benefits certainly transfer to a single heatmap with a cutting plane and a juxtapositioned scatterplot visualization.

5 Conclusion

The presented quantitative experiment assessed the performance of *EduClust*, a web teaching framework for conveying the functionality of clustering algorithms, in a real-world university context. Students were more confident about their understanding of the respective topic and considered the tool more useful in comparison to traditional teaching material. Additionally, we presented an initial prototype of a VR version of the learning platform that strives to exploit potentially beneficial properties of immersive space in the educational domain. Even though we can imagine various merits of the transfer, a quantitative evaluation of the immersive system is subject to future research. In the near future, we try to incorporate collaboration aspects in our *VRClust* software. Students and lecturers are assigned to different roles and join the same teaching environment.

Acknowledgements. The work was partially supported by funds from the University of Konstanz as part of the Young Scholar Fund (YSF).

References

1. Alhalabi, W.: Virtual reality systems enhance students' achievements in engineering education. *Behav. Inf. Technol.* **35**(11), 919–925 (2016)
2. Berthold, M.R., et al.: Knime-the konstanz information miner: version 2.0 and beyond. *ACM SIGKDD Explor. Newsl.* **11**(1), 26–31 (2009)
3. Chen, Q., Chen, Y., Liu, D., Shi, C., Wu, Y., Qu, H.: Peakvizor: visual analytics of peaks in video clickstreams from massive open online courses. *IEEE Trans. Vis. Comput. Graph.* **22**(10), 2315–2330 (2016). <https://doi.org/10.1109/TVCG.2015.2505305>
4. Chen, Y., Chen, Q., Zhao, M., Boyer, S., Veeramachaneni, K., Qu, H.: Dropoutseer: visualizing learning patterns in massive open online courses for dropout reasoning and prediction. In: 2016 IEEE Conference on Visual Analytics Science and Technology (VAST), pp. 111–120, October 2016. <https://doi.org/10.1109/VAST.2016.7883517>
5. Chen, Y., Chen, Q., Zhao, M., Boyer, S., Veeramachaneni, K., Qu, H.: Dropoutseer: visualizing learning patterns in massive open online courses for dropout reasoning and prediction. *IEEE Trans. Vis. Comput. Graph.* **23**(1), 111–120 (2017)
6. Cheng, K.H., Tsai, C.C.: A case study of immersive virtual field trips in an elementary classroom: Students' learning experience and teacher-student interaction behaviors. *Comput. Educ.* **140**, 103600 (2019)
7. Firat, E.E., Denisova, A., Laramée, R.S.: Treemap literacy: a classroom-based investigation. In: Romero, M., Sousa Santos, B. (eds.) *Eurographics 2020 - Education Papers*. The Eurographics Association (2020). <https://doi.org/10.2312/eged.20201032>
8. Frank, E., Hall, M., Holmes, G., Kirkby, R., Pfahringer, B., Witten, I.H., Trigg, L.: Weka-a machine learning workbench for data mining. In: Maimon, O., Rokach, L. (eds.) *Data mining and knowledge discovery handbook*, pp. 1269–1277. Springer, Boston (2009). https://doi.org/10.1007/978-0-387-09823-4_66
9. Fuchs, J., Isenberg, P., Bezerianos, A., Miller, M., Keim, D.: Educlust-a visualization application for teaching clustering algorithms. In: *Eurographics 2019–40th Annual Conference of the European Association for Computer Graphics*, pp. 1–8 (2019)
10. Halim, S., Koh, Z.C., Loh, V.B.H., Halim, F.: Learning algorithms with unified and interactive web-based visualization. *Olympiads Inf.* **6**, 53–68 (2012)
11. Harris, N.: K-means visualization using a scatterplot with dividing planes (2014). <https://www.naftaliharris.com/blog/visualizing-k-means-clustering/>. Accessed Nov 2021
12. Hinneburg, A., Keim, D.A.: An efficient approach to clustering in large multimedia databases with noise. In: *Proceedings of the Fourth International Conference on Knowledge Discovery and Data Mining*, pp. 58–65. KDD 1998, AAAI Press (1998)
13. Kaufman, L., Rousseeuw, P.J.: *Finding groups in data: an introduction to cluster analysis*, vol. 344. Wiley, Hoboken (1990). <https://doi.org/10.1002/9780470316801>
14. Kerren, A., Stasko, J.T.: *Algorithm animation*. In: *Software Visualization*, pp. 1–15. Springer, Cham (2002)
15. Kraus, M., et al.: Assessing 2d and 3d heatmaps for comparative analysis: An empirical study. In: *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, pp. 1–14 (2020)
16. Kraus, M., Weiler, N., Oelke, D., Kehrer, J., Keim, D.A., Fuchs, J.: The impact of immersion on cluster identification tasks. *IEEE Trans. Vis. Comput. Graph.* **26**(1), 525–535 (2019)

17. Marsden, E., Torgerson, C.J.: Single group, pre-and post-test research designs: some methodological concerns. *Oxford Rev. Educ.* **38**(5), 583–616 (2012)
18. Olmos-Raya, E., Ferreira-Cavalcanti, J., Contero, M., Castellanos, M.C., Giglioli, I.A.C., Alcañiz, M.: Mobile virtual reality as an educational platform: a pilot study on the impact of immersion and positive emotion induction in the learning process. *EURASIA J. Math. Sci. Technol. Educ.* **14**(6), 2045–2057 (2018)
19. Roussou, M.: Learning by doing and learning through play: an exploration of interactivity in virtual environments for children. *Comput. Entertainment (CIE)* **2**(1), 10–10 (2004)
20. Schwab, M., et al.: booc. io: an education system with hierarchical concept maps and dynamic non-linear learning plans. *IEEE Trans. Vis. Comput. Graph.* **23**(1), 571–580 (2017)
21. Seo, S.h., Kim, E., Mundy, P., Heo, J., Kim, K.K.: Joint attention virtual classroom: a preliminary study. *Psychiatry Inv.* **16**(4), 292 (2019)
22. Shaffer, C.A., et al.: Algorithm visualization: the state of the field. *ACM Trans. Comput. Educ. (TOCE)* **10**(3), 9 (2010)
23. VidaSystems: Google expeditions (2021). <https://artsandculture.google.com/project/expeditions>. Accessed Nov 2021
24. Virginia Tech: Algoviz (2021). <https://research.cs.vt.edu/AVresearch/>, <https://research.cs.vt.edu/AVresearch/>. Accessed Nov 2021
25. Wu, T., Yao, Y., Duan, Y., Fan, X., Qu, H.: Networkseer: visual analysis for social network in moocs. In: 2016 IEEE Pacific Visualization Symposium (PacificVis), pp. 194–198, April 2016
26. Yee, S., Chu, T.: r2d3.us (2021). <http://www.r2d3.us/> Accessed Nov 2021



Function Execution Log Based Judgment System for Arduino Learning Practice

Kangbok Seo and Woojin Lee^(✉)

School of Computer Science and Engineering, Kyungpook National University,
Daegu, Republic of Korea

dating1227@gmail.com, wojin@knu.ac.kr

Abstract. The automatic judgment system, which has been widely used in recent programming lectures, is being developed mainly in a language that is performed based on a console. Embedded systems such as Arduino require more effort from instructors to guide students, but research on systems that support them is slow. In order to support this, this paper intends to support instructors and students by developing an Arduino practice judgment system based on a function execution log. The Arduino practice judgment system can perform Arduino practice judgment by processing the Fritzing output and the source code execution result in the form of a function execution log rather than the actual Arduino HW. In addition, it was shown that the system can be applied to actual lectures to make lectures more efficient.

Keywords: Automated assessment · Programming education · Automated feedback · Arduino practice

1 Introduction

Recently, as the number of students taking programming classes increases, the work of instructors conducting the classes also increases. To support this, development and research on automatic judgment systems are being conducted in various ways [8]. However, in the case of automatic judgment systems, console-based languages such as C and JAVA are mainly developed and used, and research on automatic judgment systems targeting embedded systems such as Arduino is progressing very slowly. In a system like Arduino, the instructor's workload is higher than that of the existing programming class because the instructor must provide feedback on the HW configuration and source code while the instructor is conducting the class. To support this, functions that automatically perform judgment and provide feedback are required, but embedded systems are difficult to implement. In order to score a system such as Arduino, you need to check the HW configuration and connection, and upload the source code to the developed HW to check the output. However, since it is difficult to automatically check the output of the actual HW, research on the automatic judgment system for embedded systems is also being conducted slowly.

Therefore, in this paper, we propose an Arduino practice judgment system based on the function execution log. The Arduino practice judgment system is a tool developed to automatically test the HW and source code developed by students in Arduino practice. First, the instructor configures the HW required by the example in Fritzing [3], writes the appropriate source code, and uploads it to the system. The system extracts the conditions necessary for HW configuration from the HW configuration file uploaded by the instructor, and creates a test case based on the function execution log obtained by executing the source code. Test cases for the generated HW configuration conditions and source code are used to score students' practice. The products uploaded by the students go through the same process as the files uploaded by the instructor, and are scored using pre-created configuration conditions and test cases. In addition, by providing feedback according to the test results to the students, it supports the students to correct errors in the practice themselves.

The structure of this paper is as follows. Section 2 deals with related research, and Sect. 3 provides the main idea and configuration of the system. Section 4 describes the system test method, Sect. 5 shows the case of applying the system to actual lectures, and Sect. 6 consists of a conclusion.

2 Related Works

Currently, research on automatic judgment systems is mainly focused on systems targeting languages that can perform tests by operating console-based [5,8]. In addition to programming languages, an automatic judgment system was also developed that compares and tests the results of a computer aided design (CAD) product-based hardware design [2] and DB query [6]. However, in the case of embedded systems such as Arduino, automatic judgment systems are being studied slowly. Model-based tests [9] and embedded system simulators [4] exist as automated test techniques required to develop an auto-judgment system. However, to apply these techniques to Arduino, it is difficult to apply them due to insufficient source code or limitations of the simulator. The currently developed ARDUINO Intelligent Tutoring System [1] does not support most of the functions supported by other Intelligent Tutoring Systems, so its use is very limited.

3 Function Execution Log Based Judgment System

3.1 System Overview

The Arduino practice judgment system is a system that automatically scores students' practice using the HW and Arduino source code configured in Fritzing [3]. The system performs practice judgment by comparing the HW configuration and test case written by the instructor with the output written by the students. For this purpose, students are required to configure the HW in Fritzing in addition to the existing Arduino practice environment. Judgment system consists of

4 modules, and each module consists of HW configuration condition generator, function execution log generator, test execution module, and feedback generator. Figure 1 shows the configuration of the Arduino practice judgment system.

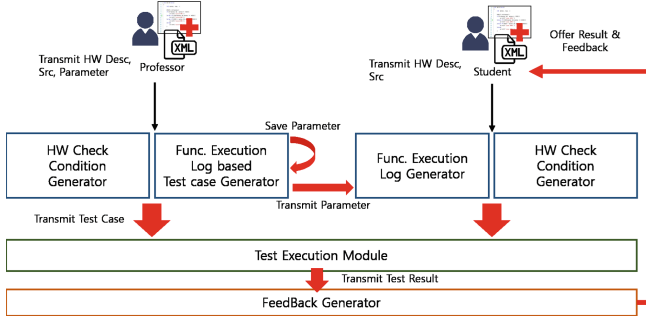


Fig. 1. Overview of Arduino practice judgment system

Arduino practice judgment system extracts HW configuration information from HW configuration condition generator and uses it to test HW list and connection information. Next, the function execution log generator has a function to create a function execution log by executing the source code of the instructor and student. At this time, the instructor enters the conditions necessary for executing the source code (ex. sensor measurement information, serial input, number of loops, etc.) according to the example and uses it to generate the test case. The source code uploaded by the student gets the execution conditions entered in advance by the instructor and executes the source code to create a function execution log. The third is the test execution module, which checks the HW configuration conditions generated by the instructor and uses the generated test cases to test the source code. HW configuration condition check is performed by comparing the extracted information, and the source code is tested in different ways depending on the type of test case generated. The system generates and provides feedback to students based on the test results performed.

3.2 Fritzing Based HW Configuration

Fritzing [3] is an open source SW developed to implement the Arduino HW configuration on a PC. In this paper, Fritzing's configuration data is used instead of Arduino's actual HW to implement the Arduino's automated test. Fritzing provides the configured HW information in XML format and includes the type of configured HW and connection information between the HW. In XML data, one port connection information is defined between `<net>` `</net>` tags to indicate HW information connected to Arduino. The HW information connected to the Arduino port is defined between `<connctor>` and `</connctor>`, and if it exists in the same `<net>` tag, it means that they are connected.

3.3 Function Execution Log

HWs used in Arduino are controlled using functions in the library, and each function receives connected Arduino port information and control commands as parameters to operate the corresponding hardware. If the parameters required for function execution and hardware information operated by the function are combined and expressed in a log, the operation of specific hardware can be expressed in a log format, and this log is called a function execution log. Figure 2 shows the structure of the function execution log generator.

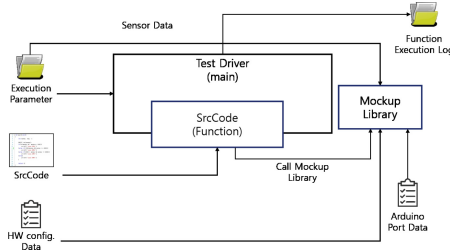


Fig. 2. Overview of function execution log generator

In order to generate the function execution log, the port information of the actual Arduino board, the mock-up library, and the test driver are required. The execution environment composed of these three components executes the uploaded source code without changing it, creating a function execution log. Mock-up library refers to a library created by re-defining functions used in Arduino as the function names are. Existing functions operate the hardware using the input parameters, but the redefined functions have the function of generating the function execution log by outputting the input parameter information to the standard output. The test driver saves the source code execution conditions (loop number, sensor data, etc.) entered by the instructor in the library and calls the `setup()` and `loop()` functions in the Arduino source code to create a function execution log.

4 Judge Method

4.1 HW Check Condition

The HW tests performed by the Arduino practice judgment system are the HW list test and the HW connection test. After the instructor configures the hardware using Fritzing and uploads the hardware configuration diagram in XML format to the server, the verification conditions to be used for the test are automatically created. The generated confirmation condition is used to test the hardware configured by the students in Fritzing, and feedback based on the test result is also generated using the confirmation condition.

The HW list and connection confirmation conditions are created by extracting the information defined in the XML. First, the HW list extracts the names of all HWs connected to the Arduino and creates a list for testing. If the instructor’s HW list matches the student’s HW list, the test is considered passed.

Next, the HW connection check condition is created by combining the information on each port of the Arduino and the connection information for each other HW port. The generated HW connection confirmation condition checks the HW connection by comparing the instructor’s condition with the student’s condition, and provides feedback accordingly to the student.

4.2 Test Case Description

Arduino is a system with the characteristics of an embedded system, and it is difficult to test Arduino SW with the test method used in the existing judgment system. Accordingly, the test case was newly defined by reflecting the characteristics of TTCN-3 (Testing and Test Control Notation version 3) [7].

No.	TTCN-3 Test Object	Arduino Test Case
1	Template(Protocol)	Partial Order between events
2	Response Time	Reaction Time between events
3	Receiving Data	HW Value Variance

Fig. 3. Types of test cases and test targets of TTCN-3

As shown in Fig. 3, the defined test case was defined by reflecting the characteristics for testing the template (protocol), response time, and received data to be tested in TTCN-3. An event used in a test case means one function execution log (HW operation). The defined test case is defined so that the sequence between events, delay value between events, and change in HW value can be checked. Figure 4 shows the composition of the newly defined test case.

ID	Contents
Format	<Event Ordering> : [Constraints](Optional) <FuncName(Param1, Param2), FuncName(Param1, Param2)>:[,] / [Param2.asc or desc]
Example 1	<digitalWrite(RED_LED, HIGH), digitalWrite(RED_LED, LOW)>
Example 2	<digitalWrite(RED_LED, HIGH), digitalWrite(RED_LED, HIGH)>:[50, 500]
Example 3	<analogWrite(RED_LED, *) , analogWrite(RED_LED, *)>:[Param2.asc]

Fig. 4. Test case configuration

As shown in the figure, the test case writing form consists of two parts: ‘<Event Ordering>’ and ‘[Constraints]’. First, ‘<Event Ordering>’ means that two events can be written between < >, and when two other events are input, the preceding event is executed first and the latter event is executed when performing a test. You can write a test case by entering two parameters in the

function execution log. Param1 means the HW name. Param2 means a number or command to be transmitted, and Param2 can be written using ‘*’ in addition to the reserved words used in the existing Arduino, which means that it does not matter which reserved words or numbers are input. Next, ‘[Constraints]’ means constraints and can be selectively entered and applied to test cases. Currently, the constraint is divided into two. The first constraint is used to check whether the delay is within the range by designating the delay range when it is necessary to check the delay between events written in the test case. The second constraint is a constraint used in the HW numerical change test case, and is used to express the constraint on the increase or decrease of the numerical value (Param2) delivered to a specific HW. In Fig. 4, 3 examples are written in the form of a defined test case, and Example 1 is a test case written using only Event Ordering. Examples 2 and 3 are examples of test cases written by applying Constraints, and the test method is applied differently depending on the applied Constraints.

5 Judge Experiment

The Arduino practice judgment system was applied to the subject called ‘IoT Practice’, which was opened in the second semester of 2021, and the number of students enrolled was 15. The applied system was a prototype, and only Partial Order between Events Test Case was used for example judgment. The system reduced the workload of the instructor, and students were able to receive feedback on their practice results more quickly. Lectures were conducted by checking the learning progress of the students through the system, and the learning progress was confirmed through data such as grading results, number of submissions, errors occurred, and submission time. The instructor can check the progress of each student and find and guide students who need additional guidance from the instructor, and students who do not can proceed with their own learning by using the feedback provided by the system. In this way, instructors can conduct more efficient lectures by using the functions provided by the system.

6 Conclusion

In this paper, an Arduino practice judgment system was developed and applied to actual lectures, and the results were shown. The Arduino practice scoring system is a system that tests the practice developed by students using the XML format hardware diagram and Arduino source code supported by Fritzing. By automating the grading and providing feedback previously performed by instructors, the burden on instructors was relieved, and more students were provided with feedback more frequently than before in the same amount of time. In the experiment conducted in this study, only one type of test case was applied, and an experiment to secure additional data is planned.

Acknowledgements. This study was supported by the BK21 FOUR project (AI-driven Convergence Software Education Research Program) funded by the Ministry of Education, School of Computer Science and Engineering, Kyungpook National University, Korea (4199990214394) and the Basic Science Research Program through the National Research Foundation of Korea(NRF) funded by the Ministry of Education (NRF-2017R1D1A3B04035880 and NRF-2018R1A6A1A03025109)

References

1. Albatish, I., Mosa, M.J., Abu-Naser, S.S.: Arduino tutor: an intelligent tutoring system for training on arduino (2018)
2. Bryan, J.A.: Automatic grading software for 2D cad files. *Comput. Appl. Eng. Educ.* **28**(1), 51–61 (2020)
3. Fritzing: Firtzing website (2022). <https://fritzing.org/>. Accessed 27 Apr 2022
4. Kapinski, J., Deshmukh, J.V., Jin, X., Ito, H., Butts, K.: Simulation-based approaches for verification of embedded control systems: an overview of traditional and advanced modeling, testing, and verification techniques. *IEEE Control Syst. Mag.* **36**(6), 45–64 (2016)
5. Luxton-Reilly, A., et al.: Introductory programming: a systematic literature review. In: *Proceedings Companion of the 23rd Annual ACM Conference on Innovation and Technology in Computer Science Education*, pp. 55–106 (2018)
6. Puspitasari, D., Arhandi, P., Saputra, P., Syaifudin, Y., Himawan, H., Sholihah, P.: Online judge MySQL for learning process of database practice course. In: *IOP Conference Series: Materials Science and Engineering*. vol. 523, p. 012046. IOP Publishing (2019)
7. TTCN-3: TTCN-3 website (2022). <https://www.ttcn-3.org>. Accessed 27 Apr 2022
8. Wasik, S., Antczak, M., Badura, J., Laskowski, A., Sternal, T.: A survey on online judge systems and their applications. *ACM Comput. Surv. (CSUR)* **51**(1), 1–34 (2018)
9. Weißleder, S., Schlingloff, H.: An evaluation of model-based testing in embedded applications. In: *2014 IEEE Seventh International Conference on Software Testing, Verification and Validation*, pp. 223–232. IEEE (2014)



A Hybrid Approach for Mitigating Learners' Rogue Review Behavior in Peer Assessment

Gabriel Badea  and Elvira Popescu  

Computers and Information Technology Department, University of Craiova, Craiova, Romania
{gabriel.badea,elvira.popescu}@edu.ucv.ro

Abstract. Rogue reviews represent an important challenge in the application of the peer assessment process in educational settings. Students sometimes provide inappropriate evaluations of their peers' work, due to laziness, malevolence, retaliation, or dishonesty. Various approaches have been described in the literature for mitigating personal bias and increasing assessment reliability. The current paper proposes an innovative mechanism for managing rogue reviews, based on a hybrid approach: combining automatic labelling of suspicious reviews by the system with manual analysis of their content by the teacher. In addition, dedicated prompts are displayed to the students, providing specific recommendations for revising potentially rogue reviews. The mechanism was integrated in an existing peer evaluation system called LearnEval. The platform was used in a pilot study whose results are reported and discussed in the paper; several lessons learned from the experience and potential improvements are also included.

Keywords: Peer evaluation · Peer assessment platform · Rogue reviews · Feedback quality

1 Introduction

Peer assessment is an educational activity applied in a wide range of areas, both formally and informally. The feedback offered by the students is a critical component for learning, being helpful in various ways: it provides multiple perspectives for authors to consider [10], each learner receives individualized feedback [6], it encourages reflection and metacognition, and fosters critical thinking skills [14]. Some practitioners consider the timeliness and diversity of the feedback even more important than its content [10]; however, ensuring a quality peer feedback is an important endeavor [7, 10, 16].

A challenging aspect in the successful application of the peer assessment process is represented by the rogue behavior of the students. Rogue reviews are inappropriate evaluations that have as source laziness, malevolence, retaliation, or dishonesty [15], affecting students' learning enthusiasm [17]. The rogue assessors assign arbitrary grades regardless of the solution quality [15]; indeed, in every course it can be assumed that a part of the learners will assign random grades during the assessment process [9]. In this context, several approaches have been proposed in the literature to mitigate the personal bias and increase assessment reliability: use of multiple reviewers for the same solution

to increase the accuracy [15, 17]; let evaluators see how other peers have assessed the same solutions they reviewed to increase self-awareness [8]; use of design prompts that alert assessors to change their reviewing standards when they submit rogue evaluations [17]; use of advanced and innovative systems that prevent rogue review behavior from the start [15].

In this paper we propose a novel hybrid approach, where automatic machine labelling is combined with manual human marking to facilitate the detection of rogue reviews, by considering aspects related to time allotted, grades assigned, or feedback content. We report on the integration of the rogue review management mechanism in an existing peer evaluation system called LearnEval, offering details regarding the implementation, context of use, a pilot study, and an initial analysis of the potentially rogue reviews. LearnEval [1, 2] is an innovative peer assessment platform that has been applied in several scenarios, especially in Project-Based Learning (PBL) settings [3]. The system allows students to assess their peers' artifacts by assigning grades and providing feedback; the students are guided during the assessment using criteria defined by the instructor; advanced mechanisms for reviewer calibration, monitoring and visualizations as well as dynamic review allocation are included in the platform.

The paper is structured as follows. Section 2 presents related work about peer assessment reliability and rogue reviewing. Section 3 describes the proposed hybrid mechanism for detecting and preventing rogue review behavior. Section 4 reports the results obtained in a pilot study performed with 52 undergraduate students. Section 5 contains a discussion about the findings, draws some conclusions and offers perspectives for future work.

2 Related Work

In the following, we present relevant works that address techniques for detecting rogue review behavior and examine peer assessment reliability.

Various ways for increasing feedback quality and avoiding the clustering of grades are presented in an early paper [7]. Several approaches are proposed, such as: deny credit unless submitting the evaluations, prevent access to one's own received feedback until the student provides feedback to peers, compute student's grade based on the grades received by the peers they reviewed (thus encouraging highly useful evaluations), and include an additional phase where the learner assesses the reviews performed by others. In addition, the accuracy of grading could be increased by requesting learners to complete a pre-certification test before evaluating their peers. Furthermore, the authors propose additional methods for preventing the clustering of grades, such as: use ranking instead of grading, or assign each student a limited number of shares that have to be distributed among the reviewed solutions.

A study to assess the accuracy and effectiveness of distributed peer assessment and to determine how often issues such as rogue reviews arise is carried out in [15]. Several factors that foster the rogue behavior are addressed, such as: laziness, retaliation, collusion, and competition. The authors use incentives to mitigate such behavior. For instance, students are demanded to provide the assessments prior to seeing their solution score. On the other hand, laziness is counteracted by reckoning the assessment process

as a course activity and tailoring the student workload appropriately. An analysis was conducted to assess the level of retaliation by correlating the review scores with the ones assigned by the authors to evaluations and the value found was low. Several cases of collusion were identified, where pairs of learners settled to assign each other high scores. However, very few proofs of rogue behavior were detected in practice. Overall, the results highlight that peer assessments are accurate in comparison to a recognized standard of evaluation.

An interesting approach which relies on analyzing the lexical sophistication of feedback comments in a large-scale study is reported in [10]. The paper examines reviewers' competence in making appropriate evaluations and the complexity of the feedback provided. The textual complexity is assessed by employing five metrics: comment length, number of distinct tokens, median word length, word frequency, and average token-type ratio. The findings show that high performing students generally write more and have a better vocabulary than the rest. In addition, the analysis of lexical sophistication highlights the fact that reviewers generally produce less complex comments than instructors. However, the authors suggest that the gap could be reduced by assigning multiple evaluators for a single submission. Thus, the effect of rogue reviews is slightly lessened by assigning grades to solutions computed based on a weighted average of the received marks.

A different mechanism is proposed in PeerStudio platform [12], which shows reviewers short tips based on the feedback they provide. These helpful tips are generated using a list of relevant words extracted from the draft submitted by the student and the assignment description. Furthermore, the system uses the number of relevant words in the feedback and its length to propose enhancements. A large amount of low-quality comments is detected by employing this simple heuristic. The platform guides the reviewer to provide the most relevant feedback for the current state of the submission by internally computing the solution quality (low, medium, high). Additionally, the authors can evaluate the received assessments and deliver messages to the staff. The approach was quite successful, with students considering 45% of the comments to be "somewhat concrete or better", offering links to helpful resources, or suggestions on how to improve the solution (while the rest of the comments were simply praise or support messages).

Statistical measures are also proposed in [17] for detecting non-consensus and radical review behavior in peer assessment. Non-consensus occurs when multiple reviewers disagree and have contrary opinions on the same aspect. Moreover, a part of the reviewers have radical behavior during assessment by repeatedly assigning low or high grades, without considering the actual solution quality. In this context, the EduPCR peer assessment platform automatically identifies non-consensus by means of the standard deviation of the grades assigned by reviewers. Teacher arbitration is triggered when such non-consensus is discovered in a group of reviewers. On the other hand, a reviewer is marked by the platform as a radical candidate when they repeatedly offer high or low grades. In such cases, a short message or an email is delivered to the instructor, who then manually inspects the grades assigned by that reviewer.

Furthermore, some papers propose innovative ways to increase peer assessment reliability. For instance, fuzzy logic is used in [4]; the approach is employed to model opinions, the opinions are further weighted based on their validity, and in the end, they

are aggregated to achieve a reliable process. Automatic classifiers are used in [13] to evaluate the quality of the peer assessment process, and various metrics for gauging the reliability and validity of the reviewers are presented. Paper [5] presents the hypothesis that allowing both reviewers and solution authors to introduce themselves and exchange messages with each other during the peer assessment process rises the feedback quality. Finally, game-like elements are employed in [11] to incentivize students to provide reliable assessments.

The current paper adds to the literature by proposing a novel mechanism for managing rogue reviews, which is based on a hybrid approach: combining automatic labelling of suspicious reviews by the system with manual analysis of their content by the teacher. The automatic detection is based on a quantitative approach, computing a score based on a set of factors which could indicate a potentially rogue review. An additional mechanism is proposed for encouraging students to carefully check their reviews and make appropriate revisions before submitting them, as described in the next section.

3 A Mechanism for Mitigating Rogue Review Behavior

The starting point of our approach is an existing peer assessment system, called LearnEval, which we proposed in [1, 2]. The platform supports a highly configurable assessment scenario, providing several functionalities for both students and instructors: calibration module, open learner model, comprehensive monitoring and visualization features, dynamic review allocation module. In this section we report on the design and integration of a mechanism for detecting and preventing rogue review behavior. More specifically, a hybrid approach is proposed, which combines automatic appraisal of the reviews by the system with human judgment. The system tags specific reviews as potentially rogue, considerably reducing the amount of reviews that need to be manually checked by the teacher. In addition to this detection mechanism, a prevention mechanism is also put in place: warning and recommendation messages are displayed to the students when they try to submit potentially rogue reviews, which can be used to improve the review content.

A quantitative approach is used for assessing the likelihood for a review to be rogue. More specifically, a *Rogue Score* is computed every time a review is submitted, considering various criteria related to: time required for performing the evaluation, grades assigned to the assessment criteria, and quality of the feedback provided. Two scores are stored for each review: an *Initial Rogue Score* computed when the student aims to send the evaluation for the first time, and a *Final Rogue Score* computed when the evaluation is actually submitted (after the student has the chance to revise/improve it, taking into account the recommendations provided by the system).

Based on our own experience as well as a review of the literature, we extracted a set of 14 criteria that could indicate a rogue evaluation. These criteria are described in Table 1, including an impact level for each of them (i.e., criterion score). The *Rogue Score* of a review is computed by adding the corresponding scores for each fulfilled criterion. Starting from this score, a set of features are provided by the rogue review mitigation mechanism in LearnEval, both for the student and the teacher, as described in the following subsections.

Table 1. Set of criteria indicating a potentially rogue review, integrated in LearnEval

Category	Criterion description	Criterion Score
Time	Review submitted in less than five minutes after it was assigned	100
Grade	Student gives the same grade for all the reviews performed for the current assignment	100
	Student gives the same maximum grade (10) for all the reviews performed for the current assignment	100
	Student gives the same minimum grade (1) for all the reviews performed for the current assignment	50
	Student gives the same median grade (7) for all the reviews performed for the current assignment	50
Feedback	Student provides similar feedback (over 90% textual similarity) for two different assessment criteria of the current review	50
	Student provides similar feedback (over 90% textual similarity) for at least four different assessment criteria (belonging to different reviews of the same assignment)	100
	Student provides similar feedback (over 90% textual similarity) with the one written by a peer, for at least three different assessment criteria	50
	Student provides feedback that contains only whitespaces	50
	Student provides feedback that does not contain any letters or digits	50
	Student provides feedback that contains less than five words	75
	Student provides feedback that contains less than five distinct words	50
	Student provides feedback that contains repeating consecutive words (at least 4 times the same word)	25
	Student provides feedback that contains repeating consecutive letters (at least 5 times the same letter)	15

3.1 Student Perspective

The rogue review prevention mechanism automatically verifies each evaluation when the student aims to submit it (i.e., clicks the submit button). In case at least one of the rogue criteria from Table 1 is satisfied, the platform prevents the submission and displays a notification message to the student, asking them to recheck the review. A list of revision recommendations, corresponding to each fulfilled criterion, is provided to the learner, as illustrated in Fig. 1. The student has the opportunity to perform the suggested improvements and resubmit the review, after explicitly acknowledging that the review was appropriately checked and revised.

However, if the student chooses to submit a review which still fulfills the rogue criteria and is marked as such by the teacher, then their reviewing skills score is automatically

Fig. 1. LearnEval rogue review prevention module: revision recommendations displayed to the student when submitting a potentially rogue review

decreased. More specifically, LearnEval models the learner's assessment capabilities by computing an aggregated score:

$$\begin{aligned} \text{ReviewingScore} = & \text{PeerBackReviewsAvg} * \text{wm1} + \text{TeacherBackReviewsAvg} * \text{wm2} \\ & + \text{AgreementWithFinalMark} * \text{wm3} + \text{CalibrationScore} * \text{wm4} - 0.5 * \text{RogueReviewsCount} \end{aligned}$$

where: *PeerBackReviewsAvg* represents the mean grade of the back-reviews received by the student from the solution authors, *TeacherBackReviewsAvg* represents the mean grade of the back-reviews received by the student from the teacher, *AgreementWithFinalMark* depicts the accuracy of the grades assigned by the student, *CalibrationScore* depicts student's reviewing capabilities at the start of the peer assessment process and *RogueReviewsCount* represents the number of reviews marked as rogue by the teacher (each entailing a penalty of 0.5 points); the weights are configured by the teacher, such that $\text{wm1} + \text{wm2} + \text{wm3} + \text{wm4} = 1$.

3.2 Teacher Perspective

The platform provides a dedicated *Reviews* page which allows the teacher to readily visualize all the submitted evaluations and their rogue scores (as illustrated in Fig. 2).

The instructor can order the reviews based on their *Rogue Score*, thus easily identifying the evaluations which are more likely to be rogue and focus their efforts on checking those first. In addition, detailed information is available about each student evaluation (as illustrated in Fig. 3), such as: time in review (interval between review assignment and review submission); grade assigned in back-review by the solution author; difference between the grade assigned by the reviewer and the final solution grade; initial and final *Rogue Score* along with a description of the fulfilled criteria (if any). The teacher can use this information, together with the actual content of the evaluation, to decide whether a review is indeed rogue. Once marked as rogue by the instructor, that review is no longer taken into account when computing the final grade of the solution.

Reviews
Reviews for Human-Computer Interaction 2020-2021

Search [] 10 [] []

Reviewer	Author	Solution URL	Assignment Name	Solution Mark	Rogue Score	Review Date	Review Category	Commands
Student Name	Solution Author	https://mega.nz...	Assignment 4	7.83	575	4/21/2021 9:3...	RLC	[] []
Student Name	Solution Author	https://mega.nz...	Assignment 1	5.73	375	3/16/2021 12:...	RMC	[] []
Student Name	Solution Author	https://mega.nz...	Assignment 2	8.35	375	3/24/2021 7:3...	RLC	[] []
Student Name	Solution Author	https://mega.nz...	Assignment 2	9.97	375	3/24/2021 9:0...	RMC	[] []
Student Name	Solution Author	https://mega.nz...	Assignment 4	6.92	375	4/21/2021 3:5...	RMC	[] []
Student Name	Solution Author	https://mega.nz...	Assignment 4	10	375	4/21/2021 3:5...	RMC	[] []
Student Name	Solution Author	https://mega.nz...	Assignment 4	9.95	375	4/21/2021 9:3...	RLC	[] []
Student Name	Solution Author	https://mega.nz...	Assignment 4	4.00	325	4/21/2021 12:...	RLC	[] []
Student Name	Solution Author	https://mega.nz...	Assignment 4	7.83	300	4/16/2021 11:...	RHC	[] []
Student Name	Solution Author	https://mega.nz...	Assignment 4	9.95	300	4/16/2021 11:...	RHC	[] []

Showing 1 to 10 of 707 entries

© 2022 - LearnEval. All rights reserved.

Fig. 2. LearnEval rogue review prevention module: teacher view of the reviews and their rogue score

4 Rogue Reviews Analysis: Pilot Study

4.1 Context of Study

LearnEval peer assessment platform integrating the proposed rogue review prevention mechanism was applied in the context of a *Human-Computer Interaction* course at the University of Craiova, Romania. The course followed a project-based learning approach, being taught to 52 4th year students, during the second semester of 2020–2021 academic year.

The task of the project was to design, build, and assess the user interface of a web application. The project was split in four milestones: the first deliverable referred to user modeling and storyboarding tasks; the second deliverable required students to build low

The screenshot shows the 'Review details' page in the LearnEval system. The page title is 'Review details' with the subtitle 'Details for the selected review'. The main content is a table with the following data:

Marked as Rogue	No
Back Review Mark	8
Difference Mark Assigned	0.17 (s1: 7.00, s2: 8.00, p: 6)
Time in Review	0 h 7 m 59 s
Submission description (as provided by reviewer)	This is a sample short description of the submission under review. This is a sample test review to show the designed prompts that are displayed to the reviewer when submitting a potentially rogue review.

Criterion Name	Review criterion 1
Description	Review criterion 1 sample description.
Mark	7
Feedback	The reviewer considered the messages that were sent by the system. The review was updated and now contains appropriate and helpful feedback for the solution author. The vocabulary is also more elaborated.

Below the table, there is a red button labeled 'Mark as Rogue'. Underneath, the text reads: 'Initial Rogue Score: 175. The review submitted initially by the student had the following issues:' followed by a list of three issues:

1. The student submitted a very fast review in less than 5 minutes.
2. The review contains less than 5 distinct words.
3. The review contains a sequence of at least 4 consecutive repeating words.

The text continues: 'The final review submitted by the student had no issues.' At the bottom, there is a blue button labeled 'Back Review'.

Fig. 3. LearnEval rogue review prevention module: teacher detailed view of a potentially rogue review

and high fidelity prototypes; the third deliverable required learners to implement the actual user interface; and the last one involved interface evaluation and usability testing.

The milestones of the project were used to create four peer assessment sessions. A session had a submission phase, where students submitted solutions (deliverables), followed by a review period. The review period was further split in two stages: a first review phase, where students had to mandatory assess three peers' solutions, and an extra review phase, where students were able to optionally assess up to three peers' solutions. The learners were granted bonus points at the end of the course based on the number of extra solutions reviewed. The assessment criteria were defined by the teacher and varied according to the requirements of the milestone. The student had to assign a grade on a scale from 1 to 10 and provide feedback for each criterion. Furthermore, a short summary of the solution was required. At the beginning of the semester students were provided with an introductory meeting in which the instructor explained the peer assessment process, including a description of the rogue review behavior, its consequences and why it should be avoided.

4.2 Results Analysis

In the following, we analyze the *Rogue Scores* values computed by the system for potentially rogue evaluations. A total of 707 reviews were submitted by the students.

Almost a quarter of these peer evaluations (i.e., 172 reviews), met at least one rogue review criterion from Table 1, thus requiring teacher's attention.

Most of these reviews resided in the [100, 200) score range, as illustrated in Table 2. The most common causes encountered were: many evaluations were sent very quickly by the students, in less than 5 min (criterion counting 100 points); and many reviewers provided similar feedback for at least two different assessment topics (criterion counting 50 points).

Table 2. Distribution of reviews based on *Rogue Score* (the percentages are computed out of the total number of reviews identified as potentially rogue by the system – i.e., 172 reviews)

Rogue score	[1, 100)	[100, 200)	[200, 300)	[300, 400)	400+
Reviews count	18 (10%)	107 (62%)	36 (21%)	10 (6%)	1 (<1%)

The teacher manually checked each potentially rogue review and marked the truly rogue ones correspondingly. Overall, 23% of the potentially rogue reviews were found to be actually rogue by the instructor, as detailed in Table 3. As can be seen, the percentage of evaluations identified as rogue by the teacher increases as the *Rogue Score* increases, thus, a higher score raises the likelihood for an assessment to be actually rogue. A significant raise in the percentage can be noticed starting with [200, 300) interval; the rogue likelihood of an evaluation with a score of at least 200 is more than 50% - hence this could be considered a cut-off point, above which additional measures could be taken by the system.

Table 3. Distribution of teacher-identified rogue reviews based on *Rogue Score* (the percentages are computed out of the number of reviews identified as potentially rogue by the system for each interval, as shown in Table 2)

Rogue score	[1, 100)	[100, 200)	[200, 300)	[300, 400)	400+	Total
Reviews count	1 (6%)	11 (10%)	20 (56%)	6 (60%)	1 (100%)	39 (23%)

In addition to the inadequate feedback, the grades provided in the rogue reviews were also less accurate. The average difference between the grade assigned by the student and the final solution grade in the reviews identified as potentially rogue by the system (but not by the teacher) was 0.81; as expected, this value is lower than in case of reviews identified as rogue by the teacher (i.e., 1.11). Furthermore, the correlation between the grade assigned by the student and the grade assigned by the instructor in the reviews identified as potentially rogue by the system (but not by the teacher) was 0.78, which is higher than the correlation in case of reviews identified as rogue by the teacher (i.e., 0.49).

While the rogue review detection process was successful, the review improvement component did not work as expected. Although the students were provided with review

revision suggestions, as described in Sect. 3.1, the learners rarely took advantage of these recommendations before resubmitting their reviews. More specifically, only 6 reviews were revised according to the system suggestions and only 2 of them were substantially improved. Therefore, additional measures and better incentives must be devised to motivate reviewers to enhance their feedback quality.

5 Discussion and Conclusion

The paper presented the integration of a rogue review mitigation mechanism in LearnEval peer assessment platform. The mechanism applies a hybrid approach where automatic machine labelling of potentially rogue reviews is combined with teacher's manual check. The advantages of the system are twofold: on one hand, the platform significantly decreases the time spent by the teacher to identify rogue reviews, by already flagging suspicious evaluations and thus reducing the search space. On the other hand, improvement recommendations are automatically displayed to the students, who have the opportunity to revise their reviews before submitting them, which could lead to a lower number of rogue reviews.

The mechanism was employed in a pilot study involving 52 students, in the context of a Human-Computer Interaction course. The system flagged around a quarter of the evaluations as potentially rogue, which significantly decreased teacher's workload. Furthermore, an initial analysis showed that the higher the *Rogue Score*, the higher the likelihood that the review was actually rogue and marked as such by the instructor; this comes as a validation of our detection mechanism and the proposed rogue criteria. Nevertheless, the scores for some criteria could be revised in light of our initial findings; an even better approach would be to make these scores configurable by the teacher, based on the specificities of each instructional scenario. In addition, a more advanced mechanism could be envisioned, based on an extended list of criteria and a more complex fuzzy logic approach.

A limitation of our study was that very few students actually followed the recommendations provided by the system in order to improve their reviews. Various approaches could be applied to address this issue. First of all, the current mechanism only displays revision suggestions, but does not prevent the submission of the review if these suggestions are not followed. Therefore, a more complex, layered approach could be proposed, based on the value of the *Rogue Score*, such as:

1. Since the percentage of reviews identified as rogue by the teacher was low in the interval $[0, 200)$, the flow could be kept unchanged below this threshold (i.e., simply prompt the student to consider the recommendations made by the system).
2. In the next interval, $[200, 300)$ range, more than half of the evaluations were identified as rogue by the teacher, thus more restrictive measures could be applied. Hence, the submission of the review should not be allowed in case its final *Rogue Score* is the same as the initial one (i.e., the student needs to address at least one of the rogue criteria). Furthermore, an automatic notification could draw teacher's attention to immediately check the potentially rogue review.

3. In the following score intervals (i.e., ≥ 300), a majority of the evaluations were marked as rogue by the teacher. Therefore, the student should not be allowed to submit a review with a *Rogue Score* over 300. However, an appeal mechanism should be made available to the student, who could ask for teacher's evaluation in case they consider their review to be a valid one.

It should be noted that the above thresholds are inferred from the current pilot study and may not be generally valid. Therefore, a configurable approach could be envisioned, in which the teacher can set the thresholds based on the particularities of the peer assessment scenario and the reviewing skills of the students. Furthermore, dedicated incentives for fostering students' motivation and encouraging them to provide higher quality feedback need to be integrated in the system. Finally, we aim to apply the improved version of the LearnEval platform in more courses and instructional scenarios and conduct a more in-depth analysis of the peer assessment quality.

References

1. Badea, G., Popescu, E.: A web-based platform for peer assessment in technology enhanced learning: student module prototype. In: Proceedings of ICALT 2019, Maceió, Brazil, pp. 372–374. IEEE Computer Society Press (2019). <https://doi.org/10.1109/ICALT.2019.00115>
2. Badea, G., Popescu, E.: Instructor support module in a web-based peer assessment platform. In: Proceedings of ICSTCC 2019, Sinaia, Romania, pp. 691–696. IEEE (2019). <https://doi.org/10.1109/ICSTCC.2019.8885687>
3. Badea, G., Popescu, E.: Using peer assessment in conjunction with project-based learning: a comparative study. In: Proceedings of ICALT 2020, pp. 376–380. IEEE Computer Society Press (2020). <https://doi.org/10.1109/ICALT49669.2020.00119>
4. El Alaoui, M., El Yassini, K., Ben-Azza, H.: Peer assessment improvement using fuzzy logic. In: Ben Ahmed, M., Boudhir, A.A., Younes, A. (eds.) SCA 2018. LNITI, pp. 408–418. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-11196-0_35
5. Gamage, D., Whiting, M., Perera, I., Fernando, S.: Improving feedback and discussion in MOOC peer assessment using introduced peers. In: 2018 IEEE International Conference on Teaching, Assessment, and Learning for Engineering (TALE), pp. 357–364 (2018). <https://doi.org/10.1109/TALE.2018.8615307>
6. Gehringer, E.F.: Strategies and mechanisms for electronic peer review. In: 30th Annual Frontiers in Education Conference (FIE 2000), pp. F1B/2–F1B/7 (2000). <https://doi.org/10.1109/FIE.2000.897675>
7. Gehringer, E.F.: Assignment and quality control of peer reviewers. In: ASEE Annual Conference and Exposition, American Society for Engineering Education, Albuquerque, USA (2001)
8. Gehringer, E., Ma, K., Duong, V.: What peer-review systems can learn from online rating sites. In: State-of-the-Art and Future Directions of Smart Learning, pp. 341–350 (2016). https://doi.org/10.1007/978-981-287-868-7_42
9. Hamer, J., Ma, K.T.K., Kwong, H.H.F.: A method of automatic grade calibration in peer assessment. In: Proceedings of the Seventh Australasian Computing Education Conference (ACE2005), Vol. 42 of Conferences in Research and Practice in Information Technology, Newcastle, NSW, Australia, Australian Computer Society, pp. 67–72 (2005)

10. Hamer, J., Purchase, H.C., Denny, P., Luxton-Reilly, A.: Quality of peer assessment in CS1. In: Proceedings of the Fifth International Workshop on Computing Education Research Workshop, Berkeley, CA, USA, 10–11 August 2009, pp. 27–36 (2009). <https://doi.org/10.1145/1584322.1584327>
11. Indriasari, T.D., Luxton-Reilly, A., Denny, P.: Investigating accuracy and perceived value of feedback in peer code review using gamification. In: Proceedings of the 26th ACM Conference on Innovation and Technology in Computer Science Education, pp. 199–205 (2021). <https://doi.org/10.1145/3430665.3456338>
12. Kulkarni, C.E., Bernstein, M.S., Klemmer, S.R.: PeerStudio: rapid peer feedback emphasizes revision and improves performance. In: Proceedings of the Second ACM Conference on Learning @ Scale, Vancouver, BC, Canada, pp. 75–84 (2015). <https://doi.org/10.1145/2724660.2724670>
13. Molina-Carmona, R., Sattore-Cuerda, R., Compañ-Rosique, P., Llorens-Largo, F.: Metrics for estimating validity, reliability and bias in peer assessment. *Int. J. Eng. Educ.* **34**(3), 968–980 (2018)
14. Politz, J.G., Patterson, D., Krishnamurthi, S., Fislser, K.: CaptainTeach: multi-stage, in-flow peer review for programming assignments. In: Proceedings of the 2014 Conference on Innovation & Technology in Computer Science Education (ITiCSE 2014), pp. 267–272. ACM, New York (2014). <https://doi.org/10.1145/2591708.2591738>
15. Reily, K., Finnerty, P.L., Terveen, L.: Two peers are better than one: Aggregating peer reviews for computing assignments is surprisingly accurate. In: Proceedings of the ACM 2009 International Conference on Supporting Group Work, ACM, Sanibel Island, Florida, USA, pp. 115–124 (2009). <https://doi.org/10.1145/1531674.1531692>
16. Staubitz, T., Petrick, D., Bauer, M., Renz, J., Meinel, C.: Improving the peer assessment experience on MOOC platforms. In: Proceedings of the Third ACM Conference on Learning @ Scale, ACM, Edinburgh, Scotland, UK, pp. 389–398 (2016). <https://doi.org/10.1145/2876034.2876043>
17. Wang, Y., Liang, Y., Liu, L., Liu, Y.: A multi-peer assessment platform for programming language learning: considering group non-consensus and personal radicalness. *Interact. Learn. Environ.* **24**(8), 2011–2031 (2016). <https://doi.org/10.1080/10494820.2015.1073748>



DeepCode: An Annotated Set of Instructional Code Examples to Foster Deep Code Comprehension and Learning

Vasile Rus^(✉), Peter Brusilovsky, Lasang Jimba Tamang, Kamil Akhuseyinoglu, and Scott Fleming

The University of Memphis, Memphis, TN 38152, USA
vrus@memphis.edu

Abstract. We present here a novel instructional resource, called DeepCode, to support deep code comprehension and learning in intro-to-programming courses (CS1 and CS2). DeepCode is a set of instructional code examples which we call a codeset and which was annotated by our team with comments (e.g., explaining the logical steps of the underlying problem being solved) and related instructional questions that can play the role of hints meant to help learners think about and articulate explanations of the code. While DeepCode was designed primarily to serve our larger efforts of developing an intelligent tutoring system (ITS) that fosters the monitoring, assessment, and development of code comprehension skills for students learning to program, the codeset can be used for other purposes such as assessment, problem-solving, and in various other learning activities such as studying worked-out code examples with explanations and code visualizations. We present here the underlying principles, theories, and frameworks behind our design process, the annotation guidelines, and summarize the resulting codeset of 98 annotated Java code examples which include 7,157 lines of code (including comments), 260 logical steps, 260 logical step details, 408 statement level comments, and 590 scaffolding questions.

Keywords: Code comprehension · Intelligent tutoring systems · Self-explanation

1 Introduction

Code comprehension, i.e., understanding of computer code, is a critical skill for both learners and professionals. Students learning computer programming spend a significant portion of their time reading or reviewing someone else's code (e.g., code examples from a textbook or provided by the instructor). Furthermore, it has been estimated that software professionals spend at least half of their time analyzing software artifacts in an attempt to comprehend computer source code. Reading code is the most time-consuming activity during software maintenance, consuming 70% of the total lifecycle cost of a software product [5, 8, 32]. O'Brien [24] notes that source code comprehension is required when a programmer maintains, reuses, migrates, reengineers, or enhances software systems.

Therefore, offering support to enhance learners' source code comprehension skills will have lasting positive effects for their academic success and future professional careers.

Our goal is to explore instructional strategies that promote deep code comprehension and learning. To this end, we present here a novel instructional resource, called DeepCode, which is a set of annotated code examples to support code comprehension and learning activities in intro-to-programming courses (CS1 and CS2). Indeed, the design of DeepCode was driven by our larger efforts on exploring instructional strategies that foster the development of code comprehension skills and the construction of accurate mental models and learning in conjunction with advanced education technologies such as conversational intelligent tutoring systems (ITSs; [33, 34, 45]) that use scaffolded self-explanations to foster the monitoring, assessment, and development of code comprehension skills for students learning to program. It is important to note that ITSs are a particular category of adaptive instructional systems (AISs) that offer both micro-level and macro-level adaptivity, as explained later.

While we developed DeepCode to serve our goal of developing an ITS for code comprehension and learning of programming concepts, the resulting codeset of annotated code examples can be used for other purposes such as assessment, problem-solving, and in various other learning activities such as studying worked-out examples or code visualizations as well as for research purposes such as exploring other instructional strategies, e.g., asking students to Explain-in-Plain-English target code examples (EiPE; 9, 22, 41).

The code examples included in the DeepCode dataset cover the vast majority of topics in a typical intro-to-programming course (CS1) - see the list of topics later. While the DeepCode instructional dataset contains Java code examples, the design principles and annotation guidelines are generally applicable to any programming language. The annotation guidelines were based on code comprehension theories, self-explanation theories, and the micro-macro adaptivity framework used by ITSs.

We are not aware of any similar resources available for code comprehension and learning activities that have been publicly released and which cover the vast majority of topics in CS1. Furthermore, one unique feature of our annotated code examples is the theory-driven annotation guidelines.

Work in the area of code comprehension targets a subset of CS1 topics and usually do not release code examples for use by others [2, 3, 27, 28]. Recent work on Explain-in-Plain-English (EiPE; 9, 24, 42) use large sets of code examples, e.g., Chen and colleagues [9] use 52 code examples scattered across various courses and course related activities: a CS1 for engineers' course (8 homework questions), 5 homework and exam problems in a 'CS1 for CS majors' course, 26 exam problems of which each student was assigned one problem on each of 5 exams, in a data structures course, and 13 additional questions as part of a paid survey that was offered to sophomore-level CS students. It should be noted that in EiPE tasks, learners/readers are asked to provide a high-level natural language (e.g., English) description of the code which is different from asking them to self-explain as detailed later. To the best of our knowledge, there was no principled way, theory-driven selection and annotation of the 52 code examples.

Other notable efforts to create questions/problems in CS Education fall into two broad categories: (1) creating traditional test questions, usually in multiple-choice format such as Canterbury Bank [31] and (2) creating more advanced learning content such as explained worked examples [7], codecasts [36], code animations [20, 38], Parson's problems [22], and code construction problems with automatic assessment [17, 39]. All these kinds of learning content are based on meaningful code examples, which could be explained, animated, or presented in a form of a problem. Surprisingly, research in this direction predominantly focused on creating authoring tools that allow end users create advanced content items leaving it to the users to create useful collections of examples or problems. Moreover, no theory-based guidelines for creating problems or examples are usually offered. In this context, our work bridges the gap between these two research direction. Similar to the motivation of question-bank developers, we focus on a collection of quality content. However, in contrast to relatively simple content in question banks, we focus on complete meaningful augmented code examples. This code examples could be used as-is in its original form or serve as a basis for creating collections of quality examples and problems using the advanced content authoring tools mentioned above.

The DeepCode codeset has the following main features meant to promote deep comprehension of code and learning of computer programming concepts:

- Explanations in the form of logical step comments capturing the domain model;
- Explanations in the form of logical step details comments capturing the program model and linking the program model and the domain model, i.e., the integrated model;
- Explanations of new concepts being introduced by each code example.
- Scaffolding hints in the form of questions for the domain model, program model, and the integrated model as well as for the new concepts (enables micro-adaptation).
- Topic ordering based on input from CS1 and CS2 instructors and prior research on programming concept difficulty and importance (enables macro-adaptation and implementation of learning strategies such as mastery learning and spacing effects).

2 Related Work

The development of the DeepCode set of instructional code examples has been guided by a number of theories and frameworks and recent advances in code comprehension and text comprehension research of which the following are the most important: reading/code comprehension theories [6, 16, 18, 19, 21, 26, 35, 37, 44], cognitive load theory [41], cognitive engagement theory [13, 14] and the ICAP framework [13], self-explanation theory [10, 12], and the intelligent tutoring framework that offers macro- and micro-adaptive instruction [45] with a focus on conversational tutoring that implement scaffolded self-explanation strategies [33] and related efforts such as the conversational tutor for program planning ProPL [24].

For instance, according to Pennington’s theory of code comprehension [25] involves the building of a domain model and of a program model, as detailed later. Accordingly, the DeepCode annotation guidelines specify adding comments, which we call logical step comments, describing the code from a domain perspective, i.e., describing the domain model. For each logical step, details of how those logical steps are implemented in a chunk of code are needed as well. This is meant to describe the program model as well as link the domain model to the program model. By the same token, based on cognitive load theory, each code example introduces only one new topic or concept and only individual statements that refer to the new concept that learners are supposed to learn are annotated with statement level comments. Concepts in other lines of code are supposed to have been mastered earlier while working on code examples corresponding to topics introduced earlier in the sequence of topics. Indeed, code examples are sequenced based on a priori defined sequence of topics and each code example is supposed to use only concepts related to previously mastered topics and the new topic, as detailed later. Scaffolding questions were also annotated for the logical step and logical step details comments in order to help with the development of scaffolding tutorial dialogues.

Of particular interest to our larger goals of building ITSs that scaffold learners’ code comprehension processes are self-explanation theories and the micro-/macro-adaptive framework for tutoring. Self-explanation theories [10, 12] indicate that students who engage in self-explanations, i.e. explaining the target material to themselves, while learning are better learners, i.e. learn more deeply and show highest learning gains. The positive effect of self-explanation on learning has been demonstrated in different science domains such as biology [11] and physics [15], math [1], and programming [3]. Furthermore, research has shown that guided self-explanation is effective too [1]. Self-explanation’s effectiveness for learning is attributed to its constructive nature, e.g., it activates several cognitive processes such generating inferences to fill in missing information and integrating new information with prior knowledge, and its meaningfulness for the learner, i.e., self-explanations are self-directed and self-generated making the learning and target knowledge more personally meaningful, in contrast to explaining the target content to others [30]. Several types of self-explanation prompts have been identified and explored such as justification-based self-explanation prompts [15] and meta-cognitive self-explanation prompts [11]. The code examples in DeepCode could be used with various prompts and we intend to do so in order to elicit from students a variety of responses to capture as much about their mental models and mental model construction processes as possible.

As already noted, our larger goal is to build an ITS for code comprehension in intro-to-programming courses. The behavior of any ITS, conversational or not, can be described using VanLehn’s two-loop framework [45]. According to VanLehn, ITSs can be described in broad terms as running two loops: the outer loop, which selects the next task to work on, and the inner loop, which manages the student-system interaction while the student works on a particular task. The outer loop provides macro-adaptivity, i.e., selects appropriate instructional topics and tasks for a learner to work on, e.g., based on their mastery level. In order to offer outer loop support, DeepCode is based on a sequence of intro-to-programming topics and instructional tasks per topics as explained later.

The inner loop of an ITS monitors students' performance through embedded assessment, updates its model of students' levels of understanding (that is, the student model), and uses the updated student model to provide appropriate scaffolding in the form of feedback and other scaffolds. Accordingly, each code example in DeepCode is annotated with hints in the form of questions which are meant to scaffold learners' comprehension processes. Conversational ITSs can use the instructional code examples and annotations, e.g., the hints, to interactively monitor, assess, and scaffold learners' comprehension and learning. This should lead to best learning outcomes, according to the Interactive, Constructive, Active, and Passive (ICAP; [13]) framework of cognitive engagement according to which interactive learning leads to best cognitive engagement.

As a way to illustrate how these various theories provided the foundations for our work, we will use a concrete example and present step by step the annotation guidelines and process with references to the underlying theories.

3 A Working Example

In order to better illustrate the guiding principles and theories underlying the development of the DeepCode annotation guidelines, we will make use throughout the paper of a concrete example related to the widely played game of Bingo - a simplified version of the game to be precise. The game is played with disposable paper boards which contain 25 squares arranged in five vertical columns and five rows. Columns are labelled 'B', 'I', 'N', 'G', 'O'. The cell in the middle is empty. Random numbers from 1 to 75 are used in the game and there are some restrictions on the range of values that can occur in each column, e.g., the 'B' column only containing numbers between 1 and 15 inclusive whereas the 'I' column containing only 16 through 30. We will work with a simplified Bingo game in which all 25 cells may contain any number between 1 and 75. Players must match rows, columns, or diagonals in randomly generated Bingo board.

Our task is to solve computationally, i.e., with the help of a computer program, the following problem: Automatically generate random boards for the (simplified) game of Bingo.

The Java code implementing the solution for this Bingo board generation task is shown in Fig. 1. Details about the annotations, i.e., the explanations added in the form of comments are provided next together with the underlying theoretical foundations. Figure 1 does not show all the annotations added because of space reasons. The missing annotations are exemplified throughout the narrative of the paper.

4 The DeepCode Annotation Guidelines

We now present a summary of the annotation guidelines for developed DeepCode, exemplifying the various elements and the underlying theories that form the foundations for those elements.

Our entire team was involved in the development of the DeepCode codset. Each team member was assigned a number of topics for which to create and annotate code examples. They were instructed to either identify code examples in various sources (textbooks, websites, etc.) and use them as they are (i.e., just focus on annotating them with comments/explanations) or modified, or create their own examples from scratch. In case a particular source was used, the annotators were supposed to specify the source as detailed later. All annotators went first through a training period and had access to a detailed annotation guidelines manual and a cheatsheet.

The general steps to create and annotate code examples for the DeepCode codeset were:

1. Create at least 4 code example for a target topic. We create 4 code examples for each topic to allow students who struggle to practice the same topic again following the master learning principle (Bloom, 1981).
2. Add metadata.
3. Identify major logical steps and add corresponding logical step comments.
4. Add logical step details describing how the logical step is carried out using programming concepts.
5. Generate statement level comments for specific lines of code referring to the newly introduced topic.
6. For each logical step and new concept references (captured by statement level comments), generate a sequence of instructional hints in the form of questions which can be used to help students understand and articulate the logical step and its details.
7. Add misconception detection information and remedial feedback.

Step 1: Code Example Creation. The general guidelines given to annotators to create code examples are given below:

- create code examples for the topics you were assigned make sure you are aware of the topic/concept ordering for the whole CS1/CS2 sequence as it is important for the code example creation and annotation. The list of topics and their order is: Preliminary Topics (Variables + Expressions + Constants + Primitive data type), Input, Math, Class, Strings, Logical Operators, If, If-else, Switch, While Loops, Do While, For loops, Nested Loops, Arrays, Two Dimensional Arrays, Array Lists, Classes + Objects, Methods, Inheritance, Exception Handling, Recursion, Sorting, and Searching.
- Each code example should focus on the topic for which the example is targeted and may rely on concepts/topics covered earlier in the ordered list of topics. This is meant to reduce the cognitive load on novices trying to learn programming. According to cognitive load theory [41], humans have a limited capacity working memory and an unlimited long-term memory. Therefore, during learning activities instructional strategies should minimize the short-term memory load and encourage the construction of knowledge structures, i.e., schemas, in long-term memory. To this end, each code example introduces one new concept/topic or subconcept/subtopic.

- The examples should have deterministic output if at all possible so that an intelligent tutoring system would be able to assess the correctness of student predictions. This may not always be possible, e.g., when input is required from the user or when a random process is involved.
- Each code example should be nicely edited using a Java/code editor that can format.
- Each code example should be built around a story, i.e., a real-life application of the code should be thought of that is meaningful to students – something they can relate to from their own life experience. This guideline plays a motivational role because using such relatable stories could lead to a more meaningful effort on learners’ part when trying to understand computational solutions to relatable challenges (as opposed to unfamiliar or abstract ones). Furthermore, using real-life applications which students can relate to it through their own life experience, minimizes the need for domain knowledge to understand the code examples (i.e., general world knowledge would suffice) should reduce the cognitive load on the learners allowing them to allocate cognitive resources on the core programming concepts to be learned. Examples of real-life applications or problems are feet to meter conversion, leap year detection, or Bingo boards.
- Code examples should compile and run as expected. Once a code example was created, compiled, and executed without errors, it needs to be augmented with metadata and instructional comments as detailed below.

Step 2: Metadata. Each code example is annotated with a header that specifies the annotator/author, topic(s), subtopic(s), source (if any), goal (what the code does, i.e., the problem it solves), input (if any), and output. The granularity of topic/subtopic is an issue in itself. The Java example in Fig. 1 does not offer all the corresponding metadata due to space reasons. We plan to publicly release the fully annotated examples as a GitHub repository once the publication of this work is being accepted.

Step 3: Identification of Logical Steps. This step is about annotating the code examples with high-level explanations describing the logical steps of the problem being solved. This guideline is based on program comprehension theories proposed over the past 50 years or so [6, 16, 19, 25, 28, 32, 34, 36, 39). A major problem with the traditional program comprehension models is that they were the result of analyzing expert programmers’ comprehension processes as opposed to novices’, i.e., individuals with no or almost no relevant knowledge. More recently, there is work addressing this issue such as Schulte and colleagues [34] who proposed an education comprehension model.

While the various models of code comprehension differ in what their main focus is, they all share the following major components [25, 34]: an external representation – external views or aids assisting the programmer in comprehending the code, a knowledge base – the programmers’ knowledge, a situation/mental model – programmer’s current understanding of the code and which is constantly updated through the assimilation process, and an assimilation process – the process through which the situation model is being updated based on the knowledge base, external representation, and the current situation model. The knowledge base and the situation model are sometimes conflated together under a broader cognitive structures label/category [34].

Given that learners lack (most of or all of) the much needed knowledge base, we face a catch-22 challenge:

In order to read and understand code the reader (learner in our case) needs a knowledge base; however, if the reader is someone who just starts learning programming then the knowledge base is empty or almost empty which means the learner needs to build their knowledge base which can be done by “looking at”, i.e., reading, code examples which means the reader must understand them which is what they are trying to achieve in the first place bringing them back full-circle. In order to transform this vicious circle into a virtuous one, external support in the form of scaffolding offered by a human or a 24/7 computer-tutor is critical, which is our larger goal.

To this end, the DeepCode codeset and the corresponding pedagogical comments were designed to compensate for the lack of a ‘knowledge base’ of students in intro-to-programming courses and offer necessary support when needed to both help students understand target code examples and learn newly introduced programming concepts and techniques.

Of particular importance to our work presented here is the distinction between the program model, the domain model, and the situation model [26]. The program model is some representation of the control-flow of the program or what we call a direct mental equivalent of the code. The domain model captures the function or goals of the program from a target domain perspective, i.e., it describes the domain problem and the solution being implemented by the code by referring mostly to objects and relations and processes and approaches of the domain and of the problem being solved. The situation model in our view captures an integrated view of both the program and domain model with an emphasis on cross-references between the two models, i.e., it contains information which is not being captured by the individual program and domain models. In fact, there is evidence that the best code readers are those who can build such an integrated situation model by seeking to understand and infer cross-references between the program and domain models.

Accordingly, we have focused on pedagogical comments that correspond to the domain model (logical level comments) and program model (logical step implementation details) and situation model (logical step details provide cross-references between the program and domain models).

The logical steps and corresponding comments describe the logical steps of the overall algorithm implemented in the code. Logical steps are meaningful, higher-level steps in the overall solution/algorithm implemented by the code. It is not necessary to describe in detail how the step is being implemented but simply indicate the meaningful purpose/functionality of each such logical code chunk in the context of the overall goal/purpose of the code. A logical step comment should be a concise sentence referring mostly to objects and relations of the domain/problem being solved as shown below (see also Fig. 1 which shows all the logical step comments).

logical_step_2: Generate 25 random numbers in the range of 1 to 75 and populate the Bingo board.

Cross-references/usage of concepts from the ‘program model’, i.e., implementation, should be avoided or kept at a minimum. The `logical_step_details` field links the problem/domain model to the implementation.

Step 4: Logical Step Details. Whereas the logical step is a very high level explanation meant to link the code to the problem/story at a very high level, the `logical_step_details` explanation provides details about how the logical step is being carried out.

The need for the `logical_step_details` is to provide a link between logical details and implementation details while keeping the logical step description high level and short (one short or medium size sentence in plain language - minimal programming language specific lingo, domain knowledge lingo is acceptable, e.g., soccer lingo, but should be kept at a minimum, if at all possible, so that all students can understand the problem and the solution. Additional explanations of domain knowledge concepts should be added if needed).

logical_step_details: Two loops are used to scan all the cells on a Bingo board. The outer loop accounts for the rows and the inner loop for all the cells in one row. For each scanned cell on the Bingo board, a random number is generated and stored in the cell.

Step 5: Statement Level Comments. Statement level comments focus on individual statements and emphasize the elements of the statement relevant to the new concept being taught. It refers more to the ‘program model’, i.e., to concepts, steps, and functions related to implementation. The statement level comment and related question should not necessarily be about the general function of the statement but rather focus on the parts related to the new concept.

stm_comment: Declare an array variable called myNumber of type integer and size 11 and allocate memory for it.

It should be noted that as another way to reduce cognitive load on learners, our guidelines are based on a code comprehension scaffolding strategy that focuses on eliciting explanations at the logical level of code understanding and of statements that refer to the new concept/topic being introduced by each code example - other statements, while important for understanding, are not explicitly explained as they refer to prior concepts which the learners should have mastered previously when those concepts were introduced previously in the sequence of concepts.

Step 6: Adding Hints in the Form of Questions and the Corresponding Answers.

In order to support the development of advanced education technologies such as ITSs that provide micro- and macro-level adaptation through interactive scaffolding, for each logical step comment and logical step details comment, we added a sequence of hints in the form of questions meant to help learners think and articulate about the logical steps and logical step details. That is, the goal is to use those hints to scaffold students’ self-explanations of the logical steps and logical step details and statement level explanations. The first question in the sequence elicits the logical step (domain model) whereas the subsequent questions should prompt learners key aspects of the logical step details (program and integrated models). For each question the corresponding answer is provided as well in order to facilitate the automated assessment of student responses to those

hints, e.g., by comparing the student responses to these ideal responses we provided for each hint using automated semantic similarity methods [33]. Figure 1 does not show the questions for any of the comments for space reasons. We illustrate below the kind of question sequences for logical step and logical step details comments.

```
/**
 * logical_step_2: Generate 25 random numbers in the range of 1 to 75 and populate
 the Bingo board.
 * logical_step_details: Two loops are used to scan all the cells on a Bingo board. The
 outer loop accounts for the rows and the inner loop for all the cells in one row. For each
 scanned cell on the Bingo board, a random number is generated and stored in the cell.
 * question_1: What does the following code block do?
 * answer_1: Generate 25 random numbers in the range of 1 to 75 and populate the
 Bingo board.
 * question_2: How many times does the outer loop execute?
 * answer_2: The outer loop iterates 5 times.
 * question_3: How many times does the inner loop execute?
 * answer_3: The inner loop executes 5 times.
 */
```

Questions and corresponding benchmark answers were generated as well for statement level comments.

```
/**
 * stm_comment: Print element of the Bingo board at position indicated by row i and
 column j.
 * question_1: Which element of the array bingoBoard is being displayed?
 * answer_1: Element of the Bingo board at position indicated by row i and column
 j is being displayed.
 */
```

Step 7: Annotating Misconceptions and Corresponding Remedial Feedback. A key instructional goal for any instruction effort, computer-based or otherwise, is to identify students' misconceptions and provide remedial feedback immediately. For our running code example, a typical misconception is the index of the last row and column of the matrix representing the Bingo board. We can trigger a question to prompt for an answer to discover the presence of the misconception in any or all the lines of code where the matrix is being referred, e.g., immediately after the bingoBoard matrix is declared or when the matrix is being scanned.

Misconception: The index of the last row of the bingoBoard matrix is 5.

Remedial feedback: The index of the last row of the bingoBoard matrix is 4 as indices run from 0 to the number of rows minus 1.

Triggering questions: What is the index of the last row of the bingoBoard matrix?


```

/**
 * SEE METADATA SECTION
 */
import java.util.Random;

public class twoDimensionalArraysBingoBoard {
    public static void main(String[] args) {

        /**
         * logical_step_1: Declare variables needed to represent the Bingo board and
         generate random numbers.
         */
        int[][] bingoBoard = new int[5][5];
        Random rand = new Random();

        /**
         * logical_step_2: Generate 25 random numbers in the range of 1 to 75 and popu-
         late the Bingo board.
         */
        for ( int i = 0 ; i < 5 ; i++ )
        {
            for ( int j = 0 ; j < 5 ; j++ )
            {
                while ( (bingoBoard[i][j] = rand.nextInt (75)) == 0 ) ;
                /**
                 * stm_comment: Print the element of the Bingo board at position indicated by
                 row i and column j.
                 */
                System.out.print( "board square [" + i + " , " + j + "]" + " = " + bingo-
                Board[i][j] + "\n" );
            }
            System.out.println( "" );
        }
        /**
         * logical_step_3: Print the Bingo board.
         */
        for ( int i = 0 ; i < 5 ; i++ )
        {
            for ( int j = 0 ; j < 5 ; j++ )
            {
                /**
                 * stm_comment: Print element of at row i and column j on the Bingo board.
                 */
                System.out.print(bingoBoard[i][j] + " " );
            }
            System.out.println( "" );
        }
    }
}

```

Fig. 1. A working example to illustrate the kind of annotations we added to all 98 of Java code examples covering all CS1 topics.

5 Discussion and Conclusions

The resulting DeepCode instructional codeset consists of 98 annotated Java code examples (at least 4 code examples per topic). More details about the codeset are shown in Table 1 in terms of total lines of code (with and without comments), total number of logical step comments and corresponding logical step details comments, total number of statement level comments, and the number of hints in the form questions.

Table 1. Descriptive statistics about the DeepCode codeset ($n = 98$ annotated Java code examples).

Metric	Total	Average
Total lines of code without comments	1631	16.81
Total lines of code with comments	7157	73.78
Logical steps	260	2.68
Logical step details	260	2.68
Statement level comments	408	4.20
Number of questions for logical steps	590	6.08

The design of the DeepCode instructional codeset was based on strong theoretical foundations which is a unique feature of it. The corresponding annotation guidelines offered as much details for the annotators as possible. Due to space reasons, we have not provided all the guidelines such as the need for the example authors to spellcheck all the comments. Furthermore, it should be noted that the guidelines are just that, guidelines. That is, they are not supposed to and in fact they cannot capture all possible cases that annotators may encounter during their code example creation and annotation.

Additionally, the guidelines leave some concepts vaguely defined, for instance, what exactly constitute a logical step. Nevertheless, we hope that the development and release of DeepCode will foster new developments in terms of additional resources and advanced educational technologies for deep code comprehension and learning of complex programming topics and ultimately help learners become successful computer professionals.

Acknowledgments. This work was supported by the National Science Foundation under award 1822816. All findings and opinions expressed or implied are solely the authors’.

References

1. Aleven, V., Koedinger, K.R.: An effective metacognitive strategy: learning by doing and explaining with a computer-based cognitive tutor. *Cogn. Sci.* **26**(2), 147–179 (2002)
2. Alhassan, R.: The effect of employing self-explanation strategy with worked examples on acquiring computer programming skills. *J. Educ. Pract.* **8**(6), 186–196 (2017)




3. Bielaczyc, K., Pirolli, P.L., Brown, A.L.: Training in self-explanation and self-regulation strategies: investigating the effects of knowledge acquisition activities on problem solving. *Cogn. Instr.* **13**(2), 221–252 (1995)
4. Bloom, B.S.: *All Our Children Learning - A Primer for Parents, Teachers, and Other Educators*. McGraw-Hill, New York (1981). ISBN 9780070061187
5. Boehm, B., Basili, V.R.: Software defect reduction top 10 list. *Computer* **34**(1), 135–137 (2001)
6. Brooks, R.: Towards a theory of the comprehension of computer programs. *Int. J. Man Mach. Stud.* **18**, 543–554 (1983)
7. Brusilovsky, P., Yudelson, M.: From WebEx to NavEx: interactive access to annotated program examples. *Proc. IEEE* **96**(6), 990–999 (2008)
8. Buse, R.P.L., Weimer, W.R.: A metric for software readability. In: *International Symposium on Software Testing and Analysis*, pp. 121–130 (2008)
9. Chen, B., Azad, S., Haldar, R., West, M., Zilles, C.: A validated scoring rubric for explain-in-plain-English questions. In: *The 51st ACM Technical Symposium on Computer Science Education (SIGCSE 2020)*, Portland, OR, USA, 11–14 March 2020. ACM, New York (2020). 7 pages. <https://doi.org/10.1145/3328778.3366879>
10. Chi, M.T.H., Bassok, M., Lewis, M.W., Reimann, P., Glaser, R.: Self-explanations: how students study and use examples in learning to solve problems. *Cogn. Sci.* **13**, 145–182 (1989)
11. Chi, M.T.H., DeLeeuw, N., Chiu, M.-H., LaVancher, C.: Eliciting self-explanations improves understanding. *Cogn. Sci.* **18**(3), 439–477 (1994)
12. Chi, M.T.H.: Self-explaining: the dual processes of generating inference and repairing mental models. In: Glaser, R. (ed.) *Advances in Instructional Psychology: Educational Design and Cognitive Science*, vol. 5, pp. 161–238. Lawrence Erlbaum Associates Publishers (2000)
13. Chi, M.T.H., Wylie, R.: The ICAP framework: linking cognitive engagement to active learning outcomes. *Educ. Psychol.* **49**, 219–243 (2014)
14. Chi, M.T.H., et al.: Translating the ICAP theory of cognitive engagement into practice. *Cogn. Sci.* **42**, 1777–1832 (2018)
15. Conati, C., VanLehn, K.: Further results from the evaluation of an intelligent computer tutor to coach self-explanation. In: Gauthier, G., Frasson, C., VanLehn, K. (eds.) *ITS 2000. LNCS*, vol. 1839, pp. 304–313. Springer, Heidelberg (2000). https://doi.org/10.1007/3-540-45108-0_34
16. Détienne, F.: *Software Design - Cognitive Aspects*. Practitioner Series. Springer, London (2002). <https://doi.org/10.1007/978-1-4471-0111-6>
17. Edwards, S.H., Murali, K.P.: CodeWorkout: short programming exercises with built-in data collection. In: *Proceedings of the 2017 ACM Conference on Innovation and Technology in Computer Science Education (ITiCSE 2017)*, pp. 188–193. Association for Computing Machinery, New York (2017). <https://doi.org/10.1145/3059009.3059055>
18. Graesser, A.C., Singer, M., Trabasso, T.: Constructing inferences during narrative text comprehension. *Psychol. Rev.* **101**, 371–395 (1994)
19. Good, J.: *Programming paradigms, information types and graphical representations: empirical investigations of novice program comprehension*. Ph.D. thesis, University of Edinburgh (1999)
20. Guo, P.J.: Online Python tutor: embeddable web-based program visualization for cs education. In: *Proceedings of the 44th ACM Technical Symposium on Computer Science Education (SIGCSE 2013)*, Denver, Colorado, USA, pp. 579–584. Association for Computing Machinery (2013)
21. Kintsch, W.: Learning from text. *Cogn. Instr.* **3**(2), 87–108 (1986)

22. Kumar, A.N.: Epplets: a tool for solving parsons puzzles. In: Proceedings of the 49th ACM Technical Symposium on Computer Science Education (SIGCSE 2018), pp. 527–532. Association for Computing Machinery, New York (2018). <https://doi.org/10.1145/3159450.3159576>
23. Lane, H.C., VanLehn, K.: A dialogue-based tutoring system for beginning programming. In: Proceedings of the Seventeenth International Florida Artificial Intelligence Research Society Conference (FLAIRS), pp. 449–454. AAAI Press (2004)
24. Lopez, M., Whalley, J., Robbins, P., Lister, R.: Relationships between reading, tracing and writing skills in introductory programming. In: Proceedings of the Fourth International Workshop on Computing Education Research, pp. 101–112. ACM (2008)
25. O’Brien, M.P.: Software comprehension – a review & research direction. Department of Computer Science & Information Systems University of Limerick, Ireland. Technical report (2003)
26. Pennington, N.: 1987. Comprehension strategies in programming. In: Soloway, E., Iyengar, S. (eds.) Empirical Studies of Programmers: Second Workshop, pp. 100–113. Ablex, Norwood (1987)
27. Recker, M.M., Pirolli, P.: A model of self-explanation strategies of instructional text and examples in the acquisition of programming skills (1990)
28. Rezel, E.S.: The effect of training subjects in self-explanation strategies on problem solving success in computer programming (2003)
29. Robins, A., Rountree, J., Rountree, N.: Learning and teaching programming: a review and discussion. *Comput. Sci. Educ.* **13**(2), 137–172 (2003)
30. Roy, M., Chi, M.T.H.: The self-explanation principle in multimedia learning. In: The Cambridge Handbook of Multimedia Learning, pp. 271–286 (2005)
31. Sanders, K., et al.: The Canterbury QuestionBank: building a repository of multiple-choice CS1 and CS2 questions. In: Proceedings of the ITiCSE Working Group Reports Conference on Innovation and Technology in Computer Science Education-Working Group Reports (ITiCSE -WGR 2013), pp. 33–52. Association for Computing Machinery, New York (2013). <https://doi.org/10.1145/2543882.2543885>
32. Rugaber, S.: The use of domain knowledge in program understanding. *Ann. Softw. Eng.* **9**(1–4), 143–192 (2000)
33. Rus, V., Sidney, D., Xiangen, H., Graesser, A.C.: Recent advances in conversational intelligent tutoring systems. *AI Mag.* **34**(3), 42–54 (2013)
34. Schulte, C., Clear, T., Taherkhani, A., Busjahn, T., Paterson, J.: An introduction to program comprehension for computer science educators. In: Proceedings of the Conference on Integrating Technology into Computer Science Education, ITiCSE, pp. 65–86 (2010). <https://doi.org/10.1145/1971681.1971687>
35. Shaft, T.M.: The role of application domain knowledge in computer program comprehension and enhancement. Unpublished Ph.D. thesis, Pennsylvania State University (1992)
36. Sharrock, R., Hamonic, E., Hiron, M., Carlier, S.: CODECAST: an innovative technology to facilitate teaching and learning computer programming in a C language online course. In: Proceedings of the Fourth ACM Conference on Learning @ Scale (L@S 2017), pp. 147–148. Association for Computing Machinery, New York (2017). <https://doi.org/10.1145/3051457.3053970>
37. Shneiderman, B., Mayer, R.: Syntactic/semantic interactions in programmer behaviour. *Int. J. Comput. Inf. Sci.* **8**(3), 219–238 (1979)
38. Sirkkiä, T.: Creating and tailoring program animations for computing education. *J. Softw. Evol. Process* **30**(2) (2018)
39. Spacco, J., Hovemeyer, D., Pugh, W., Emad, F., Hollingsworth, J.K., Padua-Perez, N.: Experiences with marmoset: designing and using an advanced submission and testing system for programming courses. In: ITiCSE 2006, pp. 13–17 (2006)

40. Soloway, E., Spohrer, J.C.: *Studying the Novice Programmer*. Lawrence Erlbaum Associates, Hillsdale (1989)
41. Sweller, J., VanMerriënboer, J.J.G., Paas, F.: Cognitive architecture and instructional design. *Educ. Psychol. Rev.* **10**, 251 (1998). <https://doi.org/10.1023/a:1022193728205>
42. Whalley, J., et al.: An Australasian study of reading and comprehension skills in novice programmers, using the bloom and SOLO taxonomies. In: *Eighth Australasian Computing Education Conference (ACE 2006)*, January 2006
43. Woolf, B.P.: *Building Intelligent Interactive Tutors: Student-Centered Strategies for Revolutionizing E-learning*. Morgan Kaufman Publishers, Burlington (2009)
44. Zwaan, R.A., Radvansky, G.A.: Situation models in language comprehension and memory. *Psychol. Bull.* **123**(2), 162 (1998)
45. VanLehn, K.: The behavior of tutoring systems. *Int. J. Artif. Intell. Educ.* **16**(3), 227–265 (2006)



Cross-Cutting Support of Making and Explaining Decisions in Intelligent Tutoring Systems Using Cognitive Maps of Knowledge Diagnosis

Viktor Uglev¹(✉) , Oleg Sychev² , and Tatiana Gavrilova³ 

¹ Siberian Federal University, Zheleznogorsk, Russia
vauglev@sfu-kras.ru

² Volgograd State Technical University, Volgograd, Russia

³ St. Petersburg University, St. Petersburg, Russia

Abstract. The paper concerns the problem of visual support of decision-making in intelligent tutoring systems and generating natural-language explanations of these decisions. A model of interpreting a learning situation and decision making about further learning on operational, tactical, and strategic levels considered from topic, competency, and learning-goal points of view. We show a case study of learning-situation visualization of analysis and synthesis of explanatory feedback for making operational, tactical, and strategic decisions. The method was evaluated with 3 groups of graduate students; the groups that received simplified Cognitive Maps of Knowledge Diagnosis along with textual explanations demonstrated higher trust in the system's decisions and were more likely to follow the recommendations. We conclude with recommendations for using cognitive maps of knowledge diagnosis for cross-cutting analysis of learning situations.

Keywords: Intelligent tutoring systems · Cognitive visualization · Decision making · Explanation of decisions · Digital learning footprint · Cognitive maps of knowledge diagnosis

1 Introduction and Related Works

Intelligent tutoring systems (ITS) routinely make decisions that should be implemented in collaboration with a human: a learner or a teacher (tutor) [11]. This collaboration mostly uses such approaches as text synthesis, speech synthesis, and cognitive visualization. But the decision logic of an intelligent tutor is complex and, often, hidden from the users. According to the conception of Explainable Artificial Intelligence (XAI) [1,9], we should strive to make the work of intelligent systems “transparent” to humans, so the process of the generalization of the raw data and the logic of inferring conclusions from them should be explained convincingly [3]. In the learning process, in particular, there are

two major axes: on the subject-domain level, each course has its specific parts requiring explanation and interpretation, on the methodical level, managing the learning process in many courses has a lot in common. We think that using the methods and tools of cognitive visualization with the elements of interactivity to support and explain decision making in ITS will increase the impact of the explanations and the level of confidence in the ITS decisions. This paper considers the process of using cognitive visualization to enhance explaining ITS decisions about methodical support of the learning process.

Generating explanations of the system's decisions is necessary when debugging and tuning the ITS reasoner when a learner needs help or feedback (initiates a dialog with the system about its recommendations), and during monitoring of the learning process by a teacher or tutor. The explanations should be logical, context-dependent, individualized, and, last not least, terminologically adapted to the communicating human [15]. Examples of the models that should be explained are the model of "perception" of a learning situation, the decision-making methods, and their results. The learning situation should be interpreted both for subject-domain data and methodical data. Generating explanatory feedback is well known in highly-structured fields of learning.

The decision-making process in ITS implies capturing the changes in the system's state (the event), reflecting it by the system's problem solver (interpretation), execution, and verbalization (if necessary). The events caused by human users and their reactions to them are captured in the digital learning footprint: the logs show who initiated the event (and their role - a learner, a teacher, or a tutor), the relevant system's component, its place in the event sequence and the kind of the event. Thus we get a digital learning footprint coordinated with the structural components (scopes) of the learning environment (see Fig. 1 a):

- nano level is the level of subject-domain learning material and activities, assessing knowledge or competency development for a specific learning unit. For these tasks, the digital footprint contains the data about frequency and time of access to the learning resources, students' successes, failures and their stability. When making pedagogical decisions, these data are, mostly, useful when helping the student to solve the current learning problem and, as a rule, are of little interest for more complex decisions, though they can serve as raw data;
- micro level is the level of learning units and topics, working together to build the logic of presentation of learning material and help acquire the course material. In addition to the frequency and time of access to the course components (i.e., the trajectory of moving among them), the data at this level captures the results of learning in knowledge and competency aspects. These components of learning digital footprint are important for intra-course decisions about individualized learning interventions;
- mezo level is the level of courses in a semester, including inter-course links in the semantic and competency aspects in a single ITS. These digital footprint data are necessary to manage the learning process to achieve statutory and personal learning goals (including visualizing group dynamics);

- macro level is the level of learning programs, academic majors, and personal learning goals. It generalizes the data from the previous levels and allows to work with an image of a graduate as a whole that can be overlaid with data about current learning and other achievements to perform integrated assessment, comparison, selection and goal setting.

Among the important elements of the learning footprint, we should mention the results of summative assignments (doing assignments, passing quizzes, and solving problems in virtual laboratory environments, etc.) that form the data to monitor the learners’ progress. When aggregating these data, they can be represented, for example, as competency development profiles built using the learning results of a single course (Fig. 1 b), or a set of courses in the given semester (Fig. 1 c), according to the method, described in [23]. Each axis represents a competency from the learning plan whose development is measured using certainty factor (ranging from -1 to 1). This allows to use quantitative estimates of competency development for learning units, topics, courses and their sets according to the desired scope of analysis.

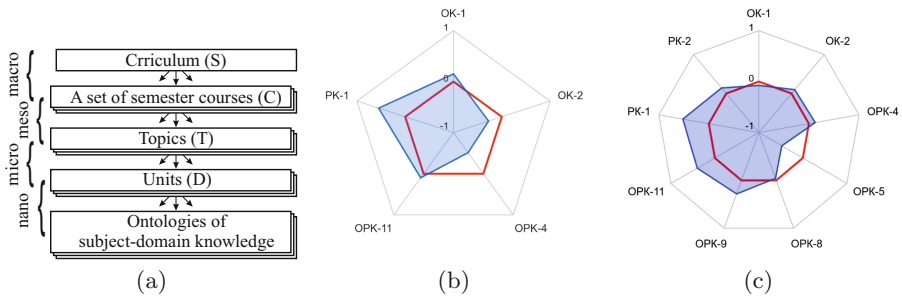


Fig. 1. Hierarchy of learning situation analysis scopes (a), individual competency profile in one course (b), and individual competency profile for a group of courses (c).

Apart from the scope and the point of view (e.g., course structure, competency development, or learning goals), we must also take into account the level of decisions making. We consider 3 levels:

- operational (the decisions are about the local learning situation, e.g. a quiz, assignment, topic, or any course element);
- tactical (the decision are about the goals of the learning module or course);
- strategic (the decisions are about a group of courses, a curriculum, or personal plans of professional development).

To make a parallel with a microscope, the scope is the lens of the eyepiece, the point of view is the color filter, the level of decision making is the light source intensity while the learning situation, represented as the digital learning

footprint, plays the role of a fixated sample. For example, if reacting to the ITS recommendation, the student asked “Why should I study the unit X ?” we need to represent the situation so that the representation will include all the necessary objects of analysis (the learning units and topics in the course), linked by cause-effect and hierarchical links. It should be colored by the current point of view (see the example in Sect. 3) and compared to its importance according to the institutional and personal expectations of the knowledge that should be acquired during the course. In this way, a complex representation of the learning situation corresponds to different levels of decision-making to make a balanced decision. We will call this approach cross-cutting because it is used not only to make and show decisions but also to generate natural-language explanations of these decisions.

The goal of our study was to assess the applicability of cognitive visualization to the cross-cutting analysis of learning situations and, using Cognitive Maps of Knowledge Diagnosis (CMKD) [21], measure the changes in the confidence level of students when cognitive visualization is used.

Automatic synthesis of explanations about the subject-domain material (nano scope) was successfully implemented by many researchers in well-structured subject domains (e.g., when teaching programming [13] or algebra [16]). In poorly-formalized subject domains, the preferred approaches are the attempts to develop unified methods of synthesis of recommendations and explanations (e.g., [12, 19]) and moving to the learning-methods level (e.g., [18, 24]), corresponding to the scope of the verbalized decision (from micro to macro level in Fig. 1 a). On the methodical level of analysis of learning situation, methods of visualization can be used to enhance the perception and trust to the ITS decisions [3]. The following methods are typically used for visualization:

- visualizing a single parameter (e.g., graphs and bar charts, [5, 20]);
- elements of pictographics (from separate indicators to complex images like Chernoff’s faces [7]);
- graphs [18, 25];
- information dashboards [4, 6, 10];
- feature maps ([8, 18], and, partly, [2, 17]).

The review of related works shows the following. First, to the best of our knowledge, there is no unified notation of cognitive visualization for all the levels and scopes of the analysis of learning situations. Secondly, combining textual recommendations and visualization remains a poorly researched ITS feature (see [3]). Thirdly, the changes between the visualization notations when the level or scope are changed are not smooth (the questions “For what?” and “Why this way?” are not answered). We did not find any method of visualization or their combination that would allow cross-cutting support of decision making for the basic decisions made in intelligent tutoring systems [11], including generating human-understandable explanations of the system’s decisions. Cognitive maps are mostly intended to be interpreted by a teacher or a tutor; they are not often supported by explanatory texts, generated by ITS.

2 Method

Our research is based on the cognitive visualization methods called Cognitive Maps of Knowledge Diagnosis (CMKD), proposed by Uglev et al. [21] for cross-cutting decision support and generating explanations in ITS. Parameters of the interpreted learning situation are overlaid on the map corresponding to the selected scopes shown in Fig. 1 a (learning units D , modules (or topics, T), and courses C). It includes the structural description of each level, a priori learning parameters of the course and its adapted (to a particular student) configuration, and different settings (e.g., structural links to skills, competencies, assignments, quizzes, etc.) The semantic links connecting the entities (D , T , and C) will reflect the cause-effect relationships between them. The form, color, and thickness of the lines, fonts, and other signs are used to code different parameters of the learning entities and data from the digital learning footprint in the relevant scope. In particular, the color of the map nodes reflects the estimates of the analysed aspect in the chosen scope. Figure 2 a-c shows CMKDs in competency development aspect for different scopes. For example, the placement of the unit $d5$ in Fig. 2 a shows that in this individualized course it stands in the fourth place ($d4$ is excluded) and can be learned in two different variants; the material in the learning units $d1$ and $d2$ is pre-requisite to learn $d4$; the importance of $d5$ in the course is low (circle form); it is included in the current analysis (from the competency development point of view) and the student's learning results are good (green color). The entity color marks the student's level of mastery of the relevant unit (topic, course): the red color signal problematic entities, green marks successes while white - uncertainty; the gray elements do not affect the analyzed aspects significantly. Map elements (units, topics, or courses) that are not included in the individualized learning trajectory are shown outside of the gray circle; they can be used as additional reference material. A set of CMKDs forms an atlas that can serve as a basis for synthesizing didactic recommendations and their explanations.

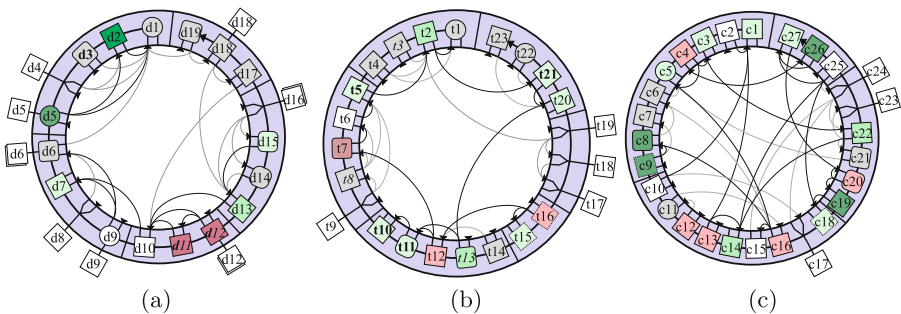


Fig. 2. An example of visualizing learning situations using CMKD on micro (a), meso (b), and macro (c) levels. (Color figure online)

The map is interactive; it allows emphasizing key parameters and shadowing secondary ones. It also can be shown in the minimal configuration (see the example below in Fig. 3 c) to let the viewer concentrate attention on the key aspects of the learning situation. The visual notation used to create CMKD is described in detail in [22]. The process of generalizing digital learning footprint data can be implemented using many different approaches from statistics to expert systems and heavy-weighted ontologies.

We evaluated using the cross-cutting cognitive visualization in the learning process of graduate students of Siberian Federal University (enrolled in the 2020 year year) majoring in “Informatics and computing.” The curriculum contains 23 courses (belonging to the breadth, depth, and elective sections), 90 credits (excluding production practice, internships, and thesis) during 3 semesters. The curriculum determines the conditions of learning for each course (the number of learning hours, the form of the final exam, is the course obligatory, etc.), the sequence of courses, and the requirements for each course (including the developed competencies; 9 of these competencies were monitored during this study). For each course, its topics and their units (lectures, formative and summative assessments) were defined based on the subject-domain knowledge including cause-effect relationships during the typical sequence of topics in the course). Most of the learning was done in the excremental ITS AESU, developed in Siberian Federal University¹.

We studied three groups of students with 8 students in each. Each student, before learning, answered a survey, stating their preferences regarding each course (i.e., the topics and competencies they found interesting and preferred practice tools if the course offered alternatives), and performed summative assignments (we analyzed students’ performance at the beginning, middle, and the end of each semester). The digital learning footprint included the data about the students’ actions within the ITS and their answers during the assessments (about 800 assessments reports; 1 350 answers per student on average). For each student and each learning course, we formed individual learning trajectories taking into account the students’ learning goals (the kinds of learning material included the learning units with high importance to the course, high importance to the student, and the pre-requisite topics necessary to understand the important topics; some of the unnecessary topics were marked as optional). So all the students had the same curriculum but studied different content and performed different assignments.

To verify our hypotheses, we used the task of synthesizing and explaining learning recommendations to the students according to the results of their summative assignments and exams. This decision is made on the micro level (inside a particular course) which requires using the information from mezo (intra-course links) and macro (learning goals) levels. The students from the first two groups (*G1* and *G2*) had the opportunity to ask the ITS for explanation both in the form of text messages and visualization of simplified CMKDs. The students of

¹ <https://aesu.ru/>.

the third (control) group *G3* did not see the visualization; they were shown only the explanation text.

We evaluated the students' level of trust in the ITS's recommendations that were generated using the cross-cutting analysis with CMKDs we used the following process:

1. after each summative assignment the system generated a number of Cognitive Maps of Knowledge Diagnosis with the learning footprint data about the current learning situation (it allows concentrating the required raw data for making further decisions);
2. the ITS solver formed hypotheses using the concentrated data from the set of CMKDs (to narrow the set of possible situations for further analysis);
3. the recommendation text was synthesized and the student could select one of the provided lines of reasoning to see the extended information, explaining the recommendation (which lets capture the initial decisions of the system and allows the student to choose which kind of explanations they want to see);
4. the intelligent core of the ITS verified the cognitive maps of the current decision-making level and its neighboring levels to find the links that were significant for explaining the recommendations by using different points of view (which allows selecting the most important aspect, e.g., knowledge, competency, or goal aspect);
5. among the significant links, the ITS solver chose the level of providing arguments (operational, tactical, or strategic) that was expected to affect the student better (to determine the arguments which, likely, will be accepted by the student as a guide to motivated action);
6. the explanation text was synthesized using the arguments which are rated as personally significant for the student placed first (which allows verbalising the explanation and relevant arguments);
7. for the students from the test groups, a simplified CMKD from the most significant point of view was generated and shown (to make the arguments more appealing using visual representation channels);
8. if the information for the generated explanation had been generalized, the student was given an option to ask the system for clarification, getting detailed arguments up to reaching the raw data (student's answers and actions in the ITS during performing the assignment).

For all the students we gathered anonymized reports with the generated dialogue, the number of times the student asked for clarification of ITS decisions, and the subsequent usage of online learning material. These data were analyzed to determine the differences between the behavior of the students in test and control groups; we also surveyed the students about their levels of trust with ITS explanations.

3 Case Study

Let's consider the process of cross-cutting generalization and visualization of these data on the example of one of the students.

Stage 1. Each summative assignment resulted in grading the students’ attempts and estimating their competency development levels. For the course “Simulation modeling”, the student selected for this example on 02 November 2021 had an individual competency profile shown in Fig. 1 b (micro level); his profile regarding all courses in the semester is shown in Fig. 1 c (meso level). CMKD for the course “Simulation modeling” is shown in Fig. 2 a; it is overlaid with the data about the level of development of competency OPK-4 “Search, generalize, concentrate and structure the information; select the important information; make conclusions” (the source data are shown in Fig. 1 b; they were processed according to the method described in [23]) and were coded by color. Similarly, Fig. 2 b shows a CMKD for mezo level demonstrating cross-course links between the topics of different courses of the third (currently analyzed) semester regarding the development level of the competency OPK-4. The CMKD for macro level (the entire curriculum for the major “Informatics and computing”) shown in Fig. 1 c demonstrate the generalization of the data from the profile of development of competency OPK-4 to the set of all courses taking into account the semantic links between them.

Changing the point of view from competency development (Fig. 2 a) to course mastery (Fig. 3 a) shows on micro level the distribution of the level of success in mastering the course material by the analyzed student. The CMKD from the learning-goals point of view is presented in Fig. 3 b. So switching between scopes and points of view allows cross-cutting visualization of data from the digital learning footprint when it is necessary to analyze the learning situation.

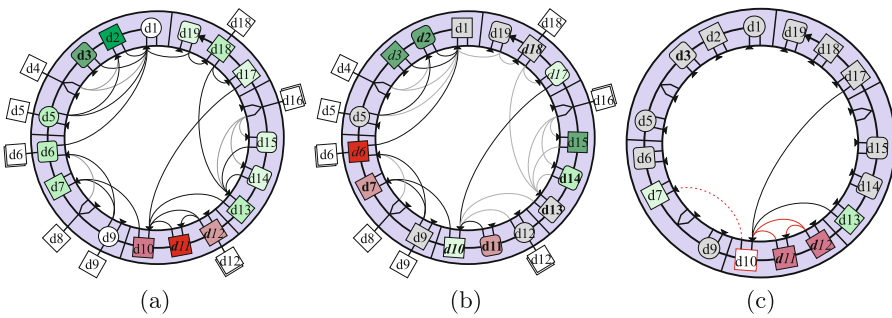


Fig. 3. An example of CMKD showing a learning situation on a micro level from different points of view: course mastery (a), learning goals (b) competency (a simplified map) (c).

Stages 2–4. A student performed the middle-semester summative assignment in the course “Simulation modeling” and, after seeing their final grade and the ITS recommendations, they initiated a dialog with the system by asking “Why I am recommended to study the topic ‘Generating a sequence of pseudo-random numbers using Monte Carlo method’ ?” (this topic is shown as the unit d10 in Fig. 2 a and Fig. 3 a, b.) The course a-priori characteristics show that the

unit $d10$ is one of the core topics of the course (it is visualized as a square) and is associated with competencies $OPK - 4$ and $PK - 2$ by the teacher. From the pre-course survey of this student, the system knows that they marked the competency $OPK - 4$ as their priority and did not choose $d10$ as a topic of interest.

Stage 5. Analyzing the data from the digital learning footprint, the ITS solver (or planner) module can generalize data and group them for later analysis. It can automatically infer the following:

- “the actual scope of the situation analysis is micro level because the dialog is initiated by the student as a reaction on recommendations to study a particular unit ($d10$) in the course”;
- “unit $d10$ is important in this course because it is one of the course’s core elements and it is linked to the following units extensively” (see the square form and outgoing links in Fig. 2 a: course structure point of view, tactical level);
- “unit $d10$ is not mastered by the student which affects studying the following units negatively” (units $d11$ and $d12$ linked with unit $d10$ have low level of mastery and the mastery of unit $d10$ is low too which is shown by their red color, even though the initial unit $d7$ is mastered well enough; see Fig. 3 a, the operative level for the course structure and competency development points of view)
- “the student is not aware of the importance of unit 10 for them but marked the competency $d10$ linked to it as a priority” (see the coloring in Fig. 2 a and Fig. 3 b and the semantic links on these maps for the tactical level from the target point of view);
- “the student’s digital learning footprint shows that they ignore the ITS recommendations in 27% of all cases and 64% of situations with similar circumstances” (according to the data from digital learning footprint for the last semester - strategic level for micro and meso scopes).

Stage 6. Based on this information, the ITS should synthesize the text that will not just explain the operational usefulness of the system’s recommendations but also increase the motivation to follow it (according to the reflexive theory of V. Lefebvre [14]) on the tactical level. The actual generated explanatory text was as following:

Unit $d10$ has high importance for this course and significantly affects the impact of learning its dependent units $d11$ and $d12$. As you noted that you want to develop competency $OPK - 4$ during this course and unit $d10$ develops the said competency, unit $d10$ is of critical importance and requires enhanced monitoring of its mastery.

This explanation mixes all the points of view: course mastery, learning goals, and competency development. It can be enhanced by generating additional text detailing the reasons behind the three key arguments (emphasized by the underlined text).

Stages 7–8. If the student selects any of the underlined arguments, the additional information (e.g., a button, hypertext or a parametric query), with a more detailed explanation, can be shown (similarly to backward-chaining reasoning) based on the same CMKDs or CMKDs of the smaller scope. This explanation should be accompanied by a visualization of the key argument as a simplified CMKD. Figure 3 c shows an example of the simplified visualization for the argument “affects the impact” from the explanatory text above; the map is a simplified map shown in Fig. 2 a.

It should be noted that for decisions that are performed automatically, each stage additionally uses the reasoning methods of rule-base expert systems.

4 Results and Discussion

Table 1 shows the results of the students from two test groups ($G1$ and $G2$) and the control group $G3$. It allows us to make two conclusions:

- the graphical support of decision making by CMKDs motivated students to ask for clarification more often (see column 3);
- the students who received ITS explanations supplied with simplified cognitive maps followed the ITS recommendations more often than those who received only textual explanations (see column 4).

Table 1. The experimental data aggregated by the student groups

Group	Student who asked for explanations of ITS decisions	Recommendations that led to asking for explanations	Following ITS recommendations
$G1$ (test)	87.5%	68.1%	84.3%
$G2$ (test)	100%	59.4%	81.2%
$G3$ (control)	87.5%	27.9%	63.3%

The Pearson’s correlation coefficient between the results of the groups $G1$ and $G2$ is 0.892 which shows a high degree of coherence between the results of the students from these groups. All the students from the experimental groups rated the feature of receiving cross-cut explanations about the ITS decisions as positive: given the statement “if students know why they receive a recommendation, it will increase their trust in the system and the likelihood to follow the system’s recommendation,” 16% of students chose “partially agree” and 84% “fully agree” on the Likert scale. These results strengthen the hypothesis that “if students know why they receive a particular recommendation, it could increase their trust in the system along with the likelihood of them following feedback provided by the system” [3].

These data were gathered during two years of study of graduate students of Siberian Federal University majoring in “Informatics and computing”; the learning process was partially individualized by changing the learning-unit composition using the experimental ITS AESU. The validity of the obtained results about the efficiency of using the cross-cut approach to decision making and explaining with CMKD in ITS can be increased by widening the numbers of participating students and the learning programs they are enrolled in. We are planning to obtain more data, by including graduate students majoring in “System analysis and management” and “Design and technology support of machine-building facilities”.

The chief findings of our study is that cross-cutting cognitive visualization supporting textual explanations of ITS decisions allowed to increase the level of trust in the ITS’s decisions. The graduate students who saw simplified CMKDs built on different scopes from different points of view followed the system’s recommendations more often and asked for more explanations.

Using these data, we can conclude the following: the system of transitions between the scopes of analysis of the learning situation and aspects of analysis does not have pre-defined “entry points” during a dialog with a student. The continuity of levels (Fig. 4 a) allows the system to initiate dialog in a wide range of learning situations and direct the explanation process either to show more details or to higher levels of abstractions as necessary. For example, considering only three previously mentioned aspects (knowledge, competency, and goal), we can get the system of transitions between CMKDs shown in Fig. 4 b. In the combined form, the trajectories of transitions can be represented in a scheme shown in Fig. 4 c. It should be noted that the intellectual core of ITS will shift the analysis focus according to the student’s wishes using the typical algorithm. Furthermore, reflection features (i.e., interpreting the situation in the context of the learner, teacher, or tutor model) will be also implemented using cognitive maps.

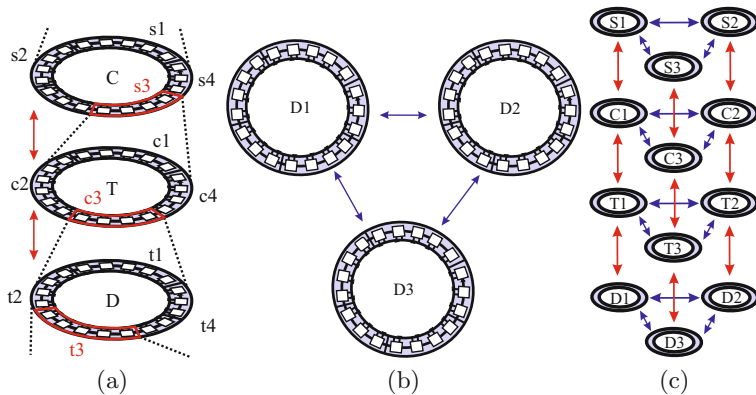


Fig. 4. The diagram of cross-cutting transitions between variants of visualization of CMKD in different scopes (a), analysis aspects (b) and the combined form (c.)

The general recommendations on using CMKDs for analysis and visualization of a learning situation are as follows:

- the process of the formalization of data about each course includes obtaining data about semantic and cause-effect links between the micro-level entities (the learning units) and competency lists, exercises for the course assessments, and the used tools (methods and software);
- the quality of decision making when using CMKDs depends directly on the quality of the pre-course survey data which requires developing standards for the survey structure and its processing;
- the individualized learning process shows a significant difference in the structure of cognitive maps even for the students of one learning group which makes it difficult to generalize the learning results by the group or scientific major;
- the analysis of the learning situation should, if possible, use not only structural but also functional visualization.

5 Conclusion

The methods of methodical support of decision making by the software reasoner in ITS should be not only unified (i.e., applicable to different learning processes) but also inspire trust in the learner. The possibilities of cross-cut generalization of the data from digital learning footprint for different levels and scopes of decision making using Cognitive Maps of Knowledge Diagnosis, given in this paper, shows the directions of enhancing the abilities of ITS. The two student groups that used simplified CMKDs had higher trust in the learning recommendations of the system and used the advice from ITS a lot more often. This makes us think that further development of this method will allow enhancing the learning process given the current trend of increasing the role of online learning.

Using CMKDs to generate explanations of the ITS decisions brings us closer to implementing XAI for ITS on the methodical level. It also reminds us about the importance of the legal regulation of the processes of accumulating and migrating data about the students' digital learning footprint, their protection, and the responsibility of teachers, institutions, and the developers of intelligent tutoring systems for the recommendations the system shows to the users.

References

1. Arrieta, A.B., et al.: Explainable artificial intelligence (XAI): concepts, taxonomies, opportunities and challenges toward responsible AI. *Inf. Fusion* **58**, 82–115 (2020)
2. Bargel, B.A., Schröck, J., Szentes, D., Roller, W.: Using learning maps for visualization of adaptive learning path components. *Int. J. Comput. Inf. Syst. Ind. Manag. Appl.* **4**(1), 228–235 (2012)
3. Bodily, R., Verbert, K.: Trends and issues in student-facing learning analytics reporting systems research. In: *Proceedings of the Seventh International Learning Analytics & Knowledge Conference*, pp. 309–318. LAK 2017, Association for Computing Machinery, New York (2017). <https://doi.org/10.1145/3027385.3027403>

4. Brusilovsky, P., Rus, V.: Social navigation for self-improving intelligent educational systems, pp. 131–145. Army Research Laboratory, December 2019. <https://www.pitt.edu/~peterb/papers/SocNav4SIS.pdf>
5. Davis, D., Jivet, I., Kizilcec, R.F., Chen, G., Hauff, C., Houben, G.J.: Follow the successful crowd: raising MOOC completion rates through social comparison at scale. In: Proceedings of the Seventh International Learning Analytics & Knowledge Conference. p. 454–463. LAK 2017, Association for Computing Machinery, New York (2017). <https://doi.org/10.1145/3027385.3027411>
6. Fleur, D.S., van den Bos, W., Bredeweg, B.: Learning analytics dashboard for motivation and performance. In: Kumar, V., Troussas, C. (eds.) ITS 2020. LNCS, vol. 12149, pp. 411–419. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-49663-0_51
7. France, L., Heraud, J.M., Marty, J.C., Carron, T., Heili, J.: Monitoring virtual classroom: visualization techniques to observe student activities in an e-learning system. In: Sixth IEEE International Conference on Advanced Learning Technologies (ICALT 2006), pp. 716–720. IEEE (2006)
8. Grann, J., Bushway, D.: Competency map: visualizing student learning to promote student success. In: Proceedings of the Fourth International Conference on Learning Analytics and Knowledge, pp. 168–172 (2014)
9. Gunning, D., Aha, D.: DARPA’s explainable artificial intelligence (XAI) program. *AI Mag.* **40**(2), 44–58 (2019)
10. Jivet, I., Scheffel, M., Specht, M., Drachsler, H.: License to evaluate: preparing learning analytics dashboards for educational practice. In: Proceedings of the 8th International Conference on Learning Analytics and Knowledge, pp. 31–40 (2018). <https://doi.org/10.1145/3170358.3170421>
11. Karpenko, A., Dobryakov, A.: Model for automated training systems. overview, science and education. *Sci. Educ.* **7**, 1–63 (2011). <https://doi.org/10.7463/0715.0193116>. (in Russian)
12. Khanal, S.S., Prasad, P.W.C., Alsadoon, A., Maag, A.: A systematic review: machine learning based recommendation systems for e-learning. *Educ. Inf. Technol.* **25**(4), 2635–2664 (2019). <https://doi.org/10.1007/s10639-019-10063-9>
13. Kumar, A.N.: Generation of problems, answers, grade, and feedback—case study of a fully Automated Tutor. *J. Educ. Resour. Comput.* **5**(3), 3-es (2005). <https://doi.org/10.1145/1163405.1163408>
14. Lefebvre, V.: Lectures’ about the theory of Reflexive Games. Cogito-Tsentr, Moscow (2009)
15. Mashbitz, E., Andrievskays, V., Komissarova, E.: Dialog in a Tutoring System. Kiev, Higher school (1989). (in Russian)
16. O’Rourke, E., Butler, E., Díaz Tolentino, A., Popović, Z.: Automatic generation of problems and explanations for an intelligent algebra tutor. In: Isotani, S., Millán, E., Ogan, A., Hastings, P., McLaren, B., Luckin, R. (eds.) AIED 2019. LNCS (LNAI), vol. 11625, pp. 383–395. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-23204-7_32
17. Pirrone, R., Cannella, V., Russo, G.: A map-based visualization tool to support tutors in e-learning 2.0. In: 2009 2nd Conference on Human System Interactions, pp. 485–490. IEEE (2009)
18. Takada, S., et al.: Toward the visual understanding of computing curricula. *Educ. Inf. Technol.* **25**(5), 4231–4270 (2020). <https://doi.org/10.1007/s10639-020-10127-1>

19. Tarus, J.K., Niu, Z., Mustafa, G.: Knowledge-based recommendation: a review of ontology-based recommender systems for e-learning. *Artif. Intell. Rev.* **50**(1), 21–48 (2017). <https://doi.org/10.1007/s10462-017-9539-5>
20. Tuah, N.M., Yoag, A., Nizam, D., Mohd, N., Chin, C.W.: A dashboard-based system to manage and monitor the progression of undergraduate it degree final year projects. *Pertanika J. Sci. Technol.* **30**(1) (2022)
21. Uglev, V., Zakharyin, K., Baryshev, R.: Cognitive maps of knowledge diagnosis as an element of a digital educational footprint and a copyright object. In: Silhavy, R., Silhavy, P., Prokopova, Z. (eds.) *CoMeSySo 2020. AISC*, vol. 1295, pp. 349–357. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-63319-6_31
22. Uglev, V., Sychev, O.: Creating and visualising cognitive maps of knowledge diagnosis during the processing of learning digital footprint. In: Cristea, A.I., Troussas, C. (eds.) *ITS 2021. LNCS*, vol. 12677, pp. 93–98. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-80421-3_11
23. Uglev, V.A., Ustinov, V.A.: The new competencies development level expertise method within intelligent automated educational systems. In: *Trends in Practical Applications of Heterogeneous Multi-agent Systems. The PAAMS Collection*, pp. 157–164 (2014)
24. Zakharova, I., Avriskin, M.: Student’s digital footprint: from data to predictions and recommendations. In: *Education Informatization and the Methods of Online Learning: Digital Technologies in Education: Proceedings of V International Conference*, pp. 120–124 (2021)
25. Zhang, S., Luo, X., Xuan, J., Chen, X., Xu, W.: Discovering small-world in association link networks for association learning. *World Wide Web* **17**(2), 229–254 (2012). <https://doi.org/10.1007/s11280-012-0171-7>



Covering Possible Reasoning Errors for Intelligent Tutoring Systems: Order of Expression Evaluation Case

Yaroslav Kamennov , Oleg Sychev  , and Yulia Orlova 

Volgograd State Technical University, Lenin Ave, 28, Volgograd 400005, Russia
o_sychev@vstu.ru

Abstract. One of the problems in computer-supported education is defining meaningful errors that students can make during a complex automated exercise. A precise set, describing what can go wrong in the student's reasoning, allows better measurement of students' knowledge, asking pointed follow-up questions to stimulate the student's thinking, and providing precise explanatory feedback. We describe a method for building a set of possible low-level errors in reasoning during solving a complex task. We demonstrate an example of using this method for the task of determining the order of evaluation of a programming language expression and discuss prospects of applying the described method.

Keywords: Intelligent tutoring systems · Error classification

1 Introduction and Related Work

When developing Intelligent Tutoring Systems (ITS) using the Mastery Learning approach, we should classify not only the knowledge that students should acquire but also possible errors. According to [10], an expert system is called ITS if it “simulates the individual learning process of a student and teacher by improving understanding”. This means not only increasing the knowledge about the subject domain but also reducing erroneous judgments. So understanding the reasons for students' mistakes is necessary both for assessing their knowledge and providing explanatory feedback to improve their mastery of the topic.

Shute and Zapata-Rivera also discuss the importance of finding and classifying student errors [11]. The most progressive approach they describe is using assessment as feedback not only to the student but to the entire system. They conclude that the ability to determine student errors is one of the most important features of ITS. Classifying students' answers accurately is also discussed in [1]. The paper indicates that determining the context and student's intention is important for the correct formation of the training program.

The reported study was funded by RFBR, project numbers 20-07-00764 and 20-07-00502.

Brusilovsky [2] provides recommendations for building formal models of subject domains based on the studied concepts. His idea of intelligent textbooks, based on subject-domain concepts [15], allows tracking students' knowledge only on the level of mastering the entire concept or topic which is sufficient to calculate overall grade but is not enough to choose the next learning task inside a topic because it does not contain the detailed information about the kinds of errors the student can make while trying to apply the given concept.

To build ITS that can determine and take into account the exact cause of the student's error, a systematic, fine-grained set of possible errors should be developed for the target domain. Unfortunately, many articles about ITS do not elaborate on methods that were used to identify possible errors (e.g., [7,9]).

We aim to close this gap in the literature by describing a method to systematically build sets of possible errors in students' reasoning, based on a formal description of the correct lines of reasoning for solving the task. We provide an example of applying this method to determining the order of evaluation of an expression and demonstrate the resulting set of low-level reasoning errors.

Most of modern ITS classify students' errors and provide feedback on them. For example, J-LATTE [3] provides three types of feedback: "simple feedback" (if the answer is correct), "hint" (points to the first error), and "all errors".

Prolog Tutor [4] calculates the student's errors by comparing the syntax of the reference code with the student's code. The article [16], describes a system that reduces code to an internal representation that represents the used concepts. The concepts that are not present in the reference template for the task are considered extraneous or missing. This approach allows to learn in detail about specific errors, but makes it difficult to find their causes.

CLARA engine [14] represents the problem solution as a graph. The student's solution is reduced to the correct form by the smallest number of corrections which are shown as hints. The same approach is used for token sequences [12] to teach constructing sentences and lines of code. Expression tutor from the Problems family [6] determines errors in the expression construction and evaluation. The set of errors is big, but the method used to build it is not described.

The work [5] presents a methodology for teaching algorithms by using flow charts. The system has a task trace panel, displaying errors in the diagram, e.g., "You can't declare node here." Martin and Mitrovich [8] present a constraint-based tutor for SQL programming language and an ontology model for logical expressions. It starts from showing only answer's correctness; then the student receives more hints until the correct solution is achieved or the student gives up.

2 Method of Identifying Possible Errors

The input data for building a set of possible errors when solving an intellectual task is the description of the correct method of reasoning to solve the task. Lines of reasoning are built using several kinds of reasoning steps: (a) **Questions**. Possible answers lead to either conclusions or the next step. (b) **Actions**. E.g., finding an object satisfying certain conditions or calculating something. It leads

to the next step and should be accompanied by a list of possible errors. (c) **Branching**. Splits the line of thought, creating several child lines. Their conclusions can be aggregated (e.g., by logical operators **or** or **and**); sometimes only one of line can result in a conclusion. (d) **Iterations**. Finding several objects satisfying certain criteria and performing all the following steps for each of them. The conclusions from iterated objects are aggregated.

Participating objects must be given identifiers, while their properties and relations between them must be named according to the subject-domain ontology. This makes lines of reasoning defined formally; they can be implemented using software reasoners or coded in imperative programming languages. Some of the objects used in the reasoning process are given as the reasoning input; others are determined during “find” **action** and **iteration** steps.

We identified several types of atomic reasoning errors: (1) choosing a wrong first step; (2) answering a **question** or performing an **action** wrongly, or choosing a wrong object during **iteration**; (3) drawing wrong conclusions from correct answers (i.e., a wrong choice of the next step after a **question** or **action**); (4) missing a branch during **branching** or missing an object during **iteration**; (5) wrong aggregation of conclusions after **branching** or **iteration**. Theoretically, the sixth type of error is asking a wrong **question** (performing a wrong **action**), but modeling all possible wrong questions gives a too big search space to use it practically. The errors of types 1 and 3 deal with the structural reasoning problems (i.e., performing the steps in the wrong order) while the errors of types 2 and 4 deal with problems in performing specific steps.

The method of building the set of possible errors goes as follows.

1. For the starting point and each **branching** create type 1 errors for each step (except the first) using only objects already known. Error formulation: “The learner starts doing X from doing Y while it should be started by Z.”
2. For each **question** step create type 2 errors for every possible pair of unequal answers (choices). “The answer to X is Y, but the learner thinks it’s Z.”
3. For each **action** and **iteration** step create type 2 errors, defined by the possible errors listed in the step. The formulation depends on the step errors.
4. For each exit of each step (i.e., an answer to a question, a branch, etc.), create type 3 errors for each step down the line of reasoning and each conclusion that makes sense, i.e. uses only objects known at that point (except the correct next step for this exit). “The learner thinks that as X, Y” (where X is an answer to the question, Y is a step).
5. For each branch of each **branching** step, create a type 4 error that this branch is missing. Error formulation: “The learner doesn’t check X.”
6. For each **iteration** step create a type 4 error for missing an object. These errors are formulated like “The learner misses X while searching for Y.”
7. For each **branching** and **iteration** with aggregation create a type 5 error for each wrong kind of aggregation (e.g., logical **or** instead of **and**). “The learner aggregated conclusions using X operator while they should use Y).”

The most numerous errors belong to types 2 and 3. The number of type 2 errors for a **question** step is $N_{ans} * (N_{ans} - 1)$ where N_{ans} is the number

of possible answers to the question. For an **action** step, the number of type 2 errors is identified when defining the step. The number of type 3 errors is smaller than it could be because some steps (**actions** finding objects and **iterations**) introduce new objects, and it's not possible to ask a **question** (perform an **action**) using an object that wasn't found yet.

3 Case Study: Order of Expression Evaluation

Let us consider a learning task of determining the order of evaluation of a programming-language expression. This task is frequently used by intelligent tutors for introductory programming: e.g., it is a subtask in Problets [6] and the Expression domain in CompPrehension [13]. To solve it, the learner should repeatedly select the next operator to evaluate. Basic rules of operator precedence and associativity give only a few high-level errors, identifiable by the learner's answer: "an unevaluated operator standing ..." (1) to the left has higher precedence; (2) to the right has higher precedence; (3) to the left has the same precedence and left associativity; (4) to the right has the same precedence and right associativity; (5) between the current operator's tokens. More complex errors caused by the strict order of operand evaluation in some operators are out of the scope of the example in this paper.

To determine if operator *X* can be evaluated, the learner must follow 3 lines of reasoning, verifying if the inner, left and rights operands are fully evaluated (if they exist) The conclusions are aggregated using the logical **and** operator. We will use the second line of reasoning, concerning the left operand, to show the results of error generation. It goes as follows:

1. Does *X* need a left operand? If no, the conclusion is **true**.
2. Find *A*, *X*'s left closest unused operand. Possible errors: *A* is to the right of *X*, *A* is unevaluated, *A* is already used, *A* is too far from *X*.
3. Find *Y*, *A*'s left closest unevaluated operator. *Y* is *X*'s competitor for operand *A*. If not found, the conclusion is **true**. Possible errors: *Y* is to the right of *A*, *Y* is already evaluated, *Y* is too far from *A*.
4. Checking effects of parentheses. These three conditions can be checked in any order as no two of them can give conclusions simultaneously.
 - (a) Is there parenthesis enclosing *Y* but not enclosing *X*? If yes, the conclusion is **false**.
 - (b) Is there parenthesis enclosing *X* but not enclosing *Y*? If yes, the conclusion is **true**.
 - (c) If *Y* is a two-token operator, does *X* stand between its tokens? If yes, the conclusion is **true**.
5. Compare precedence of *X* and *Y*. If *X*'s precedence is higher, the conclusion is **true**. If *Y*'s precedence is higher, the conclusion is **false**.
6. What is the associativity of *X* and *Y* (it must be the same as they have equal precedence)? If *X* and *Y* are left-associative operators, the conclusion is **false**. If *X* and *Y* are right-associative operators, the conclusion is **true**.

Table 1. Does the left operand block the evaluation of X? Possible low-level errors, part 1.

N	Error description	Error type
11	The learner attempts to find the left operand for X without verifying if the left operand is needed	1
12	X does not need a left operand, but the learner thinks it needs	2
13	X needs a left operand, but the learner thinks it does not	2
14	The learner thinks that as X needs the left operand, X cannot be evaluated	3
15	The learner thinks that as X needs the left operand, X can be evaluated	3
16	The learner thinks that as X does not need the left operand, X cannot be evaluated	3
17	The learner chose A (possible left operand of X) to the right of X	2
18	The learner chose A too far from X	2
19	The learner chose a used operand as A	2
110	The learner chose an unevaluated operator as A	2
111	The learner did not find any unevaluated operator to the left of A, but there is one	2
112	The learner thinks that as there is no unevaluated operator to the left of A, X cannot be evaluated	3
113	The learner thinks there is an unevaluated operator to the left of A, but there is none	2
114	The learner thinks that as there is an unevaluated operator to the left of A, X cannot be evaluated	3
115	The learner thinks that as there is an unevaluated operator to the left of A, X can be evaluated	3
116	The learner chose a too far operator as the operator competing for A (Y)	2
117	The learner chose a token to the right of A as Y	2
118	The learner chose an evaluated operator as Y	2
119	Finding the left competing operator (Y), the learner analyzes the associativity of X and Y	3
120	Finding the left competing operator (Y), the learner compares precedence of X and Y	3
121	The learner did not check if X is placed between Y's tokens	4
122	Y has 2 tokens, but the learner thinks it has 1	2
123	Y has 1 tokens, but the learner thinks it has 2	2
124	The learner thinks that as Y has 2 tokens, X can be evaluated	3
125	The learner thinks that as Y has 2 tokens, X cannot be evaluated	3
126	The learner chose a token with the wrong type as the second token of Y (Y2)	2
127	The learner chose a too close token as Y2	2
128	The learner chose a too far token as Y2	2
129	The learner thinks that as X is between Y1 and Y2, X cannot be evaluated	3
130	The learner thinks that as X is not between Y1 and Y2, X can be evaluated	3
131	The learner thinks that as X is not between Y1 and Y2, X cannot be evaluated	3

Table 2. Does the left operand block the evaluation of X? Possible low-level errors, part 2.

N	Error description	Error type
l32	The learner thinks that as X is enclosed in parentheses of which Y is not, X cannot be evaluated	3
l33	The learner takes into account parentheses that enclose both X or Y	2
l34	The learner takes into account parentheses that do not enclose both X or Y	2
l35	The learner did not check if X should be evaluated before Y because of parentheses	4
l36	The learner did not check if Y should be evaluated before X because of parentheses	4
l37	The learner thinks that as Y is enclosed in parentheses of which X is not, X can be evaluated	3
l38	After checking for parenthesis, the learner analyzes the associativity of X and Y	3
l39	X has a higher precedence than Y, but the learner thinks they have equal precedence	2
l40	X has a higher precedence than Y, but the learner thinks X has a lower precedence than Y	2
l41	X has a lower precedence than Y, but the learner thinks they have equal precedence	2
l42	X has a lower precedence than Y, but the learner thinks X has a higher precedence than Y	2
l43	Precedence of X equals precedence of Y, but the learner thinks X has a higher precedence than Y	2
l44	Precedence of X equals precedence of Y, but the learner thinks X has a lower precedence than Y	2
l45	The learner thinks that as X has a higher precedence than Y, X cannot be evaluated	3
l46	The learner thinks that as X has a lower precedence than Y, X can be evaluated	3
l47	The learner thinks that as X has a higher precedence than Y, associativity must be checked	3
l48	The learner thinks that as X has a lower precedence than Y, associativity must be checked	3
l49	The learner thinks that X and Y are left associative, but they are right associative	2
l50	The learner thinks that as X and Y are left associative, X can be evaluated	3
l51	The learner thinks that X and Y are right associative, but they are left associative	2
l52	The learner thinks that as X and Y are right associative, X cannot be evaluated	3

Applying the method described in Sect. 2 to this line of reasoning, we built the error sets shown in Tables 1 and 2. The errors concerning missing lines of reasoning and wrong aggregation are shown in Table 3. Some of type 2 errors (erroneous answers to questions) depend on the class of the considered operator(s), so in practice, there will be separate instances of these errors for some operators. E.g., the error 12 will have an instance for every operator that does not need a left operand (e.g., prefix unary operators.) The errors like 139 depend on two operators (X and Y) so an instance of this error should be created for each pair of operators where X has higher precedence than Y.

Table 3. Missing branches and aggregations errors for determining the order of evaluation

N	Error description	Error type
g1	The learner does not check that if X's inner operand exists and is fully evaluated	4
g2	The learner does not check that if X's left operand exists and is fully evaluated	4
g3	The learner does not check that if X's right operand exists and is fully evaluated	4
g4	The learner concluded that X cannot be evaluated in one of the branches, but thinks that X can be evaluated	5
g5	The learner concluded that X can be evaluated in all the branches, but thinks that X cannot be evaluated	5

In practice, the number of possible low-level reasoning errors is smaller than the described set because some errors are not applicable to all situations. Many of them are mutually exclusive (e.g., 12 and 13). Consider the expression $a + b \& c$ in the C programming language; a student makes an error by selecting the $\&$ (X) operator as the first. The competing operator Y here is the $+$ operator. The wrong conclusion is done in the line of reasoning containing 52 errors (see Tables 1 and 2), but only 16 of them (and two general errors g2 and g4) are applicable: 11, 13, 15, 17, 18, 111, 115, 119, 120, 121, 123, 138, 141, 142, 146, and 148. Other errors are not applicable because X and Y have 1 token each, there are no parentheses in the expression, there are no operators to the left of X, there is only one possible candidate for Y, the binary $\&$ operator has lower precedence than the $+$ operator, etc. These 16 low-level errors are indistinguishable by the student's answer to the high-level task; the ITS can either update the information about the possible errors using pre-determined probabilities of making different kinds of low-level errors or ask a series of follow-up questions to determine the particular reasoning error the student made.

Once an error is identified, ITS can generate a specific explanatory message for it. E.g., for the error 138 (type 3) the message is "Before checking operators' associativity you must compare their precedence," while for the error 141 (type

2) it is “The addition operator has lower precedence than the multiplication operator.” It can help fix misconceptions that the student has.

4 Evaluation

To evaluate the resulting error set, we showed it to 4 experts who taught expressions as a part of introductory programming courses. They were asked two questions on the Liker scale. Answering the question “Does the error set cover all the possible errors that a student can make while determining the order of evaluation of an expression?” two experts said that it covers all the possible errors; two others said most of the possible errors, but they could not name any uncovered error. Answering the question about the level of detail, two experts selected “sufficient level of detail” and two choose “A bit more detailed than necessary.” In free-text answers, some of the experts noted that some of the identified errors belong to the pre-requisite knowledge (e.g., handling parentheses) and so can be removed from this error set. This supports our hypothesis that the proposed method can be used to create full, fine-grained sets of reasoning errors in subject domains. The experts selected the following errors as most frequent: **g1** (3 experts); **g2**, **g3**, **126**, **139**, **141**, and **142** (2 experts each).

5 Conclusion

When developing ITS, too little attention was given to generating sets of errors, systematically describing wrong ways of thinking. The papers about ITS rarely contain full lists of identified errors and the methods used for identifying them. This can lead to omitting some kinds of errors or basing the tutor’s feedback on the errors that are determined by the algorithm the ITS is based on without prying further into their causes. When follow-up questions were used (e.g., in [13]), they were manually created by a subject-domain expert and so prone to omissions and errors. This paper intends to fill this gap and stimulate discussion on how error sets for subject domains should be built systematically.

Our method is applicable to any subject domain where reasoning, required to find a correct answer, can be represented as a decision tree, consisting of questions, actions, branching and iterations. We used it for problems in teaching programming like expression evaluation, determining data types, tracing control-flow statements, building access expressions, etc.; we consider using it in other well-defined areas like word order in English language. The resulting error sets can be used to: (1) measure learners’ progress in acquiring mastery of the topic; (2) select the next learning task in adaptive systems; (3) generate systematic sets of follow-up questions to find the exact cause of the learner’s error; (4) classify automatically generated questions by knowledge required to solve them.

Further work will include developing a program module for generating error sets from formalized descriptions of lines of reasoning, developing methods of generating follow-up questions based on these error sets, and implementing them in an intelligent tutoring system.


References

1. Afzal, S., Shashidhar, V., Sindhgatta, R., Sengupta, B.: Impact of tutor errors on student engagement in a dialog based intelligent tutoring system. In: Nkambou, R., Azevedo, R., Vassileva, J. (eds.) ITS 2018. LNCS, vol. 10858, pp. 267–273. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-91464-0_26
2. Brusilovsky, P.: Domain Modeling for Personalized Guidance, pp. 165–183. Army Research Laboratory (2016)
3. Holland, J.: A Constraint-based ITS for the Java Programming Language. Master's thesis, University of Canterbury, Christchurch, New Zealand (2009). <https://doi.org/10.26021/1309>
4. Hong, J.: Guided programming and automated error analysis in an intelligent Prolog tutor. *Int. J. Hum.-Comput. Stud.* **61**(4), 505–534 (2004). <https://doi.org/10.1016/j.ijhcs.2004.02.001>
5. Hooshyar, D., Ahmad, R.B., Yousefi, M., Fathi, M., Horng, S.J., Lim, H.: SITS: a solution-based intelligent tutoring system for students' acquisition of problem-solving skills in computer programming. *Innov. Educ. Teach. Int.* **55**(3), 325–335 (2016). <https://doi.org/10.1080/14703297.2016.1189346>
6. Laengrich, M., Schulze, J., Kumar, A.N.: Expression tasks for novice programmers: turning the attention to objectivity, reliability and validity. In: 2015 IEEE Frontiers in Education Conference (FIE). IEEE, October 2015. <https://doi.org/10.1109/fie.2015.7344070>
7. Li, D., Zhou, H.H.: An intelligent tutoring system with an automated knowledge acquisition mechanism. In: 2015 IEEE International Conference on Computational Intelligence & Communication Technology, pp. 88–91. IEEE, February 2015. <https://doi.org/10.1109/cict.2015.97>
8. Mitrovic, A., Martin, B., Suraweera, P.: Intelligent tutors for all: the constraint-based approach. *IEEE Intell. Syst.* **22**(4), 38–45 (2007). <https://doi.org/10.1109/mis.2007.74>
9. Ramesh, V.M., Rao, N.J., Ramanathan, C.: Implementation of an intelligent tutoring system using moodle. In: 2015 IEEE Frontiers in Education Conference (FIE), pp. 1–9. IEEE, October 2015. <https://doi.org/10.1109/fie.2015.7344313>
10. Rathore, A.S., Arjaria, S.: Intelligent Tutoring System, pp. 121–144. IGI Global (2020). <https://doi.org/10.4018/978-1-7998-0010-1.ch006>
11. Shute, V., Zapata-Rivera, D.: Intelligent systems. In: Peterson, P., Baker, E., McGaw, B. (eds.) *International Encyclopedia of Education (Third Edition)*, pp. 75–80. Elsevier, Oxford, third edition edn. (2010). <https://doi.org/10.1016/B978-0-08-044894-7.00247-5>
12. Sychev, O.A., Mamontov, D.P.: Automatic error detection and hint generation in the teaching of formal languages syntax using correctwriting question type for moodle lms. In: 2018 3rd Russian-Pacific Conference on Computer Technology and Applications (RPC), pp. 1–4, August 2018. <https://doi.org/10.1109/RPC.2018.8482125>
13. Sychev, O., Penskoj, N., Anikin, A., Denisov, M., Prokudin, A.: Improving comprehension: intelligent tutoring system explaining the domain rules when students break them. *Educ. Sci.* **11**(11), 719 (2021). <https://doi.org/10.3390/educsci11110719>
14. Wang, M., Wu, W., Liang, Y.: A novel intelligent tutoring system for learning programming. In: 2020 International Conference on Development and Application Systems (DAS), pp. 162–168. IEEE, May 2020. <https://doi.org/10.1109/das49615.2020.9108925>

15. Wang, M., Chau, H., Thaker, K., Brusilovsky, P., He, D.: Knowledge annotation for intelligent textbooks. *Technol. Knowl. Learn.* 1–22 (2021). <https://doi.org/10.1007/s10758-021-09544-z>
16. Weragama, D., Reye, J.: The PHP intelligent tutoring system. In: Lane, H.C., Yacef, K., Mostow, J., Pavlik, P. (eds.) *AIED 2013. LNCS (LNAI)*, vol. 7926, pp. 583–586. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-39112-5_64



Handshape Recognition in an Educational Game for Finger Alphabet Practicing

Tomasz Kapuscinski^(✉) 

Department of Computer and Control Engineering, Faculty of Electrical and Computer Engineering, Rzeszow University of Technology,
W. Pola 2, 35-959 Rzeszow, Poland
tomekkap@kia.prz.edu.pl

Abstract. A vision-based method of handshape recognition was developed and used in a simple educational game designed to practice the finger alphabet. It uses a deep neural network to determine the two-dimensional skeleton of the hand and the genetic algorithm to recognize its shape. The classification was carried out by defining and solving an optimization problem, in which the skeleton corresponding to the unknown shape is subjected to an affine transformation and then adjusted to the previously prepared set of templates. The method was tested using the leave-one-subject-out validation protocol on the author's dataset of Polish Finger Alphabet letters and the publicly available Microsoft Kinect and Leap Motion Dataset. Based on the developed algorithm, a simple educational game was prepared, the purpose of which is to practice hand dexterity in showing complex shapes appearing in the finger alphabet.

Keywords: Handshape recognition · Finger alphabet · Educational game

1 Introduction

The finger alphabet, also called the manual or hand alphabet, is the representation of the letters using handshape and/or its movement. It is used by deaf people during the so-called fingerspelling (dactylography) as a complement to sign language. Words for which there is no distinct gesture in sign languages are fingerspelled. Handshapes from the finger alphabet, shown for different positions and orientations, are also an important component of other dynamic signs. There are about 40 finger alphabets in the world.

Learning sign language usually starts with the finger alphabet because it requires mastering the technique of forming fingers into precise configurations. At the beginning of learning users often make mistakes, which, if they persist in

This project is financed by the Minister of Education and Science of the Republic of Poland within the “Regional Initiative of Excellence” program for years 2019–2022. Project number 027/RID/2018/19, amount granted 11 999 900 PLN.

the future, may limit the ability to communicate freely. The most common mistakes include excessive muscle tension (the so-called ‘hard hand’), unnecessary accentuation of a sign by moving the arm back and forth, and a tendency to look at the hand while performing the signs. Special rehabilitation exercises are being developed to achieve proper precision, softness, and fluidity of movements. After a few weeks of regular practice, the muscles relax, and the fingers gain much greater efficiency [31].

Children can be encouraged to do such exercises by offering them participation in an educational game, which is the motivation to carry out the work described in this article. In the prepared game, the precision of handshapes presentation is achieved by the recognition method based on the comparison with prerecorded templates, while the speed and smoothness are stimulated by introducing an element of competition.

The structure of this paper is as follows. Section 2 provides the research background and relevant literature references. The proposed method of handshapes recognition is described in Sect. 3. Its experimental evaluation is reported in Sect. 4. The educational game developed to practice the finger alphabet is described in Sect. 5. A summary and a proposal for further work are given in Sect. 6.

2 Related Works

Various educational tools for learning finger alphabets and sign languages are described in the literature. There are known solutions for national sign languages: American (ASL) [9, 11], Brazilian (Libras) [2, 17], British (BSL) [6], Indian (ISL) [10, 23], Korean (KSL) [15], Costa Rican (LESCO) [24], Macedonian (MSL) [1], Malaysian (BIM) [30], Mexican (LSM) [8], Pakistani (PSL) [25], Persian (PSL) [20], Portuguese (LGP) [7], Thai (ThSL) [26, 28], Turkish (TSL) [12, 32], Ukrainian (USL) [13].

Most of the developed tools are used to memorize handshapes and movements and to improve the skills of understanding and interpreting signed messages, e.g. [1, 2, 8, 12–14, 16, 24–26, 28]. However, there are also tools that stimulate users to make gestures on their own, e.g. [6, 7, 9–11, 15, 17, 23, 30, 32]. Frequent exercises involving the repetition of gestures allow to obtain the appropriate dexterity and flexibility of the hand and enable smooth signing.

Solutions from the second group require automatic recognition of handshapes and/or dynamic hand gestures. There are applications that use dedicated equipment in the form of special gloves [9, 15], vision-based solutions [6, 10, 11, 17, 20, 23, 30] or a combination of both approaches [7]. The disadvantages of solutions using special gloves are cost, availability, and unnatural way of interaction. For example, wiring can load certain parts of the hand, affecting the degree of muscle tension and smoothness of movement. This can lead to developing incorrect habits when learning gestures.

Solutions with the use of humanoid robots [12, 16, 18, 20, 26, 32] or artificial, anthropomorphic hands [14] have also been developed. They are attractive,

especially for the youngest users, fascinated by advanced technology. Their disadvantages, however, are the high cost and availability of advanced technology.

Most vision systems that use color cameras impose severe restrictions on the background, lighting, and clothing. There are also solutions using RGB-D sensors [7, 11, 32] or motion controllers, e.g. LeapMotion [6]. They use good quality depth data and work in complex, heterogeneous backgrounds, with unstable lighting. However, RGB-D cameras are only embedded in certain laptop models.

Therefore, among the solutions offered, those that use typical color cameras are particularly attractive. Their use involves the need to solve the problem of variable background and lighting algorithmically. The dynamically developing technology of deep learning can be used for this purpose. The solution presented in this paper falls within this group.

The advantages of the proposed solution are as follows: (i) no extraordinary restrictions on lighting, background, and clothing; (ii) working with a limited set of templates - easy to adapt to other finger alphabets; (iii) ordinary, inexpensive equipment: laptop and built-in camera; (iv) customizable architecture - easy to change the graphics, theme of the game, and adapt it to children of all ages.

3 Method Description

Color images, obtained using an ordinary laptop camera and depicting the upper part of the finger-spelling user, were used. In these images, two-dimensional hand skeletons were determined. The classification was carried out by solving an optimization problem in which the skeleton of an unknown handshake is subjected to an affine transformation, including translation, rotation, and scale change, and then it is matched against the set of the previously prepared template skeletons (Fig. 1).

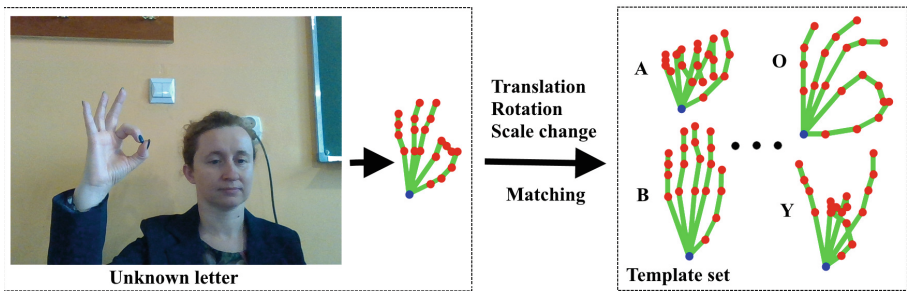


Fig. 1. Idea of the method (Color figure online)

The Pompeiu-Hausdorff distance [3, 27] was used to measure the distance between skeletons treated as sparse point clouds, composed only of points constituting the nodes of the skeleton. The approach in which skeletons are transformed before matching should guarantee effective recognition even when the template set is limited. Details are given in 3.1, 3.2, and 3.3.

3.1 Determining Hand Skeletons

Two-dimensional hand skeletons were determined using the OpenPose library [4, 5, 29, 33]. The model shown in Fig. 2 (c) was adopted. It consists of nodes that are analogs of the joints (red circles in Fig. 2 (c)). They are connected by rigid sections corresponding to bones (green segments in Fig. 2 (c)). Such models are generated by running an image through a deep pre-trained convolutional neural network to obtain a series of heat maps (one for each node, Fig. 2 (b)).

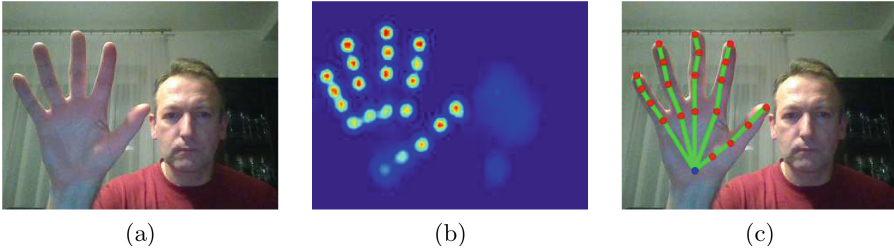


Fig. 2. Determining hand skeleton: (a) input image, (b) accumulated and normalized heatmaps, (c) skeleton superimposed on the image (Color figure online)

3.2 Pompeiu-Hausdorff Distance

Let P and Q denote two point clouds in metric space with the metric d . The Pompeiu-Hausdorff distance is defined as:

$$h(P, Q) = \max \{d_1, d_2\} \quad (1)$$

where: $d_1 = \sup_{p \in P} d(p, Q)$, $d_2 = \sup_{q \in Q} d(q, P)$, $d(p, Q) = \inf_{q \in Q} d(p, q)$, \sup , \inf denote supremum and infimum (Fig. 3).

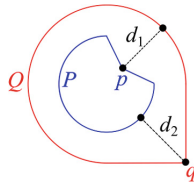


Fig. 3. Pompeiu-hausdorff distance (Color figure online)

3.3 Optimization Problem Definition

The following vector of decision variables was used:

$$X = [a, b, \alpha, s_x, s_y]^T \quad (2)$$

where:

a, b - the coordinates of the translation vector in the image plane,

α - the rotation angle in the image plane around the skeleton center of gravity,

s_x, s_y - the coefficients of scale change, respectively, along the x and y axes.

The Pompeiu-Hausdorff distance to the ‘closest’ template was selected as the fitness function:

$$J(X) = \min_{i=1..n} h(T(X)U, S_i) \quad (3)$$

where:

$h(P_1, P_2)$ - the Pompeiu-Hausdorff distance between the point clouds P_1 and P_2 ,

$T(X)$ - the transformation operator (translation, rotation, and scale change) defined by elements of X ,

U - the point cloud corresponding to the skeleton of the hand representing an unknown letter,

S_i - the point cloud corresponding to the skeleton of the i -th template,

n - the number of templates.

The following constraints were also adopted:

$$LB \leq X \leq UB \quad (4)$$

where:

$LB = [a_{min}, b_{min}, \alpha_{min}, s_{x_{min}}, s_{y_{min}}]^T$, $UB = [a_{max}, b_{max}, \alpha_{max}, s_{x_{max}}, s_{y_{max}}]^T$
- lower and upper limits of the parameters defining the transformation of the point cloud.

The optimization task was defined as minimizing:

$$\begin{aligned} J_{min} &= \min_{LB \leq X \leq UB} J(X) \\ \hat{X} &= \arg \min_{LB \leq X \leq UB} J(X) \end{aligned} \quad (5)$$

After the optimization task is solved, the recognized class is determined by a label \hat{l} corresponding to the ‘closest’ template, obtained when calculating $J(\hat{X})$:

$$\hat{i} = \arg \min_{i=1..n} h(T(\hat{X})U, S_i) \quad (6)$$

$$\hat{l} = L(\hat{i}) \quad (7)$$

where: L , $\dim L = n$ - the set of labels corresponding to the templates. The genetic algorithm was used to solve the optimization problem [21].

4 Method Evaluation

The experiments were conducted on two datasets: (i) the dataset of PFA handshapes and (ii) the Microsoft Kinect and Leap Motion Dataset, using the leave-one-subject-out (l-o-s-o) protocol. Details are given in 4.1–4.4.

4.1 Datasets Used

The dataset of PFA handshapes contains recordings of 22 classes shown ten times by ten users. It was prepared in cooperation with the Subcarpathian Association of the Deaf. 24-bit RGB images with a resolution of 640×480 were obtained using a regular color camera built into the laptop (Fig. 4).



Fig. 4. Selected images from the dataset of PFA handshapes

Microsoft Kinect and Leap Motion Dataset contains recordings of ten handshapes shown ten times by fourteen users. It contains, among other data, 24-bit RGB images with a resolution of 1280×960 . The dataset is available on the Internet and is often used as benchmark data to compare different recognition methods [18, 19, 22].

The leave-one-subject-out (l-o-s-o) validation protocol (l-o-s-o) was used in the experiments. It consists in selecting the data recorded by one user as a test set and the remaining data as the training set. The procedure is repeated, selecting a different user each time, and the obtained results are averaged. This protocol usually gives worse recognition results because the individual gestures differ significantly. However, it is a more reliable assessment method. It estimates how the system will perform when gestures are executed by an unknown user, not involved in the preparation of the training data.

4.2 Parameter Settings

A computer with an Intel(R) Core(TM) i7-6700K CPU @ 4.00GHz processor, equipped with 64 GB of RAM and two TITAN RTX and GeForce GTX 750 Ti graphics cards was used. The experiments were conducted in the Matlab R2021b environment.

Table 1 lists the genetic algorithm settings used in the experiments.

Table 1. Genetic algorithm settings

Parameter name	Description	Value
<i>MaxGenerations</i>	Maximum number of iterations before the algorithm halts	100
<i>PopulationSize</i>	Size of the population	1000
<i>UseParallel</i>	Compute fitness function in parallel	True
<i>UseVectorized</i>	Specifies whether functions are vectorized	False
<i>Display</i>	Level of display	Off

The sensitivity of the method to changes in the size, position, and orientation of the hand depends on the optimization constraints settings used in the experiments (Table 2). The increased range of the translation vector, given in parentheses, applies to Microsoft Kinect and Leap Motion Dataset, which contains higher-resolution images.

Table 2. Optimization constraints settings

Variable	Value	Variable	Value
a_{min}	-320 (-640)	a_{max}	320 (640)
b_{min}	-240 (-480)	b_{max}	240 (480)
α_{min}	-180	α_{max}	180
$s_{x_{min}}$	0.75	$s_{x_{max}}$	1.25
$s_{y_{min}}$	0.75	$s_{y_{max}}$	1.25

A small, unrepresentative template set may require wider ranges of changes to be set. Note, however, that setting the ranges too wide extends the search domain, which may require increasing *PopulationSize* and/or *MaxGenerations* and thus increasing the system response time.

4.3 Results

The PFA handshapes dataset was refined by eliminating erroneous performances, and handshapes shown by seven people, labeled P1, P2, ..., P7, were selected for further experiments. The obtained results of l-o-s-o validation are shown in Table 3. The results of l-o-s-o validation for Microsoft Kinect and Leap Motion Dataset are shown in Table 4.

Table 3. Results of l-o-s-o validation for the PFA handshapes dataset

Iteration	Template set	Test set	Recognition rate [%]
1	P2, P3, P4, P5, P6, P7	P1	86.36
2	P1, P3, P4, P5, P6, P7	P2	84.09
3	P1, P2, P4, P5, P6, P7	P3	85.00
4	P1, P2, P3, P5, P6, P7	P4	80.00
5	P1, P2, P3, P4, P6, P7	P5	85.45
6	P1, P2, P3, P4, P5, P7	P6	85.91
7	P1, P2, P3, P4, P5, P6	P7	81.36
Average			84.03

Table 4. Results of l-o-s-o validation for microsoft kinect and leap motion dataset

Iteration	Template Set	Test Set	Recognition Rate [%]
1	P2, P3, P4, P5, P6, P7, P8, P9, P10, P11, P12, P13, P14	P1	88.00
2	P1, P3, P4, P5, P6, P7, P8, P9, P10, P11, P12, P13, P14	P2	82.00
3	P1, P2, P4, P5, P6, P7, P8, P9, P10, P11, P12, P13, P14	P3	85.00
4	P1, P2, P3, P5, P6, P7, P8, P9, P10, P11, P12, P13, P14	P4	87.00
5	P1, P2, P3, P4, P6, P7, P8, P9, P10, P11, P12, P13, P14	P5	81.00
6	P1, P2, P3, P4, P5, P7, P8, P9, P10, P11, P12, P13, P14	P6	83.00
7	P1, P2, P3, P4, P5, P6, P8, P9, P10, P11, P12, P13, P14	P7	86.00
8	P1, P2, P3, P4, P5, P6, P7, P9, P10, P11, P12, P13, P14	P8	85.00
9	P1, P2, P3, P4, P5, P6, P7, P8, P10, P11, P12, P13, P14	P9	81.00
10	P1, P2, P3, P4, P5, P6, P7, P8, P9, P11, P12, P13, P14	P10	84.00
11	P1, P2, P3, P4, P5, P6, P7, P8, P9, P10, P12, P13, P14	P11	87.00
12	P1, P2, P3, P4, P5, P6, P7, P8, P9, P10, P11, P13, P14	P12	84.00
13	P1, P2, P3, P4, P5, P6, P7, P8, P9, P10, P11, P12, P14	P13	83.00
14	P1, P2, P3, P4, P5, P6, P7, P8, P9, P10, P11, P12, P13	P14	84.00
Average			84.29

4.4 Discussion

The developed method was tested on two datasets with different characteristics. The PFA handshapes dataset contains more classes. Some letters require a lot of skill, therefore their performances differ significantly from the ideal shapes presented in the sign language dictionaries. On the other hand, Microsoft Kinect and Leap Motion Dataset are characterized by greater variability in the individual appearances of the people participating in the recordings. In both cases, gestures are shown in typical lighting conditions, on a very demanding background containing objects with chrominance similar to the skin color. Sometimes the hand appears against the background of the face. Despite this, in both cases, a satisfactory recognition rate greater than 84% was achieved.

Two error sources were noticed: (i) misclassification of the letters shown imprecisely or under wrong hand orientation, (ii) incorrect skeleton determination, especially for the visually similar shapes. The first one should encourage the user to show gestures carefully. To deal with the latter one, tuning the OpenPose model using the transfer learning method, applying other deep learning-based models, and using 3D skeletons are planned.

The results obtained for Microsoft Kinect and Leap Motion Dataset exceed by almost 4% points the results obtained by the authors of the database, when they used only the skeleton data obtained with the LeapMotion sensor [18]. After adding the features determined on the basis of RGB-D images acquired by the Kinect sensor, they achieved a recognition rate of 91.28%.

It should be emphasized that the developed method does not require any special controllers or RGB-D sensors, as it determines 2D skeletons in ordinary color images using deep learning techniques. Moreover, unlike solutions available in Kinect or LeapMotion, it is open and can be further refined using custom, user-specific data.

Another advantage of the method is the ability to effectively recognize handshapes also when the set of templates is small. Tests of the educational game described in Chap. 5 have shown that this is possible even when only one pattern for each considered shape is available. This is because the skeletons corresponding to the recognized shapes undergo an affine transformation involving translation, rotation, and scale change over a wide range. However, this means an increase in the recognition time, which on the computer used in the tests was about 5 s.

5 Educational Game

Based on the presented method, a simple educational game was prepared, the aim of which is to encourage children to practice hand dexterity and learn the PFA letters. It is about the correct and quick execution of a given shape. Correctness is verified by comparing the shown shapes with the saved templates. Speed is stimulated by the introduction of an element of competition.

The screen has been divided into several configurable areas (Fig. 5 (a)). Media files presented in the areas of stimulus, player and competitor rewards are loaded from the appropriate folders. Thanks to this, the game can be easily modified

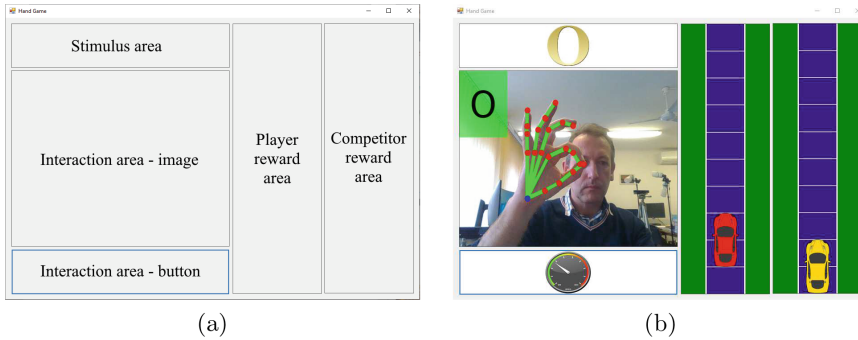


Fig. 5. Educational game: (a) screen layout, (b) current version of the game after correct recognition of the letter O

and adapted to the individual needs of the child without having to recompile the code.

In the current version, a stimulant in the form of a letter is presented in the stimulus area. In the interaction area, a preview from the camera is shown. After showing the required handshape, the user presses the button and the recognition begins using the method described in Sect. 3. The results of the recognition in the form of the detected skeleton model and the recognized letter are overlaid on the displayed image (Fig. 5 (b)). If the handshape is correctly recognized, the image in the reward area is updated. In the current version of the game, the red car is moved forward. The application can be run on two networked computers. The results of the second player are presented in the competitor reward area.

The results were presented at the scientific seminar ‘IT Systems Supporting the Deaf’ conducted by the author at the Subcarpathian Association of the Deaf and assessed during several working meetings. Tests carried out by the deaf users have shown that handshapes shown carelessly or inconsistently with the registered templates, based on the sign language dictionary, were not accurately classified. This observation suggests that the presented game may stimulate players to correctly and carefully perform the letters of the finger alphabet.

The tests also revealed that the method accuracy strongly depends on the correctness of the skeleton determination. The presence of outliers along the length of the phalanges may indicate an abnormal, damaged skeleton. In that case, the procedure is repeated for the next acquired image frame.

Response time is also a key factor influencing the rating of the game. The software version has been prepared, in which the skeletons are determined using a graphics card, and the genetic algorithm is run in parallel on a multicore processor.

6 Conclusions and Future Work

A method of recognizing PFA letters using color images and the OpenPose library was developed. The classification was carried out by solving an opti-

mization problem in which a skeleton of an unknown handshape is subjected to an affine transformation and then matched to a previously prepared base of reference skeletons. The method was tested using the l-o-s-o protocol on the dataset of PFA handshapes and Microsoft Kinect and Leap Motion Dataset. A simple educational game for learning the PFA letters using the developed method was prepared. The consecutive, necessary research step will be a formal assessment of the game's impact on potential users.

Possible directions of further research are: (i) use of three-dimensional skeletons; (ii) introduction of flexible models in which, in addition to global affine transformation, local changes in the length of phalanges and angles between them are also taken into account; (iii) testing other, faster global optimization algorithms; (iv) preparing a version with the use of libraries whose license allows for sharing and/or commercialization of the game; (v) designing alternative graphic variants and formal evaluation of their usability.

Acknowledgments. The author would like to express his deepest gratitude to the Subcarpathian Association of the Deaf for their kind assistance and support and members of the Scientific Circle of Human-Computer Interaction GEST for their help in the game frontend development, its presentation, and evaluation.

References

1. Ackovska, N., Kostoska, M.: Sign language tutor-rebuilding and optimizing. In: 2014 37th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO), pp. 704–709. IEEE (2014)
2. Antunes, D.R., Rodrigues, J.D.: Endless running game to support sign language learning by deaf children. In: Antona, M., Stephanidis, C. (eds.) HCII 2021. LNCS, vol. 12769, pp. 25–40. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-78095-1_3
3. Birsan, T., Tiba, D.: One hundred years since the introduction of the set distance by Dimitrie Pompeiu. In: Ceragioli, F., Dontchev, A., Futura, H., Marti, K., Pandolfi, L. (eds.) CSMO 2005. IIFIP, vol. 199, pp. 35–39. Springer, Boston, MA (2006). https://doi.org/10.1007/0-387-33006-2_4
4. Cao, Z., Hidalgo, G., Simon, T., Wei, S.E., Sheikh, Y.: Openpose: realtime multi-person 2d pose estimation using part affinity fields. *IEEE Trans. Pattern Anal. Mach. Intell.* **43**(1), 172–186 (2019)
5. Cao, Z., Simon, T., Wei, S.E., Sheikh, Y.: Realtime multi-person 2d pose estimation using part affinity fields. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 7291–7299 (2017)
6. Economou, D., Russi, M.G., Doumanis, I., Mentzelopoulos, M., Bouki, V., Ferguson, J.: Using serious games for learning British sign language combining video, enhanced interactivity, and vr technology. *J. Univ. Comput. Sci.* **26**(8), 996–1016 (2020)
7. Escudeiro, P., et al.: Virtual sign-a real time bidirectional translator of Portuguese sign language. *Procedia Comput. Sci.* **67**, 252–262 (2015)
8. Estrada-Cota, I., Carreño-León, M.A., Sandoval-Bringas, J.A., Leyva-Carrillo, A.A.: "manos que hablan": tool for teaching-learning the Mexican sign language for children with or without hearing disabilities. In: 2020 3rd International Conference of Inclusive Technology and Education (CONTIE), pp. 173–179. IEEE (2020)

9. Hernandez-Rebollar, J.L., Elsakay, E.I., Alanís-Urquieta, J.D.: Acelespell, a gestural interactive game to learn and practice finger spelling. In: Proceedings of the 10th International Conference on Multimodal Interfaces, pp. 189–190 (2008)
10. Joy, J., Balakrishnan, K., Sreeraj, M.: SignQuiz: a quiz based tool for learning fingerspelled signs in Indian sign language using ASLR. *IEEE Access* **7**, 28363–28371 (2019)
11. Kamnardsiri, T., Hongsit, L.O., Khuwuthyakorn, P., Wongta, N.: The effectiveness of the game-based learning system for the improvement of American sign language using kinect. *Electron. J. E-Learn.* **15**(4), 283–296 (2017)
12. Kose, H., Yorganci, R.: Tale of a robot: humanoid robot assisted sign language tutoring. In: 2011 11th IEEE-RAS International Conference on Humanoid Robots, pp. 105–111. *IEEE* (2011)
13. Krak, I., Kryvonos, I., Wojcik, W.: Interactive systems for sign language learning. In: 2012 6th International Conference on Application of Information and Communication Technologies (AICT), pp. 1–3. *IEEE* (2012)
14. Krastev, A., Lekova, A., Dimitrova, M., Chavdarov, I.: An interactive technology to support education of children with hearing problems. In: Proceedings of the 15th International Conference on Computer Systems and Technologies, pp. 445–451 (2014)
15. Lee, Y., Min, S., Yang, H., Jung, K.: Motion sensitive glove-based Korean fingerspelling tutor. In: 2007 International Conference on Convergence Information Technology (ICCIT 2007), pp. 1674–1677. *IEEE* (2007)
16. Luccio, F.L., Gaspari, D.: Learning sign language from a sanbot robot. In: Proceedings of the 6th EAI International Conference on Smart Objects and Technologies for Social Good, pp. 138–143 (2020)
17. Madeo, R.C.B.: Brazilian sign language multimedia hangman game: a prototype of an educational and inclusive application. In: The proceedings of the 13th International ACM SIGACCESS Conference on Computers and Accessibility, pp. 311–312 (2011)
18. Marin, G., Dominio, F., Zanuttigh, P.: Hand gesture recognition with leap motion and kinect devices. In: 2014 IEEE International Conference on Image Processing (ICIP), pp. 1565–1569. *IEEE* (2014)
19. Marin, G., Dominio, F., Zanuttigh, P.: Hand gesture recognition with jointly calibrated leap motion and depth sensor. *Multimedia Tools Appl.* **75**(22), 14991–15015 (2016)
20. Meghdari, A., Alemi, M., Zakipour, M., Kashanian, S.A.: Design and realization of a sign language educational humanoid robot. *J. Intell. Robot. Syst.* **95**(1), 3–17 (2019)
21. Mitchell, M.: *An Introduction to Genetic Algorithms*. MIT Press, Cambridge (1998)
22. Multimedia Technology and Telecommunications Laboratory, University of Padova: Hand Gesture Datasets. <https://lstm.dei.unipd.it/downloads/gesture/>. Accessed: 14 Jan 2022
23. Nanaware, T., Sahasrabudhe, S., Ayer, N., Christo, R.: Fingerspelling-Indian sign language training tool. In: 2018 IEEE 18th International Conference on Advanced Learning Technologies (ICALT), pp. 330–334. *IEEE* (2018)
24. Naranjo-Zeledón, L., Chacón-Rivas, M., Peral, J., Ferrández, A., Gil, D.: Improvement of a sign language learning reinforcement tool through phonological proximity. In: Visvizi, A., Troisi, O., Saeedi, K. (eds.) *RIIFORUM 2021*. SPC, pp. 99–107. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-84311-3_10

25. Parvez, K., et al.: Measuring effectiveness of mobile application in learning basic mathematical concepts using sign language. *Sustainability* **11**(11), 3064 (2019)
26. Pluempitiwiriyaewej, C., Changsnit, P., Chevapatr, P., Ranong, S.N.: Fing: thai fingerspelling practice application. In: 2017 6th ICT International Student Project Conference (ICT-ISPC), pp. 1–4. IEEE (2017)
27. Rockafellar, R.T., Wets, R.J.B.: *Variational Analysis*, vol. 317. Springer, New York (2009). <https://doi.org/10.1007/978-3-642-02431-3>
28. Silanon, K.: Thai finger-spelling computer-assisted instruction for hearing and speaking impaired children. In: Proceedings of the international Convention on Rehabilitation Engineering & Assistive Technology, pp. 1–4 (2016)
29. Simon, T., Joo, H., Matthews, I., Sheikh, Y.: Hand keypoint detection in single images using multiview bootstrapping. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1145–1153 (2017)
30. Siong, T., Nasir, N., Salleh, F.: A mobile learning application for Malaysian sign language education. In: *Journal of Physics: Conference Series*, p. 012004. IOP Publishing (2021)
31. Szczepankowski, B.: Signed Polish in School 1 (in Polish: *Jezyk migany w szkole 1*). WSiP (2008)
32. Uluer, P., Akalın, N., Köse, H.: A new robotic platform for sign language tutoring. *Int. J. Soc. Robot.* **7**(5), 571–585 (2015)
33. Wei, S.E., Ramakrishna, V., Kanade, T., Sheikh, Y.: Convolutional pose machines. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 4724–4732 (2016)



Comparing Alternative Approaches to Debriefing in a Tool to Support Peer-Led Simulation-Based Training

Sandra Katz¹ (✉), Patricia Albacete¹, John Gallagher², Pamela Jordan¹, Thomas Platt³, Scott Silliman¹, and Tiffany Yang⁴

¹ Learning Research and Development Center, University of Pittsburgh, Pittsburgh, PA, USA
katz@pitt.edu

² School of Nursing, University of Pittsburgh, Pittsburgh, PA, USA

³ School of Health and Rehabilitation Sciences, University of Pittsburgh, Pittsburgh, PA, USA

⁴ School of Medicine, Department of Pediatrics, University of Pittsburgh, Pittsburgh, PA, USA

Abstract. This poster describes an early-stage project. It introduces MedDbriefer, a tablet-based tool that allows small groups of paramedic students to practice realistic prehospital emergency care scenarios. While two or more students collaborate as members of an emergency medical service (EMS) team, a peer uses the tablet's checklists to record the team's actions. The system then analyzes the event log to provide an automated debriefing on the team's performance. Although debriefing is purported to be one of simulation-based training's most critical components, there is little research to guide human and automated debriefing. We are implementing two approaches to automated debriefing and will compare their effectiveness in an upcoming randomized controlled trial.

Keywords: Simulation-based training · Debriefing · Healthcare training

1 Project Goals

The coronavirus pandemic highlighted the dire consequences of an international shortage of paramedics and other emergency medical service (EMS) providers [1, 2]. This poster introduces MedDbriefer, a tablet-based tool that allows small groups of EMS students to engage in simulated prehospital emergency care scenarios and participate in an automated debriefing on their performance.

Across the healthcare professions, students who struggle to acquire clinical reasoning and psychomotor skills can rarely get the supplemental simulation-based training (SBT) they need to pass certification exams. Instructors who are trained to facilitate simulated scenarios during course labs are in short supply [3, 4]. Many instructors are themselves active healthcare providers, which limits the time that they can devote to teaching. If successful, MedDbriefer could help to reduce the shortage of EMS providers and, ultimately, other healthcare professionals.

Even if instructors were plentiful, little is known about how to guide them in conducting an effective debriefing [5, 6], which is often deemed to be simulation-based training's

most critical component [5, 7–9]. Our research goal is to extend the knowledge base on debriefing. Toward that end, we are implementing two approaches to automated debriefings in MedDbriefer. One approach mirrors that taken in state of the art computer-based healthcare systems: a step-by-step narrative of students' actions during the scenario, coupled with color-coded (red-green-yellow) textual feedback [10]. The second, experimental approach adapts one of several debriefing protocols developed to help human SBT instructors to structure their debriefings [5, 11, 12]. These protocols actively engage students in the process of reflecting on and critiquing their performance, more so than do narrative system-led debriefings. However, there is no empirical evidence that a protocol-based approach to automated debriefing would be more beneficial for learning than a narrative walkthrough of students' actions. An upcoming randomized controlled trial will examine this question.

2 MedDbriefer

As in most healthcare professions, becoming a paramedic requires mastery over a circumscribed body of domain knowledge, clinical reasoning skills, team coordination skills, and numerous psychomotor skills (e.g., intubating a patient's airway) [13, 14]. MedDbriefer focuses on developing clinical reasoning and decision-making skills. It is a web-based application designed to run on a tablet, so that it ultimately can be used for scenario-based practice just about anywhere: in a simulation lab, breakout room, dorm room, etc., without the need for simulation equipment or a human instructor.

MedDbriefer's scenarios are adapted from those included in standard EMS training curricula, such as the *Prehospital Trauma Life Support* scenario bank. They exercise clinical reasoning skills by requiring students to: (1) perform a rapid but thorough patient assessment in order to gather clinical findings and other pertinent information (e.g., vital signs such as heart rate and blood pressure; the events leading up to an injury or illness); (2) interpret these findings to identify life-threats requiring immediate attention and less serious problems to address if time permits (e.g., minor wounds); and (3) determine what interventions to perform and how to perform them.

For example, one scenario involves a patient who experienced a lawnmower rollover accident. He presents with an amputated foot; severe blood loss; low blood pressure; pale, cool, diaphoretic skin; decreasing consciousness; and numerous bruises and lacerations. These findings indicate that the patient is in hypovolemic shock, a serious condition that should be managed immediately by applying a tourniquet to stop blood loss, intubating the patient's airway, administering high flow oxygen and intravenous (IV) fluids, and keeping the patient warm. Various decisions govern effective execution of these interventions, such as: the need for sedation prior to intubation, type and size of airway adjunct, type and dosage of fluids to administer intravenously.

Simulation-based training during live, instructor guided labs that use practice scenarios such as this one typically assigns one student to play the role of team leader while one or more peers serve as team member(s)—for example, an emergency medical technician (EMT) who assists the lead paramedic. The scenario can be conducted in various ways, depending on the focus of instruction. If the instructor wants students to practice psychomotor skills as well as non-technical skills (e.g., team coordination;

clinical reasoning and decision making), the instructor will direct the EMS team to fully assess and treat the simulated patient (e.g., a manikin or peer), using lab equipment and supplies. However, if the focus is on non-technical skills, the instructor might opt to have students “voice treat” the simulated patient. Voice treating entails verbalizing the assessment and treatment actions the team leader would perform, how he would perform them, which actions he would delegate to a team member, etc. In addition to its role in training, voice treating is used for assessment (e.g., on part of the National Registry of Emergency Medical Technicians Paramedic certification exam). Although students often mime actions using readily available equipment (e.g., a stethoscope), voice treating obviates the need for students to perform interventions, thereby allowing them to focus on clinical reasoning and decision-making skills instead of psychomotor skills.

The screenshot shows the MedDbriefer interface for a scenario named 'M2CA'. The left sidebar contains a navigation menu with sections: BSI, Scene Size-up, Primary Survey (GENERAL IMPRESSION, AIRWAY, BREATHING, CIRCULATION, TRANSPORT DECISION, TRAUMA EXPOSE), History Taking, Secondary Survey (HEAD, NECK, CHEST, ABDOMEN, PELVIS, EXTREMITIES, POSTERIOR THORAX, LUMBAR AND BUTTCKS), and Reassessment Plan. The main content area is titled 'MedDbriefer SCENARIO: M2CA' and includes a 'CHECKLIST' section with a dropdown arrow. The checklist contains a note: 'State what you are looking/listening/feeling for while checking circulation.' Below this is the 'Checked pulse' section with the question 'Which pulse?' and radio buttons for 'carotid', 'radial' (selected), and 'other'. There are checkboxes for 'rate' (value 130, highlighted in yellow), 'rhythm', 'quality', and 'checked skin'. The 'checked skin' section has checkboxes for 'color' (cyanotic), 'temperature' (cool), 'condition (moisture)' (diaphoretic), and 'Did a gross blood sweep'. Below the checklist is the 'Patient Status' section with tabs for 'Interventions Status', 'Vitals', 'SAMPLE', and 'OPQRST'. A table lists vital signs and current values, with a 'Requested' button for each: Pain (Unable to access), P (130, weak radial pulses), Spo2 (78 % /RA), R (38, shallow; L5 clear and equal with crepitus on right), ET/CO2 (64 mm Hg), Glucose (86 mg/dl(4.8 mmol / l)), Temp (96.5 F(35.8 C)), Skin (Cyanotic, diaphoretic), and GCS (E-2, V-2, M-4), PERRLA. The right sidebar is titled 'Interventions' and contains a list of intervention categories: Airway, Breathing, Circulation, Transfer to ambulance, Transport, Spinal motion restriction, Manage wounds and specific injuries, Administer Medications, IVs, and Ongoing Assessment and Management Plan.

Fig. 1. MedDbriefer’s observer interface. Assessment checklists and findings at left; intervention checklists at right. Prompts for further detail in italics; current callout highlighted in yellow. (Color figure online)

MedDbriefer is being developed to support this approach to scenario-based practice that focuses on non-technical skills, with one exception: students will be able to use the system on their own, without an instructor. A student who is not part of the EMS team (the session “Observer”) will use MedDbriefer’s checklists to record the team’s stated actions. As shown in Fig. 1, MedDbriefer’s observer interface presents two main checklists: one to record the team’s (stated) assessment actions, the other to record their

(stated) interventions. Interspersed throughout these menus are prompts for the Observer to issue to the EMS team if they fail to provide sufficient detail while voice treating the “patient”: a peer, manikin (if available), doll, or other tangible object. For example, the Circulation menu includes a prompt to the team leader to specify which pulse he is checking (carotid pulse, radial pulse, etc.).

MedDbriefer analyzes the event logs that the peer Observer produces, to provide feedback during and after the scenario. During the scenario, the system provides feedback on the team’s actions: initial findings and updated findings that result from treatment interventions. For example, when the team leader states that he is checking the patient’s pulse and the Observer checks this action, MedDbriefer displays a callout for the Observer to issue (e.g., 130 beats per minute, highlighted in yellow; Fig. 1). A limited simulation feature determines when patient findings should change from the “initial” values pre-specified in the scenario description to their “good” or “bad” values. This decision requires a representation of the interventions, or lack thereof, that would improve or downgrade patient findings, respectively. For example, if the EMS team “performs” interventions necessary to manage shock and then requests a pulse reading to determine if the patient’s condition is improving, MedDbriefer will display a callout of the “good” (or improved) pulse rate specified in the scenario description. Otherwise, MedDbriefer will display the scenario’s “bad” (unchanged or worsened) pulse rate.

After the EMS team completes a scenario, MedDbriefer analyzes the event log to generate an automated debriefing. At this stage of the project, we represent a correct solution for each clinical problem that the simulated patient presents (e.g., hypovolemic shock, an obstructed airway), to enable the system to assess which actions are correct and incorrect in the recorded event log. Each solution is represented as a set of findings that should suggest to the EMS team that the problem needs to be addressed, appropriate interventions to address that problem, interventions that would be contra-indicated according to state EMS protocols, and explanations about why this is the case. Relevant findings, along with partial ordering constraints, suggest when events (both assessment actions and treatment interventions) should take place relative to each other. For example, the team needs to have discovered certain patient findings before they can recognize that a clinical problem exists and begin to manage it. Overall, this analysis executes a limited form of plan recognition. In a later stage of the project, we will add rules to generate solutions automatically.

MedDbriefer uses this analysis to generate appropriate debriefing feedback. For example, if the team leader failed to state that he would ventilate the “patient” using a bag valve mask attached to high flow oxygen, the debriefing will state findings that indicated the need to ventilate and oxygenate the patient, such as slow respiratory rate. If the team leader failed to assess the patient’s respiratory rate, this missed assessment will be included in the feedback.

3 Two Approaches to Automated Debriefings

We are implementing and will compare the effectiveness of two approaches to automated debriefing in a randomized controlled study whose aim is to address the question: *Is it more effective to structure a debriefing by having students step through a chronological*

narrative of their actions during the scenario, with embedded feedback, or by following a standardized debriefing protocol? The former approach is commonly implemented in computer based SBT systems, such as the American Heart Association’s *HeartCode BLS* [10]. Although several simulation researchers and practitioners have advocated the use of protocols to structure human instructor-led debriefings—for example, Gather-Analyze-Summarize (GAS) [15], TeamGAINS [16], and DEBRIEF [11, 12]—there is little empirical evidence to support this practice [5, 11, 12].

As a step towards addressing this gap in simulation-based training research, we will conduct a study to compare a version of MedDbriefer that implements a system-led, narrative approach to automated debriefing with one that implements an adaptation of the DEBRIEF protocol, which stands for: **D**efine the debriefing rules; **E**xplain the learning objectives; specify the performance **B**enchmarks; **R**eview what was supposed to happen; **I**dentify what actually happened; **E**xamine why; and **F**ormalize the “take home” points [11, 12].

The chief difference between these approaches to debriefing lies in the extent to which they engage students in active reflection on, and critiquing of, their performance. In the narrative approach, a human instructor or tutoring system critiques each step of a student’s (or student team’s) solution, as in HeartCode BLS [10]. In contrast, when human facilitators implement protocol-based debriefings—which have not yet been automated—they encourage students to play a more active role. This approach is illustrated by the US military’s implementation of the DEBRIEF protocol for battlefield training [11] and a proposed adaptation of DEBRIEF for simulation-based training in healthcare [12]. Specifically, students are prompted to assess whether their solution met a set of performance standards (“benchmarks”), and then consider why it fell short of meeting certain standards. As such, DEBRIEF affords a more active and interactive approach to post-practice debriefings than does the narrative approach.

We are adapting DEBRIEF for inclusion in MedDbriefer and, ultimately, other healthcare SBT systems. Abundant research demonstrates the superiority of active and interactive approaches to learning and instruction over more passive approaches [17]. This research therefore suggests that the DEBRIEF protocol-based version of MedDbriefer will predict higher learning gains than the narrative version.

The DEBRIEF protocol does not uniformly engage students in active and interactive learning, neither in live implementations of this protocol nor as implemented in MedDbriefer. Its first three components are didactic and realized in MedDbriefer using canned text. During “D”, MedDbriefer states the goals of the debriefing and outlines what students will do to achieve these goals. During the first “E”, the system lists the main learning objectives (“expectations”) that the scenario was designed to achieve—for example, to provide practice with recognizing and managing hypovolemic shock. During “B”, MedDbriefer specifies performance benchmarks for each objective. For example, effective shock management entails administering oxygen at a flow rate of 15 L per minute, administering the correct dosage of fluids intravenously (commensurate with the patient’s weight), and performing several other interventions to spec.

These DEBRIEF components set up a framework for the more active and interactive components that follow. During “R”, MedDbriefer presents a summary of a sample expert solution, which reviews the clinical problems that the EMS team should have

discovered and the interventions they should have performed to manage these problems. Selected terms and phrases contain hyperlinks that allow students to request additional information. For example, selecting the link in, “The team lead recognizes that this patient is in hypovolemic shock”, would display findings that indicate shock.

The next two components (“I” and “E”) lie at the core of student-system interaction and leverage the same analysis of event logs that drive the system’s critique in the narrative version. During “I”, MedDbriefer focuses on each clinical problem that a scenario provides practice with diagnosing and managing, and prompts students to identify which performance benchmarks they met or failed to meet. MedDbriefer compares students’ self-critique with its assessment of the scenario event log, to identify benchmarks that students incorrectly rated as achieved (i.e., “false positives”) and the reverse (i.e., “false negatives”). It then provides feedback to address (truly) missed benchmarks. This feedback is similar in content to that provided on the same errors in the narrative version’s debriefings. During the second “E”, MedDbriefer prompts students to examine and explain why they missed selected priority benchmarks, in their own words. To scaffold this reflective process, MedDbriefer suggests a few likely causes. For example, if students failed to perform any interventions that would indicate they recognized the need to manage shock, the system will suggest: “I forgot to do assessment actions that would have indicated shock” and “I misinterpreted (an) assessment finding(s).” Finally, during “F”, MedDbriefer prompts students to state (formalize) a few lessons learned from the debriefing, also in their own words.

Currently, the observer interface is operational and narrative debriefings are generated. Implementation of the adapted DEBRIEF protocol is partially completed. In an upcoming study to compare these two approaches to automated debriefing in MedDbriefer, students enrolled in the EMS program at the University of Pittsburgh will be randomly assigned to a debriefing condition. Each participant will complete a scenario-based and written pretest and then voice treat the patient presented in four intervention scenarios. These scenarios require identification and management of several clinical problems. For logistical and control purposes, an EMS instructor or expert (not a student) will play the role of Observer, using MedDbriefer to record participants’ stated actions, issue callouts, prompt for further detail, etc. Participants will engage in an automated debriefing after each scenario, structured according to their assigned condition. They will then take a written and scenario-based posttest that is similar in content to the pretest. The pre- and post-tests target the clinical reasoning and decision-making skills exercised in the intervention scenarios.

Acknowledgements. This research is supported by grant IIS 2016018 from the National Science Foundation. The ideas and opinions expressed are those of the authors and do not necessarily represent the views of the NSF. We greatly appreciate the contributions of University of Pittsburgh EMS instructors Stuart Prunty, Samuel Seitz, and Keith Singleton; Karen Kornblum for advice on interface design; NSF Research Experiences for Undergraduates interns Anna Audley, Trenton Husick, Emily Miller, Collin O’Connor, Austin Oldham, and Zachary Smith; student research assistants Priya Gupta, Lily Nong, Odinakachi Ugwanyi, and Alissa Yen; and the EMS students whose participation in simulation-based training sessions that we observed, videotaped, and analyzed informed MedDbriefer’s design and content.

References

1. Friese, G.: AAA study sets a benchmark for turnover in the EMS industry. *EMS1* (2018)
2. Amiry, A.A., Maguire, B.J.: Emergency medical services (EMS) calls during covid-19: early lessons learned for systems planning (a narrative review). *Open Access Emergency Med. OAEM* **13**, 407 (2021)
3. McKenna, K.D., et al.: Simulation use in paramedic education research (SUPER): descriptive study. *Prehosp. Emerg. Care* **19**, 432–440 (2015)
4. Boet, S., et al.: Looking in the mirror: self-debriefing versus instructor debriefing for simulated crises. *Crit. Care Med.* **39**(6), 1377–1381 (2011)
5. Cheng, A., Eppich, W., Sawyer, T., Grant, V.: Debriefing: the state of the art and science in healthcare simulation. In: *Healthcare Simulation Education: Evidence, Theory, and Practice*, pp. 158–164 (2017)
6. Mariani, B., Cantrell, M.A., Meakim, C., Prieto, P., Dreifuerst, K.T.: Structured debriefing and students' clinical judgment abilities in simulation. *Clin. Simul. Nurs.* **9**(5), e147–e155 (2013)
7. Cook, D.A., et al.: Mastery learning for health professionals using technology-enhanced simulation: a systematic review and meta-analysis. *Acad. Med.* **88**(8), 1178–1186 (2013)
8. Issenberg, S., et al.: Features and uses of high-fidelity medical simulations that lead to effective learning: a BEME systematic review. *Med. Teach.* **27**(1), 10–28 (2005)
9. Tannenbaum, S.I., Cerasoli, C.P.: Do team and individual debriefs enhance performance? A meta-analysis. *Hum. Factors* **55**(1), 231–245 (2013)
10. Oermann, M.H., et al.: Advantages and barriers to use of HeartCode BLS with voice advisory manikins for teaching nursing students. *Int. J. Nurs. Educ. Scholarship* **7**(1), article no. 26 (2010). <https://doi.org/10.2202/1548-923X.1949>
11. Sawyer, T., Eppich, W., Brett-Fleegler, M., Grant, V., Cheng, A.: More than one way to debrief: a critical review of healthcare simulation debriefing methods. *Simul. Healthc.* **11**(3), 209–217 (2016)
12. Sawyer, T.L., Deering, S.: Adaptation of the US Army's after-action review for simulation debriefing in healthcare. *Simul. Healthcare* **8**(6), 388–397 (2013)
13. Ebbs, P., Gonzalez, P.: A need to balance technical and non-technical skills. *J. Paramedic Pract. Clin. Monthly Emergency Care Prof.* **11**(3), 98–99 (2019)
14. Von Wyl, T., et al.: Technical and non-technical skills can be reliably assessed during pandemic simulation training. *Acta Anaesthesiol. Scand.* **53**(1), 121–127 (2009)
15. Phrampus, P.E., O'Donnell, J.M.: Debriefing using a structured and supported approach. In: Levine, A.I., DeMaria, S., Schwartz, A.D., Sim, A.J. (eds.) *The comprehensive textbook of healthcare simulation*, pp. 73–84. Springer, New York (2013). https://doi.org/10.1007/978-1-4614-5993-4_6
16. Kolbe, M., et al.: TeamGAINS: a tool for structured debriefings for simulation-based team trainings. *BMJ Qual. Saf.* **22**(7), 541–553 (2013)
17. Chi, M.T., Wylie, R.: The ICAP framework: linking cognitive engagement to active learning outcomes. *Educ. Psychol.* **49**(4), 219–243 (2014)



Ontological Reference Model for Piloting Procedures

Marc-Antoine Courtemanche^(✉), Ange Tato, and Roger Nkambou

Université du Québec à Montréal, Montréal, Canada
courtemanchem.marc-antoine@courrier.uqam.ca,
{nyamen.tato.angea.drienne,nkambou.roger}@uqam.ca

Abstract. Knowledge modeling in the context of an intelligent educational system remains a key research issue. Indeed, intelligent learning systems need a high-quality knowledge formalization of the domain theory related to the context. Ontology-based languages allow a detailed semantic description of knowledge that can support the intelligent system in understanding the domain theory. In this paper, we present an ontology-based approach for modeling complex procedural knowledge related to aircraft piloting procedures. We have developed a reference model for piloting tasks with a strong semantic grounding to the domain theory. The decomposition of a reference procedure in a task ontology with links to the domain ontology for supporting the execution that is highly related to the context will be detailed. This work is a preliminary step to an intelligent tutoring system aims at assisting pilots and at providing useful feedback during piloting task.

Keywords: Ontology · Task ontology · Domain ontology · Intelligent tutoring systems · Piloting procedures · Knowledge representation

1 Introduction

Air travel is of particular interest in terms of quality, speed, security, and comfort when it comes to travel ways. However, in terms of piloting an aircraft, it can be very stressful and difficult for humans behind cockpits [6, 14]. The piloting task is a very challenging and complex domain since the pilot needs to analyze a large set of information while piloting the aircraft. Moreover, the pilot has to execute many difficult tasks taking into account multiple parameters of the execution environment. Having non-human assistant could be a great advantage to support the execution of tasks. It is thus important to develop an intelligent agent that will use domain knowledge and understand how piloting procedures are done, to assist and coach pilots in their tasks.

A complex task is composed of specific knowledge that can be difficult to understand and transfer. To capture the dynamics associated with complex knowledge, we need a formal way to represent it. This formalization has to

The original version of this chapter was revised: The original version did not include the correct acknowledgement. The correction to this chapter is available at https://doi.org/10.1007/978-3-031-09680-8_36

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2022, corrected publication 2022
S. Crossley and E. Popescu (Eds.): ITS 2022, LNCS 13284, pp. 95–104, 2022.
https://doi.org/10.1007/978-3-031-09680-8_9

capture the specific elements in order to ease its manipulation. Our main goal is to formalize the procedures associated with the execution of a flight. Our motivation is to learn the associated actions and tasks with their relationships to the execution environment. A high level of detail is needed in order to permit the automatic manipulation of knowledge. We are approaching this modeling objective through the use of semantic web technologies. From this perspective, we are proposing an ontological knowledge base [4] as a reference model. Specifically, a task ontology will be the base structure for the formalization of our terminological tasks decomposition. This task ontology is the general framework that specifies all the parameters, constraints, and links required for the procedural execution.

Since tasks and sub-tasks refer to knowledge elements related to the environment parameters, the decomposition of the tasks is supported by multiple links to a domain ontology for considering the environment in the resolution process. The domain ontology is a formalization of the domain theory (environment structure, flight parameters, etc.). Mapping a task ontology to a domain ontology is a key issue for providing the semantic grounding for task execution. Hence, by making the resolution process highly related to the execution environment, we are establishing an effective framework for assessing and monitoring the task execution. The proposed solution will be used as the expert knowledge part of a synthetic pilot/intelligent assistant which aims at coaching pilots during flights.

The paper is organised as follow. Section 2 presents a short description of related work on procedural knowledge modeling for educational purposes. Section 3 provides more detail about the structure of the ontological reference model. Section 4 present the task and domain ontologies and and discusses their relationships. The paper end with some concluding remarks and future work associated with the reference model.

2 Task Execution Model: Issues and Related Work

We are taking the knowledge formalization with a specific perspective. At another level, the reference model will be used for monitoring the execution of the tasks by a pilot. With this objective in mind, our model has to be able to automatically execute the task structure in order to support its ability to monitor an external execution. We are seeing this intelligent tutor as a coach that can support the task execution in a simulation environment. The goal of the intelligent system is not to replace the pilot, but rather to assist him in his task.

Our knowledge formalization permits the execution monitoring by the exploitation of the semantic properties specific to the ontologies. We can see the task ontology as a structure containing a set of production rules that are used to evaluate the sequence of execution. For Mitrovic et al. [11] production rules are the procedural knowledge that can be linked to declarative knowledge in order to monitor the execution of a task. Therefore, for each of the problems in the environment, we have a set of production rules supporting the resolution process. These production rules are used in the model and knowledge tracing algorithms.

While the model tracing compares what the student is doing according to the reference model, the knowledge tracing evaluates the performance of the student learning [7].

We are seeing our learning model as a set of production rules that can follow the student execution. The rules can capture the parameters of the execution in comparison with the reference procedure. A complete set of rules are formalized in order to support a wide range of situations. These rules are implemented through tasks graphs which are allowing the choice of the right solution for the problem space. By using the ontology for the formalisation of the task, we are semantically anchoring its execution to the domain theory. Ontology engineering for building an educational system is a way to overcome some problems that can explain the low quality of some automated learning systems. The interest of using an ontology-based structure for the representation of the rules has been demonstrated by Mizoguchi and Bourdeau [12] for supporting factual knowledge. In this work, we are exploiting productions rules based on an ontology for formalizing a knowledge that has the characteristic of being procedural.

In the field of air traffic, some ontologies have been proposed with a different perspectives. Aghdam et al. [2] implemented an ontology for flight safety messages aiming to prevent air accidents, human errors and enrich flight interactions. The final goal of their ontology was to be incorporated into the foundation of an aeronautical telecommunication network (ATN) aviation network. Stefaniadis et al. [16] proposed an aviation domain ontology named ICARUS ontology, which aims at facilitating the semantic description and integration of information resources since in the aviation industry the data are complex and can be derived from heterogeneous data sources. Sheng et al. [15] in contrary, proposed an approach to decision-making processes in Air Traffic Management based on an ontology including concepts and instances of trajectories and meteorology. None of this work used ontologies for actively assisting pilots and providing direct feedback about the task execution based on the reference theory.

3 Structure of the Reference Model

The proposed reference model has multiple parts. It is essential to understand the goal and links of each sub-part of the structure to appreciate the full potential of the proposition. In the end, the reference model is a set of ontological files but several other knowledge representation tools and manipulation methods had to be exploited to reach this final state. Despite the fact that ontology is not frequently used at its full potential, this knowledge representation formalism has many advantages to offer. Concretely, the reference model is based on Web Ontology Language (OWL) [17] for the formalization of the concepts. The advantage of using such a language is in the possibility of representing a knowledge that is logical, and complex with efficient relationships between the elements [18]. The semantic characteristics specific to this language is supporting this high level of detail in the knowledge formalization. For a better understanding of the characteristics of this language, it is important to understand what

is ontology. Several authors have attempted to define the concept of ontology which can be found in multiple variations. Since there are multiple ways to approach the concept, we had to select a definition consistent with our needs. This short and general definition gives a good viewpoint of the general objectives: “An ontology is an explicit specification of a conceptualization” [3]. Since we are approaching the concept with an automated manipulation perspective, this second definition seems to be more specific: “Ontologies are consensus-based controlled vocabularies of terms and relations with associated definitions, which are logically formulated to promote automated reasoning” [9]. Although other definitions are available, we are keeping in mind that ontologies are formal representations of a concept with its associated semantics properties and descriptions of specific knowledge. With this knowledge representation perspective in mind, it is clear that the possibility of interpretation and manipulation by a machine makes ontologies interesting.

Since ontologies have been selected for the representation of the expert knowledge about the procedures for piloting an aircraft, we need a process for capturing this expert knowledge and formalizing it with a high level of detail before translating it to the task ontology. As the first step of the process, we choose to use a visual knowledge representation tool to capture the elements of the procedures which have the characteristic of being dependent on sequential order, time-restricted, and highly related to the dynamic environment. The Unified Modeling Language (UML) has been retained since it is a standardized formal representation language that can capture and support the formalization of the specific characteristics of the knowledge that we are trying to capture. From this standardized language, we choose to use the statechart diagrams that have the ability to capture the dynamic environment with a set of a complex activity workflow sequence [8]. For us, this specific visual representation has the advantage of being solely visual and easily interpreted by an ontologist and an expert pilot. This interpretation by the expert is essential since we are anticipating the need for a validation process that will require a tool that can be easily interpreted and manipulated by someone who has no prior knowledge of ontology. Once knowledge is visually formalized and validated by the experts, the following step is to translate the knowledge visually represented to the task ontology by assertion. This translation is crucial for the quality of the reference model. The final representation has to be rich in detail by describing the characteristics of the knowledge with the functionalities of the OWL language. The reference model is a set of two separated, but highly related ontologies. The first is a task ontology containing the terminological concepts and their relations allowing to structure an execution model. It is the main framework around which all procedures will be built. For supporting the objectives of a task ontology, we selected the following definition: “The term “task ontology” can be interpreted in two ways : (1) Task-subtask decomposition together with task categorization such as diagnosis, scheduling, design, etc. and (2) An ontology for specifying problem-solving process.” [5]. This task ontology will act as the base framework for decomposing the sequence related to the specific knowledge.

More details about the structure of this ontology can be found in Sect. 4. For supporting the task representation, a domain ontology will also be provided. This second ontology aims at representing the environment of the aircraft which is our domain theory integrating the specific knowledge for operating an aircraft. For supporting the goal of the domain ontology with the perspective of using it with a task ontology, we selected this definition: “Domain ontologies describe the vocabulary related to a generic domain, like medicine. Task ontologies describe the vocabulary related to a generic task, like diagnosing or treatment prescription.” [1].

4 Results

The task and domain ontologies are the final versions of the proposed reference model. In this section, we give more details about the structure of both ontologies. As previously stated, the ontologies are formalized with OWL which is an ontology language that permits a rich representation of knowledge.

4.1 The Task Ontology

The task ontology is a central element of the reference model. It is the basic structure around which the other elements of the reference model will be built. This terminological decomposition structure allows linking the information required for the execution of the tasks. In other words, the goal is to identify the characteristics specific to the context of execution and link this information for efficient decomposition of the knowledge that is highly related to the domain theory. Obviously, the objective of a task ontology is different from one context to the other. For this reason, our task execution model is specific to the characteristics of a piloting environment. This definition captures the important aspects of a task ontology: “Task ontology is a system of vocabulary for describing the problem-solving structure of all the existing tasks domain-independently. It is obtained by analyzing task structures of real-world problems. The design of the task ontology is done in order to overcome the shortcomings of generic tasks and half weak methods while preserving their basic philosophies. The ultimate goal of task ontology research includes providing vocabulary necessary and sufficient for building a model of human problem-solving processes.” [13]. Figure 1 is a visual representation of the task ontology with a specific terminology for the piloting environment and characteristics of the procedural knowledge. This terminological set is the T Box containing the terminological axioms.

The general idea behind this ontology is to be able to decompose tasks and sub-tasks with dependency links to capture the knowledge associated with the execution. In other words, each task can have a super-task, a sub-task, a responsible for the execution, a person able to execute the task, a precondition, an execution status, an action, and a constraint. There is a short description of each of the terminological axiom:

- **Task:** this is the main concept of the model. It contains all the necessary tasks for the execution of the procedures.
- **Capability:** since a task can be executed by the pilot or the co-pilot, this concept specifies the capabilities associated with each task.
- **Person:** this concept specifies the individual which in our context it can be the pilot or the co-pilot.
- **Precondition:** assertions of this concept are also members of the concept task. The objective is to integrate a sequence of execution.
- **ExecutionStatus:** this concept contains the status of execution of each task.
- **Action:** it contains the links with the domain ontology specifying the action to be applied to the domain ontology. This execution is specific to each task.
- **Constraint:** since some of the tasks may not be executed before reaching a specific state of the environment, this concept specifies some characteristics with semantic links to the domain ontology.

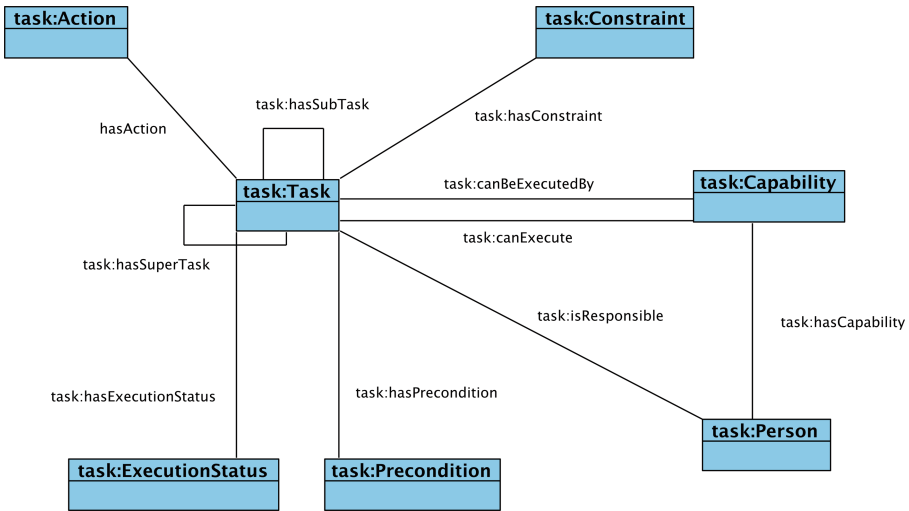


Fig. 1. Task ontology

Before asserting the procedures (A-Box) to the task ontology (T-Box), each procedure and associated sequence are visually represented for supporting a high level of detail of decomposition and a need for validation by the expert. Figure 2 is an excerpt of the visual representation that can be found in the takeoff procedure. Each box is specific to an aircraft state and the arrows are the tasks ID that initiated a change from one state to the other. In this sequential schematization, the tasks ID, actions ID and constraints ID that will later be asserted in the task ontology structure for capturing the dynamic sequence of the knowledge. In Fig. 2, the arrow from the first to the second box has the action ID 1004. Associated to this task, the action ID (increasing thrust to 50%)

allow the aircraft to reach the second state. The sequence of tasks is captured by the precondition which specifies the task that has to be executed before the execution of the next action. In Fig. 2, task ID 1006 has the task with ID 1004 as precondition. In other words, it specifies the sequence of execution. This sequence of execution is exploited by semantic web rules languages (SWRL), a language that introduces a semantic syntax not supported by OWL[10]. Equation 1 can be applied in Fig. 2 for the execution of task ID 1004 and action ID 3.

$$\begin{aligned}
 & \text{task} : \text{Task}(?t) \wedge \text{task} : \text{hasPreconditionPermission}(?t, \text{task} : \text{PreconditionOk}) \\
 & \wedge \text{task} : \text{hasAction}(?t, ?a) \wedge \text{task} : \text{hasActionParameter}(?a, ?ap) \\
 & \wedge \text{task} : \text{hasActionValue}(?a, ?av) \\
 & \rightarrow \text{task} : \text{hasExecutionStatus}(?t, \text{task} : \text{Executed}) \\
 & \wedge \text{task} : \text{hasActionParameter}(\text{task} : \text{LiveExecution}, ?ap) \\
 & \wedge \text{task} : \text{hasActionValue}(\text{task} : \text{LiveExecution}, ?av)
 \end{aligned} \tag{1}$$

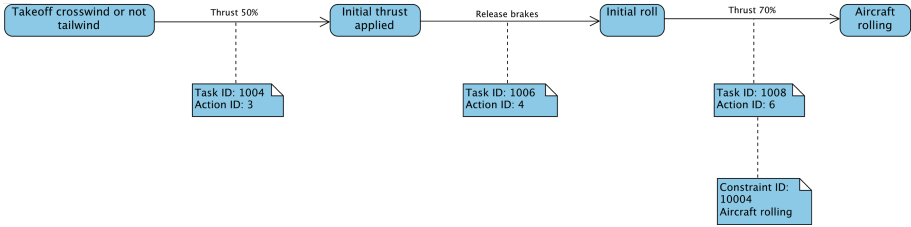


Fig. 2. Excerpt of the takeoff procedure

4.2 The Domain Ontology

The domain ontology aims at representing the knowledge and specific terminology related to the context of the application. This knowledge representation can be considered as a set of concepts related to a specific domain. In our context, the domain ontology is the representation of the piloting environment. The environment is used as a reference by the task ontology. Links from the task ontology support the evaluation of the environment by the task structure in order to integrate the dynamic characteristic specific to the knowledge. The links to the domain ontology support the sequence of tasks previously captured in the statechart diagrams. For supporting the execution shown in Fig. 2, the constraint ID 10004 is directly related to the domain ontology. This constraint limits the execution of task ID 1008 to a positive value of the aircraft ground speed. By adding an environmental constraint and a precondition to the task, its execution is therefore framed by the reference theory. Each constraint is directly linked to the specific parameter in the domain ontology.

Since the domain ontology is the representation of the environment, different terminological categories (concepts) specify some specialization of the general environment. There is a short description of each of the terminological groups:

- **Environment:** this is the main general concept of the domain structure. It is only linking the other subcategories of the environment presented (inside, outside and systems).
- **Aircraft inside environment:** this concept is a specialization of the general environment. The inside environment is considered to be the cockpit with all the associated instruments.
- **Aircraft outside environment:** also considered as a specialization of the general environment, this concept is the formalization of the parameter of the actual environment outside the aircraft. The specialization associated with this environment contains information about the aircraft position, weather conditions, etc.
- **Aircraft systems:** this generalization of the general environment is specific to the aircraft systems that are not found in the cockpit. The engines and other mechanical parts are associated with this concept.

5 Conclusion

In this paper, we presented a reference model for capturing the knowledge associated with piloting procedures. This reference is based on an ontological model that has the capacity to capture and formalize an explicit representation of the knowledge. Our main contribution is the task ontology which is a specific structure able to capture and formalize the task decomposition of specific knowledge related to aviation piloting procedures. For supporting the procedures with the context of execution, a second ontology specific to the domain of aviation has been proposed. This domain ontology is an explicit formalization of the environment in which the pilots operate. Specifically, the environment consists of the inside environment or the cockpit, the outside environment, and the aircraft systems. For capturing the specific and complex knowledge about piloting procedures, we used statechart diagrams to visually represent the procedures. This visual representation allowed to capture the dynamic characteristics of the knowledge in a way humans can easily interpret and manipulate. It is also a good tool for involving domain experts in the validation process of the ontology.

The reference model is the first part of a future work perspective. In fact, it will be integrated (as a procedural memory) into a cognitive intelligent agent that will coach (by giving recommendations) pilots during flights. The very next step is to integrate a manipulation tool to support the automatic execution of the tasks. The semantic property of the reference model is included in this future perspective where the automatic execution will have to integrate the tasks that are sequentially organized and restricted by the constraints. Another work in progress is the extension of the task ontology to introduce the cognitive dimension related to each task. Experimentation is running in order to characterize each action in terms of expected cognitive behaviour and the result will be integrated in the task terminology set. The third perspective is more related to the domain ontology. Since the domain is a representation of the execution environment, at one point we will need to link this environment to a simulation device.

This environment integration has to be possible on multiple platforms from a desktop to more realistic and complete systems. The first implementation will be tested on X Plane and gradually move to the environment that Bombardier and CAE are using. The task ontology is adequately flexible and can be used with different simulation equipment requiring only minor changes. We have a working and solid task and theory model supporting the decomposition of piloting procedures for supporting our future work perspective.

Acknowledgment. We acknowledge the support of CRIAQ, the Natural Sciences and Engineering Research Council of Canada (NSERC), CAE, Bombardier, and BMU.

References

1. Abrahão, E., Hirakawa, A.R.: Task ontology modeling for technical knowledge representation in agriculture field operations domain. In: 2017 Second International Conference on Information Systems Engineering (ICISE), pp. 12–16. IEEE (2017)
2. Yousefzadeh Aghdam, M., Kamel Tabbakh, S.R., Mahdavi Chabok, S.J., kheyrabadi, M.: Ontology generation for flight safety messages in air traffic management. *J. Big Data* **8**(1), 1–21 (2021). <https://doi.org/10.1186/s40537-021-00449-3>
3. Gruber, T.R.: A translation approach to portable ontology specifications. *Knowl. Acquisition* **5**(2), 199–220 (1993)
4. Guarino, N., Oberle, D., Staab, S.: What is an ontology? In: Staab, S., Studer, R. (eds.) *Handbook on Ontologies*. IHIS, pp. 1–17. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-540-92673-3_0
5. Ikeda, M., Seta, K., Kakusho, O., Mizoguchi, R.: Task ontology: Ontology for Building Conceptual Problem Solving Models, pp. 126–133 (1998)
6. Insaurrealde, C.C., Blasch, E.: Uncertainty in avionics analytics ontology for decision-making support. *J. Adv. Inf. Fusion* (2019)
7. Koedinger, K.R., Anderson, J.R., Hadley, W.H., Mark, M.A., et al.: Intelligent tutoring goes to school in the big city. *Int. J. Artif. Intell. Educ.* **8**(1), 30–43 (1997)
8. Larman, C.: *Uml 2 et les design patterns: analyse et conception orientées objet* (2005)
9. Lin, Y., Xiang, Z., He, Y.: Towards a semantic web application: ontology-driven ortholog clustering analysis. In: *Proceedings of the International Conference on Biomedical Ontology*, pp. 33–40 (2011)
10. MacLarty, I., Langevine, L., Bossche, M.V., Ross, P.: Using swrl for rule-driven applications. **9** (2009)
11. Mitrovic, A., Koedinger, K.R., Martin, B.: A comparative analysis of cognitive tutoring and constraint-based modeling. In: Brusilovsky, P., Corbett, A., de Rosis, F. (eds.) *UM 2003*. LNCS (LNAI), vol. 2702, pp. 313–322. Springer, Heidelberg (2003). https://doi.org/10.1007/3-540-44963-9_42
12. Mizoguchi, R., Bourdeau, J.: Using ontological engineering to overcome common AI-ed problems. *J. Artif. Intell. Educ.* **11**, 107–121 (2000)
13. Mizoguchi, R., Vanwelkenhuysen, J., Ikeda, M.: Task ontology for reuse of problem solving knowledge. *Towards Very Large Knowl. Bases Knowl. Build. knowl. Sharing*, **46**(59), 45 (1995)

14. Perry, D.H., Burnham, J.: A flight simulation study of difficulties in piloting large jet transport aircraft through severe atmospheric disturbances (1967)
15. Sheng, Y., Chen, X., Mo, H., Chen, X., Zhang, Y.: An ontology for decision-making support in air traffic management. In: Liang, Q., Wang, W., Mu, J., Liu, X., Na, Z., Chen, B. (eds.) *Artificial Intelligence in China*. LNEE, vol. 572, pp. 458–466. Springer, Singapore (2020). https://doi.org/10.1007/978-981-15-0187-6_55
16. Stefanidis, D., et al.: The icarus ontology: a general aviation ontology developed using a multi-layer approach. In: *Proceedings of the 10th International Conference on Web Intelligence, Mining and Semantics*, pp. 21–32 (2020)
17. W3C: Owl 2 web ontology language : structural specification and functional-style syntax (second edition) (2012)
18. W3C: Web ontology language (owl) (2012)



Towards Adaptive Coaching in Piloting Tasks: Learning Pilots' Behavioral Profiles from Flight Data

Ange Tato, Roger Nkambou^(✉), and Gabrielle Joyce Nana Tato

Université du Québec à Montréal, Montréal, Canada
{nyamen_tato.angea_drienne,nkambou.roger}@uqam.ca

Abstract. The use of machine learning techniques for safety analysis, incident and accident investigation, fault detection and also for the study of piloting behaviors has gained popularity within the aviation community. This paper present a methodology that uses machine learning for the analysis of piloting behaviors in order to highlight behavioral profiles of piloting (pilot gain) going beyond the norm 'high gain/low gain. Moreover pilot profiles are learned not from the whole data, but from meaningful flight segments to better characterize the change in behaviour style during a flight. These profiles can then be referred to experts in the field for their use in several operational frameworks (piloting assistance, training in a piloting task, monitoring of a piloting activity, etc.).

Keywords: User modeling · Machine learning · Aircraft dynamics · Piloting behaviors

1 Introduction

Experienced flight test engineers will tell you that when it comes to testing pilots, there are two types: low gain and high gain. Typically, the low-gain pilot makes inputs that are considered smoother, and usually at smaller amplitudes than the high-gain pilot. Every seasoned flight test professional is able to identify test pilots who are considered high-gain or low-gain pilots [12]. In technical terms, the gain is what engineers call the ratio of an error response. Pilot gain describes the level of aggressiveness in pilot control activity. It depends on training, aircraft dynamics, the task at hand, stress level, and also individual temperament [6].

One interesting question is how can we automatically identify high and low-gain pilots to ensure a flight test program? What parameters should be used to categorize a driver? Can we find categorizations other than high gain/low gain? Knowing that data (state of the aircraft, the flight path, the immediate environment around the aircraft, the weather and terrain information, and the pilots' input to control the aircraft, even additional sensors such as eye tracking devices and biological monitor can also be added to determine the condition of

The original version of this chapter was revised: The original version did not include the correct acknowledgement. The correction to this chapter is available at https://doi.org/10.1007/978-3-031-09680-8_36

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2022, corrected publication 2022
S. Crossley and E. Popescu (Eds.): ITS 2022, LNCS 13284, pp. 105–114, 2022.
https://doi.org/10.1007/978-3-031-09680-8_10

pilots) can inform us on new different categorizations that could also be useful. This paper presents unsupervised machine learning algorithms powered by flight data provided by one of the civil aviation main actors (CAE¹) which first tries to highlight the most important parameters in the categorization of pilots. Then, to explore the potential existence of a more advanced categorization than the existing one (high gain/low gain) by developing an operational model for the classification of behaviors in piloting tasks. This classification will help in the understanding of pilots' behaviours in general (e.g. simulate a specific profile in different conditions to evaluate how one pilot will act on the same conditions). It will also help the development of a future intelligent assistant to best coach a pilot during a piloting task.

2 Related Work

The field of machine learning is largely benefiting from the availability of computing power and a large volume of collected data. They have been widely used in the air traffic domain mainly for detecting abnormal events and operations during flights. In this sense, Li et al. [10], instead of using predefined criteria to identify risks, they used a cluster-based anomaly detection to detect abnormal flights from flight data (365 B777 flights and 25,519 A320 flights), which can support domain experts in detecting anomalies and associated risks from routine airline operations. Sheridan et al. [17] did the same but by using Density-Based Spatial Clustering of Applications with Noise (DBSCAN) for the detection of anomalous flights. On the contrary Fernandez et al. [3] performed a descriptive analysis that employed clustering techniques to aid in extracting patterns and correlations within data and also in identifying clusters of similar observations.

None of the above mentioned work presented focused on the study of the pilot itself. The ability to learn and forecast pilot profile/behaviour within a cockpit as well as taking automatic action when necessary could become a powerful tool for aviation as well [4]. This research present a methodology to identify the most significant parameters from flight data for a useful classification of pilot profile (skill, style, aptitude, etc.). Few studies have been conducted with the goal of extracting pilot behaviour from flight data. Camille Bodin in her work identifies using machine learning techniques (logistic regression, support vector machines and neural networks, classification) flight maneuvers from real flight test data. These extracted maneuvers are labeled in terms of actions performed (rolling roll, 360° rolls, etc.) not behavioral profiles [1]. “Numerical measurement of pilot gain has always been a kind of dark art.” [5]. A study summarizes an approach for the validation of several potential pilot gain measurements in the time and frequency domain based on pilot models and associated results. The validation is based on data from a simulator study that was performed with 12 experimental test pilots and 12 operational pilots who varied their pilot gain/aggressiveness on command during a closed-loop tracking task [13]. This study presents potential measures

¹ CAE is the major training partner of aviation professionals, airlines, large fleet operators, and aircraft manufacturers the world over. It has the largest civil aviation network in the world.

of pilot gain which are grouped into three different classes: time domain-based measurements, frequency domain-based measurements, and pilot model-based measurements. These methods may require specially designed tasks, while in flight testing this additional load may not be accepted if its sole purpose is the determination of the pilot's gain. Garcia Lorca et al. [4] proposed a characterization of pilot profiles through non-parametric classification of flight data, using kmeans clustering technique. They used data from takeoffs and landings for feature identification then classified in pairs for varying number of clustering centroids representing different profiles. The extracted flight profiles separated data from each flight, based on specifics score similarity.

Few studies have focused on pilot skill level extraction and prediction (expert vs novice) such as the work proposed by Nittala et al. [14]. They proposed a system to predict the skill level and workload of pilots using machine learning algorithms such as support vector machine (SVM), logistic regression with LASSO (L1), etc. Pilot's heart rate variability and flight control data including pilot inputs such as throttle and aileron from 15 pilots each flying the dame 5 pre-defined routes on X-plane were used as input data. Their results indicate that the flight control data alone are sufficient to provide a near-perfect classification of a pilot's skill level into expert or novice. Kasarskis et al. [8] and Wiggins et al. [19] on the contrary, propose a study where they compare expert and novice pilots behaviors/performance during simulated flight. They used flight data as well as eye-tracking data. Dideriksen et al. [2] proposed a Deep Neural Networks (DDN) to predict pilot student performance during flight. They used data from 30 pilots flying multiple flight maneuvers in a simulator and live jet and over 50+h of live flight (approximately 1.2 million data points) and recorded the cognitive state of the same pilots.

Pilot dynamics research has shown that a pilot can change behavioral strategy when faced with a change in aircraft dynamics, resulting in different values of the pilot's "gain" [9]. It is therefore very important to take into account the parameters external to the piloting actions (environmental parameters, parameters related to the aircraft, etc.) in order to best determine the pilot gain and to optimally help the test community in flight. As far as we know, none of the existing work proposed the automatic extraction of profiles based on flight data that is divided into flight segments to better address the change in behavioral strategy. Moreover, the extracted profiles are not predefined categories but rather new categories found from unsupervised learning.

3 Methodology

The study focused on the take-off task, a choice suggested by the experts given the critical nature of this phase in a flight and also the magnitude of the data. The idea is of course to eventually reach a model that covers a complete flight. A result of a concerted effort with experts in the field makes it possible to highlight the segments of a take-off task, which are: verification of thrust symmetry, take-off thrust, take-off for the climb phase, flap retraction, etc.

The data (telemetry) used in our work is collected from CAE Level D full flight simulators, and are takeoffs (normal and abnormal) of approximately 90 pilots. The data included plane orientation, speed, pilot inputs, contextual data, etc. The flight data (which are sequential) is associated with a temporal index (the tenth of a second) representing the variation of the parameters over time. We present the steps of our methodology to extract meaningful pilots' profiles from sequential data.

Step 1: Preprocessing

The preprocessing step of the methodology involves:

- Parameter selection down
It is the process of removing the less important parameters (parameters with low variances) and keeping only those relevant for the development of the pilot profiling model. The selection made it possible to keep 125 parameters out of the 195 that we had in the raw data.
- Standardization
Variables measured at different scales do not contribute equally to model fit and model learning function. Thus, to deal with this potential problem, standardization is generally used before fitting the model, which is our case. The raw data has been standardized using the scikit-learn python library.

Step 2: Dimension reduction + data encoding

Given the volume and the sequential aspect of data, exploring, identifying behavioral groups, and understanding the relationships between functionality becomes difficult. To address these challenges, the dimensionality of the data must be reduced to minimize the number of variables considered random. There are several techniques for dimensionality reduction such as PCA (Principal Components Analysis), FDA (Fisher Discriminant Analysis), t-SNE, etc. For time-series (sequential) data reduction, the recurrent neural network (RNN) architecture and its long short term memory (LSTM) variant have been shown to be more accurate than statistical methods [16].

The technique that we used is therefore LSTM Autoencoders (implementation of an auto-encoder for sequential data) [18]. In a certain sense, this technique forces the model to derive the most important features from sequential data using as few parameters as possible. It takes into account the inner sequential structure of the data to extract the most interesting features. Since the original K-means does not take as input sequential data, it is important to encode sequential data to non-sequential data before passing it to the K-means algorithm. This is indeed the goal of the LSTM-Autoencoder: transform and reduce sequential data to non-sequential data without losing much information.

Step 3: Clustering

“Clustering is the process of grouping similar objects into different groups, or more precisely, partitioning a data set into a subset that corresponds to a defined measure of distance” [15]. Cluster analysis is a commonly used data-mining technique to identify common patterns in a dataset. The technique that we used for clustering is the K-means algorithm. The goal of this algorithm is to find clusters in the data, with the number of clusters represented by the variable K . The algorithm works repetitively to assign each data point to one of K clusters based on the features provided. Compared to other algorithms, it improves classification accuracy and ensures that information about a particular problem domain is available. It also deals with fewer parameters, construct a model that is interpretable and works well on fewer data compared to neural networks techniques which could have also been used in this context for clustering. The results of the k-means clustering algorithm on the data are:

- The centroids (average piloting behavior) of the clusters (2 to 3 clusters or groups depending on the experiment were found after evaluation using the elbow technique [11]), which can be used to label new data.
- Labels for training data (each data point is assigned to a single cluster).

Step 4: Post-processing

Once the clusters have been obtained, this step consists of analyzing and validating them. It is indeed a task of giving labels to each group according to the piloting behaviors observed. Work with the experts is on-going with the aim of analyzing the values of parameters extracted for each groups. The final goal being to interpret each cluster of pilots and labeling them from a human view.

4 Experiment 1: General Clustering

This first experiment is performed on data obtained from routine airline operations. These data consist of 90 take-offs with 195 parameters collected including 67 takeoffs carried out under normal conditions and 23 with engine failures. This experiment consists of highlighting the different piloting profiles in a general way, that is to say taking into account flights with and without breakdowns, all the flight parameters without distinction of flight segments (state of the aircraft, entry drivers, context). The goal was to verify if we were first able to extract the well-known low and high gain profiles, and second to compare if the profiles extracted from the whole flight are different from the ones extracted from flight segments.

Prepossessing : The analysis of data shows that takeoffs are recorded over intervals ranging from 34 s to 294 s. Equal time series (meaning that the data must be recorded over an equal time interval) is a requirement for data to be sent to an LSTM Auto-encoder. Therefore, the study was carried out on 3 scenarios:

- Minimum interval (34s): we adjust all the data so that all the pilot has 34s of take-off time. We are aware that in this first scenario, we lose a lot of information which can lead to insignificant results.
- Maximum interval (294s): we adjust all the data so that all the pilots have 294s of take-off time. We are aware that in this second scenario, we introduce a lot of noise in the data since for pilots that did not reach the 294s, we padded (with 0) their data.
- Median interval (91s): this scenario is the less 'aggressive one' and also the more intuitive one since we cut out and padded a few data that means less data loss and noise.

4.1 Findings

The dimensionality reduction was not possible in the second scenario (timestamp = 294S) due to the lack of information. This case was not considered in the final solution. We find that the lower the learning rate, the greater the losses. We will use lower learning rates for dimension reduction. For better learning it is important to have the training epochs neither too large (overfitting) nor too small (underfitting) [7]. We opted to keep the number of epochs at 300.

Several experiments were performed on the flight data set. The optimal configuration in our case for data reduction is obtained with the first scenario: timestamp = 34 s, epochs = 300 epochs and learning rate = 0.00001. This data encoding makes it possible to initiate the notion of clusters (groups or control profiles).

Step 3: Clustering The experiment brought out three groups of pilots. We are particularly interested in the action of the pilot on the rudder pedals to maintain the heading of the aircraft on the axis of the runway for different clusters. Pilots in the first and the second extracted clusters (Fig. 1) will oscillate less (low gain) than those in the third cluster (high gain) (Fig. 2). However, we keep in mind that the results are only based on the first 34s of a takeoff task and that some pilots could not fit in those clusters as they could spend more time on this task. Also, taking a fixed time to cluster pilots can be misleading, since some actions are not done at the same time and with the same amount of time. This is the reason why we opted for a solution with a takeoff task that has been segmented.

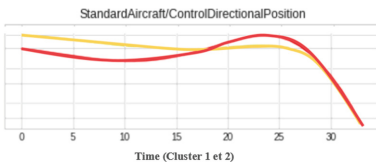


Fig. 1. Cluster 1 & 2

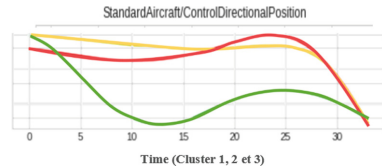


Fig. 2. Cluster 1,2 & 3

5 Experiment 2: Clustering According to Flight Segments

A concerted effort with experts in the field leads to the conclusion that in reality, different profiles are observed according to the different phases of flight. It becomes more interesting to highlight the pilot profiles according to the flight phases. This experiment focuses on the so-called rotation phase. We will only display the results of this phase which lasts about 13s in view of the data we have at our disposal.

This phase begins a few seconds before the indicated speed reaches the speed VR (VspeedVr). When the indicated airspeed reaches VR, the pilot will pull the side stick (pitch stick) in such a way as to achieve a pitch rate (StandardAircraft/PitchRate) of $3^\circ\text{C}/\text{s}$ towards a pitch angle (StandardAircraft/PitchAngle) of 15°C (maximum) this phase ends as soon as the aircraft leaves the ground. The idea here is to observe the pilot behavior on the pitch stick to achieve a certain pitch rate towards a pitch angle

5.1 Results

We have extracted 2 profiles from this phase:

- group 1 (see Fig. 3): this cluster tends to represent low gain pilots because once the indicated airspeed reaches VR speed, the pitch stick is handled a little less aggressively (slight oscillation). We also note that the parameters have a slightly higher value compared to the second cluster. This second observation needs further analysis by experts to help characterize this behavior. The least visible lines represent each of the pilot entries that are placed in that group and the most visible line represents the centroid.
- group 2 (see Fig. 4): Unlike pilots in the first group, pilots in this group tend to exert more pressure on the side stick (pitch stick) and lower parameter values compared to the first group. Figure 5 shows a clearer distinction between the two groups in their way of manipulating the side stick (pitch stick).

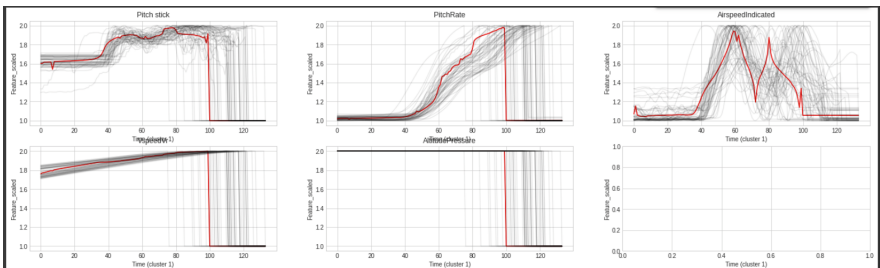


Fig. 3. Group 1 on the rotation phase. Five parameters were selected for this phase. The x-axis represents the time in ms and the y-axis represents the parameter value.

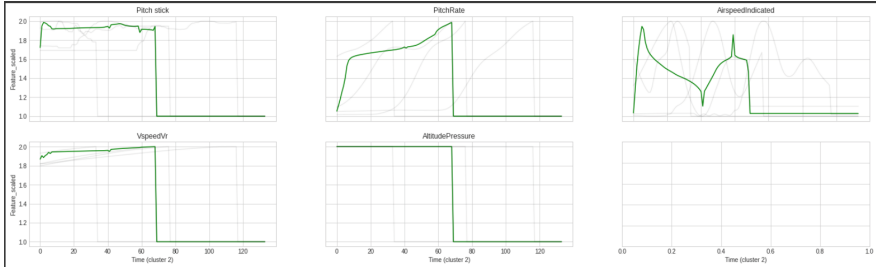


Fig. 4. Group 2 on the rotation phase. Five parameters were selected for this segment. The x-axis represents the time in ms and the y-axis represents the parameter value.

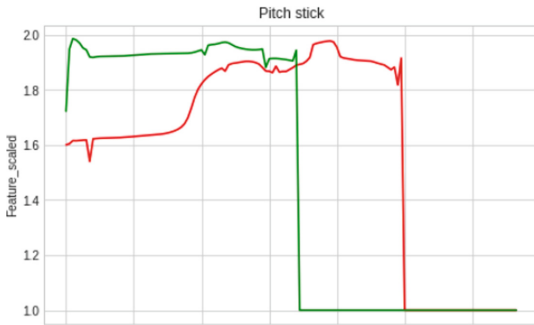


Fig. 5. Difference between cluster 1 and cluster 2: Pitch stick

6 Conclusion and Future Work

In this paper, we have developed and used an automatic learning methodology for the analysis of piloting behaviors in order to highlight behavioral profiles of piloting. The model developed was tested on flight data provided by one of our partners (CAE). The first test was carried out on a data set of 90 A320 take-offs taking into account all the parameters (state of the aircraft, pilots, etc.). The results showed that the proposed method was certainly capable of highlighting categories of pilots on this data set, but with results that were not too relevant since the takeoffs were not all carried out over the same number of times. The second test was carried out on a data set of 67 A320 takeoffs (taking place under normal conditions). In this test, the profiling of pilots was based on the notion of flight segment, which makes it possible to have a very precise time interval and target parameters to be interested in, in particular the Procedure of the pitch stick to counter the nose-up effect of the take-off thrust adjustment. This approach is a more interesting and relevant for future tasks. Implementing the methodology on the data set reveals various interesting clusters within the analyzed data. It shows 2 behavioral profiles of piloting which represent in a way the level of aggressiveness during the intervention of pilots (control of direction).

The brand new step of this study consists in giving a meaningful label to each extracted cluster for each phase of flight. This will be done with the help of experts in the field. The learned profiles are intended to be used for simulation of the specific type of pilot in different conditions (this will help the liberalization stage) and to develop a smart assistant that will coach and give recommendations to pilots based on their current behavior style. We will also extend this work to other flight phases as well as to other types of aircraft. This will increase the probability of discovering new interesting profiles.

Acknowledgment. We acknowledge the support of CRIAQ, the Natural Sciences and Engineering Research Council of Canada (NSERC), CAE, Bombardier, and BMU.



References

1. Bodin, C.: Automatic flight maneuver identification using machine learning methods (2020)
2. Dideriksen, A., Williams, J.: Machine learning and physiological metrics enhance performance assessment. In: Naweed, A., Bowditch, L., Sprick, C. (eds.) ASC 2019. CCIS, vol. 1067, pp. 129–138. Springer, Singapore (2019). https://doi.org/10.1007/978-981-32-9582-7_10
3. Fernández, A., et al.: Flight data monitoring (FDM) unknown hazards detection during approach phase using clustering techniques and autoencoders. In: Proceedings of the Ninth SESAR Innovation Days, Athens, Greece, pp. 2–5 (2019)
4. Lorca, F.G., Gururajan, S., Belt, S.: Characterization of pilot profiles through non-parametric classification of flight data. In: AIAA Information Systems-AIAA Infotech@ Aerospace, p. 0914 (2017)
5. WR Gray. A boundary avoidance tracking flight test technique for performance and workload assessment. In Proceedings of the 38th Symposium of Society of Experimental Test Pilots, San Diego (2007)
6. Hess, R.A.: Simplified approach for modelling pilot pursuit control behaviour in multi-loop flight control tasks. Proc. Instit. Mech. Eng. Part J. Aerosp. Eng. **220**(2), 85–102 (2006)
7. Jabbar, H., Khan, R.Z.: Methods to avoid over-fitting and under-fitting in supervised machine learning (comparative study). Comput. Sci. Commun. Instrum. Dev. **70** (2015)
8. Kasarskis, P., Stehwien, J., Hickox, J., Aretz, A., Wickens, C.: Comparison of expert and novice scan behaviors during VFR flight. In: Proceedings of the 11th International Symposium on Aviation Psychology, vol. 6. Citeseer (2001)
9. Klyde, D., Brenner, M., Thompson, P.: Wavelet-based time-varying human operator models. In: AIAA Atmospheric Flight Mechanics Conference and Exhibit, p. 4009 (2001)
10. Li, L., Das, S., John Hansman, R., Palacios, R., Srivastava, A.N.: Analysis of flight data using clustering techniques for detecting abnormal operations. J. Aerosp. Inf. Syst. **12**(9), 587–598 (2015)
11. Marutho, D., Handaka, S.H., Wijaya, E., et al. The determination of cluster number at k-mean using elbow method and purity evaluation on headline news. In: 2018 International Seminar on Application for Technology of Information and Communication, pp. 533–538. IEEE (2018)

12. Mitchell, D.G., Klyde, D.H.: Defining pilot gain. *J. Guidance Control Dyn.* **43**(1), 85–95 (2020)
13. Niewind, I.: A new approach for the validation of potential pilot gain measures. In: *EuroGNC 2013, 2nd CEAS Specialist Conference on Guidance, Navigation and Control* (2013)
14. Nittala, S.K.R.: Pilot skill level and workload prediction for sliding-scale autonomy. In: *2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pp. 1166–1173. IEEE (2018)
15. Omran, M.G., Engelbrecht, A.P., Salman, A.: An overview of clustering methods. *Intell. Data Anal.* **11**(6), 583–605 (2007)
16. Sagheer, A., Kotb, M.: Unsupervised pre-training of a deep LSTM-based stacked autoencoder for multivariate time series forecasting problems. *Sci. Rep.* **9**(1), 1–16 (2019)
17. Sheridan, K., et al.: An application of dbscan clustering for flight anomaly detection during the approach phase. In: *AIAA Scitech 2020 Forum*, p. 1851 (2020)
18. Srivastava, N., Mansimov, E., Salakhudinov, R.: Unsupervised learning of video representations using lstms. In: *International Conference on Machine Learning*, pp. 843–852. PMLR (2015)
19. Wiggins, M., Stevens, C., Howard, A., Henley, I., O’Hare, D.: Expert, intermediate and novice performance during simulated pre-flight decision-making. *Aust. J. Psychol.* **54**(3), 162–167 (2002)



LORD: A Moodle Plug-in Helps to Find the Relations Among Learning Objects

Rita Kuo¹ , Radomir Wasowski², Ted Krahn², and Maiga Chang² 

¹ New Mexico Institute of Mining and Technology, Socorro, NM, USA

rita.mcs1@gmail.com

² Athabasca University, Edmonton, AB, Canada

wasowski@ualberta.ca, tedkrahn@gmail.com, maiga.chang@gmail.com

Abstract. Learning Management System (LMS) is widely used in higher education. Researchers have proposed methods to analyze the relations among learning objects (i.e., re-sources/activities) of a course in the LMS and then construct the graph structure for the learning objects. Student's learning behaviour in the LMS can be represented and analysed in graph, i.e., the Learning Object Graph (LOG). With the LOGs represent different students' learning behaviours, plug-in is designed to cluster students into groups based on their learning behaviours. Such method requires the relations among learning objects can be identified and measured accurate and properly. This research explains how the LORD (Learning Object Relation Discovery) Moodle plug-in measures the similarity between two learning objects, with the help of WordNet and Natural Language Processing, according to their content in English, French and Hindi to create a more reasonable and objective Learning Object Graph (LOG) that can be used to represent students' sequential behaviours among learning objects.

Keywords: Behaviour analysis · WordNet · Semantic similarity · Munkre's Assignment Algorithm · Visualization · Learning path

1 Introduction

Learning analytics is an Educational Technology research area that focuses on analyzing the data about learners and their context in order to optimize learning and the corresponding environments [3]. Researchers adopt learning analytics systems to predict students' performance, such as retention and dropout in the course, the completion of the course, etc. [4]. Most of the learning analytics are implemented on the learning management systems [6].

The similarity calculation between learning objects is widely used in the recommender systems in the learning analytics research. To tell learners which course or learning materials is best for them according to their interests, the recommender systems usually determine the similarity between learning objects (content-based approach) or between the selections of learning objects by students (collaborative-filtering approach) [1]. Researchers have designed a Behaviour Analytics Moodle Plug-in [8] to analyze students' learning behaviours on Moodle that uses the collaborative-filtering approach to cluster students in groups based on their past learning behaviour.

The Behaviour Analytics clustering research strongly relies on the Learning Object Graph (LOG) that represents all the relations among learning objects and requires teachers to adjust the LOG by themselves based on their perceptions toward the various learning objects designed in their course. This research considers reducing teachers' workload by providing them a pre-analyzed LOG according to the content-based similarity calculation results of any two given learning objects.

Section 2 reviews the existing Behaviour Analytics Moodle Plug-in research as well as the text similarity calculation methods. Section 3 explains the learning object similarity calculation method designed and proposed by this research. The Moodle plug-in that implements the proposed method is introduced in Sect. 4. Section 5 reveals the evaluation plan and summarizes the work done.

2 Research Background

2.1 Behaviour Analytics Moodle Plug-in

The Behaviour Analytics (BA) Moodle Plug-in [8] is a graph-based student behaviour representation and analysis tool in Moodle. As Fig. 1 shows, the plug-in first analyzes the learning resources/activities on Moodle and constructs a Learning Object Graph (LOG) according to the structure of learning resources/activities managed by the teachers. Next, students' interactions on learning objects will be retrieved by the plug-in and the students' behaviour graphs will then be generated; the centroid of each graph is determined by using centroid decomposition [5, 13]. Teachers are able to group students based on students' behaviour graphs with the built-in k-mean algorithms [9] that the plug-in has and understand students' behaviour patterns via the plug-in.

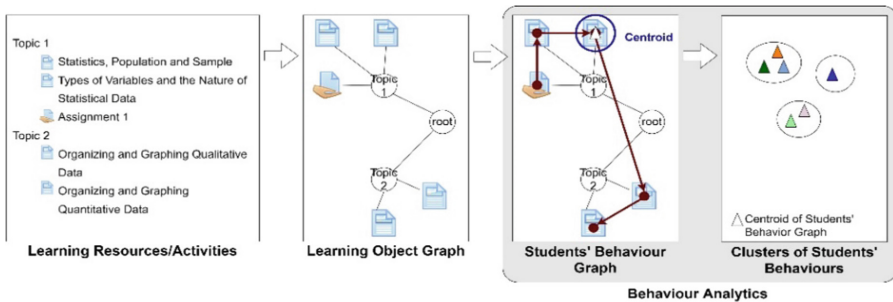


Fig. 1. The system flow of the Behaviour Analytics Moodle Plug-in [8]

The Learning Object Graph is integrated in the Behaviour Analytics Moodle Plug-in in the past study [8]. Through the plug-in, the teachers are able to display students' behaviour as Fig. 2(a) shows. The plug-in can also calculate the centroids of students' behaviour graphs as the triangles in Fig. 2(b). The centroids are used to cluster students into groups as Fig. 2(c) shows. Teachers can use the information to understand students' behaviour patterns and deliver different feedback to students in each group.

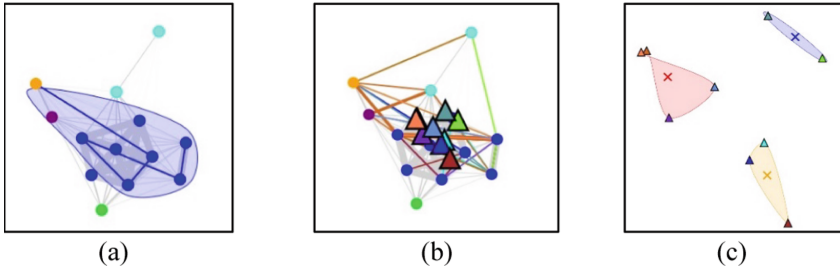


Fig. 2. The screenshots of the Behaviour Analytics Moodle Plug-in: (a) generating the students' behaviour graph; (b) finding the centroids of the students' behaviour graphs; and (c) clustering students in group according to the centroids of students' behaviour graphs.

However, the learning resources/activities graph structure is based on the section organization in the Moodle course. If there are no sections organizing the resources/activities, the materials will be formed as a meaningless one-level tree structure. Moreover, the teachers need to spend a lot of time and efforts to pre-arrange a LOG through reviewing the learning objects designed in their courses before they can run the student clustering function. In order to reduce teachers' burden, this research proposes a method that determines the similarity between any given two learning resources/activities with Natural Language Processing techniques and implements the Learning Object Relation Discovery (LORD) Moodle plug-in as a support package to the BA Moodle Plug-in. The BA Moodle Plug-in could use LORD to construct a Learning Object Graph (LOG) that teachers might consider to be reasonable stand ground for reaching to the final LOG they can use for clustering students.

2.2 Semantic Similarity

Semantic similarity can be used in the content-based similarity measure by identifying the shared information between two concepts [11]. String-based, corpus-based, and knowledge-based are the three major approaches in semantic similarity research [14]. Jaccard Similarity, Levenshtein distance, and n-gram are the common methods in the string-based method. Corpus-based similarity usually checks words' co-occurrence to measure the similarity between words; the meanings of the words are ignored. On the contrary, the knowledge-based approach measures word's similarity according to the semantic information in the knowledge representation, such as *WordNet*.

WordNet groups the synonyms in a synonym set, or a *synset* [10]. A short definition of the synset and the usage example is stored with the synset. Moreover, the synsets are connected with each other based on the semantic relations, such as hyponymy and hypernym, meronymy and holonymy etc. Many studies use *WordNet* to determine the similarity among words. For example, Sheeba and Krishnan [12] analyzed learners' interests with semantic-based representation of *WordNet* based on their frequently used documents.

3 Similarity Calculation

In order to determine the semantic similarity between learning resources/activities, this research designs a Word & Sentence Natural Language Processing (WS-NLP) Similarity Service that uses *WordNet* lexical database as the knowledge graph to calculate the similarity between words, sentences, paragraphs, and documents. Figure 3 shows the workflow of the WS-NLP Similarity Service.

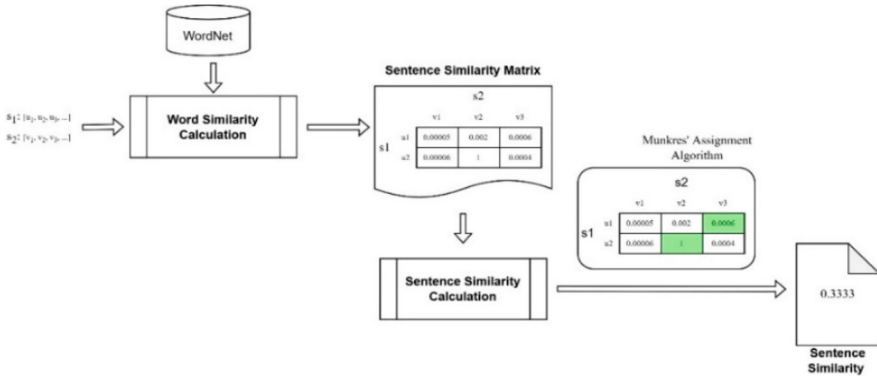


Fig. 3. The workflow of the WS-NLP similarity service

If two synsets have closer distances, the synsets have higher semantic similarity. Take *dog*, *corgi*, and *bear* in Fig. 4 for example, there is only one edge-distance between *dog* and *corgi*, but the distance between *dog* and *bear* is 3-edge-distance. The result shows that *dog* and *corgi* have higher semantic similarity than *dog* and *bear*. It indicates that the similarity is the reciprocal for the edge-distance between two synsets.

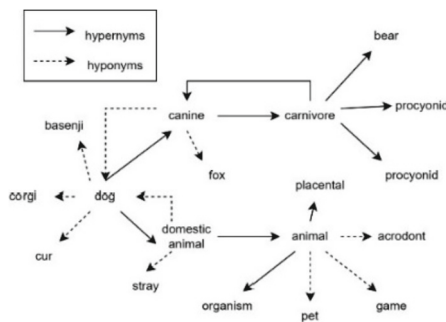


Fig. 4. An example of synsets and their relations in the WordNet

The similarity service first looks for which synset a given word belongs to in the WordNet. With the identified two synsets, the Word Similarity Calculation module in Fig. 3 uses Uniform-Cost Search [2] to traverse the synsets in WordNet to find the

shortest path. After the shortest path is found, the similarity between two words is the reciprocal of the edge-difference of the found shortest path. Take *dog* and *bear* in Fig. 4 for example. The shortest path between two synsets is:

dog → *canine* → *carnivore* → *bear*.

The edge-difference from *dog* to *bear* in the shortest path is 3. Therefore, the similarity of *dog* and *bear* is $1/(3 + 1) = 0.25$.

Figure 5 shows a matrix that represents the similarity between words in two sentences are calculated. The matrix is then sent to the Sentence Similarity Calculation module (see Fig. 4) to determine the similarity between two sentences with Munkre’s Assignment Algorithm [7] – a combinational optimization algorithm to find the optimal pairing between two sets, to match the most similar words in two sentences.

		S ₂		
		static	fields	methods
S ₁	constant	0.00005	0.002	0.0006
	fields	0.00006	1	0.0004

Fig. 5. The sentence similarity matrix and the matching (in green) is calculated by the Munkre’s Assignment Algorithm (Color figure online)

Take sentence s_1 : “constant fields” and sentence s_2 “static fields and methods” as examples, the Word Similarity Calculation module ignores the word “and” in s_2 and generate the Sentence Similarity Matrix (see Fig. 5). The Sentence Similarity Calculation module applies the Munkre’s Assignment Algorithm to find the two best matchings between words in individual sentences: “constant” in s_1 and “methods” in s_2 as well as “field” in s_1 and “field” in s_2 . Because the number of words in two sentences are not even, the remaining words (e.g., “static” in s_2) will not be matching to any other words.

The Sentence Similarity Calculation module calculates the average similarity of matchings to determine the sentence similarity. Following the example above, the maximum number of words in the two sentences is three (in s_2); therefore, the similarity between the two sentences is

$$\frac{1 + 0.0006 + 0}{3} = 0.3335.$$

The similarity service uses the same way to calculate the similarity between paragraphs and even articles.

4 Learning Object Relation Discovery Moodle Plug-in

Instead of creating the graph only based on the learning object structure, the research develops the Learning Object Relation Discovery (LORD) Moodle Plug-in that adopts the WS-NLP Similarity Service to determine the distance among learning objects and generate a Learning Object Graph based on the content analysis results. When a course

has the LORD Moodle plug-in installed and enabled, the LORD block can be seen as Fig. 6 shows. The block summarized how many learning activities and connections among the learning activities exist in the course. The teachers can click the “View graph” link in the block to check the relations between learning activities.

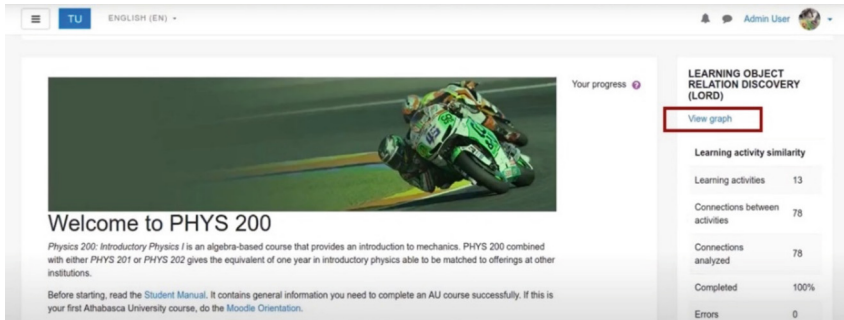


Fig. 6. The screenshot of the LORD Moodle plug-in

Figure 7 shows the interface after teachers clicking the “View graph” link in the LORD block. When teachers click the “Regenerate graph” button, the LORD will generate the Learning Object Graph based on the calculation results of the similarity among the learning objects. If the checkbox “Allow changes?” is checked, then the teachers are able to further drag and drop any nodes on the graph if they believe the relations among the nodes are inappropriately found by the LORD.

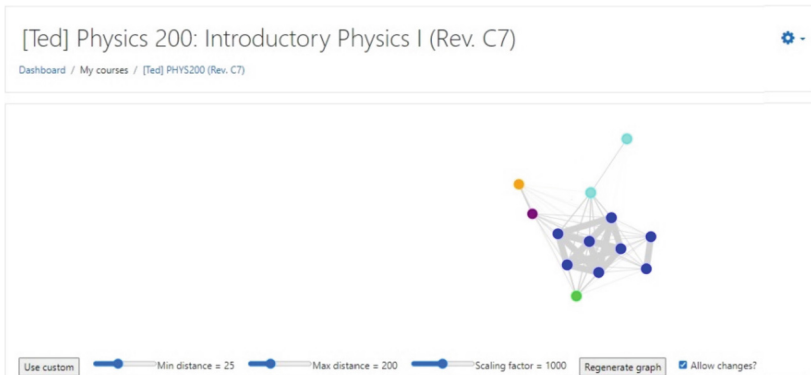


Fig. 7. The Learning Object Graph generated by the LORD.

The updated LOG will be saved automatically as the customized LOG. The button “Use custom” will show the customized LOG. When teachers switch the view to the customized LOG, the button “Use custom” is changed to “Use generated” for them to switch back to the system-generated LOG.

If the teachers are wondering how the system calculates the similarity, they can left-click a node on the graph and then right-click another node – the LORD will show the similarity calculation result on the bottom of the page as Fig. 8 shows. The LORD compares not only the names of the learning objects and also the text content of the objects. Take the first comparison matrix in Fig. 8 for example, the LORD removes the number 3 in the “Lab Report 3” learning object first and then compares the similarity between words in “Lab Report” and “Course Evaluation Survey”. The overall similarity on the top of the figure is calculated based on the method introduced in Sect. 3.

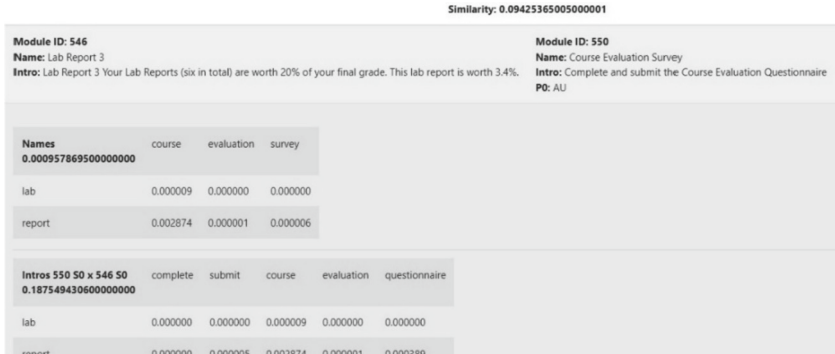


Fig. 8. The screenshot of the sentence similarity matrix comparing two learning objects in LORD

5 Conclusion and Future Works

This research proposes and develops the Word & Sentence Natural Language Processing (WS-NLP) Similarity Service to measure the similarity in text. The service integrates WordNet as the knowledge base and uses the Uniform-Cost Search to find the shortest path between given words in order to determine their similarity. The Munkre’s Assignment Algorithm is adopted to calculate the similarity between sentences, paragraphs, and documents.

With the similarity service, the research creates the LORD Moodle Plug-in to be a support package of an existing Behaviour Analytics Moodle Plug-in. The LORD Moodle plug-in retrieves the information of the learning objects in a Moodle course and sends them to the similarity service to find the similarity between learning objects. The Learning Object Graph is then constructed based on the calculated similarities. Teachers are able to rearrange the LOG freely and use the custom LOG in the BA Moodle Plug-in to cluster students in groups as they did.

The research team is now conducting evaluation by working with two professors who are teaching undergraduate courses: Physics I, Introduction to Statistics and Methods in Applied Statistics, and Statistics and Methods in Applied Statistics, in a university in North America as well as one professor who is teaching graduate level course, Introduction to English for Academic Purposes, in a university in Asia.

The professors are using the Behaviour Analytics with and without the proposed LORD Moodle Plug-ins for their classes in the previous semester. They are asked to use the original BA generated LOG and the LORD generated LOG in the BA Moodle Plug-in to cluster students and verify whether or not the clustering results are appropriate by moving students from/to a proper group. The research team will evaluate the LORD's usability through the comparisons of the precision, recall, and f-measure of the clustering results and professors' given system usability scale score.

References

1. De Medio, C., Limongelli, C., Sciarrone, F., Temperini, M.: MoodleREC: a recommendation system for creating courses using the moodle e-learning platform. *Comput. Hum. Behav.* **104**, 106168 (2020)
2. Felner, A.: Position paper: Dijkstra's algorithm versus uniform cost search or a case against Dijkstra's algorithm. In: *International Symposium on Combinatorial Search*, vol. 2, no. 1, pp. 47–51 (2011)
3. Ferguson, R.: Learning analytics: drivers, developments and challenges. *Int. J. Technol. Enhanced Learn.* **4**(5–6), 304–317 (2012)
4. Ifenthaler, D., Yau, J.-K.: Utilising learning analytics to support study success in higher education: a systematic review. *Educ. Tech. Res. Dev.* **68**(4), 1961–1990 (2020). <https://doi.org/10.1007/s11423-020-09788-z>
5. Jordan, C.: Sur les assemblages de lignes. *Journal für die reine und angewandte Mathematik* **70**, 185–190 (1869)
6. Kew, S.N., Tasir, Z.: Learning analytics in online learning environment: a systematic review on the focuses and the types of student-related analytics data. *Technol. Knowl. Learn.* 1–23 (2021)
7. Kuhn, H.W.: The Hungarian method for the assignment problem. *Naval Res. Logist. Q.* **2**(1–2), 83–97 (1955)
8. Kuo, R., Krahn, T., Chang, M.: Behaviour analytics - a Moodle Plug-in to visualize students' learning patterns. In: Cristea, A.I., Troussas, C. (eds.) *ITS 2021. LNCS*, vol. 12677, pp. 232–238. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-80421-3_25
9. Lloyd, S.: Least squares quantization in PCM. *IEEE Trans. Inf. Theory* **28**(2), 129–137 (1982)
10. Miller, G.A., Beckwith, R., Fellbaum, C., Gross, D., Miller, K.J.: Introduction to WordNet: an on-line lexical database. *Int. J. Lexicogr.* **3**(4), 235–244 (1990)
11. Resnik, P.: Using information content to evaluate semantic similarity in a taxonomy. *arXiv preprint cmp-lg/9511007* (1995)
12. Sheeba, T., Krishnan, R.: A semantic approach of building dynamic learner profile model using wordnet. In: Pati, B., Panigrahi, C.R., Buyya, R., Li, K.-C. (eds.) *Advanced Computing and Intelligent Engineering. AISC*, vol. 1082, pp. 263–272. Springer, Singapore (2020). https://doi.org/10.1007/978-981-15-1081-6_22
13. Smart, C., Slater, P.J.: Center, median, and centroid subgraphs. *Netw. Int. J.* **34**(4), 303–311 (1999)
14. Sunilkumar, P., Shaji, A. P.: A survey on semantic similarity. In: *2019 International Conference on Advances in Computing, Communication and Control (ICAC3)*, pp. 1–8, IEEE (2019)



Deep Knowledge Tracing on Skills with Small Datasets

Ange Tato and Roger Nkambou^(✉)

Université du Québec à Montréal, Montréal, Canada
{nyamen.tato.angea.drienne,nkambou.roger}@uqam.ca

Abstract. Deep Knowledge Tracing (DKT), as well as other machine learning approaches, is biased toward data used during the training step. Thus, for problems where we have few amounts of data for training, the generalization power will be low, and the models will tend to work well on classes containing many samples and poorly on those with few. This situation is quite common in educational data where some skills are very difficult to master while others are very easy. As a result, there will be less data on students who correctly answered questions related to difficult skills, but also on those who provided incorrect answers to questions related to easy skills. In those cases, the DKT is unable to correctly predict the student's answers to questions associated with these skills. To improve DKT performance under these conditions, we have developed a two-fold approach. Firstly, the loss function is modified so that some skills are masked to force the model's attention on those that are difficult to generalize. Secondly, to cope with the limited amount of data on some skills, we proposed a hybrid architecture that integrates a priori (expert) knowledge with DKT through an attentional mechanism. The resulting model accurately tracks student Knowledge in the Logic-Muse Intelligent Tutoring System (ITS), compared to the traditional Bayesian Knowledge Tracing (BKT) and the original DKT.

Keywords: Deep Knowledge Tracing · Bayesian Knowledge Tracing · Knowledge Tracing

1 Introduction

Modeling students' knowledge is a fundamental step when building intelligent tutoring systems (ITS). Knowledge Tracing, a popular approach to modeling learner knowledge, aims at modeling how students' knowledge evolves during learning [7]. There exist many solutions to estimate that probability such as the Bayesian Knowledge Tracing (BKT) and the Deep Knowledge Tracing (DKT).

BKT [7] is a special case of the Hidden Markov Model where student knowledge is represented as a set of binary variables. Observations are also binary: a student gets a problem either right or wrong [29]. However, there is a certain probability (G , the Guess parameter) that the student will give a correct response. Correspondingly, a student who does know a skill generally will give a

correct response, but there is a certain probability (S , the Slip parameter) that the student will give an incorrect response. The standard BKT model is thus defined by four parameters: initial knowledge, learning rate (learning parameters), slip, and guess (mediating parameters). It has been successfully used in a variety of systems including computer programming [11], reading skills [4], logical reasoning [24] etc. Using a Bayesian network sometimes implies manually defining apriori probabilities and manually labeling student interactions with relevant concepts. Also, the binary response data used to model knowledge, observations and transitions impose a limit on the kinds of exercises that can be modeled. DKT has been proposed as a good alternative to overcome BKT limits.

Deep learning has been successfully applied in many domains including images recognition [9], Natural Language Processing [2,6] and more recently in education for modeling student knowledge. DKT [20] uses an LSTM (Long Short Term Memory) to predict student performance based on the pattern of their sequential responses. DKT observes knowledge at both the skill level, and the problem level, observing the correctness of each problem. At any time step, the input layer of the DKT is the student performance on a single problem of the skill that the student is currently working on. In other words, the skill and correctness of each item are used to predict the correctness of the next item, given that problem's skill [31]. Rather than constructing a separate model for each skill as BKT does, DKT models all skills jointly [12,20]. It has been shown that DKT can robustly predict whether or not a student will solve a particular problem correctly given the accuracy of historic solutions [26,31]. However many recent works have pointed out some issues with the DKT such as: the model only considers the knowledge components of the problems and correctness as input, neglecting the breadth of other features collected by computer-based learning platforms [31]. This problem was solved by incorporating more features into the input of the model and by incorporating an auto-encoder network layer to convert the input into a low dimensional feature vector. Other issues were pointed out by Chun-Kit et al. [28] which are (1) the model fails to reconstruct the observed input; As a result, even when a student performs well on a skill component, the prediction of that skill mastery level decreases instead, and vice versa; (2) the predicted performance for skills across time-steps is not consistent. As a solution, they augmented the loss function with regularization terms that correspond to 'reconstruction' and 'waviness'. This solution is similar to our first contribution.

Skills with limited data denotes skills that are very difficult to master (there are few data on students that have mastered these skills) and skills that are very easy to master (there are few data on students that have not mastered these skills). Like other machine learning techniques, the DKT is biased towards the data seen during the training phase due to its data-driven approach. Therefore, the generalization performance of the model depends on the training data. For problems (or skills) that are difficult to master, which means a rare occurrence of correct answers, the DKT is unable to accurately predict the student performance on questions associated with those skills. In the same vein, for skills

that are easy to master, which means a rare occurrence of incorrect answers, the DKT also fails to accurately predict the student performance on questions associated with those skills. In machine learning domain, this problem is known as the class imbalance problem [10] where there are fewer occurrences of data for a certain class which results in sub-optimal performance.

In the educational field, a priori expert knowledge is usually available and generally used to build systems such as Intelligent Tutoring Systems (ITS). Expert knowledge can be available through books or previously built models (such as rules-based models). We believe that this a priori expert knowledge, sometimes acquired over decades of intense research, cannot be dismissed and ignored. Sometimes, a priori expert knowledge can be available but is not always sufficiently accurate. Nevertheless, even inaccurate models can provide useful information that should not be dismissed [30]. In general, employing a fully data-driven approach to train deep neural networks requires the acquisition of a huge amount of data, which might not always be practical or realistic due to economic reasons or the complexity of the process it entails. We show that combining a priori expert knowledge and data-driven methods using the attentional mechanism constitutes a suitable approach towards the design of hybrid deep learning architecture.

In this paper, we put forth an approach that uses attentional mechanism [27] and capitalizes on the availability of expert knowledge (through a Bayesian Network built by experts) to overcome the problem of having few data when training machine deep learning models. We also propose to leverage the problem of skills with limited data by using a custom loss function for the DKT, where we mask skills with many samples and give weight to skills with few samples. The main contributions of this work can be summarized as follows: (1) An extension that improves the original DKT in the prediction of skills with limited data; (2) The incorporation of a priori knowledge (when available) using attention mechanism in a deep learning architecture. We applied the proposed solution to the prediction of the logical reasoning performance of students.

2 Brief Review of the Deep Knowledge Tracing

DKT takes as input sequences of exercise-performance (e_t, p_t) pairs presented one trial at a time. The model then predicts the knowledge state, based on the current hidden state. The hidden layer of the LSTM represents the latent encoding of knowledge state, based on the current input and previous latent encoding of knowledge state. It represents the latent knowledge state of a student, resulted from his past learning trajectory.

To train the model, the exercise-performance needs to be converted into a sequence of fixed length input vectors x_t . x_t is a one-hot encoding of (e_t, p_t) that represents the combination of which exercise (skill involved) was answered and the real answer given by the student. For student s with a sequence of exercise-performance of length T , the DKT model maps the inputs (x_1, x_2, \dots, x_T) to the output y_t which is a vector of length equals to the number of skills. Each entry

of y_t represents the predicted probability that the student will correctly answer exercises from that particular skill. The training objective is the negative log-likelihood of the observed sequence of student responses under the model. The loss function is as follows [20]:

$$L = \sum_t \ell(y_t^\top \delta(e_{t+1}), p_{t+1}) \quad (1)$$

$\delta(e_{t+1})$ is the one-hot encoding of which exercise is answered at time $t + 1$. ℓ is the binary cross entropy.

3 Penalization of Loss Function

The performance prediction on skills with little data can be compared to unbalanced problems in machine learning. There are multiple strategies to deal with class imbalance such as resampling the data by under-sampling the majority class or oversampling the minority class [19]. However, over-sampling can easily introduce undesirable noise with overfitting risks; on the other hand, under-sampling is often preferred but may remove valuable information, which we can't afford because of the few amounts of data. Another well-known strategy is cost-sensitive learning, which assigns higher misclassification costs to the minority class than to the majority class [23]. Our proposed solution falls into the category of cost-sensitive learning, where we assigned a higher cost for skills with few samples. In other words, during the training, we force the model to pay more attention to floor/ceiling skills.

The main idea is to treat the loss as a weighted average where the weights are specified by parameters λ_i with $i \in [0, n]$ where n is the number of parts to add to the original loss. We added a regularization term in the loss function which corresponds to the application of a mask to the original loss to ignore skills with many samples. The result of the mask has 2 parts: the very difficult skills and the very easy skills. Each part of the mask is multiplied by regularization parameters (or weights) λ_1 and λ_2 respectively. These factors are the penalties applied to the model. Penalizing the DKT model imposes an additional cost when making prediction mistakes on the minority class during training. These penalties bias the model to pay more attention to the minority class (correct answers on skills difficult to master and incorrect answers on skills easy to master). Now we will explain how we compute the new loss.

If we were to evaluate a DKT model for the prediction of a knowledge state on only one skill k , the loss function would be written as follows:

$$L_k = \sum_t \ell(y_{k,t}, p_{k,t+1}) \quad (2)$$

where $y_{k,t}$ is a vector of length equals to the number of skills with 0 on all the entries except for the entry k . It represents the predicted probability at time t that a student would correctly answer a problem related to that specific skill

(k) at time $t + 1$. $p_{k,t+1}$ denotes the real answer (performance) given at time $t + 1$ by a student on a problem related to the skill k . For the model to be able to make good predictions on skills with few data, we first extracted those skills using statistics (skills with a very small number of correct answers and skills with a very high number of correct answers). We could have also used the DKT itself (by running the DKT and then identifying skills where the precision and recall are low for each of the correct/incorrect classes). We then apply a mask (this is easily done with any programming language) on the loss function whose purpose is to hide skills with many samples to only keep what we want the model to focus on. We then run the model with a new loss function:

$$L = \sum_t \ell(y_t^T \delta(e_{t+1}), p_{t+1}) + \lambda_1 \sum_t \sum_{kF} \ell(y_{kF,t}, p_{kF,t+1,1}) + \lambda_2 \sum_t \sum_{kC} \ell(y_{kC,t}, p_{kC,t+1,0}) \quad (3)$$

Parameters λ_1 and $\lambda_2 \geq 0$ are weights that we apply to the mask and kF , kC denote respectively all skills that are difficult to master (floor skills) and all skills that are easy to master (ceiling skills). The digit 1 in $p_{kF,t+1,1}$ denotes correct answers and digit 0 in $p_{kC,t+1,0}$ denotes incorrect answers. If $\lambda = 0$, the model becomes the original DKT. The more the value λ is high, the more the model will be biased towards skills with few samples (floor/ceiling skills). The choice of the value of λ is thus important. $y_{k,t}$ and $p_{k,t+1}$ are the vectors y_t and p_{t+1} respectively, where we only keep values related to the skill k . Thus $p_{kF,t,1}$ refers to a vector of length equals to the number of skills with 0 on all the entries except for entries kF and where the real answers given are correct at time $t + 1$. $\delta(e_{t+1})$ is the one-hot encoding of which exercise is answered at time $t + 1$. The new loss provides another way to balance the data.

4 Combining a Bayesian Network with the DKT

BN is a graphical model used to model process under uncertainty by representing relationships between variables in terms of a probability distribution [21]. BN allows inferring the probability of mastering a skill from a specific response pattern [18]. The structure and the parameter or probability distributions are provided by experts or learned using algorithms such as Expectation-Maximization. In this work, we only consider BNs that are built by experts. BNs have been successfully used to model knowledge state of learners [15, 16, 24] or learner affect [22]. There are many contexts where a lot of data or expert knowledge (e.g. medicine) are available. How can the DKT benefit from that? We want to take advantage of expert knowledge when available. We also suppose that the model accuracy could increase as expert knowledge is not biased towards rare data, which can be very helpful in the case of few amounts of data.

The inner architecture of a neural network makes it difficult to incorporate domain knowledge into the learning process [13]. Our solution is to force the model to pay attention to what the expert knowledge says about the current

input x . The goal is not to incorporate how the expert knowledge is processed, but rather its final prediction about the input. Since attention [27] is a memory-access mechanism, it fits well in this context where we want the model to have access to the expert knowledge during learning [1], as a memory. Thus, the model will pay attention to what the expert knowledge says before taking any decision. Attention-based recurrent networks have been successfully applied to a wide variety of tasks, such as machine translation [3], handwriting synthesis [8], speech recognition [5], etc. By integrating the expert knowledge (here a BN) in the DKT using the attention mechanism, the model iteratively processes the a priori knowledge by selecting relevant content at every step. In the attention mechanism presented by Luong et al. [14] (specifically the global attentional model), the attentional vector is computed from the target hidden state h_t and the input hidden state. Instead of the input hidden state, we will have the data coming from expert knowledge which will be used to compute the context vector C_t (that we will call expert-side context vector) (see Fig 2 in [14]). Thus, given the hidden state y_t (the prediction) of the DKT, and the expert-side context vector c_t^e , we employ a concatenation layer to combine the information from both vectors to produce the attentional hidden state a_t as follows:

$$a_t = \tanh(W_c [c_t^e; y_t]) \quad (4)$$

The attentional vector a_t along with the expert prediction e and the y_t are then fed through a Dense layer to produce the predictive knowledge state y_t' . Now, the expert-side context is computed as follows:

$$\begin{aligned} \text{score}(e_k, y_t) &= e_k \cdot y_t \cdot W_a + b \\ \alpha_{t,k} &= \frac{\exp^{\text{score}(e_k, y_t)}}{\sum_{j=1}^s \exp^{\text{score}(e_j, y_t)}} \\ c_t^e &= \sum_k \alpha_{t,k} \cdot e \end{aligned} \quad (5)$$

where $1 \leq k \leq s$, e is the current knowledge state predicted by the expert, y_t is the current knowledge state predicted by the DKT and s is the number of skills. The parameter e represents a vector of length equals to the number of skills where each entry represents the predicted probability that the student will answer correctly to exercises from that particular skill, given by the BN. e_k is of size 1 and W_a , W_c , W_s , y_t , e and a_t are of size s the number of skills. Figure 1 shows in detail this global process.

5 Experiments

Our goal is to create an accurate learner model in an ITS called Logic-Muse. The current learner model implemented in Logic-Muse uses a BN [hidden] built from expert knowledge.

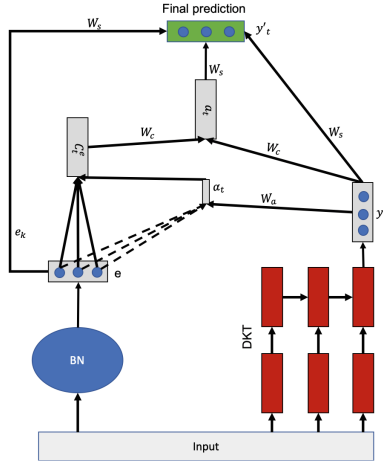


Fig. 1. Global attentional hybrid model—at each time step t , the model infers an alignment weight vector $\alpha_{t,k}$ based on the current predicted knowledge y_t and all entries of the expert knowledge vector. c_t^e is then computed as the weighted average, according to $\alpha_{t,k}$, over each of the entries of the expert knowledge vector.

5.1 Logic Muse

Logic-Muse is a web-based Intelligent Tutoring System that helps learners improve logical reasoning skills. Logic-Muse includes a learning environment that uses various meta-structures to provide reasoning activities on various contents [hidden]. The expert model implements logical reasoning skills and knowledge as well as related reasoning mechanisms (syntactic and semantic rules of the given logical system). The model of (valid and invalid) inference rules are encoded as production rules, and the semantic memory of the target logic is encoded in a formal OWL ontology and connected to the inference rules. The first version of Logic-Muse focuses on propositional logic. Logic-Muse learner model goal is to represent, update and predict the learner’s state of knowledge based on her/his interaction with the system. It has multiple aspects including the cognitive part that essentially represents the learner’s knowledge state (mastery of the reasoning skills in each of the six reasoning situations that have been identified thanks to the experts). The cognitive state is generated from the learner’s behavior during his interactions with the system, that is, it is inferred by the system from the information available. It is supported by a Bayesian network [hidden] based on domain knowledge, where influence relationships between nodes (reasoning skills) as well as prior probabilities are provided by the experts. Some nodes are directly connected to the reasoning activities such as exercises. The skills involved in the BN are those put forward by the mental models’ theory to reason in conformity to the logical rules. There are 16 skills directly observable (linked to the exercises) and 12 latent skills. There is a total of 48 exercises.

5.2 Dataset

294 participants participated in this study. They all completed the 48 logical reasoning exercises. In our dataset, each line of data represents each participant (a total of 294 data and a sequence length of 48). The amount of data is very few to train a deep learning model. However, combined with expert knowledge, we will see a substantial difference in the results. The exercises were encoded using skills that are directly observable, which means that the questions related to the same skill are encoded with the same Id (1~16). The skills with few data are determined by a comparison of the average of correct answers obtained for each skill. In Table 1, we averaged all the answers on each skill. The skills difficult to master (floor) are those with the lowest average value and the skills easy to master (ceiling) are those with the highest average. The second and the last part of our new loss function involve those skills. Since the LSTM only accept a fixed length of vectors as the input, we used one-hot encoding to convert student performance into a fixed length of vectors whose all elements are 0 except for a single 1. The single 1 in the vector indicates two things: which skill was answered and if the skill was answered correctly.

Table 1. Distribution of responses over skills—Skills difficult to master (Average < 0.4) and skills easy to master (Average > 0.9) are in **bold**.

Skills	N	Average	Standard dev
MppFd	294	0,9456	0,16078
MppMd	294	0,898	0,23726
MppCcf	294	0,907	0,2394
MppA	294	0,9615	0,16066
MttFd	294	0,8435	0,26646
MttMd	294	0,7925	0,29985
MttCcf	294	0,7494	0,33326
MttA	294	0,8401	0,28974
AcMa	294	0,424	0,38072
AcFa	294	0,3039	0,3652
AcCcf	294	0,3345	0,40801
AcA	294	0,2823	0,41038
DaMa	294	0,407	0,37389
DaFa	294	0,3027	0,35081
DaCcf	294	0,381	0,40662
DaA	294	0,305	0,42077

5.3 Results

To assess our proposed solutions, we ran 3 models: the DKT, the DKT where we applied a mask to the loss function (DKTm), and the DKTm with apriori knowledge (DKTm+BN). We used 20% of the data for testing and 15% for validation. The BN alone gave 65% of global accuracy. The result is evaluated using the *F1score* metric on each skill (treated as 2 classes - correct and incorrect answers) being predicted and the overall accuracy. The models were evaluated in 20 different experiments and the final results were averaged. In all our experiments, we set λ_1 and $\lambda_2 = 0.10$. Our implementation of the DKTm+BN model in Tensorflow using Keras backend was inspired by the implementation¹ done by Khajah et al. [12]. Our code is also available on GitHub² for further research.

The results (for skills that are difficult to master) are shown in Figs. 2 and 3. As expected, the new DKTm (Accuracy = 0.8) outperforms the original DKT (Accuracy = 0.74) on all the skills being predicted. Furthermore, the DKTm enhanced with BN (Accuracy = 0.82) outperforms all other models on predicting skills with little data (good answers on skills difficult to master). For skills that are easy to master (e.g. **MPP**), all the models always predict that students will give correct answers (F1score of incorrect answers is 0 for DKT and almost 0 for the other models) even after applying the weighted loss. This is because, on the 294 data, we have for example only 6 incorrect answers for the **MPP_FFD** skill. We tested the models with high values for λ_2 and we got values of f1score equal to around 0.6 for correct answers and around 0.7 for incorrect answers. This result can be satisfying in other contexts but in the context of logical reasoning where it is established that the **MPP** is a skill that is always well mastered, 0.6 as an f1score for correctly predicting a correct answer is not acceptable. That is why we kept $\lambda_2 = 0.10$. However, the solution stays valid for data where the ratio $r = \text{number of correct answers} / \text{number of questions answered}$ or $r = \text{number of incorrect answers} / \text{number of questions answered}$ on a skill is not too small (as in this case) and is less than 0.5. For skills that are difficult to master (Figs. 2 and 3), there is a huge difference between the DKT and the other models. The DKT is unable to track correct answers on skills difficult to master (see Fig. 3). This behavior cannot be accepted since the knowledge tracing of students who perform well on those skills will fail. Thus it is important to make sure that the final model is accurate for all the skills. For the prediction of wrong answers on skills difficult to master (see Fig. 3), we can notice that the DKTm and the DKTm+BN still behave better than DKT which means that the penalty added to the loss function does not affect the predictive capacity of the original DKT.

We noticed that predictions are not sometimes consistent with the reality as other works have also highlighted [28]. The model fails to reconstruct the observed input. As a result, even when a student performs well on a skill, the prediction of that skill's mastery level decreases instead, and vice versa. Also, the predicted performance across time steps is not consistent. When a student gives

¹ <https://github.com/mmkhajah/dkt>.

² <https://github.com/angetato/Deep-Knowledge-Tracing-On-Skills-With-Limited-Data>.

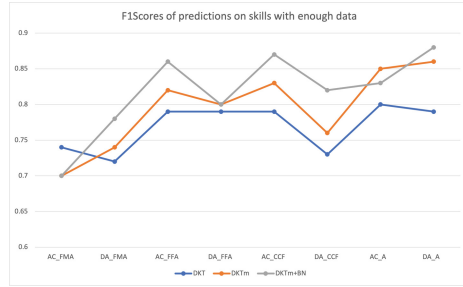


Fig. 2. The DKT, the DKTm and the DKTm+BN on skills that are difficult to master—We repeated the experiments 20 times. For each skill, we computed the value of the f1score for the prediction of incorrect answers (enough data).

correct answers to a skill k , the DKT does not sometimes update the current state of knowledge on that skill (the skill stays low or is updated very slowly). The problem can be addressed by adding regularization terms to the loss function of the original DKT as suggested by [28]. It can also be partially solved when adding the a priori knowledge as we noticed during our experiments. However, we stay confident in the fact that if the a priori knowledge is more accurate (which is not our case as the accuracy of the BN is 0.65) we will get better results.

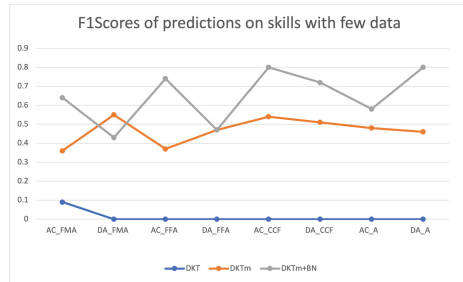


Fig. 3. Prediction score of correct answers on skills that are difficult to master (limited data) for the three models: DKT, DKTm and DKTm+BN.

The idea of using the attention mechanism to incorporate expert knowledge into NN is novel and can be used in other domains such as text classification or in medicine where there is a lot of expert knowledge available. For example, we could think of a classifier using a neural network combined with a rule-based system playing the role of expert knowledge.

6 Conclusion

In this paper, we proposed two simple, effective, and intuitive techniques to improve the DKT on the prediction of floor and ceiling skills which are skills that are very difficult and easy to master respectively. The first technique consists of applying a penalty to the loss function, for making incorrect predictions on skills with few samples. The second solution aims at incorporating a BN (expert knowledge) in the DKT using the attention mechanism. At the same time, we introduced a new way of using the attention mechanism, to allow neural networks to take into account expert knowledge (when available) in their training and decision process. We tested the solution on a dataset that is unbalanced. The results showed that the DKT is unable to accurately track skills with limited data, compare to the DKTm and the DKTm+BN.

During the experiments, we noticed that, for skills that are very easy to master, all the 3 models were unable to track incorrect answers, since the ratio $r = \text{incorrect answers} / \text{total of questions answered}$ was very low. Even with a penalty, we were not able to significantly improve the DKT model on the skills involved. However, with the combination of the BN, the results were noticeable. In this paper, we have set the regularization parameters λ_1 and λ_2 with a fixed value but for future work, we will do a grid search to find the best values.

We are aware that the lack of data might be a bias to our results since deep learning architectures perform better on larger datasets. However, the floor/ceiling skills problem can still occur even with a large dataset. Thus, we believe that the solutions we have proposed can work perfectly on larger datasets. We will do further experiments on the integration of expert knowledge into neural network architectures. We also plan to test our techniques on larger and or public datasets such as ASSISTments dataset.

References

1. Ange, T., Roger, N., Aude, D.: Hybrid deep neural networks to predict socio-moral reasoning skills. In: Proceedings of the 12th International Conference on Educational Data Mining, pp. 623–626 (2019)
2. Ange, T., Roger, N., Aude, D., Claude, F.: Semi-supervised multimodal deep learning model for polarity detection in arguments. In: 2018 International Joint Conference on Neural Networks (IJCNN), pp. 1–8. IEEE (2018)
3. Bahdanau, D., Cho, K., Bengio, Y.: Neural machine translation by jointly learning to align and translate. arXiv preprint [arXiv:1409.0473](https://arxiv.org/abs/1409.0473) (2014)
4. Beck, J.E., Chang, K.: Identifiability: a fundamental problem of student modeling. In: Conati, C., McCoy, K., Paliouras, G. (eds.) UM 2007. LNCS (LNAI), vol. 4511, pp. 137–146. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-73078-1_17
5. Chorowski, J.K., Bahdanau, D., Serdyuk, D., Cho, K., Bengio, Y.: Attention-based models for speech recognition. In: Advances in Neural Information Processing Systems, pp. 577–585 (2015)
6. Collobert, R., Weston, J.: A unified architecture for natural language processing: deep neural networks with multitask learning. In: Proceedings of the 25th International Conference on Machine learning, pp. 160–167. ACM (2008)

7. Corbett, A.T., Anderson, J.R.: Knowledge tracing: modeling the acquisition of procedural knowledge. *User Model. User-Adap. Inter.* **4**(4), 253–278 (1994)
8. Graves, A.: Generating sequences with recurrent neural networks. *arXiv preprint [arXiv:1308.0850](https://arxiv.org/abs/1308.0850)* (2013)
9. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 770–778 (2016)
10. Huang, C., Li, Y., Change Loy, C., Tang, X.: Learning deep representation for imbalanced classification. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5375–5384 (2016)
11. Kasurinen, J., Nikula, U.: Estimating programming knowledge with bayesian knowledge tracing. In: *ACM SIGCSE Bulletin*, vol. 41, pp. 313–317. ACM (2009)
12. Khajah, M., Lindsey, R.V., Mozer, M.C.: How deep is knowledge tracing? *arXiv preprint [arXiv:1604.02416](https://arxiv.org/abs/1604.02416)* (2016)
13. Lu, H., Setiono, R., Liu, H.: Effective data mining using neural networks. *IEEE Trans. Knowl. Data Eng.* **8**(6), 957–961 (1996)
14. Luong, M.T., Pham, H., Manning, C.D.: Effective approaches to attention-based neural machine translation. *arXiv preprint [arXiv:1508.04025](https://arxiv.org/abs/1508.04025)* (2015)
15. Martin, J., VanLehn, K.: Student assessment using bayesian nets. *Int. J. Hum Comput Stud.* **42**(6), 575–591 (1995)
16. Nguyen, L., Do, P.: Combination of bayesian network and overlay model in user modeling. In: Allen, G., Nabrzycki, J., Seidel, E., van Albada, G.D., Dongarra, J., Sloat, P.M.A. (eds.) *ICCS 2009. LNCS*, vol. 5545, pp. 5–14. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-01973-9_2
17. Nkambou, R., Brisson, J., Kenfack, C., Robert, S., Kissok, P., Tato, A.: Towards an intelligent tutoring system for logical reasoning in multiple contexts. In: Conole, G., Klobučar, T., Rensing, C., Konert, J., Lavoué, É. (eds.) *EC-TEL 2015. LNCS*, vol. 9307, pp. 460–466. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-24258-3_40
18. Nkambou, R., Mizoguchi, R., Bourdeau, J.: *Advances in Intelligent Tutoring Systems*, vol. 308. Springer, Berlin (2010). <https://doi.org/10.1007/978-3-642-14363-2>
19. Oquab, M., Bottou, L., Laptev, I., Sivic, J.: Learning and transferring mid-level image representations using convolutional neural networks. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1717–1724 (2014)
20. Piech, C., et al.: Deep knowledge tracing. In: *Advances in Neural Information Processing Systems*, pp. 505–513 (2015)
21. Russell, S.J., Norvig, P.: *Artificial Intelligence: a Modern Approach*. Pearson Education Limited, Malaysia (2016)
22. Sabourin, J., Mott, B., Lester, J.C.: Modeling learner affect with theoretically grounded dynamic bayesian networks. In: D’Mello, S., Graesser, A., Schuller, B., Martin, J.-C. (eds.) *ACII 2011. LNCS*, vol. 6974, pp. 286–295. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-24600-5_32
23. Shen, W., Wang, X., Wang, Y., Bai, X., Zhang, Z.: Deepcontour: a deep convolutional feature learned by positive-sharing loss for contour detection. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3982–3991 (2015)

24. Tato, A., Nkambou, R., Brisson, J., Kenfack, C., Robert, S., Kissok, P.: A bayesian network for the cognitive diagnosis of deductive reasoning. In: Verbert, K., Sharples, M., Klobučar, T. (eds.) EC-TEL 2016. LNCS, vol. 9891, pp. 627–631. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-45153-4_78
25. Tato, A., Nkambou, R., Brisson, J., Robert, S.: Predicting learner’s deductive reasoning skills using a bayesian network. In: André, E., Baker, R., Hu, X., Rodrigo, M.M.T., du Boulay, B. (eds.) AIED 2017. LNCS (LNAI), vol. 10331, pp. 381–392. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-61425-0_32
26. Wang, L., Sy, A., Liu, L., Piech, C.: Deep knowledge tracing on programming exercises. In: Proceedings of the Fourth (2017) ACM Conference on Learning@ Scale, pp. 201–204. ACM (2017)
27. Xu, K., et al.: Show, attend and tell: Neural image caption generation with visual attention. In: International Conference on Machine Learning, pp. 2048–2057 (2015)
28. Yeung, C.K., Yeung, D.Y.: Addressing two problems in deep knowledge tracing via prediction-consistent regularization. arXiv preprint [arXiv:1806.02180](https://arxiv.org/abs/1806.02180) (2018)
29. Yudelson, M.V., Koedinger, K.R., Gordon, G.J.: Individualized bayesian knowledge tracing models. In: Lane, H.C., Yacef, K., Mostow, J., Pavlik, P. (eds.) AIED 2013. LNCS (LNAI), vol. 7926, pp. 171–180. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-39112-5_18
30. Zappone, A., Di Renzo, M., Debbah, M., Lam, T.T., Qian, X.: Model-aided wireless artificial intelligence: Embedding expert knowledge in deep neural networks towards wireless systems optimization. arXiv preprint [arXiv:1808.01672](https://arxiv.org/abs/1808.01672) (2018)
31. Zhang, L., Xiong, X., Zhao, S., Botelho, A., Heffernan, N.T.: Incorporating rich features into deep knowledge tracing. In: Proceedings of the Fourth (2017) ACM Conference on Learning@ Scale, pp. 169–172. ACM (2017)

**Algorithms for Prediction,
Recommendation and Classification
in Learning Systems**



A Learning Analytics Approach to Build Learner Profiles Within the Educational Game OMEGA+

Deepak Chandrasekaran¹, Maiga Chang², and Sabine Graf²(✉)

¹ Rajalakshmi Institute of Technology, Chennai, Tamil Nadu, India

² Athabasca University, Edmonton, Canada

{maigac, sabineg}@athabascau.ca

Abstract. Educational games can act as excellent learning environments, where learners play and learn at the same time. However, typically, once a game has been developed, it is launched and then maybe evaluated for learning effectiveness but details on how learners actually use the game as well as how they play and learn in the game are rarely investigated. In addition, which groups of learners are more attracted or less attracted by the game is seldom looked at. However, such investigations are essential to ensure that the game is used in the way it was intended, that the game is fun and provides learning opportunities at the same time, that learners can benefit the most from the game and to make the game interesting for many different groups of players. In this paper, we introduce a learning analytics approach that builds learner profiles based on learners' characteristics and behaviour in the educational game OMEGA+. The approach is rather generic and can be easily adapted to other educational games. By using the proposed learning analytics approach, clusters of learners are built that provide insights into how learners use the game, how they play and how they learn. In addition, when considering demographic attributes when analysing the clusters, insights can be gained on which groups of learners are more and which groups are less attracted to the game.

Keywords: Game-based learning · Educational games · Learning analytics · Game learning analytics · Clustering · Learner profiling

1 Introduction

Educational games have high potential in helping students to learn in a fun way. However, similar to online courses, in order to improve such game-based learning environments and ensure that students can benefit most from them, it is important to understand aspects such as how learners use the game, how they behave in it, how much they play in comparison to how much they learn, etc. In addition, understanding which groups of learners/players like the game most and who is not so much attracted to the game, provides the possibility

The authors acknowledge the support of Canadian Internet Registration Authority (CIRA)'s Community Investment Program, the National Science and Engineering Research Council of Canada (NSERC) [RGPIN-2020-05837], and Mitacs (Globalink program).

to expand and tailor the game to those underrepresented player groups to enable them to benefit from the educational game too.

Learning analytics is a fast-emerging research area, which deals with “the measurement, collection, analysis and reporting of data about learners and their contexts, for purposes of understanding and optimizing learning and the environments in which it occurs” [1]. Most research in learning analytics looks at online courses as learning environments, several consider social environments (e.g., discussion forums, social network sites, etc.) but only a relatively small number of works conduct learning analytics research in game-based learning environments. (i.e., Alonso-Fernández et al. [2] have conducted a comprehensive systematic literature review on such papers).

In this paper, we propose a learning analytics approach to build learner profiles based on learners’ characteristics and behaviours in the educational game OMEGA+ [3, 4]. Such profiles can then be used to learn more about how students play and learn in the game as well as how to improve the game design to attract groups of learners that are underrepresented.

According to a systematic literature review on research papers that use learning analytics/data science approaches on game data from educational games, only seven papers exist that focus on building learner profiles based on data from educational games [2]. These papers focus on two directions: First, some research has been conducted on building learner/player profiles to identify certain information from player data (similar to learner modelling). For example, Denden et al. [5] identified personality traits from player behaviour. Another example is the work by Loh and Sheng [6], where authors created a Maximum Similarity Index that represents how (dis)similar the performance of novice players is, compared to expert players within a ‘multiple-solution’ serious game environment. The other direction includes research on building learner/player profiles based on performance. Examples of such research include the works by Slimani et al. [7], Lazo et al. [8] and Polyak et al. [9], where players are clustered into performance groups.

In this paper, we propose a learning analytics approach that also uses clustering but with the purpose of analysing how learners play, how they learn and how to improve the game to make it more attractive for players/learners who are currently not so attracted to the game. In order to do so, the proposed approach is based on a comprehensive learners/player profile based on multiple learner/player characteristics and behaviours. The proposed approach has been designed for the educational game OMEGA+ but can be easily adapted to other educational games. To verify our approach, a detailed example with simulated data is provided.

This paper is structured as follows: Sect. 2 provides a brief overview on OMEGA+. Section 3 introduces the proposed learning analytics approach and Sect. 4 demonstrates the approach through an example. Section 5 then concludes the paper.

2 OMEGA+

OMEGA+ (the former version of the game was called OMEGA) [3, 4] is an online educational game that aims at improving four meta-cognitive skills while learners are playing. Those skills are: (1) problem solving, (2) associative reasoning, (3) planning

and organization and (4) accuracy and evaluation. In the game, players play matches consisting of a set of three subgames against each other. Overall, there are ten different subgames, each focusing on improving a particular meta-cognitive skill.

In each match, players are scored by how they performed individually through the increase of their meta-cognitive skills and how they performed against their opponent through increase/decrease of their points. For each subgame played, a performance score is calculated that shows how well the player played that subgame. This performance score is then translated into a meta-cognitive skill score of the meta-cognitive skill associated with the respective subgame. To compare players with each other, all performance scores of the subgames within a match are summed up. The winner receives points and the loser loses points, allowing a ranking based on points. The number of won/lost points depends on the overall points the players have before the match.

Besides points and meta-cognitive skill scores, the game entails several other motivational features. Each subgame has 10 difficulty levels where players upgrade to the next level once their average performance over the last 10 times in the subgame is above 70%. Players are also presented with an overall game level, which is the average value of all subgame levels. The game's currency (Ω) is earned for every subgame played depending on how well it is played. If players log in multiple days in a row, they get a bonus to earn more currency per played subgame. Every player is represented by a robot avatar and the earned currency can be used to purchase robot parts to upgrade the player's robot avatar. In addition, a learning analytics dashboard is provided for players to monitor and investigate their game behaviour [10]. Players can unlock 48 badges that are linked to game activities, such as logging in for consecutive days, winning matches, and using the learning analytics dashboard. The game also features a leaderboard with multiple rankings (e.g., by points, metacognitive skill scores, available currency and several other metrics). Players can also send friendship requests to other players. When they play a match, they can then choose to either play against a friend who is currently online or be matched with a random player.

3 Learning Analytics Approach

The proposed approach retrieves relevant data from the game's database and uses a clustering algorithm to classify the data into different groups. Those groups can then be visualized and analysed with respect to their significant characteristics and behaviours to improve our understanding on how players play and learn in OMEGA+ and which groups are more or less attracted to the game. The proposed approach has been implemented in Python using Google's Colaboratory (Colab). The approach consists of four steps, which are explained in the following subsections in more detail.

3.1 Data Retrieval

In this research, we use three different categories of attributes: player details, player possessions, and player activities. Those categories include the following attributes:

Player Details

- **GameLevel:** The game level presents the average difficulty level the player reached in all subgames.
- **Points:** Points represent how well a player played matches against other players.
- **AgeRange:** When creating a player account, players are asked to optionally provide their age range (e.g., 18–24 years, 25–34 years, etc.).
- **Gender:** Another information that players can provide optionally when creating an account is the gender.
- **AllowFriend:** This attribute shows whether the player has enabled or disabled friend request. If this option is enabled, other players can send friend request.
- **ProblemSolvingSkills:** The player's problem-solving metacognitive skills are calculated as a percentage value of his/her performance in the Bypass and Viroid subgames. Only increases in those skills are recorded. All the other metacognitive skills are calculated in the same way.
- **AssociativeReasoningSkills:** It represents how well the player performs in Associative Reasoning subgames, which are Crossplay, Pattern Hacker and Pirate Hunter.
- **PlanningOrganizationSkills:** It represents how well the player performs in Planning and Organization subgames, which are CR2k, Evacres and Weekend Barista.
- **EvaluateAccuracySkills:** It represents how well the player performs in Accuracy and Evaluation subgames, which are Card Swap and Delivery Dash.
- **AveragePerformance (1–10):** These 10 attributes (one for each subgame) represent the average performance achieved by the player when playing the respective subgame the past 10 times.

Player Possessions

- **Currency:** The in-game currency (Ω) is awarded to a player for each played subgame within a match based on their performance and difficulty level.
- **RobotParts:** This attribute represents the number of robot parts the player has purchased. The player can buy different parts of the robot using in-game currency after fulfilling certain requirements (e.g., earning a certain badge, etc.).
- **TotalBadges:** This attribute represents the total number of badges a player earned.
- **TotalFriends:** This attribute represents how many friends a player has in the game.

Player Activities

- **TotalMatches:** This attribute represents the total number of matches played.
- **TotalTime:** It represents the total time spent by the player in the game.

- **SurveysCompleted:** The game contains a few surveys to evaluate the game with respect to its ability to improve meta-cognitive skills of the player, usability and others. This attribute shows whether the player completed any surveys and if so, how many surveys the player completed.
- **LeaderboardLog:** This attribute represents the number of times the player checks the different leaderboards in the game. More accurately, this attribute counts the clicks in the leaderboard area that players use to look at different leaderboards or different configurations of the leaderboards.
- **AnalyticLog:** This attribute represents the number of times the player checks the learning analytics dashboard in the game. The learning analytics dashboard contains (1) line graphs, which show metacognitive skill scores with various filters and visualization options and (2) scatter plots, which show performance scores, again with various filters and visualization options. More accurately, this attribute shows the total number of visualizations created in the learning analytics dashboard.

Code has been implemented that retrieves data regarding the proposed attributes for every player from the game's database.

3.2 Data Preparation

After retrieving the data, it is checked for null values. Some data of attributes in the player possessions and player activities categories may contain null values for some players. Most machine learning algorithms cannot work with missing data [11]. Therefore, any null values were replaced with 0 or mean, depending on the attributes.

In addition, most machine learning algorithms do not perform well when numerical attributes have different scales. Therefore, standardization was used to bring values of different attributes on the same scale. In particular, the `StandardScaler` transformer feature [12] of the Scikit-Learn library [13] was used to standardize the data. Accordingly, standardization is calculated by subtracting the mean value and then dividing by the standard deviation, so the resulting distribution has unit variance.

3.3 Algorithm

To build learner profiles, the k-means clustering algorithm was used. This algorithm was selected because it is guaranteed to converge, easily adapts to new examples and assigns every player into a cluster [11].

The number of clusters should be specified for the algorithm. To find the optimum number of clusters for the given data, the following steps are performed [11]: first, the model's inertia is calculated for several potential numbers of clusters (i.e., 2 to 9) and plotted on a graph. The inertia of the model is the sum of the squared distance between each instance and its closest centroid [11]. As a result, such graph often contains an inflection point called the elbow after which the inertia decreases much more slowly. Second, another graph is plotted showing the silhouette score of the model for each potential number of clusters. The silhouette score is the mean silhouette coefficient over all the instances [11]. A higher silhouette score is preferred for the optimal number of clusters [11]. Third, by comparing and analysing the elbow value (from the inertia

graph) and the silhouette score (from the silhouette score graph), the optimum number of clusters is determined.

Once the optimal number of clusters is determined, the k-means algorithm is executed with that number of clusters and the results of the model, representing which data point belongs to which cluster, is stored.

3.4 Visualization and Analysis

To visualize the high dimensional data, a dimensionality reduction algorithm, namely the Principal Component Analysis (PCA) algorithm [11, 14], is used. The clusters are then visualized in 2D and 3D using python's matplotlib library for a better understanding of the results.

After the results are visualized, each learner in each cluster is analysed and compared with the learners in the same cluster and neighbouring clusters to understand the significant characteristics and behaviour represented in each cluster. In addition, each principal component of the PCA (represented on the axes of the visualizations) is investigated to find out what it represents. Such analysis provides insights into how learners behave in the game, how they play and how they learn. In addition, when looking at demographic attributes such as age range and gender in each cluster, insights can be gained into who is more attracted by the game and who is not.

4 Validation

This section presents a validation of our approach using simulated data from 47 players to demonstrate the different steps in the approach and potential outcomes. The data are not real but were modelled based on the behaviour and activities of beta-testers. As such, the results represent a realistic example to verify our approach, demonstrate how it works and show which kind of insights it can provide.

After the data extraction and preparation, the optimal number of clusters for the k-means algorithm is determined by plotting the model's inertia and silhouette score in a graph for 2 to 9 clusters (see Figs. 1 and 2). Given that the inertia value decreases slowly after 4 or maybe 5 clusters (Fig. 1) and 4 clusters have a greater silhouette score than 5, the optimum number of clusters for this data is 4. Accordingly, the k-means algorithm is executed using 4 clusters and the results are stored.

Then, the Principal Component Analysis (PCA) dimensionality algorithm is used to transform the high dimensional data into 2D and 3D visualizations (see Figs. 3 and 4). This is done by identifying the hyperplane that lies closest to the data, and then projecting the data onto it [11].

When analysing our exemplary data, the following can be found: The x-axis may represent some sort of overall activity status in the game, where learners on the lower end are rather passive (i.e., having less matches played, less possessions, less activities, less skills) and learners on the upper end are very active in the game. The y-axis may represent learning effectiveness where learners on the lower end play a lot but improve their skills only little (i.e., high amount of time in the game, high number of matches played, a lot of activities on leaderboards and learning analytics dashboard, but relatively

low meta-cognitive skills, relatively low performance in subgames, etc.) while learners on the upper end play little but improve their skills a lot. The z-axis may represent some sort of social status in the game, where learners on the lower end may be less social (i.e., not allow friend requests, have fewer friends, play less matches, complete no or few surveys, etc.) and learners on the higher end or more social.

As such, learners in each cluster may be characterized as follows:

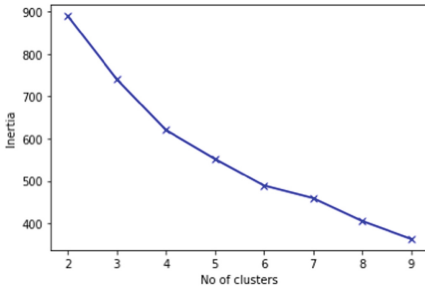


Fig. 1. Inertia graph

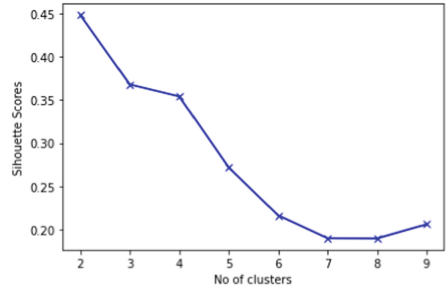


Fig. 2. Silhouette graph

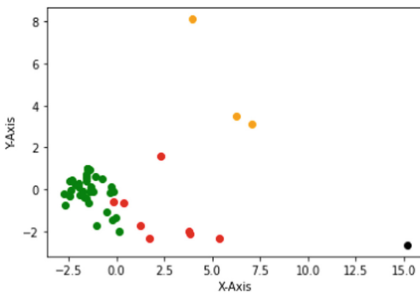


Fig. 3. Visualization of clusters in 2D

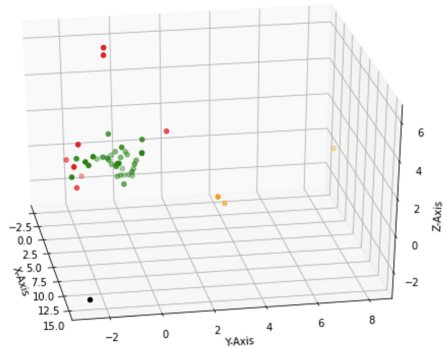


Fig. 4. Visualization of clusters in 3D

- **Green Cluster:** Learners in this cluster are rather passive given their activity status. They have not played too many matches, do not have a lot of possessions, do not do many activities, and have lower skills. They may be novice players who are not so familiar with the game yet. Their learning effectiveness is low to medium, showing that given the amount of time and activities they do, their skills are somewhat improving. This is again in line with novice players who still need to get familiar with the game. With respect to their social status, they are somewhat social. Again, this is in line with novice players who do not have that many friends yet but are building their social status.
- **Red Cluster:** Learners in the red cluster are in general more active than learners of the green cluster, however, their learning effectiveness is in general lower. This means that while they seem to spend more time in the game and use different game features,

their skills are not improving as much as we would expect. With respect to the social status, some learners are very social while others are not.

- **Orange Cluster:** Learners in the orange cluster are more active than learners in the red cluster. But in contrast to the red cluster, their learning effectiveness is rather high. This means that while they spend a lot of time in the game and use a lot of its features, they also have high performance in the subgames and improve their skills a lot. With respect to the social status – similar to the red cluster – some learners are quite social while others are not.
- **Black Cluster:** Only one learner was assigned to this cluster. This learner seems to be extremely active, but his/her learning effectiveness is rather low. This means that although the learner spends a lot of time in the game and uses a lot of features, he/she does not improve his/her skills as it would be expected. This could be because he/she might be more distracted by some of the features (e.g., spending hours on looking through different leaderboards).

While those descriptions of axes and clusters are just exemplary, they demonstrate well how powerful this approach can be in finding out more about how learners use and play in the game and how/whether they benefit from the game. In addition, demographic attributes such as age range and gender can be used to further analyse the clusters (if those attributes are not already dominant in the principal components).

For example, we may see in the data that the distribution of male and female players is similar in the green cluster, where we have mainly players who just started to play and/or are not very active in the game. However, when looking at the other clusters, where we have players who are more active, we see that the percentage of female learners compared to male learners is significantly lower than it is in the green cluster. This may show that female players are not as active in the game and do not benefit much from the game due to their low activity status. Such findings can then be used to improve the game design to attract those learner groups (i.e., female learners) and add features that may make the game more interesting for them.

5 Conclusions

This paper presents a learning analytics approach to build learner profiles in the educational game OMEGA+. The profiles are created through cluster analysis and consider a variety of features related to learners' characteristics, possessions in the game and their activities in the game. The approach has been validated with simulated data from 47 players to demonstrate the insights and benefits this approach can provide.

Most related works focus either on identifying new information (e.g., personality traits, new performance metrics, etc.) from behaviour in an educational game [e.g., 5, 6] or on clustering based on performance in an educational game [e.g., 7–9]. However, the clusters in this approach are built by considering not only performance but a variety of other learner characteristics, their possessions in the game as well as their activities in the game. By considering such a diverse set of attributes when building the clusters/groups of learners, insights into how learners use the game, how they play and how they learn can be gained. In addition, by considering demographic attributes, investigations can

be conducted into the attractiveness of the game for different groups of learners. Such insights can be used to improve the game design, on one hand, to ensure that it is used the way it was intended and really provides learners with learning opportunities that are fun for them and, on the other hand, to broaden the reach of the game and make it attractive for more diverse groups of learners.

Future work will deal with using our approach on real data to learn more about the effectiveness and reach of OMEGA+. In addition, future work will deal with adapting our approach to other educational games and using it with real data for those games.

References

1. Siemens, G., Gašević, D.: Special issue on learning and knowledge analytics. *Educ. Technol. Soc.* **15**(3), 1–163 (2012)
2. Alonso-Fernández, C., Calvo-Morata, A., Freire, M., Martínez-Ortiz, I., Fernández-Manjón, B.: Applications of data science to game learning analytics data: a systematic literature review. *Comput. Educ.* **141**, 103612 (2019)
3. Chang, M., Graf, S., Corbett, P., Seaton, J., McQuoid, S., Ross, T.: OMEGA: a multiplayer online game for improving user's meta-cognitive skills. In: *Proceedings of the IEEE International Conference on Technology for Education (T4E 2019)*, pp. 178–185. IEEE Computer Society, Goa (2019)
4. OMEGA+. <https://omega.athabascau.ca>. Accessed 12 Apr 2022
5. Denden, M., Tlili, A., Essalmi, F., Jemni, M.: Implicit modeling of learners' personalities in a game-based learning environment using their gaming behaviors. *Smart Learn. Environ.* **5**(1), 1–19 (2018). <https://doi.org/10.1186/s40561-018-0078-6>
6. Loh, C.S., Sheng, Y.: Maximum similarity index (MSI): a metric to differentiate the performance of novices vs. multiple-experts in serious games. *Comput. Hum. Behav.* **39**, 322–330 (2014)
7. Slimani, A., Elouaai, F., Elaachak, L., Bakkali Yedri, O., Bouhorma, M., Sbert, M.: Learning analytics through serious games: data mining algorithms for performance measurement and improvement purposes. *Int. J. Emerg. Technol. Learn.* **13**(1), 46–64 (2018)
8. Lazo, P.P.L., Anareta, C.L.Q., Duremdes, J.B.T., Red, E.R.: Classification of public elementary students' game play patterns in a digital game-based learning system with pedagogical agent. In: *Proceedings of the 6th International Conference on Information and Education Technology (ICIET 2018)*, pp. 75–80. ACM, New York (2018)
9. Polyak, S.T., von Davier, A.A., Peterschmidt, K.: Computational psychometrics for the measurement of collaborative problem solving skills. *Front. Psychol.* **8** (2017)
10. Seaton, J.X., Chang, M., Graf, S.: Integrating a learning analytics dashboard in an online educational game. In: Tlili, A., Chang, M. (eds.) *Data Analytics Approaches in Educational Games and Gamification Systems*. *SCI*, pp. 127–138. Springer, Singapore (2019). https://doi.org/10.1007/978-981-32-9335-9_7
11. Géron, A.: *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow - Concepts, Tools, and Techniques to Build Intelligent Systems*, 2nd edn. O'Reilly Media Inc., Canada (2019)
12. Buitinck, L., et al.: API design for machine learning software: experiences from the scikit-learn project. *arXiv preprint arXiv:1309.0238* (2013)
13. Pedregosa, F., et al.: Scikit-learn: machine learning in Python. *J. Mach. Learn. Res.* **12**, 2825–2830 (2011)
14. Jolliffe I.T., Cadima, J.: Principal component analysis: a review and recent developments. *Philos. Trans. Roy. Soc. A Math. Phys. Eng. Sci.* **374**(2065) (2016)



Design and Evaluation of a Competency-Based Recommendation Process

Louis Sablayrolles^{1,2}(✉), Marie Lefevre², Nathalie Guin², and Julien Broisin¹

¹ IRIT, Université de Toulouse, CNRS, Toulouse INP, UT3, Toulouse, France
{Louis.Sablayrolles, Julien.Broisin}@irit.fr

² Univ Lyon, UCBL, CNRS, INSA Lyon, LIRIS, UMR5205, 69622 Villeurbanne, France
{Marie.Lefevre, Nathalie.Guin}@liris.cnrs.fr

Abstract. The purpose of recommending activities to learners is to provide them with resources adapted to their needs, to facilitate the learning process. However, when teachers face a large number of students, it is difficult for them to recommend a personalized list of resources to each learner. In this paper, we are interested in the design of a system that automatically recommends resources to learners using their cognitive profile expressed in terms of competencies, but also according to a specific strategy defined by teachers. Our contributions relate to (1) a competency-based pedagogical strategy allowing to express the teacher's expertise, and (2) a recommendation process based on this strategy. This process has been experimented and assessed with students learning Shell programming in a first-year computer science degree. The first results show that (i) the items selected by our system from the set of possible items were relevant according to the experts; (ii) our system provided recommendations in a reasonable time; (iii) the recommendations were consulted by the learners but lacked usability.

Keywords: Competency-based approach · Recommender system · Pedagogical strategy

1 Introduction

Recommender systems are a big focus of Technology-Enhanced Learning (TEL). They allow to provide relevant items [1] structured in sets or sequences, according to several constraints (e.g., new items, the most relevant items, or items linked to a learning path). We are interested in this article in recommender systems dedicated to learners.

Our context is the competency-based approach, and in particular, recommendations of learning resources based on learners' competencies. We focus on recommendation of ordered lists of resources to students. We studied similar works to design our proposition, and in particular the PERSUA2 model [2] which allows a teacher to configure the recommendation process according to her pedagogical approach. These works led us to propose our contributions to answer our research question: **How to propose a recommendation process based on a competency framework and corresponding to pedagogical practices?** Based on [2] and similar works, we define a pedagogical strategy independent of level and discipline in order to provide students with a personalized

recommendation which is compliant with teachers' pedagogy. We then design a recommendation process able to apply the strategy using a set of competencies linked together by semantic relations. Finally, we experimented our propositions on a real teaching context.

This article is structured as follows. First, we introduce the ComPer project in which our work take place, as well as our motivations. After a review of prior works on competency-based recommendation, we present the ComPer strategy and our recommendation process. Next, we describe the experiment in computer education we conducted to evaluate this recommendation process. Finally, we analyze and discuss the results of this evaluation before concluding and exposing future evolutions.

2 Project and Motivation

Our work take place in the ComPer project, whose purpose is to design models and tools to implement a competency-based approach to support personalized learning. The ComPer project environment is represented in Fig. 1.

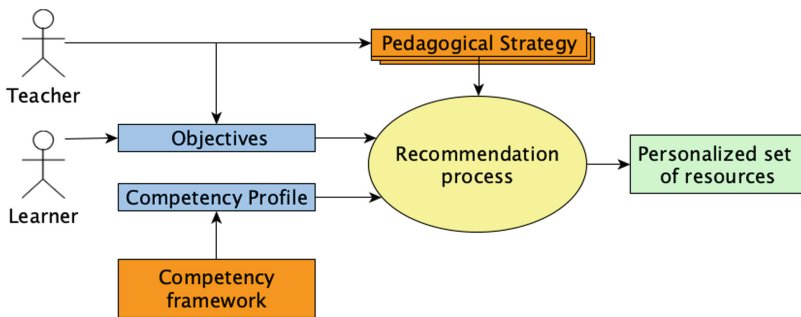


Fig. 1. The ComPer project environment

A *competency framework* (see Fig. 2 for an example) is a hierarchical set of knowledge, skills and competencies (KSC) linked together by semantic relations such as prerequisite, composition, or complexification. For example, a possible complexification of the skill “Performing multiplications with integers” could be “Performing multiplications with floats”. The objective of these relationships is to add meaning to the links between different KSCs. A competency framework is defined by domain experts or teachers who can also link pedagogical resources (i.e., courses, exercises) to one or more KSCs. This competency framework will have been defined by a preliminary work of the teaching team to model the subject. A learner *competency profile* matches with a competency framework, augmented by a mastery level for each KSC. The different mastery levels can be calculated automatically using data captured from activities of learners (e.g., success or failure when completing exercises related to the framework), or manually defined by teachers. The learning session context is defined by the *objectives* selected by the learner (i.e., one or more KSCs she wants to master) or by the teacher, but also by constraints such as the maximum number of resources to be recommended or the

maximum working time. The *recommendation process* we propose takes into account all the above elements, and uses the teachers' expertise through the *pedagogical strategy* they define to make the recommendation process as close as how they would recommend the pedagogical resources to their learners.

Due to the objectives of the project, our recommendation process must respect the following constraints: (C1) it must allow the expression of diverse pedagogical practices while remaining executable; (C2) it must be explainable to both learners and teachers; (C3) it must be configurable by the teacher. The first constraint (C1) is due to the fact that there are different competency frameworks that can be defined for different learning levels or subjects. The second constraint (C2) will help learners to understand why they get this recommendation based on their profile and for teachers to understand how the algorithm provided this recommendation. Finally, the third constraint (C3) stems from our goal of offering the teacher control over the system she uses, in order to offer more transparency.

The work presented in this paper focuses on the conception of the recommendation process and its evaluation in an authentic learning context. Before detailing our contributions, we present in the next section an overview of the existing approaches for competency-based personalization.

3 Competency-Based Personalization

Competency-based personalization processes can be divided into three main families: algorithms based on machine learning approaches; algorithms focusing on non-model-based approaches; and algorithms implementing model-based approaches.

In machine learning approaches, neural models can be combined as in [3] where a SVM generates learning paths before validation by a LSTM. In [4], the authors use probabilities to predict relationships between concepts, and an item/item matrix built by a Bayesian approach is used to recommend the best activities to learners. Other approaches use the learning paths of the best performing students to perform a clustering using learning paths defined by experts, or to use ant colony optimization [3]. However, all these approaches cannot be used in our context due to their lack of explainability (C2) and understandable configuration (C3).

Some works try to include the expertise of teachers in the personalization process. In [5], a system generating sequences of activities allowing to work on the pre-requisites of a skill was designed. This system, in order to make the recommendations, relies on the mistakes made by learners and on exercises that teachers have associated to these mistakes. However, this work only considers the pre-requisites and does not address other relations such as composition or complexification which are necessary in our context. On the other hand, the formalization of pedagogical scenarios based on pedagogical intentions (e.g., discover and reinforce) [6] can allow teachers to configure a system that gives them the opportunity to build a course based on a teaching strategy. This takes into account the activities to be carried out, the necessary resources, the interactions between individuals as well as the place where the scenario is carried out. However, these scenarios apply to a set of individuals, but in our context, we are only interested in an individual recommendation. Moreover, we only want to recommend an ordered list of educational

resources. Finally, the work of [7] offers a recommendation of pedagogical resources related to the course concepts, which are grouped in the form of a tree of possible learning paths. The learner is recommended the learning path defined by the teacher that will allow her to reach the highest increase of mastery estimated by the algorithm. These works show the possibility offered to teachers to link the concepts of the course using various relations such as pre-requisites or compositions. They can also drive the recommendation with pedagogical intentions. However, in our context, these works do not cover the diversity of all the semantic relations offered by our competency framework. Moreover, the defined pedagogical intentions seem interesting but not complete.

Closer to the constraints of our context, we found the following model-based work. The authors of [8] use the Cb-KST approach on a framework of competencies linked by compositional or prerequisite relationships. They then recommend a number of resources to the learner from a set of competencies chosen by the teacher (i.e., the objective) according to difficulty parameters she defined. This recommendation can be modified using several predefined strategies (Progression, Reinforcement, Deepening). However, the strategies defined cannot be configured according to the needs of the teacher, who can only choose the one most suited to her needs.

In the work by [9], which proposes a competency framework structured through composition relationships, the authors use a multi-agent system to recommend resources associated to the element closest (defined using a distance measure) to the one the learner wants to work on. In [10], the domain is modeled according to an ontology of different elements (concepts, notions, knowledge, and subjects) that are linked together by a compositional weight relationship. The model of the learner is built using a Bayesian approach on trace data. Using this profile, tasks related to a target concept are recommended to the learner. However, it can be complicated for teachers to set up a framework with weights for each relationship.

The need for generating recommendations that are explicit to both students and teachers (C2) makes automatic approaches unusable in our context. In the context of the project, the experts model the domain but not the learning paths, so it is impossible for us to use the learning path generation approaches. Our recommendation is only generated from the learner profile at the time of the recommendation. The difficulty to parameter and explain the general behavior of a statistical approach can be a barrier for us (C2 and C3). Thus, in order to recommend resources to learners according to their competency profile and objectives, we identify the following issues: (i) how to reproduce the teacher's expertise through a pedagogical strategy; (ii) how can the semantic relationships of the competency framework be used within the pedagogical strategy; (iii) how a recommendation process can use these pedagogical strategies to recommend resources to a student. In other words, we try to tackle the following research question: **(RQ) How to propose a recommendation process based on a competency framework and corresponding to pedagogical practices?** We make the following assumptions: (h1) our approach allows teachers to provide guidance on the recommendation process; (h2) it is possible to use a competency framework as a structure to provide recommendations; (h3) the recommendations provided satisfy teachers.

In our context, it seems interesting to reuse the concepts of *learner objective, intention* and *pedagogical strategy* so as to respect the project's constraints (C1, C2, C3).

Finally, based on the work by [2], we will use the concept of “pedagogical strategy”, i.e. “a set of rules which allows selection of activities according to the set of available activities and to the profiles of learners”, where a “rule” is of the form IF [conditions on the profiles] THEN [activities to be proposed] ELSE [other activities to be proposed]. To answer the research question, our approach consisted in (i) defining a pedagogical strategy independent of the level and subject taught; (i) proposing a recommendation process that applies the pedagogical strategy to a learner’s competency profile based on her objectives. Our approach will then show how it is possible to provide recommendations, part of a competency-based approach, through a process that corresponds to pedagogical practices.

4 Personalized Recommendation of Resources

We followed a user-centered approach to design our recommendation process. Using a non-standardized questionnaire, we asked the teachers involved in the project some questions about their personalization needs, and debriefed the answers with them. We then designed a first pedagogical strategy (i.e., the ComPer strategy) and a recommendation process able to use this strategy. To illustrate our proposal in the remaining of this section, we take as an example an extract of the Shell programming framework illustrated on Fig. 2 and used in the experiment described in Sect. 5.

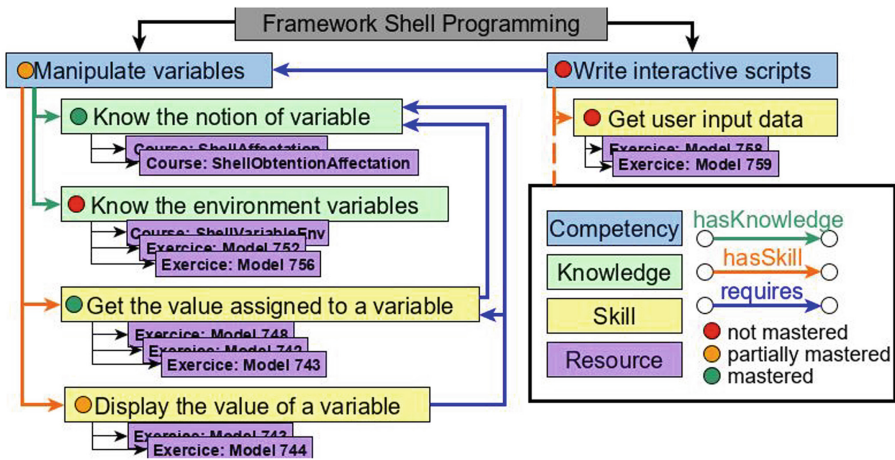


Fig. 2. Extract of a Shell programming profile

4.1 Strategy ComPer - A Strategy Independent of Level and Subject

Our personalization strategy uses intentions. They qualify the way in which the teacher and/or the learner wishes to work on a node N of the competency framework. We defined six intentions in the ComPer project. The **Prerequisite** intention consists in working on

themes that are prerequisites for N. The **Discovery** intention allows students to work on N for the first time. The **Support** intention enables to work on N, which has already been studied but is not mastered. The **Revising** intention makes students work on N in case of an exam, for example. The **Deepening** intention consists in working on nodes that are complexifications of N. Finally, the **Default** intention is used when the learner does not know how she wants to work on N.

Based on these intentions, it is possible to define instructional strategies characterized by different behaviors. In the case of the ComPer strategy, working an objective N with the **Pre-requisite** intention consists in working on the KSCs that are either prerequisites of N or pre-requisites of one of its descendants. To do this, our algorithm (1) retrieves the set of KSCs composing the goal node N by following the relationships *hasSkill*, *hasKnowledge*, and *isComposedOf*; (2) selects, for all components, the pre-requisite nodes by following the relationship *requires*. The set of unmastered pre-requisites are then selected so that the learner can work on them. If we take as example the framework defined in Fig. 2 with [*Write interactive scripts*, *Pre-requisite*] as the objective, our algorithm will check if the competency *Manipulate variables* is mastered. Since it is not mastered, the algorithm selects resources associated to knowledges and skills attached to that competency that are not mastered (*Know the environment variables* and *Display the value of a variable*).

In the case of the ComPer strategy, working an objective N with the **Discovery** intention consists in working the KSCs that are descendants of N and that the learner has not yet worked on. If we take as example the framework defined in Fig. 2 with [*Write interactive scripts*, *Discovery*] as the objective, the algorithm will select the descendant skills and knowledges of this node that the learner has not yet worked on.

The ComPer strategy presented above is independent of the learning level and the subject to be taught, as it has been defined on top of the structure of the competency model, exploiting the semantic relationships. Therefore, for any competency framework conform to this model, the ComPer strategy can be applied. In the ComPer project, physics, French, and computer science frameworks have been defined.

4.2 Recommendation Process

The objective of the recommendation process is to produce an ordered sequence of resources relevant to the learner. As illustrated in Fig. 2, the competency model we use has a graph structure. Due to the high complexity of graph exploration, and according to the interactions with the project teachers, we chose to treat KSCs and resources in two separate phases, in order to facilitate the configuration and explainability of our system. More precisely, our process is composed of 3 main phases.

The **Selection phase** chooses which KSC of the framework the learner needs to work on, according to her competency profile and the objectives that have been defined. During this phase, tags are placed on each selected node to indicate why it should be worked on, for the purpose of explaining the recommendation. The **Ordering phase** sorts the selected nodes so that the learner works on the most important ones first. Finally, the **Resource phase** consists in retrieving the resources (e.g., exercises, courses) to be recommended to the learner for each of the selected nodes. These three phases are applied independently for each objective formulated by the teacher or the student.

The KSC **Selection phase** consists of three steps. First, the algorithm follows the composition relations (*i.e.*, *hasKnowledge*, *hasSkill*, *isComposedOf*) to extract the sub-components (KSC) of the objective. Then, if needed (like with the *Pre-requisite* intention), transversal relations are followed (e.g., *requires/isRequiredBy*, *isLeverOfUnderstandingOf/isUnderstoodBy* or *isComplexificationOf/isComplexifiedBy*). Finally, we process the KSC resulting from the last step: for each node, the algorithm selects the node if it validates some condition(s) of the selection rule (e.g., is the node mastered, partially mastered, or not mastered). During the selection phase, a priority is also applied to each node. It is calculated based on the priority of the parent node, the relationship(s) that was/were followed, the node's depth relative to the goal node, and the selection rule applied. The priority is calculated using the formula (1). We defined two functions: $f_{\text{hierarchical}}$ calculates a value from the path taken by the algorithm from the objective node to the current node, and f_{node} calculates a value from mastery values associated to the current node. The parameter α allows to balance these 2 functions.

$$Priority_{node} = (1 - \alpha) \times f_{\text{hierarchical}}(node) + \alpha \times f_{\text{node}}(node) \quad (1)$$

The priority associated to each selected node is used during the **Ordering phase** to sort the nodes. This phase consists in regrouping the multiple selections of nodes and ordering them using the priorities.

Finally, using the ordered list of KSCs, the **Resource phase** retrieves the resources associated with these nodes in three steps: (1) *R.Selection* retrieves the set of resources related to the selected nodes by following the relationships *hasLearning* and *hasTraining* between a Knowledge/Skill and a resource; (2) *R.Ordering* sorts the selected resources to provide, for example, courses before exercises; (3) *R.Restriction* removes, if needed, some of the selected resources to conform with time or number constraints.

5 Experiment in Shell Programming

The experiment presented here aims at evaluating the ComPer strategy in a first real learning context: computer learning in higher education.

5.1 Experimental Settings

The experiment we conducted to evaluate our proposal was based on the Lab4CE platform dedicated to computer education [11]. The competency framework and the associated resources were designed by the teacher in charge of the module. It included a total of 96 nodes (11 Competencies, 24 Knowledge, 19 Skills and 42 resources) linked together by a total of 193 relations.

The experiment was conducted in five phases: (i) lecture; (ii) practical work session; (iii) competency profiles update by manual evaluation; (iv) generation of a recommendation for each learner according to the objectives of the week, defined by the teacher in charge of the module; (v) use of the recommendation by the learner. The recommendations consisted of a set of resources including lectures and exercises. The lecture resources were extracts from one or two slides of the lecture. The exercises were models created by the teacher in charge of the module on the ASKER platform [12]. The

instructional strategy was a combination between lecture and the ComPer strategy. The experiment took place during 6 weeks with 180 students enrolled in a first-year computer science degree, in the context of a Shell programming learning module involving 6 teachers.

Each week was organized as follows: first, the teacher in charge of the module gave a lecture. Then, a distance learning session with one of the six teachers of the module was carried out with a series of exercises to complete. The Lab4CE platform was accessible online at any time, and the students could continue their work throughout the week. On Friday evening, the logs of the platform were extracted in order to update the competence profiles of each learner. The automatic update from the logs was not yet operational, so it had to be done manually by two teachers for all the learners for each concept involved in the exercises. On Sunday evening, the recommendation process implementing the ComPer strategy allowed to generate all the recommendations according to the objectives provided by the teacher in charge of the module. Each recommendation consisted of an ordered list of a variable number of resources. So, learners could, starting on Monday of the next week, visit the recommended resources or not.

Finally, concerning the settings of the ComPer strategy, at the request of the teacher in charge of the module, the selection procedure corresponding to the Discovery intention has been deactivated and the resources have been ordered by giving priority to courses over exercises.

5.2 Collected Data

During this experimentation, we collected the learners' logs on the Lab4CE platform when they performed the practical work. We also collected the generation logs of each recommendation. By means of a non-standardized questionnaire, we then asked the 6 teachers of the learning module to provide their recommendations, which will be labeled as baseline recommendations, for two learner profiles and two objectives, in order to compare them with the algorithm's recommendations. Also, we asked them to evaluate the recommendations provided by our system for two others profiles and two objectives using a 5-value Likert scale.

5.3 Analysis Performed on the Collected Data

To evaluate our process, we used the framework by [13]. It includes three categories of criteria to assess a recommendation system in TEL: technical, educational, and social network. Let us note that, since no social data are available in the context of the ComPer project, our evaluation does not address the social category. The framework suggests three technical criteria: (i) **accuracy** measures how close a set of predicted ranking of items for a user differs from the expert's true ranking of items; (ii) **coverage** measures the proportion of recommended items among all possible items; (iii) **performance** observes if a recommendation is provided in a reasonable time frame. The educational category includes four main criteria: (i) **effectiveness** is a sign of the total amount of completed, visited, or studied resources during a learning phase; (ii) **efficiency** indicates the time that learners needed to reach their learning goals; (iii) **satisfaction** reflects the individual

satisfaction of the learners with the given recommendations; (iv) **dropout rate** mirrors the numbers of learners that dropout during the learning phase.

Table 1. Evaluation criteria and metrics of the recommendation process

	Criterion	Metrics
Technical criteria	Accuracy [14]	<ul style="list-style-type: none"> – Recall(r) indicates the proportion of the k-recommended items ($k = 4$ here) that are relevant to a user (i.e., correctly recommended items), to all items relevant to a user – Precision(p) indicates the proportion of the k-recommended items that are relevant to a user – F1-score (F1) calculates the harmonic mean of r and p – nDCG is a ranking quality metric that calculates usefulness scores (gains) of items based on their relevance and position in a list of k-recommended items – Recommendation qualitative analysis
	Coverage	– Percentage of recommended nodes and resources
	Performance	– Time to generate the recommendation (median, Q1, Q3)
Educational criteria	Effectiveness	– Mean percentage of recommended items that have been accessed
	Efficiency	– Compares the level of proficiency of users who accessed the recommended items versus users who did not access the recommended items
	Satisfaction	– Quantitative analysis of a questionnaire delivered to the students
	Dropout rate	<ul style="list-style-type: none"> – Percentage of users who never accessed the recommended items – Percentage of users who accessed the recommended items after week 1, and who did not access them after week 2 and 3

To measure the accuracy of the recommendation process, we first performed a manual clustering of the recommendations of weeks 2 and 3, and selected 4 representative learner profiles per week; we chose weeks 2 and 3 because they were located in the middle of the experiment and allowed to evaluate the post-starting accuracy. For the analysis of data collected over the teacher questionnaire, we relied on the metrics detailed in Table 1, to compare the expert recommendations with those of our algorithm. In addition, we computed the average of the teachers' evaluations regarding the quality of the recommendations generated by our system. Finally, we performed a qualitative analysis by representing the recommendation of our process and of the experts in the competency framework to see how close recommendations are, and compared the results between algorithm/expert and expert/expert. Regarding the educational criteria, we relied on the

logs of the learners' learning activities, the recommendation logs and a questionnaire delivered to learners after the experiment.

6 Results and Discussion

In this section, we present the results of the experimental data. We detail and discuss the results obtained for each of the criteria of Table 1.

Table 2. Accuracy metrics of our recommendation system for week 2 and 3

	Algorithm/expert				Expert/expert			
	Week 2		Week 3		Week 2		Week 3	
	Mean	Median	Mean	Median	Mean	Median	Mean	Median
p@4	0.65	0.75	0.54	0.50	0.54	0.50	0.44	0.50
r@4	0.65	0.75	0.54	0.50	0.54	0.50	0.44	0.50
f1@4	0.65	0.75	0.54	0.50	0.54	0.50	0.44	0.50
NDCG@4	0.60	0.54	0.52	0.50	0.61	0.61	0.52	0.50

Results in Table 2 show that our system is able to select KSCs a little better than teachers. However, the teachers agree more with each other about the order of these KSCs than our system does with them.

Concerning the evaluation of the relevancy of the recommendations by the teachers, the average score for the two weeks was 3.24/5 with 3.83 for week 2 and 2.66 for week 3. Furthermore, a qualitative analysis using a representation of the process recommendations and the experts' recommendations showed that the selected nodes were close to each other. However, the order in which they were selected was quite different between our process and the experts and between the experts.

For the coverage metrics, as shown in Fig. 3, our process has during the first two weeks recommended few nodes to work on (average coverage of 14% for the first and 13% for the second). While in weeks 3 and 4, the average coverage slightly increases (respectively 19% and 15%). In our case, the lower the coverage value the better because that means that the recommendation process can select the better resources that will help the student to improve. This low value shows that our system is able to choose some items for the learner among a number of items. Concerning the performance of our algorithm, we notice that the more the experiment is in progress, the lower the average time of recommendation is. We can see, for example, that the median recommendation times for weeks 1 and 2 are quite high (9 s and 5 s respectively) compared to those of weeks 3 and 4 (0.06 s and 0.01 s respectively). Indeed, after further investigations, we noticed that the part of our algorithm responsible for tracking its own behavior (for analysis purposes) prevented the recommendation tasks to be executed correctly.

The platforms on which activities were hosted did not allow us to obtain precise interaction data, so we were not able to calculate the educational criteria except for

satisfaction using a questionnaire. For that questionnaire provided to learners, a total of 137 responses were recorded of which 99 were complete. The major positive point of learners about the recommendations was the relevance of having course and exercises in the same place. However, there were mixed opinions on the number of resources and the relevance of the recommendations, which does not allow us to identify a general trend. On the whole, the recommendation system has been consulted but its usefulness and usability have not been perceived by all students. With regard to the satisfaction of the learners about recommendations, the feedback is mixed. Further experiments are needed to determine whether the feedback is due to the recommendations themselves, or to their incorporation into the platform.

The evaluations of the technical criteria show that our process recommends items as relevant as the experts. However, each teacher orders the selected KSCs according to her expertise. This problem with the ordering may explain the average scores of the experts regarding our recommendations. Other qualitative studies based on questionnaires and interviews have to be conducted to identify which strategies are used by teachers to order the nodes delivered to learners.

The results also show that our algorithm is able to provide a recommendation in an acceptable time frame for a user. The more information the algorithm has about the learner's mastery of the KSCs, the shorter the generation time.

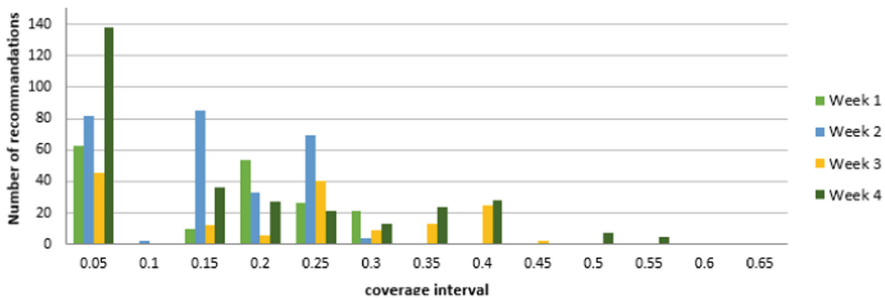


Fig. 3. Repartition of nodes coverage by week

Regarding the results of the KSC coverage, the peak at the low values can be explained by the fact that the rules concerning the discovering intention have been deactivated, or that the learners have already mastered the nodes attached to the objectives. Thus, not having been able to recommend resources to assess the learner's mastery, the coverage was low. The coverage results obtained are however to be nuanced as they strongly depend on the number of objectives of the week, as well as on the number of descendants of each of these objectives. This number of descendants increases over the weeks (20, 11, 22, 47) because the objectives to be worked on have more pre-requisites. In general, the coverage is around 25% of the nodes. This might be due to the fact that this framework is not deep and has not many nodes.

To summarize, the results show that (i) the items selected by our system from the set of possible items were relevant according to the experts (validating hypothesis h3); (ii) our system provided recommendations in a reasonable time; (iii) the recommendations

were consulted by the learners but lacked usability. So, we were able to propose a contribution that generate recommendations for learners based on their competency profile and objectives using a 3-phases process (validating hypothesis h2). To reflect pedagogical practices, our recommendation process relies on a pedagogical strategy that can be set up by a teacher as it was done during the experimentation we conducted (validating hypothesis h1).

7 Conclusion and Future Works

In this paper, we designed, experimented and evaluated a competency-based recommendation system. In particular, we (i) defined intentions to modulate the personalization according to the needs of the learner; (ii) conceived a strategy that uses the structure of the competency framework to recommend activities based on the learner's profile and teacher's specific requirements; (iii) designed a recommendation process in three phases that applies the pedagogical strategy. The results show that (i) our recommendation process and the ComPer strategy recommend items relevant according to the experts; (ii) the recommendation is done in a reasonable amount of time; (iii) the recommendations were consulted by the learners. However, our study is limited by the fact that some educational criteria could not be obtained as well as by the presence of a unique experimental field, which does not allow us to support, generalize or contradict the results obtained.

For the next part of our work, we want to continue to evaluate our ComPer strategy and the recommendation process with different disciplines and educational levels, with more complex competency frameworks. Precisely, we will study how to improve the way recommendations are provided to learners by conducting new experiments that will confirm or not the results and conclusions obtained. We will also investigate how to facilitate configuration of the strategy. Indeed, we plan to make the ComPer strategy customizable by teachers using a dedicated user interface.

To do so, we believe that the ontological structure of the competency framework and the design of the recommendation process in the form of rules will make it possible to provide explanations on the recommendations made to both teachers and learners. These explanations are expected to allow teachers to understand how the parameters of the strategy act within the recommender system, so thus to allow them to adapt the strategy to their exact needs.

Acknowledgement. The work presented in this article was funded by the ANR within the ComPer project ANR-18-CE38-0012.







References

1. Ricci, F., Rokach, L., Shapira, B.: Recommender systems: introduction and challenges. In: Ricci, F., Rokach, L., Shapira, B. (eds.) *Recommender Systems Handbook*. Springer, Boston (2015). https://doi.org/10.1007/978-1-4899-7637-6_1
2. Santos, O.C., Kravcik, M., Pérez-Marín, D.: Personalization approaches in learning environments. In: Ardissono, L., Kuflik, T. (eds.) *UMAP 2011*. LNCS, vol. 7138, pp. 117–121. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-28509-7_12

3. Diwan, C., Srinivasa, S., Ram, P.: Automatic generation of coherent learning pathways for open educational resources. In: Scheffel, M., Broisin, J., Pammer-Schindler, V., Ioannou, A., Schneider, J. (eds.) EC-TEL 2019. LNCS, vol. 11722, pp. 321–334. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-29736-7_24
4. Desmarais, M.C., Meshkinfam, P., Gagnon, M.: Learned student models with item-to-item knowledge structures. *UMUAI* **16**(5), 403–434 (2006)
5. Nabizadeh, A.H., Gonçalves, D., Gama, S., Jorge, J., Rafsanjani, H.N.: Adaptive learning path recommender approach using auxiliary learning objects. *Comput. Educ.* **147**, 103777 (2020)
6. Auzende, O., Giroire, H., Le Calvez, F.: Error driven adaptive training sequences. In: IEEE ICALT 13th, ICALT 2013, Beijing, China, pp. 219–220. IEEE (2013)
7. Emin, V., Pernin, J.-P., Guéraud, V.: Model and tool to clarify intentions and strategies in learning scenarios design. In: Cress, U., Dimitrova, V., Specht, M. (eds.) EC-TEL 2009. LNCS, vol. 5794, pp. 462–476. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-04636-0_44
8. Melero, J., El-Kechai, N., Labat, J.-M.: Comparing two CbKST approaches for adapting learning paths in serious games. In: Conole, G., Klobučar, T., Rensing, C., Konert, J., Lavoué, É. (eds.) EC-TEL 2015. LNCS, vol. 9307, pp. 211–224. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-24258-3_16
9. Paquette, G., Mariño, O., Rogozan, D., Léonard, M.: Competency-based personalization for massive online learning. *Smart Learn. Environ.* **2**(1), 1–19 (2015). <https://doi.org/10.1186/s40561-015-0013-z>
10. Mediani, C., Abel, M.-H., Djoudi, M.: Towards a recommendation system for the learner from a semantic model of knowledge in a collaborative environment. In: Amine, A., Bellatreche, L., Elberrichi, Z., Neuhold, E.J., Wrembel, R. (eds.) CIIA 2015. IAICT, vol. 456, pp. 315–327. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-19578-0_26
11. Broisin, J., Venant, R., Vidal, P.: Lab4CE: a remote laboratory for computer education. *Int. J. Artif. Intell. Educ.* **27**(1), 154–180 (2015). <https://doi.org/10.1007/s40593-015-0079-3>
12. Guin, N., Lefevre, M.: An authoring tool based on semi-automatic generators for creating self-assessment exercises. In: 14th International Conference on Computer Supported Education, 22 April 2022, Online (Portugal), pp. 181–188 (2022)
13. Drachsler, H., Hummel, H.G.K., Koper, R.: Identifying the goal, user model and conditions of recommender systems for formal and informal learning. *J. Digit. Inf.* **10**(2), 4–24 (2009)
14. Kopeinik, S., Kowald, D., Lex, E.: Which algorithms suit which learning environments? A comparative study of recommender systems in TEL. In: Verbert, K., Sharples, M., Klobučar, T. (eds.) EC-TEL 2016. LNCS, vol. 9891, pp. 124–138. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-45153-4_10



A Classification Approach to Recognize On-Task Student's Behavior for Context Aware Recommendations

Lisa Roux^(✉) , Thierry Nodenot , Patrick Etcheverry , Pantxika Dagorret ,
Christophe Marquesuzaa , and Philippe Lopisteguy 

Université de Pau Et Des Pays de L'Adour, E2S UPPA, LIUPPA, Anglet, France
{eroux, thierry.nodenot, patrick.etcheverry, pantxika.dagorret,
christophe.marquesuzaa,
philippe.lopistegu}@iutbayonne.univ-pau.fr

Abstract. The increasing development of e-learning systems has raised the necessity to apply recommender systems with the aim of guiding learners through the various courses, activities, etc. at their disposal. The learner-oriented approaches allow the recommendations to fit the user's needs as precisely as possible. Nevertheless, due to the multiplicity of possible educational situations and individual particularities, offering adaptive recommendations and diversity is still a major challenge. In order to improve this aspect and provide the learner with recommendations appropriate to both their current specific needs and general profile, we focus on a hybrid system whose knowledge will be augmented through the learner's activity and results. This system will base its analyses and future recommendations according to the evolving student's profile and behaviour during the task. For that purpose, a first step is to categorize the on-task student's behaviour. This paper focuses on this problem and proposes a model, provided by educational sciences, on which the recognition process could be based.

Keywords: Learner behaviour model · E-learning · Supervised classification · Recommender systems

1 Introduction

In recommender systems (RS), two important components need to be considered: the users and the items. The main challenge is to recommend a short selection of appropriate items (i.e. items that fit the user's needs or interests) to a given user, in order to help them to choose from a wide variety of items. In e-learning, the recommendation is mostly based on the content of the items and the collected data that describes the online user's behaviour. Providing the learner with appropriate learning recommendations is a major challenge, since each learner has specific needs and way of learning, depending on their own interests and cognitive involvement, and the learning situation.

In order to provide personalized recommendations taking into account the evolution of the on-task learner's behaviour, we propose in this paper a learner model relying

on educational sciences and aiming at describing the cognitive strategies of a learner during a problem-solving activity. This learner model is part of a RS architecture that we detailed in [1]. The learner model presented in this study has been chosen for its genericity: it is expected that it can be trained and used for any problem-based learning [2] in practicum-centered situations (i.e. practical exercises, projects) [3]. Each situation is described as a series of tasks through the guidelines provided to the students, a work environment with the suitable software so that the students can perform the task and possibly collaborate, a set of resources (e.g. numerical documents, work files produced by the students), success criteria for each task, an available support through a LMS comprising aids, resources, tips, components for a solution, course supplements, etc.

This paper focuses on the student's current behaviour classification and is organized as follows: related works on recommendation approaches and e-learning RS is given in Sect. 2; after having described the cognitive model on which the proposed solution is based and the overall functioning of this solution, Sect. 3 show the first results that we obtained; Sect. 4 outlines conclusions of this study.

2 Related Works

E-commerce RS are mostly based on matrix factorization [4]. A matrix of n users and m items is used to represent the data on which the calculations will be done, each matrix cell corresponding to the rating given to item i by the user u . Collaborative and content-based filtering are widely used methods because they are both efficient and easy to implement, but the knowledge based filtering and hybrid approaches can also be employed. The education RS are based on the same filtering methods in a variety of approaches, from matrix factorization (e.g. [5, 6]) to complex classification methods, such as fuzzy-tree [7], convolutional neural networks [8], neuro-fuzzy technics [9, 10], etc. In some works, deep attention has been paid to allow the teacher to implement their own rules [9]. When used, the learner model is mostly based on their knowledge background [9], learning objectives, past activities [11], learning style [12], knowledge level, and sometimes several of them [7, 9, 13]. These models can be categorized as follows: the models based on the student's learning trajectory and those based on their educational needs.

However, so far, it appears that none of the existing e-learning RS make recommendations based on both aspects. Moreover, using the methods of e-commerce RS in the education area raises the question of pedagogical efficiency and some ethical issues. Similarly, the selection of the learner model and the choice of basing (or not) the recommendations on a teaching model are crucial stakes that require careful attention. In accordance with the proposals that we have made in previous researches [14] that explored the ethical and epistemological issues of e-learning recommender systems, the learner model presented in this paper is provided by the educative sciences. It has to fulfill two requirements [1]:

1. The classification of students must be meaningful.
2. It must be pedagogically useful and relevant.

3 Proposed Solution

3.1 Description of the student's Behaviour: The Mode of Reasoning and the Degree of Activity

When a learner is working on a pedagogic task, they can be more or less confident, proactive, hesitating, etc. and the teacher generally takes into account the signs of ease when they want to help and guide them: they recommend to read a document, see another similar task, etc. depending on the student's needs, that is, how much they feel comfortable with the issues to be addressed during the task. However, although it is well documented that the learners' emotions when completing a task have a significant impact on their performances and stance (i.e. perseverance, dropping out) [15, 16], we can hardly automatically determine how they feel during the task: one's feelings are internal thus very difficult to assess for an outside observer.

In some papers, the emotion recognition is operated through various technical or technological devices (e.g. computer vision [17, 18]), smart cushion [19], voice recognition [20]) but, since we want our system able to work with reduced hardware costs, we chose to base our system on a learner model that describes their behaviour – thus on an observable learner activity – and that comes from the educational sciences.

In the problem-solving activities, learners alternate between two main modes of reasoning, which are exploration and exploitation.

Exploration: This strategy aims at experimenting with new alternatives or at acting on the environment in order to generate new stimuli [21]. In our experiments, we expect that the learners undertake manipulative actions on tools, environment, available objects, etc., trying and testing different combinations.

Exploitation: This strategy consists in the use of existing knowledge (declarative, procedural) in a given situation. It aims to use it in order to build hypotheses, ideas, and strategies to suit the problem-situation they meet, then get involved in a reality-check. In our experiments, we expect that the learners test their hypotheses/approaches (e.g. their specifications) so that they can validate them or not.

The relationship between the exploitation of old certainties and the exploration of new possibilities is a central concern of studies in adaptive processes [21]. Sometimes learners have to take time to exploit existing knowledge, but they also may have to explore the situation in different ways in order to handle and test the various means at their disposal, so that they can develop new knowledge then raise new solutions [22].

There are two main advantages coming from this model. First, although there can be ambiguous situations difficult to categorize, there are easily observable practice patterns that allow a good class definition for machine learning. Moreover, these behaviours, although objectively observable, can be an expression of the students' ease during the task. For example, whether a learner continues to use the exploitation problem-solving strategy while facing difficulties that they fail to overcome, one possible explanation might be that they lack confidence in exploring new insights, new ways of using the available tools, etc. On the contrary, a learner who would keep trying to explore without success could lack both theoretical and practical knowledge about the environment or

the tools. In the former situation, the learner could be encouraged to perform easy tasks in order to build confidence, while in the latter, they could be advised to strengthen the knowledge and skills necessary to complete the task in order to build appropriate hypotheses. In this way, appropriate recommendations can be made to the student, based on the history of their modes of reasoning during the current task, combined with two other information: the amount of time they stay in the same mode and the degree of difficulty encountered. The latter is calculated through a questionnaire.

3.2 Learning Steps

Our system requires two intertwined learning steps: the former to determine the classification rules of the student's current reasoning mode, the latter to define the function used to identify their current activity level given their reasoning mode. We composed six indicators, which have been selected because they fulfil three requirements:

1. They can be obtained by aggregating data describing the mouse and keyboard actions of the student
2. They have to participate to describe both the student's current reasoning mode and activity degree
3. They have to be generic (independent to the task), because the final recommender system is expected to work for any problem-solving task.

In our proposal, the student's behaviour is described by a 7-dimensional vector:

Frequency of Document Shift: This indicator shows whether the student regularly shifts back and forth across the available pedagogical and work documents. It is the average of the number of times the student has launched or clicked on another document than the active one in five minutes.

Frequency of the Work Verification: This indicator shows whether the student regularly checks the validity of their work. In the case of a programming task, it is calculated by dividing the number of times the student has launched the compilation operation by the number of complete structures in their program. This is an optional indicator, since not all software programs can offer such a functionality.

Rate of Activity: This indicator assesses whether the student intensively uses the keyboard and the mouse (e.g. clicking, rolling the mouse wheel, using the scrollbar), in comparison with the average students' rate of activity (number of actions per minute).

Rate of Writing: It indicates how much the student writes, on average, in comparison with the average students' rate of writing (number of written character per minute).

Frequency of Erasure: This assesses the student's certainty about what they write. It is defined as the number of times the student presses an erase-key per minute.

Rate of the Working Document Elasticity: This information is described by two indicators. The frequency of erasure is insufficient to evaluate adequately the student's production progress. Indeed, when we detect that they have pressed an erase-key, we do not know whether they have erased only one character or a bigger selected extract of the text. This indicator reflects the degree of variability of the size of the working document S_i . S_i is measured at regular time intervals (30 s). The indicators are calculated as follows, based on the average size of a document and its standard deviation:

$$m = \frac{\sum_{i=1}^n |S_i - S_{i-1}|}{t} \quad (1)$$

$$R_e = \sum_{i=1}^n \left| \frac{|S_i - S_{i-1}|}{t_i - t_{i-1}} - m \right| \times \frac{t_i - t_{i-1}}{n} \quad (2)$$

3.3 Classification and Evaluation

Data collection ranges from the acquisition, at runtime, of the different sources of traces (collection per student's workstation) to the gathering of these traces into an object-model that can further be exploited (queries, mining). At the end of that process, described in [23], in average 500 macro-interactions per student were collected. We obtained the archives of 60 students' actions. In order to classify them, we labelled manually these data, by watching the 60 students working on a programming task.

Several classification strategies have been investigated in our work. Since the student's activity depends on their current problem-solving strategy, we considered the cascade hybridization method, which performs the problem-solving strategy classification followed by a regression of the activity.

Concerning the classification step, due to the specificities of our data sample (i.e. small, based on surveys carried out by human observers), only classification methods that can be applied on small samples and robust to noisy measurements may be used. For the first step, which consists in classifying the data into two classes (i.e. exploitation and exploration), three different classification methods were therefore considered: Random Forrest (RF), Naïve Bayes (NB), KNeighbors (KNN). Indeed, KNN is mostly used for small samples with few explanatory variables, since the algorithm speed slows as the number of observations increases and the results are less accurate with many characteristics. NB does not require huge amount of data neither, and is not sensitive to irrelevant characteristics. Although it is efficient on large dataset, RF can also deal with little ones. Moreover, it reduces overfitting, which is particularly harmful for noisy datasets. We used 75% of the original dataset to train the model and 25% to test it. Due to the scattered data distribution, we used KNN with $k = 2, 3, 4, 5$. We calculated the error rate, which computes how much items are wrongly classified among the testing data set, and the f1-score for each classification. The results are reported in Fig. 1 and show that the classification is better when $k = 3$. Basically, the f1-score is the harmonic mean between precision and recall [24, 25], which are respectively how many of a class is found over the whole number of elements of this class and how many are correctly classified among that class:

$$\text{Recall} = \frac{\text{number of correctly recommended items}}{\text{number of interesting items}} \quad (3)$$

$$\text{Precision} = \frac{\text{number of correctly recommended items}}{\text{number of recommended items}} \tag{4}$$

$$\text{f1 - score} = \frac{2 \times (\text{Recall} + \text{Precision})}{\text{Recall} + \text{Precision}} \tag{5}$$



Fig. 1. Evaluation of k-NN classification with k = 2, 3, 4, 5

We compared the results by using those three classification methods, that is 3-NN, RF, NB. As Fig. 2 shows, results are better with 3-NN, although they look quite satisfying in RF and NB, with a quite little error rate. In order to confirm these observations, we used SHAP, which is a game theoretic approach to explain the output of any machine learning model. It connects optimal credit allocation with local explanations using the classic Shapley values from game theory and their related extensions [26].

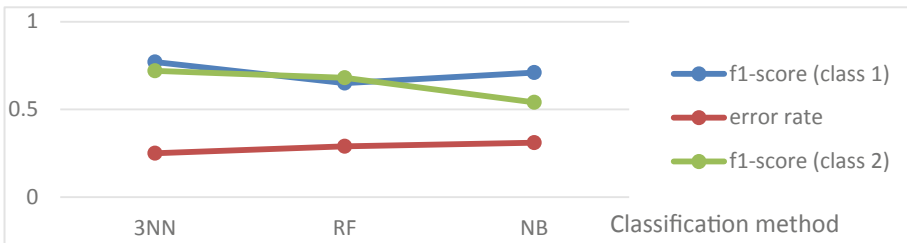


Fig. 2. Evaluation of 3-NN, RF, and NB classifications

The results obtained using SHAP shed new light on these first results. Figure 3 shows that the characteristics used to classify data are not really decisive for 3-NN, whereas the distinction of classes based on the same characteristics seem relevant for RF (Fig. 4).

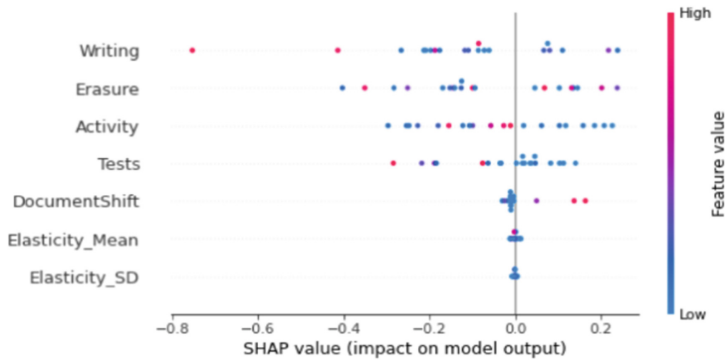


Fig. 3. SHAP on 3-NN classification

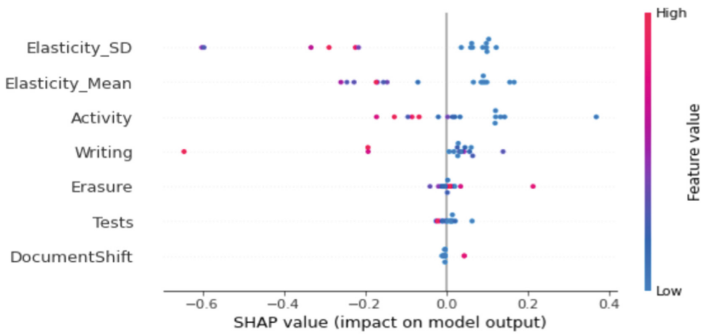


Fig. 4. SHAP on Random Forest classification

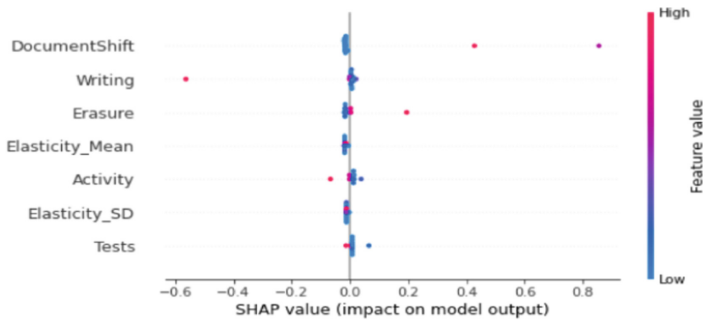


Fig. 5. SHAP on Naïve Bayes classification

The classification provided by NB does not appear to be very conclusive (Fig. 5), since the characteristics seem not discriminating, except for a very reduced number of individuals. All these results lead us to think that our model is quite promising, especially using 3-NN and RF, but has to be enhanced, notably through improvements to the dataset, in particular by increasing the number of observations, and testing classifications using multiple versions of our existing characteristics. Indeed, we can use some descriptors that, although repetitive, will describe data from different perspectives. Then, an important task will be to select the most appropriate information. For example, in these first tests, the calculation of the working document elasticity is based on the number of lines of the work document. Yet, this information can be expressed in bytes, number of words, number of lines, number of meaningful blocks (e.g. in the context of a programming task, it could be a variable declaration, a condition block, a function). Although we have the intuition that the number of meaningful blocks would be more significant, we have to ensure our hypothesis through some data feature selection methods. Similarly, the student's activity can be expressed in either an atomic (e.g. clicks, rolling, writing a character) or a larger level with meaningful actions (e.g. document shift, move in the document, sequence writing).

4 Conclusion and Perspectives

In this article, we proposed an approach to recognize a learner's behaviour during a task. It is based on a student model that comes from the educational sciences. In relation with this model, we presented the characteristics used to describe and classify the learner's behaviour and applied three different classification methods (K-NN, Random Forest, Naïve Bayes) to examine whether the chosen model and characteristics can provide a sound foundation for classifying the students' behaviours and recognize them. The results are quite encouraging, but the datasets are still very small and other tests have to be performed with larger ones. Thus the next step is to collect vast amounts of data.

For that purpose, an automatic labelling tool appears to be necessary, since manual categorization is extremely time-consuming. Our idea is to ask the learners to answer, each 15 min, a little questionnaire of five questions. The automatic labelling will be based on their replies. In this way, once we have collected enough data, we will be able to select the most appropriate classification method and the definitive characteristics. Indeed, combined with our process of automatic collection and fusion of traces, this labelling tool will contribute to feed our recommender system with rich and numerous traces of students. By processing the keyboard and mouse actions of the students, features of their professional gestures, indicators on the content of the files that they have produced or modified, etc., we think that we can aim for a classification, then a RS intelligible for the teacher and their students. The coming months will allow us to confirm these ambitions and first results, firstly for programming task. Then we will extend our range by applying these methods for students of others educational fields involved in problem-solving situations and working on specific software.

References

1. Roux, L., Dagorret, P., et al.: A multi-layer architecture for an e-learning hybrid recommender system. In: 18th International Conference on Cognition and Exploratory Learning in Digital Age (2021)
2. Wood, D.F.: Problem based learning. *Bmj* **326**(7384), 328–330 (2003)
3. Gary, K.: Project-based learning. *Computer* **48**(9), 98–100 (2015)
4. Bell, R., Koren, Y., et al.: Matrix factorization techniques for recommender systems. *IEEE Comput.* **42**(8), 30–33 (2009)
5. Thai-Nghe, N., Thai-Nghe, N., et al.: Predicting Student Performance in an Intelligent Tutoring System. Doctoral dissertation, University of Hildesheim (2012)
6. Salehi, M.: Application of implicit and explicit attribute based collaborative filtering and BIDE for learning resource recommendation. *Data Knowl. Eng.* **87**, 130–145 (2013)
7. Wu, D., Lu, J., et al.: A fuzzy tree matching-based personalized e-learning recommender system. *IEEE Trans. Fuzzy Syst.* **23**(6), 2412–2426 (2015)
8. Shu, J., Shen, X., Liu, H., Yi, B., Zhang, Z.: A content-based recommendation algorithm for learning resources. *Multimedia Syst.* **24**(2), 163–173 (2017). <https://doi.org/10.1007/s00530-017-0539-8>
9. Sevarac, Z., Devedzic, V., et al.: Adaptive neuro-fuzzy pedagogical recommender. *Expert Syst. Appl.* **39**(10), 9797–9806 (2012)
10. Kinshuk, A., Nikov, A., et al.: Adaptive tutoring in business education using fuzzy backpropagation approach. In: Proceedings of the Ninth International Conference on Human–Computer Interaction, vol. 1, pp. 465–468 (2001)
11. Khribi, M.K., Jenni, M., et al.: Automatic Recommendations for e-learning personalization based on web usage mining techniques and information retrieval. In: 2008 Eighth IEEE International Conference on Advanced Learning Technologies, pp. 241–245 (2008)
12. Qomariyah, N.N., Fajar, A.N.: Recommender system for e-learning based on personal learning style. In: 2019 International Seminar on Research of Information Technology and Intelligent Systems (ISRITI), pp. 563–567 (2019)
13. Dwivedi, P., Bharadwaj, K.K.: Effective trust-aware e-learning recommender system based on learning styles and knowledge levels. *J. Educ. Technol. Soc.* **16**(4), 201–216 (2013)
14. Roux, L.: Recommender Systems in e-learning: axiological, epistemic, and practical implications of the designing choices. *Bonn International Symposium*, Bonn, December 9th– 10th (2021)
15. Pekrun, R., et al.: Academic emotions in students' self-regulated learning and achievement: a program of qualitative and quantitative research. *Educ. Psychol.* **37**(2), 91–105 (2002)
16. Pekrun, R., Linnenbrink-Garcia, L.: Academic emotions and student engagement. In: Christenson, S., Reschly, A., Wylie, C. (eds.) *Handbook of Research on Student Engagement*. Springer, Boston (2012). https://doi.org/10.1007/978-1-4614-2018-7_12
17. Bosch, N., D'mello, S., et al: Detecting student emotions in computer-enabled classrooms. *IJCAI*, pp. 4125–4129 (2016)
18. Tonguç, G., Ozkara, B.O.: Automatic recognition of student emotions from facial expressions during a lecture. *Comput. Educ.* **148**, 103797 (2020)
19. Gravina, R., Li, Q.: Emotion-relevant activity recognition based on smart cushion using multi-sensor fusion. *Inf. Fusion* **48**, 1–10 (2019)
20. Mannepalli, K., Sastry, P. N., et al. Emotion recognition in speech signals using optimization based multi-SVNN classifier. *Journal of King Saud University-Computer and Information Sciences* (2018)
21. March, J.G.: Exploration and exploitation in organizational learning. *Organ. Sci.* **2**(1), 71–87 (1991)

22. Axtell, C.M., Holman, D.J., et al.: Shopfloor innovation: Facilitating the suggestion and implementation of ideas. *J. Occup. Organ. Psychol.* **73**(3), 265–285 (2000)
23. Roux, L., Nodenot, T., et al.: Supervised classifications and process to detect exploitation and exploration student's behaviours in higher vocational Education. In: 22nd IEEE International Conference on Advanced Learning Technologies (2022)
24. Salton, G., McGill, M.J.: *Introduction to Modern Information Retrieval*. McGraw-Hill, New York (1983)
25. Braslavski, P., et al. (eds.): *RuSSIR. CCIS*, vol. 573. Springer, Cham (2016). <https://doi.org/10.1007/978-3-319-41718-9>
26. Štrumbelj, E., Kononenko, I.: Explaining prediction models and individual predictions with feature contributions. *Knowl. Inf. Syst.* **41**(3), 647–665 (2013). <https://doi.org/10.1007/s10115-013-0679-x>



Automated Classification of Argumentative Components in Students' Essays

Qian Wan, Scott Crossley^(✉) , and Yu Tian

Georgia State University, Atlanta, GA 30302, USA
{qwan1, scrossley, ytian9}@gsu.edu

Abstract. Multiple approaches have been proposed for the automated classification of argumentative components. However, few studies have focused on argumentation in students' essays and how automated classification can support the development of automated writing evaluation (AWE) systems in intelligent tutoring systems. In this study, linguistics features (related to positionality, semantic similarity, part-of-speech tags, named entity, and syntactic dependency) were obtained from 314 essays written by first-year college students. These features were used to build an algorithm to classify 2264 argumentative components found in the essays into four categories (final claim, primary claim, data, and other). Results indicated a Random Forest model (using five repeats of 10-fold cross-validation) achieved an overall F1-score of 0.78 in the classification and that the positionality, semantic similarity, and syntactic dependency features played the most critical roles. The algorithm can help inform the development of AWE algorithms to drive feedback on argumentative essays and help developing writers improve their argumentation.

Keywords: Argumentative writing · Students' essays · Argumentative component classification · Natural language processing

1 Introduction

Writing argumentatively is a complex but critical task to successful academic/argumentative writing. The teaching and learning of argumentative writing are one of the most significant challenges many teachers and students face [9]. In the field of argument mining, many researchers have explored the possibility of automatically identifying and classifying argumentative components (e.g., major claim, claim, and premise) in various areas such as legal decision support [17, 20], information retrieval [7, 8] and debating [2].

However, there has been less research on automated argument components classification in educational setting [e.g., 11, 23, 24]. Currently, few writing evaluation systems

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/978-3-031-09680-8_16.

provide student writers feedback about the overall structure and the use of argumentative components in argumentative essays. Automated argument classification algorithms could be incorporated into writing support/tutor systems such as Writing-Pal [e.g., 15], which could help students improve their writing skills by providing guidance, strategy instruction, and feedback on the use of argumentation.

This study presents a content-based method to automatically classify argumentative components in students' essays using linguistics features related to the position of arguments within an essay, number of named entities, length of arguments, semantic similarity, part-of-speech tags, and syntactic dependencies. The following research questions guided this study:

1. What are linguistic features predictive of the argumentative components (i.e., final claim, primary claim, data, and other non-annotated texts) in students' argumentative essays?
2. What are the implications for generating automated feedback in students' argumentative writing?

2 Background

2.1 Argument Annotation

Work to annotate argumentation in educational setting has focused on classification schemes. For instance, Stab and Gurevych [23, 24] argument scheme contains three categories: claims, major claims, and premises where a major claim is the central component of an argument supported by several claims. The claims indicate the author's opinions about a specific topic, and the premises correspond to the reasons provided by the author to persuade readers of the claims. Other schemes are generally based on a modified Toulmin's model [26] that reflect the common use of argumentative components in students' essays [18, 22, 25]. For instance, Nussbaum and Kardash's [18] argument scheme categorized argumentative components in students' writing into five classes: final claim (the main idea or conclusion on the topic being discussed), primary claim (several reasons the author claims to support the final claim), counterclaim (the potential opposing opinions to the final claim), rebuttal (a claim that the author uses to refute the counterclaims), and supporting reason or example (reasons or examples the author provides to back up the final or primary claims). Similarly, Qin and Karabacak [22] and Stapleton and Wu [25] adopted a scheme that comprised six argumentative components: claim, data, counterargument claim, counterargument data, rebuttal claim, and rebuttal data.

However, only Stab and Gurevych [23, 24] released publicly available datasets to support their argumentation scheme. Yet the annotation scheme they used does not best represent the sophistication of structures in advanced learners' essays as found in Toulmin-based models. While more sophisticated schemes [18] are able to capture structural features in students' writing, the writing samples associated with the schemes are either small or not publicly accessible, limiting their use in developing and testing models of argument classification that could be included in automated systems.

2.2 Classification of Argumentative Components

Once corpora are available, models that classify argumentative components (i.e., automatically labeling each argumentative component based on an argument scheme) can be developed. A range of linguistic features has been used in classification models of argumentative components. These include the location of argumentative components in the text (i.e., at the beginning, middle, or end of an essay), part-of-speech (POS) tags, lexical features (e.g., frequent unigrams, bigrams, and trigrams), and syntactic features including syntactic dependency (i.e., the relations between two words in a sentence) [see, e.g., 1, 2, 16, 19, 20, 23, 24]. In addition, studies have examined the contribution of discourse indicators (e.g., *on the contrary*, *otherwise*, *however*, *for example*, *in conclusion*) in the classification of argumentative components [e.g., 4, 24].

Regarding classification performance, previous studies showed overall F1-scores ranging from 0.35 to 0.83 in classifying multiple argumentative components such as major claims, claims, and premises [e.g., 1, 2, 7, 17, 20, 23, 24]. Previous studies also reported that n-gram, syntactic, discourse indicator, and POS features made the strongest contributions to classification results [2, 16]. However, the accuracy results are not always comparable because previous studies have adopted various annotation schemes and operationalized feature sets differently.

3 Method

3.1 Data

The corpus we used for this study comprises 314 argumentative essays written by undergraduate students at a public university in the United States. The student authors were native speakers of English. Two prompts from retired test banks of the Scholastic Assessment Test (SAT) were used (see supplementary online information for details¹). The first prompt was about originality and uniqueness, while the other was about heroes versus celebrities. The average length of these essays was 354 words ($SD = 118.2$), with an average type-token ratio of 0.52 ($SD = 0.07$). In average, these essays contain 21 sentences ($SD = 7.42$) and 4 paragraphs ($SD = 1.38$).

3.2 Annotation

Normed human raters structurally annotated the essays for argumentative components. We adopted a modified Toulmin's argument model that contains four categories of argumentative components: *final claim*, *primary claim*, *counterclaim*, *rebuttal*, *data*, and *other*, as introduced in [18, 22, 25]. Specifically, the introductory/opening and the conclusion parts were categorized as in the *other* category. The definitions of these elements are presented in Table 1.

¹ <https://github.com/wanqian0202/ITS-2022-Automated-Classification-of-Argumentative-Components-in-Students-Essays>.

Table 1. Definitions and Examples of Argumentative components

Component	Definition
Final Claim	An opinion or conclusion on the main question
Primary Claim	A claim that supports the final claim
Counterclaim	A claim that refutes another claim or gives an opposing reason to the final claim
Rebuttal	A claim that refutes a counterclaim
Data	Ideas or examples that support primary claims, counterclaims, or rebuttals
Other	Any text that doesn't fall into any of the above categories, including introductory sentences and conclusions

Two annotators coded the essays on the web-based text annotation platform TagTog². The two annotators were native English-speaking undergraduate students majoring in applied linguistics at a public university in the United States. Before annotation, a norming session was held to help the annotators achieve consistency in annotations. During the norming process, the annotators were given sample argumentative essays ($N = 36$) from another corpus to annotate, and they discussed the categories of instances until an agreement was reached. Once normed, the two annotators worked independently and coded the 314 essays in the opposite order to avoid recency effects. During the annotating process, the two annotators coded the 314 essays independently for argument categories. In case of disagreement, a third expert annotator adjudicated the classification by comparing the annotations from both annotators and deciding on the boundary and category of the argumentative component. Inter-rater reliability calculated using Fleiss's [6] Kappa for all the annotations was 0.58 ($p < 0.001$), indicating fair to a good agreement.

Since the instances of counterclaims ($N = 19$ or less than 1% of the data) and rebuttals ($N = 20$ or less than 1% of the data) were rare in the current corpus, we merged the categories into the category of primary claim. Thus, the categories for the remaining argumentative components were *final claim*, *primary claim*, *data*, and *other*. Table 2 shows the class distribution among the 2264 argumentative components and each class's proportion in the corpus. The corpus and all annotations are available at https://github.com/scrosseye/Claim_Identification_Corpus.

² <https://www.tagtog.net>

Table 2. Number and Percentage of Different Argumentative Components

Argumentative Component	Count	Percentage
Final Claim	315	14%
Primary Claim	591	26%
Data	774	34%
Other	584	26%
Total	2264	100%

3.3 Selected Text Features for Classification

Baseline Feature (Positionality). In the current study, we used positionality features as the baseline for model comparisons because it is a widely adopted basic feature from previous studies [e.g., 20, 24]. Additionally, we found that there were significant differences in their positionality in the paragraphs and the essays among different types of argumentative components. Therefore, there is a tendency that the organization of argumentative components would follow certain positional patterns.

Two types of normalized positional features for each argumentative component were calculated: the normalized position of the argumentative component in the essay and the normalized location of an argumentative component in the paragraph in which it appeared. These were computed as the ratio of the component position in the paragraph/essay (measured by the number of characters) to the total number of characters of the paragraph/essay.

Named Entity Feature. We used the spaCy package [10] to calculate the number of named entities in each argumentative component as the named entity feature. These named entities included *persons*, *places*, and *organizations*. We presume that *data* would contain more and specific types of named entities than other types of argumentative components, but this feature has not been used in previous studies.

Semantic Similarity Features. Semantic similarity reports the relationship between text fragments (i.e., to what extent the two fragment differ or have similarities in meanings). To calculate the semantic similarity score, a transformer model can be used to encode the fragments and obtain their embeddings. Then a similarity metric (e.g., cosine similarity) can be used to compute the similarity score between fragments. In the current study, we used the *Universal Sentence Encoder*³ [5] as the encoder to calculate semantic similarities between the argumentative components.

We specifically calculated the semantic similarity scores between an argumentative component and (1) its preceding component and (2) its succeeding component. Further, we calculated (3) the difference between the semantic similarity scores of a component and its preceding component and its succeeding component ($SS_{\text{current and preceding}}$

³ <https://tfhub.dev/google/universal-sentence-encoder-large/5>

– $SS_{\text{current and succeeding}}$) and the absolute value of the difference ($|SS_{\text{current and preceding}} - SS_{\text{current and succeeding}}|$). To avoid the absence of values in the modeling stage, if an argumentative component was the first or last component in the essay, we used value 0 as placeholders for features (1) and (2). Such an approach has not been used before in argument classification, but we presume that adjacent arguments can be classified based on semantic similarity or differences.

Part-of-Speech Features. We calculated the number and percentage of fine-grained POS tags ($N = 45$) for each argumentative component using spaCy [10]. The percentage of a POS tag was defined as the number of a POS tag found in an argumentative component divided by the total number of POS tags in the component.

Syntactic Dependency Features. We also used the spaCy package to automatically label the syntactic dependencies ($N = 45$) in each argumentative component and calculated the number of and percentage for each type of dependency relation found within the components.

Discourse Indicator Features. We merged the lists of discourse indicators reported in previous studies [4, 12, 24]. The merged list contained nine categories of discourse relations: sequence, temporal situation, causal relations, similarity relations, contrast and expectation, clarifying statements, summary, interruptions, and thesis (see supplementary online information for details). We calculated the total number of discourse indicators from each discourse relation type for each argumentative component in our corpus. The percentage of indicators was calculated as the number of discourse indicators divided by the number of words in the argumentative component.

3.4 Experimental Setup

Feature Reduction. To avoid multicollinearity, we conducted correlation analyses among all the derived features (one versus all). Highly correlated variables ($r > 0.699$) were then removed (see supplementary online information for the 35 features and their descriptions used in final modeling).

Modeling. We used the scikit-learn library [21] to implement the training and testing process. We used random forest, logistic regression, and support vector machines (SVM) classifiers for each feature set. Five repeats of the 10-fold cross-validation were implemented to estimate the performance of our machine learning algorithm. We used the default hyper-parameters provided by the scikit-learn modules. Before developing the logistic regression and the SVM models, all features were standardized using the StandardScaler module. We performed data standardization for the logistic regression and the SVM models, for these classifiers can be sensitive to the range of the data points. Specifically, we applied a multinomial logistic regression classifier with an L1 penalty to optimize the models.

4 Results

4.1 Model Evaluation

Comparing different feature sets, all models with an enhanced feature set outperformed the baseline feature set (see supplementary online information for details). The results also indicate that the random forest models generally outperformed the other models, regardless of the used feature sets. Results of two-sample t-tests of accuracy extracted from all random forest models showed that adding in named entity, semantic similarity, part-of-speech, and syntactic dependency features led to significant improvement in classification performance (see supplementary online information for details).

We used F1-score to measure the performance of models with different feature sets. The best model used a random forest classifier with the positionality, named entity, semantic similarity, POS, syntactic dependency feature set, reaching an overall weighted F1-score of 0.78 (see Table 3 and 4 for the overall and by-class model performance). Table 5 presents a confusion matrix for the best model. This model performed best in classifying the non-annotated text, followed *data*, *primary claim*, and *final claim*.

Table 3. Overall Classification Precision, Recall, F1-score

Measures	Accuracy	SD Accuracy	Precision	Recall	F-score
Random Forest	0.79	0.03	0.78	0.79	0.78
Logistic Regression	0.75	0.03	0.74	0.75	0.74
Support Vector Machines	0.76	0.03	0.75	0.76	0.75

Table 4. By-class Classification Precision, Recall, F1-score

Class	Precision	Recall	F1
Final Claim	0.60	0.39	0.48
Primary Claim	0.74	0.80	0.77
Data	0.86	0.87	0.86
Other	0.81	0.88	0.84

Table 5. Confusion Matrix Retrieved from the Best Model

		Predicted			
		Final claim	Primary claim	Data	Other
Actual	Final claim	0.39	0.26	0.07	0.28
	Primary claim	0.08	0.8	0.11	0.01
	Data	0.01	0.08	0.87	0.04
	Other	0.05	0.04	0.04	0.88

Note. The row labels represent the true classes, and the column labels are the predicted classes. Values are the proportions, and each row sums up to 100%

4.2 Feature Importance

In Table 6, we present the top 10 most important variables and their feature importance values based on the best model. The feature importance values measured the relative importance of the features used to fit the model. The values were the mean and standard deviation of accumulation of the impurity decrease within each tree that was automatically calculated and reported by scikit-learn according to [3].

Table 6. Feature Importance Values Extracted from the best model

Feature	Importance value
Normalized location of the component in the essay	0.20
Semantic similarity difference between a component's preceding and succeeding component	0.13
Number of roots (of syntax trees) in the component	0.13
Semantic similarity between a component and its preceding component	0.10
Semantic similarity between a component and its succeeding component	0.10
Number of coordinating conjunctions in the component	0.05
Number of adverbial modifiers in the component	0.05
Number of compound modifiers in the component	0.03
Number of named entities in the component	0.03
Number of adverbial clause modifiers in the component	0.02

One-way ANOVA analyses were performed to compare whether there were differences of the location, the semantic similarity features, the number of roots (of syntax trees), and the number of named entity among different types of argumentative components. Results indicated that there were significant differences in the location, $F(3, 2260) = 142, p < 0.001$, the semantic similarity between one component and its preceding component, $F(3, 2260) = 80.66, p < 0.001$, the semantic similarity between one component and its succeeding component, $F(3, 2260) = 77.58, p < 0.001$, and the absolute

value of the semantic similarity difference (between a component's preceding and succeeding component), $F(3, 2260) = 514.7, p < 0.001$. Specifically, *final claims* usually precede *primary claims* in an essay, while *data* usually occurs later after *primary claims*. In terms of semantic similarity, in general, *data* had higher semantic similarity with its preceding component than *final claims* did, while it had lower semantic similarity with its succeeding component than *final claims* and *primary claims* did. It was also found that *final claims* had the highest semantic similarity difference between their preceding and succeeding components, while *primary claims* and *data* had lower semantic similarity differences between their surrounding components.

Significant differences were also found in the number of roots (of syntax trees), which was a feature highly correlated with word count. Usually, *primary claims* contained the least number of roots, followed by *final claims*, while *data* contained the greatest number of roots. Significant differences were also found in the number of named entities, $F(3, 2260) = 120.7, p < 0.001$, such that *data* contained the greatest number of named entity among all types of argumentative components, followed by *primary claims*.

5 Discussion

5.1 Classification Performance

In this study, we developed an algorithm to automatically classify the types of 2264 argumentative components in students' essays using various linguistics features. The purpose of the study was to explore automated argument classification and how it can be incorporated into writing tutor systems, which could help students improve their writing skills by providing guidance and feedback in argumentative writing.

Since relatively fewer studies have focused on students' argumentative essays and these studies have used different annotation schemes for argumentation, comparing the accuracy results from this study with the previous research is difficult. However, we can surmise by the results of the t-tests that adding in the named entity, semantic similarity, part-of-speech, and syntactic dependency features led to significant improvement in the classification accuracies. This confirmed that POS and syntactic dependency features are effective in predicting the classes of argumentative components, which was in line with previous studies [e.g., 1, 7, 24].

Previous studies [e.g., 1, 13] suggested that semantic similarity outperformed syntactic features in identifying the relationships among argumentative components. Our results expanded these findings and found that semantic similarity features also performed well in classifying argumentative components. This seems plausible given that the transitions between different argumentative components sometimes can be implicit and reflected by semantic information rather than explicit discourse markers, as discussed in [23]. As has been found in [14], around 26% of the evidence relations in the Rhetorical Structure Theory Discourse Treebank are based on explicit discourse markers. Thus, most relations may be the result of semantic distinction and not the product of explicit discourse markers. Our semantic similarity features were able to identify that *data* had higher semantic similarity with its preceding component than *final claims* and *primary claims* did and that *data* tended to have lower semantic similarity with its succeeding component. This indicated that *data* usually was more adherent to previous

components than succeeding components in meaning, despite the use of explicit discourse markers. It is likely that our approach captured the semanticity of the implicit transitional expressions and the explicit discourse markers.

This leads to another difference between our results and previous studies, which is that the discourse indicator features calculated here did not play an important role in the classification task, while they were important contributors in previous studies [e.g., 1, 23, 24]. However, many factors could lead to the differences reported here. For example, the genres and topics of the texts, as well as the task and the level of the writers, may have led to argumentative elements that did not rely strongly on discourse indicators [e.g., 14].

Our results also indicated that named entity recognition contributed to the argumentative component classification. Specifically, *data* contained the greatest number of named entities among all types of argumentative components, while *final claims* and *primary claims* contained the least number of named entities. This seems plausible given that *data* was defined to be the ideas or examples that support the claims, and the more frequent use of named entity might indicate more specific examples and sophisticated content. Our findings are in contrast to [16], who reported that named entity features contributed very little in identifying arguments; however, [16] used different methods to extract the named entities.

5.2 Implications for Generating Feedback for Students' Argumentative Writing

The models developed in this study will be integrated into a writing tutor system in order to provide student writers with detailed feedback on their use of argumentative components. Specifically, our algorithm will be used within the tutor system to predict argumentation structures (e.g., *final claims*, *primary claims*, and *data*) and highlight situations where a student may have failed to include clearly stated claims or provide persuasive evidence to warrant the claims (for example). Based on the models, feedback will be provided to students to help them revise their writing. For example, if the algorithm detects an absence of a *final claim* in an essay, feedback will be generated to the writer that a clear position and substantive arguments are needed.

The models can also identify the linguistic features (e.g., named entity, POS, syntactic dependency, and semantic similarity) used within each argumentative component. For instance, since we have found that *data* usually contains more named entities, the algorithm will suggest to writers to use more specific examples to support the claims when named entities are lower than expected. The information provided by the models might also inform the quality of the argumentative essays, as suggested in [11], although future research is needed here.

What is unique about the models presented here is that they focus specifically on argumentation, an element of writing that many writing tools provide only cursory information about or ignore completely. Knowing the importance of argumentation, the models developed in this paper can provide student writers with increased opportunities for deliberate practice during the writing process and make strong impacts on writer development.

6 Conclusion

In this study, we developed an algorithm to automatically classify the types of argumentative components in students' essays. We extracted linguistics features related to the component position, named entity, semantic similarity, part-of-speech, syntactic dependency, and discourse indicators from 314 essays to classify 2264 argumentative components of the essays. Our results indicated the Random Forest model achieved an overall weighted F1-score of 0.78 in the classification.

Our findings confirmed that positionality, part-of-speech, and syntactic features are important in the classification of argumentative components. Further, the results also expanded existing research by confirming the use of named entity recognizers and the semantic similarity features strongly contributed to the classification models. We also found that implicit information contained in semantic similarity features likely plays a more critical role than initially expected.

These results and their interpretation should be taken with caution, though. It is important to note that the corpus used for this study was relatively small and comprised essays from only two prompts. To obtain classification results of higher reliability, we plan to test our algorithms on other existing argumentation mining corpora, especially those containing more topics [e.g., 23, 24], and validate the performance of our classification algorithm. Additionally, the algorithms need to be tested on essays from different writing tasks (narrative writing, expository writing, or research reports) from a wider age range and grade levels. Overall, however, we think that the results of this study should help inform the development of writing evaluation algorithms to generate feedback on argumentative essays and gain a better understanding of the linguistic features related to argumentative composition. Importantly, these algorithms can be incorporated into intelligent tutoring systems to provide enhanced feedback to learners.

References

1. Aker, A., et al.: What works and what does not: classifier and feature analysis for argument mining. In: Proceedings of the 4th Workshop on Argument Mining. pp. 91–96 (2017)
2. Al Khatib, K., Wachsmuth, H., Hagen, M., Köhler, J., Stein, B.: Cross-domain mining of argumentative text through distant supervision. In: Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pp. 1395–1404 (2016)
3. Breiman, L., Friedman, J.H., Olshen, R.A., Stone, C.J.: Classification and regression trees, vol. 432, pp. 151–166. Wadsworth. International Group, Belmont (1984)
4. Burstein, J., et al.: Computer analysis of essay content for automated score prediction: A prototype automated scoring system for GMAT analytical writing assessment essays. ETS Res. Rep. Ser. **1998**, i–67 (1998)
5. Cer, D., et al.: Universal sentence encoder. arXiv preprint [arXiv:1803.11175](https://arxiv.org/abs/1803.11175) (2018)
6. Fleiss, J.L., Levin, B., Paik, M.C., et al.: The measurement of interrater agreement. Stat. Methods Rates Report. **2**, 22–23 (1981)
7. Habernal, I., Gurevych, I.: Exploiting debate portals for semi-supervised argumentation mining in user-generated web discourse. In: Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing. pp. 2127–2137 (2015)

8. Habernal, I., Eckle-Kohler, J., Gurevych, I.: Argumentation Mining on the Web from Information Seeking Perspective. In: *ArgNLP* (2014)
9. Hirvela, A.: Preparing English language learners for argumentative writing. L2 writing in secondary classrooms: Student experiences, academic issues, and teacher education, pp. 67–86 (2013)
10. Honnibal, M., Montani, I.: spaCy 2: natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing. *To appear*. **7**, 411–420 (2017)
11. Klebanov, B.B., Stab, C., Burstein, J., Song, Y., Gyawali, B., Gurevych, I.: Argumentation: Content, structure, and relationship with essay quality. In: *Proceedings of the Third Workshop on Argument Mining (ArgMining2016)*, pp. 70–75 (2016)
12. Knott, A., Dale, R.: Using linguistic phenomena to motivate a set of coherence relations. *Discourse Process*. **18**, 35–62 (1994)
13. Lawrence, J., Reed, C.: Argument mining: a survey. *Comput. Linguist.* **45**, 765–818 (2020)
14. Marcu, D., Echihiabi, A.: An unsupervised approach to recognizing discourse relations. In: *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pp. 368–375 (2002)
15. McNamara, D.S., et al.: The Writing-Pal: Natural language algorithms to support intelligent tutoring on writing strategies. In: *K-12 Education: Concepts, Methodologies, Tools, and Applications*, pp. 780–793. IGI Global (2014)
16. Merity, S., Murphy, T., Curran, J.R.: Accurate argumentative zoning with maximum entropy models. In: *Proceedings of the 2009 Workshop on Text and Citation Analysis for Scholarly Digital Libraries (NLPIR4DL)*, pp. 19–26 (2009)
17. Mochales, R., Moens, M.-F.: Argumentation mining. *Artif. Intell. Law*. **19**, 1–22 (2011)
18. Nussbaum, E.M., Kardash, C.M., Graham, S.E.: The effects of goal instructions and text on the generation of counterarguments during writing. *J. Educ. Psychol.* **97**, 157 (2005)
19. Ong, N., Litman, D., Brusilovsky, A.: Ontology-based argument mining and automatic essay scoring. In: *Proceedings of the First Workshop on Argumentation Mining*, pp. 24–28 (2014)
20. Palau, R.M., Moens, M.-F.: Argumentation mining: the detection, classification and structure of arguments in text. In: *Proceedings of the 12th International Conference on Artificial Intelligence and Law*, pp. 98–107 (2009)
21. Pedregosa, F., et al.: Scikit-learn: Machine learning in Python. *J. Mach. Learn. Res.* **12**, 2825–2830 (2011)
22. Qin, J., Karabacak, E.: The analysis of Toulmin elements in Chinese EFL university argumentative writing. *System* **38**, 444–456 (2010)
23. Stab, C., Gurevych, I.: Identifying argumentative discourse structures in persuasive essays. In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 46–56 (2014)
24. Stab, C., Gurevych, I.: Parsing argumentation structures in persuasive essays. *Comput. Linguist.* **43**, 619–659 (2017)
25. Stapleton, P., Wu, Y.A.: Assessing the quality of arguments in students' persuasive writing: a case study analyzing the relationship between surface structure and substance. *J. Engl. Acad. Purp.* **17**, 12–23 (2015)
26. Toulmin, S.E.: *The Uses of Argument*. Cambridge University Press, Cambridge (1958)



Evaluating the Effect of Imperfect Data in Voice Emotion Recognition

Mahsa Aghajani, Hamdi Ben Abdessalem^(✉), and Claude Frasson

Département d'Informatique et de Recherche Opérationnelle,
Université de Montréal, Montréal H3C 3J7, Canada
{mahsa.aghajani, hamdi.ben.abdessalem}@umontreal.ca,
frasson@iro.umontreal.ca

Abstract. Speech is a dominant way of communication among humans which carries both information and emotion of the speaker. A wide variety of applications can utilize the emotions in the speech for serving humans' needs more efficiently. Regarding the classification nature of a voice emotion recognition system, using machine learning algorithms is a practical solution. One important step in designing a machine learning classifier is to choose the appropriate datasets. There are several well-known and commonly used voice emotion datasets which are recorded from actors who are usually experts in reflecting their emotion in the speech. Knowing that the users of an application, for example tutoring systems, are not necessarily experts in exposing their emotion, we face a question: how much efficient are the classifiers which are trained with the perfect data in predicting the emotions from normal and non-experts' voices? In this study we tried to answer this question by testing several trained classifiers on a custom speech emotion dataset, gathered from non-actors' voices. We also aimed for evaluating the effect of using this custom dataset in the training phase of the classifiers on their final accuracies. Our experiments show that although including parts of our custom data in the training phase of the classifiers leads to lower validation performance measures, a boost in the test accuracy is obtained. That is classifiers which were trained with a combination of perfectly recorded datasets and our custom dataset, had higher performance measures in predicting the emotions from non-actors' voices.

Keywords: Voice emotion recognition · Machine learning · Brain computer interface · Affective computing

1 Introduction

Speech is a dominant way of communication among humans which carries both information and emotion of the speaker. Computers can benefit from perceiving emotions to figure out human requirements more robustly. One viable instance of benefitting from emotions in the voice are tutoring systems. During COVID-19, the significance of having efficient tutoring systems became more obvious. One of the most important challenges that a tutoring system faces is customization based on the user's feedbacks. One way of

acquiring honest feedback without putting much burden on the users is to extract their emotion from their voice while the user is interacting with the tutoring system. User's recognized emotion in this type of applications can be used in a variety of ways: first the user's emotions can be used as accurate feedback about the material learned in each lesson. Second, the arrangement of the future material or the whole learning path can get customized based on the emotion that is reflected from the learner's audio feedback. Regarding all of these, embedding a voice emotion recognition system in a tutoring application, leads to more efficiency in perception of the learner's needs and feedbacks.

In this study we want to answer the question that how much accurate is a voice emotion classifier which is built by using perfect data in its training phase in recognizing the emotions from non-actors. Also, we aim to investigate the possibility of improving a voice emotion classifier's accuracy by using imperfect data in the classifiers training phase. In this study, by imperfect data, we are referring to speech from non-actors who may not be experts in reflecting their true emotion by their voice.

For answering these questions, first we are going to build several voice emotion classifiers, capable in recognizing 7 emotions in user's voice. These 7 emotions are anger, happiness, sadness, fear, disgust, surprise and neutrality. We retrieved this set of emotions from the discrete emotional model theory presented by [2].

For evaluating the effect of leveraging imperfect data on the accuracy of classifiers in recognizing non-actors' emotions, we are going to build classifiers using different combinations of several well-known perfect speech emotion datasets, with and without using a custom speech emotion dataset. The properties of this dataset are explained in detail in Sect. 2.1. After training the classifiers, we are going to report their accuracies in recognizing the emotion from speech on our imperfect data.

The outline of this paper is as follows: In Sect. 2 we describe our methodology by introducing the datasets that we have used in this work which are three commonly used datasets together with our custom dataset, extracted features and machine learning models. Section 3 is about explaining the experiments and reporting their results. Section 4 concludes our work and discusses our future works that may lead to improvements.

2 Methodology

For building voice emotion classifiers, like any other classification task, we must first decide about the dataset, the feature set and the classification models. In this study we have trained and tested the classifiers under two different approaches: first, to train the classifiers only with perfect data. Second, to use part of our custom dataset together with all perfect datasets for training classifiers. In both of these approaches, after training the classifiers, we have tested them with the same part of our custom dataset.

In the following sections, we will provide details about our custom dataset and the process of gathering it. Then the feature sets are introduced.

2.1 Datasets

In all machine learning applications, selecting the proper dataset is extremely important. There are many different datasets for voice emotion recognition. In our work, three

well-known and commonly used datasets and our custom dataset are used for training and validating the models. The well-known datasets are Toronto Emotional Speech Set (TESS) [3], Ryerson Audio-Visual Database of Emotional Speech and Song (RAVDESS) [4] and Surrey Audio-Visual Expressed Emotion (SAVEE) Database [5].

Our custom dataset is the custom dataset that we gathered for this study. The reason for building this dataset is that usually the emotions in non-actor's voices are not as clearly reflected as actor's voices. This dataset is recorded from 4 speakers, 2 males and 2 females, with an average age of 30. Before recording the audio files from speakers, we needed to trigger a desired emotion in them. We used International Affective Picture System (IAPS) for triggering the emotions. IAPS is a database of pictures designed to provide a standardized set of pictures for studying emotion and attention [6]. We could collect 25 audio files for each of the 7 emotions, which led to a total of 175 audio files.

For evaluating the effect of using imperfect data, we have followed two different approaches in choosing the training dataset. First, Using TESS, RAVDESS and SAVEE only; Second, using TESS, RAVEDESS, SAVEE and part of our custom dataset.

2.2 Feature Sets

We have used the feature set provided in the INTERSPEECH 2013 computational paralinguistics challenge [7]. This feature set is commonly used among related studies of recognizing emotions from voice and is composed of 6373 voice features. We have used a tool named openSMILE [8] for extracting these feature sets from our audio dataset.

2.3 Classifiers

We used several widely used classifiers in voice emotion recognition studies. Support Vector Machines (SVMs) are supervised classifiers trying to classify the data using a hyperplane or a set of hyperplanes in a high dimensional space and popular in emotion speech recognition tasks [9, 10]. Ensemble methods such as Random Forest and Bagging are also popular among voice emotion recognition studies [11, 12]. Since leveraging deep learning models is a promising technique in recognizing emotions from voice [13, 14], We have also tried to include deep learning classifiers such as Recurrent Neural Network (RNN) and Convolutional Neural Network (CNN) in our investigation in this study. We have trained and evaluated an RNN composed of two hidden layers using Long Short Term Memory (LSTM) to deal with gradients exploding and gradients vanishing [15]. The CNN is composed of seven layers including 1D convolution layers with different activation layers.

3 Results and Discussion

For validating the models, we have followed two different approaches based on the dataset used for training the models; In the first approach, the training dataset is composed of TESS, RAVDESS and SAVEE datasets, which are all common speech emotion recognition datasets, recorded by actor speakers. In the second case, we added part of our custom dataset to the previous datasets. In all these approaches, the trained classifiers are tested by the same part of our custom dataset.

3.1 Considering TESS, RAVDESS and SAVEE Datasets

In this case, classifiers were trained with TESS, RAVDESS and SAVEE datasets. As it can be seen in Fig. 1, RNN and gradient boosting have the best average accuracies of 95.1% and 85.9% respectively. These classifiers have also the overall best performance measures among all the classifiers which shows that we can rely on deep learning and ensemble methods for speech emotion prediction applications.

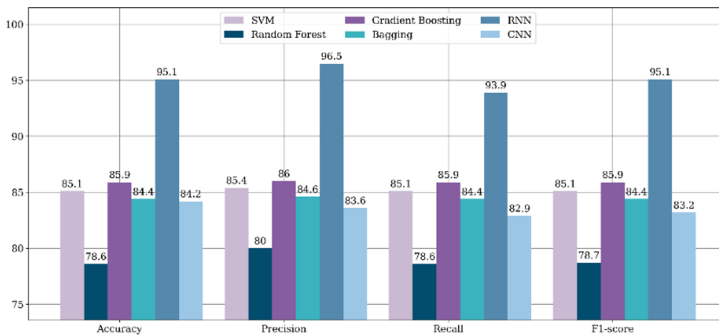


Fig. 1. Average performance measures of classifiers trained with INTERSPEECH 2013 on the validation set.

Figure 2 shows the average performance measures of the classifiers on the test set, which is composed of audio files recorded from non-actors and non-experts in reflecting their emotion in their voice.

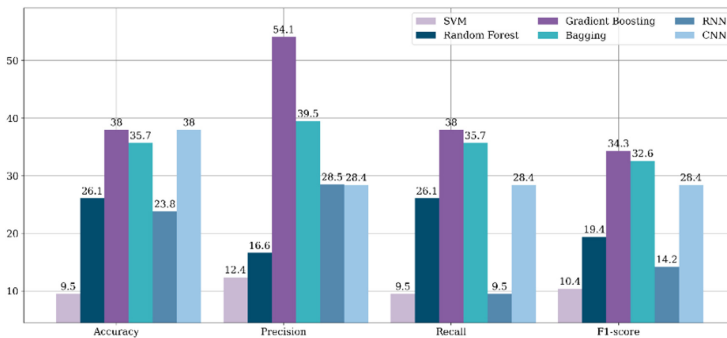


Fig. 2. Average performance measures of classifiers trained with INTERSPEECH 2013 on the test set.

Figure 2 depicts that there is a noticeable decrease in the performance measures of the classifiers while trying to predict the emotions from normal peoples' voices. While the trained RNN had the validation accuracy of 95.1%, its accuracy is 23.8% in predicting the emotions on the test set which is part of our custom data. This shows despite being accurate on the perfectly recorded data from actors, these classifiers are not capable enough in recognizing the emotions from ordinary people. In the next scenario, we will use part of our custom data in the training phase of the classifiers.

3.2 Considering TESS, RAVDESS, SAVEE and Part of the Custom Dataset

In this scenario, parts of the custom dataset are used in the training phase of the classifiers. Considering Fig. 3, RNN has the best validation accuracy of 93.8% among classifiers. After RNN, the best accuracies range from 83% to 85% and belongs to SVM, Gradient Boosting, Bagging and CNN. Now we should notice the decrease in the validation accuracy of RNN compared to the first scenario.

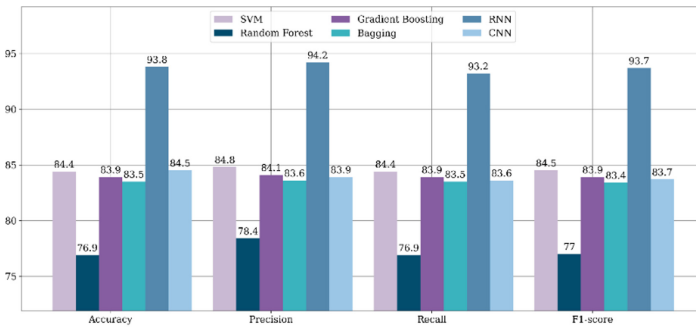


Fig. 3. Average performance measures of classifiers trained with INTERSPEECH 2013 on the validation set.

Figure 4 shows the average performance measures of the classifiers on the test set. In this figure, it can be observed that despite the decrease on the validation accuracy of RNN, its test accuracy is increased. Comparing Figs. 2 and 4, we can see that after adding parts of our custom data to the training set, RNN's accuracy in recognizing the emotions on a separate part of the custom dataset increased by 19%. Also Fig. 4 shows that in this scenario, Gradient Boosting and CNN have both the best accuracy of 52.3%, which is approximately 14% more than their test accuracy in the first case.

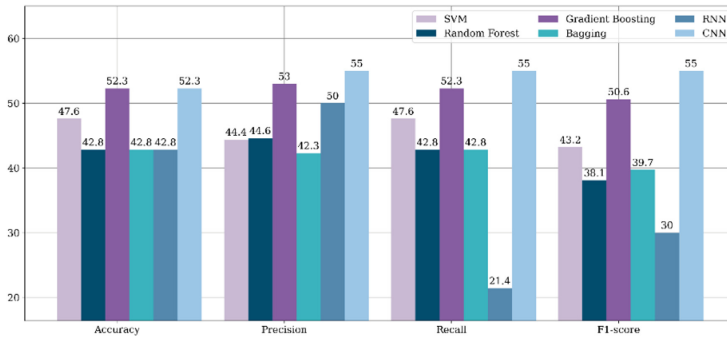


Fig. 4. Average performance measures of classifiers trained with INTERSPEECH 2013 on the test set.

4 Conclusion and Future Work

In this study, we tried to address one of the challenges in tutoring systems which is getting feedback from the users. Considering this application, we should note that the target users of a tutoring system are not necessarily experts in reflecting their emotions in their voices. Therefore, we aimed to answer the question of how efficient are voice emotion recognition systems which are trained with perfect data in recognizing the emotions from ordinary and non-actor people. Our experiments showed that using parts of the imperfect dataset in the training phase of the classifiers, improves their final performance measures in recognizing the emotions from speakers who are not necessarily experts in reflecting their emotion in their voices.

Since in real case scenarios, emotions may not always be reflected in the speaker's voice noticeably, getting help from spoken words may help in predicting the true emotion. Regarding this, we are going to use speech to text conversion techniques. This approach may decrease the current confusion rate between anger and happiness, since although the audio signals have a lot of similar features, but the spoken words are normally very different.

Acknowledgment. We acknowledge NSERC-CRD (National Science and Engineering Research Council Cooperative Research Development), Prompt, and BMU (Beam Me Up) for funding this work.

References

1. Jordan, M.I., Mitchell, T.M.: Machine learning: Trends, perspectives, and prospects. *Science* **349**(6245), 255–260 (2015). <https://doi.org/10.1126/science.aaa8415>
2. Ekman, P., Friesen, W.V., Ellsworth, P.: *Emotion in the Human Face: Guidelines for Research and an Integration of Findings*. Elsevier (2013)
3. Pichora-Fuller, M.K., Dupuis, K.: Toronto emotional speech set (TESS). *Scholars Portal Dataverse* (2020). <https://doi.org/10.5683/SP2/E8H2MF>

4. Livingstone, S.R., Russo, F.A.: The Ryerson audio-visual database of emotional speech and song (RAVDESS): a dynamic, multimodal set of facial and vocal expressions in North American English. *PLoS ONE* **13**(5), e0196391 (2018)
5. Haq, S., Jackson, P.J.B.: Machine audition: principles, algorithms and systems. In: Wang, W. (ed.), pp. 398–423. IGI Global, Hershey PA (2010)
6. Lang, P.J., Bradley, M.M., Cuthbert, B.N.: International affective picture system (IAPS): instruction manual and affective ratings. The center for research in psychophysiology, University of Florida (1999)
7. Schuller, B., et al.: The INTERSPEECH 2013 computational paralinguistics challenge: social signals, conflict, emotion, autism (2013)
8. Eyben, F., Wöllmer, M., Schuller, B.: Opensmile: the munich versatile and fast open-source audio feature extractor. In: Proceedings of the 18th ACM international conference on Multimedia, pp. 1459–1462 (2010)
9. Shen, P., Changjun, Z., Chen, X.: Automatic speech emotion recognition using support vector machine. In: Proceedings of 2011 International Conference on Electronic Mechanical Engineering and Information Technology, vol. 2, pp. 621–625, August 2011. <https://doi.org/10.1109/EMEIT.2011.6023178>
10. Schuller, B., Rigoll, G., Lang, M.: Speech emotion recognition combining acoustic features and linguistic information in a hybrid support vector machine-belief network architecture. In: 2004 IEEE International Conference on Acoustics, Speech, and Signal Processing, vol. 1, p. I–577, May 2004. <https://doi.org/10.1109/ICASSP.2004.1326051>
11. Arya, R., Pandey, D., Kalia, A., Zachariah, B.J., Sandhu, I., Abrol, D.: Speech based emotion recognition using machine learning. In: 2021 IEEE Mysore Sub Section International Conference (MysuruCon), October 2021, pp. 613–617 (2021). <https://doi.org/10.1109/MysuruCon52639.2021.9641642>
12. Noroozi, F., Sapiński, T., Kamińska, D., Anbarjafari, G.: Vocal-based emotion recognition using random forests and decision tree. *Int. J. Speech Technol.* **20**(2), 239–246 (2017). <https://doi.org/10.1007/s10772-017-9396-2>
13. Tian, L., Moore, J., Lai, C.: Recognizing emotions in spoken dialogue with hierarchically fused acoustic and lexical features. In: 2016 IEEE Spoken Language Technology Workshop (SLT), pp. 565–572, December 2016. <https://doi.org/10.1109/SLT.2016.7846319>
14. Kaya, H., Fedotov, D., Yeşilkanat, A., Verkholyak, O., Zhang, Y., Karpov, A.: LSTM Based Cross-corpus and Cross-task Acoustic Emotion Recognition (2018). <https://doi.org/10.21437/Interspeech.2018-2298>
15. Pascanu, R., Mikolov, T., Bengio, Y.: On the difficulty of training recurrent neural networks. In: International Conference on Machine Learning, pp. 1310–1318 (2013)



Application of 3D Human Pose Estimation for Behavioral Reproduction

Kodjine Dare, Hamdi Ben Abdessalem^(✉), and Claude Frasson

Département d'Informatique Et de Recherche Opérationnelle, Université de Montréal, Montréal
3C 3J7, Canada

{kodjine.dare, hamdi.ben.abdessalem}@umontreal.ca,
frasson@iro.umontreal.ca

Abstract. Human pose estimation is a challenging problem in computer vision and shares all the difficulties of object detection. This task is particularly problematic when it comes to the estimation of the human pose in three dimensions. In fact, while the perception of an object in three dimensions is easy for humans due to the mastering of 3D mental model through years, this task is harder to replicate in computer vision. In this paper, we describe an approach that aims at estimating poses from video with the objective of reproducing the observed behaviors by a virtual avatar. We are motivated by two main objectives. First, we aim at the estimation of 3D poses in video. We use a fully convolutional model based on temporal (dilated) convolutions over 2D keypoints to achieve the estimation of these 3D poses. Secondly, we set the objective to transfer 3D coordinates for previously estimated poses to a virtual avatar. The idea behind this transfer is to reproduce observed behavior in a video by a virtual avatar. Our strategy of employing temporal convolutions in lifting 2D keypoints to 3D keypoints yields better results than previous methods that attempt similar reconstruction. Finally, we use an image-based position transfer to recreate the behaviors derived from the video on the skeleton of a virtual avatar. With our approach we create learning avatar that could be used in different applications.

Keywords: 3D pose estimation · Virtual avatar · Behavior reproduction · Video estimation · Learning avatar

1 Introduction

Human Pose Estimation (HPE) revolves around detecting the positions of keypoints of the human body through media analysis (images and videos) to derive a skeleton. We understand by pose, the ensemble of human keypoints.

We have identified two objectives for our work. First, we focus on estimating the pose from videos to extract information such as the coordinates of keypoints of the human body. Extracting and storing these coordinates provides a collection of data that will be relevant for the next step. The second objective of our work consists in using the coordinates of the part of the body extracted for the reproduction of the behavior with

an avatar. This objective aims to make it possible to transfer human movements to an animated avatar to highlight the behavioral dynamics of the human from the keypoints extracted.

Achieving these two goals can lead to multiple implications. We explore a possible application in the context of Alzheimer’s disease. Alzheimer’s disease is an irreversible and progressive brain disorder that slowly destroys memory and thinking skills and affects behavior. Patients with Alzheimer’s disease can sometimes have specific behaviors (walking, balancing or otherwise) that could be observed by video camera at different times of the day, extracted and reconstructed in a virtual avatar. This avatar would serve as a training model to educate medical staff to recognize an episode in patients with Alzheimer’s disease and improve interaction with them. We make the hypothesis that utilizing dilated temporal convolutions on 2D keypoints provides better results for the estimation of poses. We also make the hypothesis that our approach provides smoother reproduction in a virtual environment.

The rest of this paper is organized as follows. In Sect. 2 we present related works. In Sect. 3 we present our approach. In Sect. 4 we explain the experiments and discuss the results in Sect. 5.

2 Related Work

Pavlakos et al. [1] formed a network with ordinal depths of human joints as constraints, whereby 2D human datasets can also be added with annotations of ordinal depths. Xu et al. [2] used a 2D pose correction module to refine unreliable 2D joints based on the kinematic model. Compared to the kinematic model, which produces human skeletons, volumetric models can recover more than just coordinates but also the human body shape and helps produce high quality mesh. An early work widely used among volumetric based methods for 3D pose estimation is the SMPL model [3]. Kanazawa et al. [4] employed adversarial learning by using a generator to predict parameters of SMPL.

Martinez et al. [5] proposed a simple and effective linear layer to lift 2D joint locations to 3D positions. Hossain and Little [6] suggested a recurrent neural network that leverages temporal information from human pose series utilizing a Long Short-Term Memory (LSTM) unit with shortcut connections.

Optimizing this work [4], our previous work [7] proposed an addition (2D keypoints coordinates) to their framework to, not only estimate behaviors from videos in three dimensions, but also reproduce said behaviors in a virtual environment by avatar. While this approach was successful in the 3D pose estimation aspect, it required a lot of time to get estimated 3D keypoints and didn’t provide smooth reproduction of behavior in virtual environment. The approach we suggest tackle the same objectives by lifting directly 2D estimation and addition of temporal convolutions.

Compared to the single-frame baseline proposed by Martinez et al. [5] and the LSTM model by Hossain and Little [6], we use temporal data by applying 2D convolutions over the time dimension, and we suggest numerous optimizations that lead to lower reconstruction error.

3 Our Approach

3.1 3D Keypoints Estimation

Given a video, our first objective is to estimate 3D keypoints and store them for further use in the reproduction by virtual avatar. To achieve the estimation, we suggest first a preprocessing of the video to detect the human in every frame and track the person in the video across the frames of the video. Secondly, we estimate 2D Keypoints from the video. Given a sequence of 2D keypoints predicted, the focus is on retrieving a series of 3D keypoints centered on the pelvis (see Fig. 2(b)).

In Fig. 1, we present the flow of our approach. First, we perform a preprocessing of the video and then we begin with the 3D estimation itself.

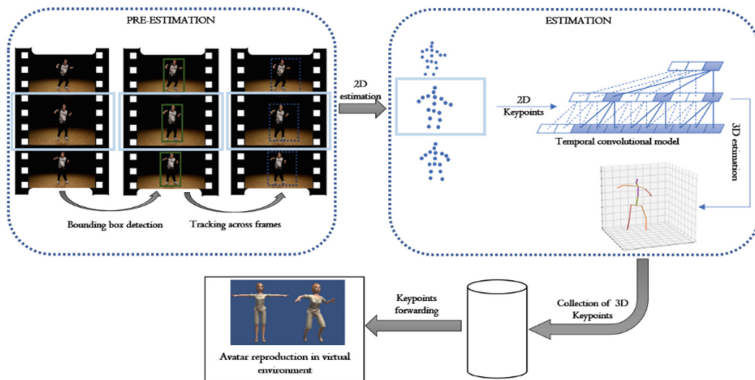


Fig. 1. Overview of proposed architecture

In the estimation phase, we use a fully convolutional architecture with residual connections that takes a series of 2D keypoints as input and transforms them through temporal convolutions. The input 2D pose sequence is represented as a 3D vector (F, J, D) , with F denoting the number of receptive fields, J represents the number of joints in each frame, and D the number of coordinate dimensions (x, y) . The temporal convolutional network used is the delated convolutional model similar to the model suggested by Pavllo et al. [8]. We use B temporal blocks where each block performs a 2D convolutions with kernel size k and dilation factor $d = k^B$, followed by a convolution with kernel size $k \times 1$. From the first Convolution to the last layer (last layer excluded), a 2D batch normalization [9] and rectified linear units [10] follows. To improve generalization, we use a dropout at the second convolution layer of blocks. Finally, the final layer generates a forecast of 3D poses for all frames in the input sequence based on both past and future data to take advantage of temporal data.

3.2 Behavior Reconstruction

Our second objective being to reproduce behaviors depicted in a video by virtual avatar, we suggest the creation of a collection of keypoints estimated in a video.

We define a behavior as the ensemble of movement depicted estimated keypoints. We have experimented the reconstruction of behaviors in a virtual environment set up in Unity through the help of a humanoid avatar by transferring the coordinates of estimated 3D keypoints to avatar character through bone rotation character to achieve the reconstruction. We present in Fig. 2 the mapping we made between estimated keypoints (on the right) and available points on avatar (on the left). The number on the avatar shows the correspondence to the estimated skeleton.

We have established a dependency of the keypoints. This dependency describes the alignment and ordering of keypoints and therefore the influence of the coordinates of one keypoint on another. Ultimately, we formulate the rotations by considering the angles formed by the keypoints and the adjacent keypoints. The keypoints of every frame are represented by a 3D vector.

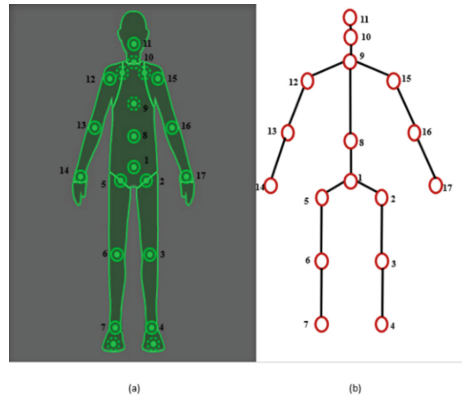


Fig. 2. Keypoints mapping between avatar and estimated skeleton.

3.3 Applications in Learning

The reconstruction of behavior by avatar introduces the notion of learning avatar. Learning avatar can be used in various domain to help individuals to learn skills. Several applications for such avatar can be found. For example, we reproduce behavior of an individual performing specific exercises to help people recovering from injuries to learn the appropriate ways to exercise. Another application is a learning system that can be built by reproducing specific behavior of patient with dementia in health domain. More specifically, we can focus on the behavior related to Alzheimer disease. To build a system in the context of Alzheimer disease, a learning avatar can be constructed by reproducing behavior of patients with Alzheimer disease such as empathy, anger, depression, agitation... This system can also be equipped with appropriate ways to interact with the learning avatar and by extension a patient with the disease. The utility of this system can be found to teach medical staff to not only recognize different behavior but also better ways to interact with patients during crisis. Such system present advantages over using recorded video of the behavior because the visualization of the behavior in 3D,

gives a better understanding of this behavior. In addition to that, the engagement with the system give to the user a perception of gaming, stimulating the learning process.

4 Experiments

We used two popular datasets for 3D pose estimation: Human3.6M [11] and HumanEva-I [12]. For the evaluation, in our experiments, we use two typical evaluation protocols. Protocol No. 1 determines the mean per joint positioning error (MPJPE), which is the average Euclidean distance between estimated and ground-truth joint locations in millimeters. Protocol No. 2 which uses a rigid alignment with the ground truth before computing the mean per joint positioning error (P-MPJPE) in millimeters.

We implemented the model by setting the number of temporal blocks B to 3 blocks and set the kernel size k to 3. For Human3.6M, we apply Amsgrad optimizer [13] with a mini-batch size of $b = 128$, and train for 80 epochs. The learning rate starts at 0.001 and then applies a learning factor $\alpha = 0.95$ in each epoch. The dropout rate p in each dropout layer is set to 0.05. For HumanEva-I, we use $b = 32$, $\alpha = 0.98$, $p = 0.5$ and train for 200 epochs.

To further improve our model during the estimation, we employ a bounding box detector [14] and rely on HRNet [15] for 2D pose estimation.

It is important to note that there is a difference in the joint setup for both datasets. In Human 3.6M we predict poses using a 17-joint skeleton and in HumanEva-I, we use 15-joints.

5 Results

3D Pose Estimation: On the dataset Human3.6M, our model achieves satisfactory performance. We present the performance on said dataset in Table 1. We represent in bold the best performance and the second best is underlined.

Table 1. Quantitative comparison on datasets

Dataset Human3.6M	Methods	Protocol No.1 (MPJPE)	Protocol No.2 (P-MPJPE)	Dataset HumanEva-I	Protocol No.2									
					Walk			Jog			Box			
					S1	S2	S3	S1	S2	S3	S1	S2	S3	
	Our previous work [8]	86.2	57.6											
	Hossain et al. [6]	58.3	44.1		Martinez et al. [5]	19.7	17.4	<u>46.8</u>	26.9	18.2	18.6	-	-	-
	Pavlo et al. [8]	46.8	36.5		Pavlo et al. [8]	<u>13.9</u>	<u>10.2</u>	46.6	20.9	13.1	13.8	28.8	33.7	32.0
	Our approach	<u>49.3</u>	<u>38.1</u>		Our approach	13.8	10.1	48.5	<u>21.3</u>	<u>14.4</u>	<u>15.6</u>	<u>24.2</u>	<u>38.7</u>	<u>35.2</u>

Our current approach shows significant improvement compared to our previous method [7] with reduction of errors almost by half. With respect to HumanEva-I dataset,

we compare our results with the performance of Martinez et al. [5] and Pavllo et al. under Protocol No.2. Our experiment showed that our model can compete with the state-of-the-art result.

Reproduction by Avatar: With the suggested approach for the pose estimation, we obtained satisfactory results that allowed smoother reproduction in the virtual environment. We present in Fig. 3 qualitative results.

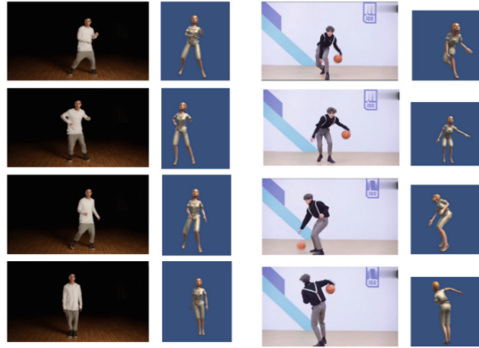


Fig. 3. Reproduction by avatar

We notice a limitation when it comes to the hands of the avatar. This limitation can be explained by the absence of keypoints related to the hands.

6 Conclusion

Through this work, we suggested a 3D pose estimation approach that provides support of temporal sequences with the objective of reproduction in a virtual environment. Our architecture proposes a preprocessing of given video by first detection of a person in frames followed by tracking of the person across the video sequence and an estimation of 2D keypoints fed to temporal convolutions. The 3D pose estimator suggested utilizes temporal information with dilated convolutions over the 2D keypoints. We compared the performance of our model with existing methods and found significant reduction of reconstruction error compared to previous methods that suggested 3D estimation for reproduction purposes. Secondly, we explored the reproduction of behaviors defined by a set of poses in the virtual environment Unity by creating a collection of keypoints and mapping of those keypoints to the available point of a humanoid avatar. We have achieved reproduction by creation of dependency in movement from one body part to the other. Finally, we presented a possible application of our work toward the creation of learning avatars that could be used in various domains.

Acknowledgment. We acknowledge NSERC-CRD (National Science and Engineering Research Council Cooperative Research Development), Prompt, and BMU (Beam Me Up) for funding this work.

References

1. Pavlakos, G., Zhou, X., Daniilidis, K.: Ordinal depth supervision for 3D human pose estimation. In: 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (2018)
2. Xu, J., et al.: Deep kinematics analysis for monocular 3D human pose estimation. In: 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2020)
3. Loper, M., et al.: SMPL: a skinned multi-person linear model. *ACM Trans. Graph.* **34**(6) (2015). p. Article 248
4. Kanazawa, A., et al.: End-to-end recovery of human shape and pose. In: 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (2018)
5. Martinez, J., et al.: A simple yet effective baseline for 3d human pose estimation. In: 2017 IEEE International Conference on Computer Vision (ICCV) (2017)
6. Hossain, M.R.I., Little, J.J.: Exploiting temporal information for 3D human pose estimation. In: Ferrari, V., Hebert, M., Sminchisescu, C., Weiss, Y. (eds.) ECCV 2018. LNCS, vol. 11214, pp. 69–86. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-01249-6_5
7. Dare, K., Ben Abdesslem, H., Frasson, C.: Extraction of 3D pose in video for building virtual learning avatars. In: Cristea, A.I., Troussas, C. (eds.) ITS 2021. LNCS, vol. 12677, pp. 512–518. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-80421-3_56
8. Pavlo, D., et al.: 3D human pose estimation in video with temporal convolutions and semi-supervised training. In: 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2019)
9. Ioffe, S., Szegedy, C.: Batch normalization: accelerating deep network training by reducing internal covariate shift. In: Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37. JMLR.org: Lille, France, p. 448–456 (2015)
10. Nair, V., Hinton, G.E.: Rectified linear units improve restricted boltzmann machines. In: Proceedings of the 27th International Conference on International Conference on Machine Learning, pp. 807–814. Omnipress: Haifa, Israel (2010)
11. Ionescu, C., et al.: Human3.6M: large scale datasets and predictive methods for 3D human sensing in natural environments. *IEEE Trans. Pattern Anal. Mach. Intell.* **36**(7), 1325–1339 (2014)
12. Sigal, L., Balan, A.O., Black, M.J.: HumanEva: synchronized video and motion capture dataset and baseline algorithm for evaluation of articulated human motion. *Int. J. Comput. Vision* **87**(1), 4 (2009)
13. Reddi, S.J., Kale, S., Kumar, S.: On the convergence of Adam and beyond. *ArXiv* (2018). [abs/1904.09237](https://arxiv.org/abs/1904.09237)
14. Redmon, J., Farhadi, A.: YOLOv3: an incremental improvement. *ArXiv* (2018). [abs/1804.02767](https://arxiv.org/abs/1804.02767)
15. Sun, K., et al.: Deep high-resolution representation learning for human pose estimation. In: 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2019)



Evaluation Test Generation Model Using Degrees of Difficulty and Keywords

Doru Anastasiu Popescu^(✉) and Mariana Madalina Nastase

Department of Mathematics and Informatics, University of Pitesti, Pitesti, Romania
dopopan@yahoo.com

Abstract. With the restrictions related to “face to face” teaching activities, they have developed in a way fast online and video teaching and evaluation mechanisms. In addition to the means developed on online evaluation-related platforms, many researchers have developed their test models. In this paper, we present a test generator model based on genetic algorithms. The tests are created starting from a certain degree of difficulty and a list of keywords that their questions must contain.

Keywords: Test generation · Genetic algorithm · Keywords · Difficulty

1 Introduction

It is a known fact that, during the Covid-19 pandemic, the educational systems were overwhelmed. Moving all activities into an online environment, not just that it restricted communication, but it opened doors to many threats. However, it also opened doors to many opportunities.

One of the most visible issues, that arise from the difficulties of a fair assessment, is the growth of cheating practices. One of the obvious solutions to alleviate this problem is to create different test sheets for each student. However, this approach is discouraging, mainly, because it is very time-consuming and not quite feasible.

Most e-learning platforms include a test generating mechanism and there are also specialized systems that facilitate assessments creation. A detailed study on the functionalities of the most used e-learning platforms and testing tools can be found in [5]. However, most of them only allow the test questions to be introduced one by one. Some of them provide a way of importing questions from files. Moodle provides a so-called, “question bank”, where previously used questions can be stored according to certain categories.

In recent years, many pieces of research have been conducted to explore different methods that could make the evaluation process easier and much more efficient. The work presented in [7] and [8] present distinct approaches on generating examination papers. The use of genetic algorithms is explored in papers [9] and [10], being closely related to the work we conducted in this paper.

The solution, this paper introduces, is a system that can generate tests, having as a “starting point” a list of keywords that links the questions to a specific topic. Also, the number of questions in each test and the difficulty level can vary, allowing the professor to create custom tests, according to the expected level of knowledge of the students and the course needs at a certain point in time, aspects on which the professor has the liberty to decide. From this point of view, the generator brings certain flexibility when it comes down to tests creation, in addition to the above-mentioned articles.

The model in this paper uses questions grouped by keywords and degrees of difficulty, other models use hierarchies of questions represented by trees such as the one presented in [3].

From the algorithmic point of view, although a genetic algorithm can be time-consuming, the randomness of its solutions and the fact that it avoids searching for the local optimal solution, like traditional procedures, makes it the most suitable approach for this case study. It also allows a hypothetical expansion in the number of parameters, which is a good foundation for future work.

The purpose of this research is to minimize the downsides of the method used (the execution time of the algorithm itself) while maximizing the strong points of the approach (the randomness, easiness of use, flexibility). This paper is a continuation of the work presented in [2,3] and [4], and it is subject to further development as it will be later discussed.

2 Initial Conditions

The algorithm was created concerning the following scenario: A professor needs one or more tests for his course. He selects several questions for each test, a level of difficulty and some keywords that represent the topics that must appear in each test. Therefore, each question has the following structure: an unique identifier, the definition of the question, answer options, the correct answer, a keywords list and a difficulty level.

Out of these items, the terms “Keyword” and “Difficulty” are the most relevant since they impact the fitness function and generation selection of the algorithm and are at the core of the solution proposed.

The term “Keyword” does not have a specific definition, since it can be anything from the name of a course to a study topic.

“Difficulty” is also rather volatile since its role can be defined by the expected knowledge level of those that are being tested, or by the purpose of the examination. Therefore, the algorithm expects the following three input parameters, concerning the composition of the generated test:

- **Keywords list** *kwl*: One or more keywords are selected from the list of all possible keywords; each question that will appear on the test must be related to at least one of the selected keywords;
- **Number of questions** *nq*: The number of questions selected for the test;
- **Difficulty** *df*: Each question selected as a fit for the test must be as close as possible to the chosen difficulty level;

And the following three serve as input for the genetic algorithm:

- **Population Size** ps : The initial size of randomly generated tests that will serve as parents of the next generation;
- **Number of generations** ng : The number of generations that the algorithm will generate;
- **Elitism Index** e : The index of mutation for each generation;

3 Algorithm Description

3.1 Terms Used and Definitions

In this section, the following term definitions and figures were prepared to achieve a better understanding of the theoretical concepts.

Definition 1: The set KWL is defined as the set containing all possible keywords that can be selected as test parameters or that can define the topic of a question.

Definition 2: A question q is defined as $q = (q_id, st, ca, d, ao, kl)$, where q_id is the unique identifier of q , st is the statement, ca is the correct answer, d is the difficulty level, ao is a set of three possible answers, kl is a set of keywords related to the question, with $kl \in KWL$ (Fig. 1);

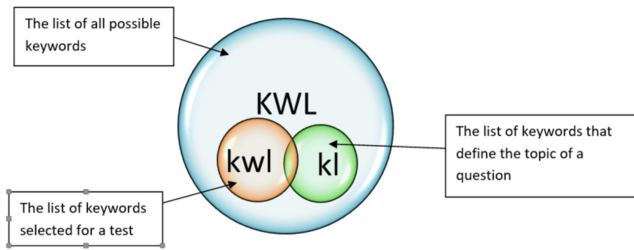


Fig. 1. Keywords lists definition

Definition 3: A chromosome C is a set of nq components, called genes. These genes are represented by questions. Therefore, a chromosome is represented by a generated test.

Definition 4: The initial population is composed of ps randomly generated chromosomes. These chromosomes are called parents on which the algorithm will apply mutations and crossover operations to build the next generation. This process of building new generations will be repeated ng times according to the elitism index e .

Note: Despite the common approach where the elitism index defines the per cent of the current population that will survive and reproduce in the next one,

in this paper the elitism index defines the mutation per cent of the individuals. Therefore, the combined per cent of modifications (mutation and crossover) that occur in the population is fixed at 50%. This approach will be better explained in Sect. 3.6

3.2 Generating Initial Population

Prerequisites: population size (ps), keywords list (kwl) and number of questions (nq).

The initial population is generated as follows:

1. We select from the database all questions which contain at least one keyword from the list of keywords kwl given as test input;
2. We randomly select nq distinct questions from the above list and save them using their unique id;
3. We repeat step 2 until we achieve ps distinct chromosomes;

The level of knowledge expected from the students expressed through the difficulty level is not used here as it might restrict the diversity of questions. However, the concept will be used in computing the fitness function.

3.3 Fitness Function

Prerequisites: A chromosome (C) and a chosen difficulty (df).

The fitness function $f(C)$ is given by the following formula:

$$f(C) = \frac{\sum_{i=1}^{nq} |d_i - df|}{nq}, \quad (1)$$

where i is a gene of the chromosome C and d_i is the difficulty of the question represented by that gene.

This function is used to measure the quality of a chromosome. The closer the value of $f(C)$ is to zero, the better the quality of the chromosome.

3.4 Mutation

Prerequisites: A chromosome (C) and a random positive integer a .

The mutation applied in this algorithm is a *random resetting*. This type of mutation implies selecting at random a position a and changing the gene in that position with a randomly generated and distinct one.

For example, let us assume that we have chromosome C :

$C = [145, 263, 387, 345, 498, 18, 897, 467, 578, 12]$

and an integer $a = 3$.

The resulting chromosome could be C' :

$C' = [145, 263, 2, 345, 498, 18, 897, 467, 578, 12]$,

where the gene in position 3 (387) is switched with a random gene that does not yet exist in the chromosome (2).

3.5 Crossover

Prerequisites: Two chromosomes (C_1 and C_2) and a random positive integer p .

The crossover operation applied in this algorithm is one point crossover as explained in [11]. Let us have the following two chromosomes $C1$ and $C2$:

$C1 = [1, 256, 34, 987, 452, 556, 45]$

$C2 = [23, 345, 765, 456, 1, 234, 999]$

and a random $p = 4$

The result will be:

$C1' = [1, 256, 34, 456, 1, 234, 999]$

$C2' = [23, 345, 765, 987, 452, 556, 45]$

However, we notice that chromosome $C1'$ has a duplicate value of 1, so we replace one of the duplicates with a random value that does not exist yet in the chromosome.

A possible result could be: $C1' = [1, 256, 34, 456, 500, 234, 999]$

3.6 Algorithm Workflow

Prerequisites: A list of keywords (kwl), the number of questions (nq), the difficulty level of the test (df), the size of the population (ps), the number of generations (ng) and the elitism index (e).

Algorithm steps:

1. Generate initial population, as described in Sect. 3.2;
2. Set an index for the first generation and keep track of the number of current generation;
3. Sort the initial population using the fitness function, as described in Sect. 3.3;
4. Apply mutation on $e\%$ randomly selected chromosomes from the population, as described in Sect. 3.4 and add the off-springs to it;
5. Apply crossover on $50 - e\%$ randomly selected chromosomes from the population, as described in Sect. 3.5 and add the off-springs to it;
6. Select chromosomes for the next generation;

Note that mutations and crossover operations generate additional 50% chromosomes that are added to the current generation. In order to restore the initial population size we select $90 * ps/100$ best chromosomes according to the fitness function, and $10 * ps/100$ randomly selected chromosomes from the remaining population.

7. Increase the current generation index;

Steps from 4 to 7 are executed repeatedly until one of the stopping conditions is satisfied.

Stopping Conditions: The following three cases are considered enough to stop the algorithm workflow;

- The index of the current generation is equal to the number of generations (nq) provided by the user;

- The value of fitness function for the considered fittest chromosome from the current generation reached 0;
- The value of fitness function for the considered fittest chromosome from the current generation did not increase for a couple of generations (ex: 10 generations without improvement of the fitness function value);

The solution is represented by the chromosome with the lowest fitness value from the last generation (Fig. 2).

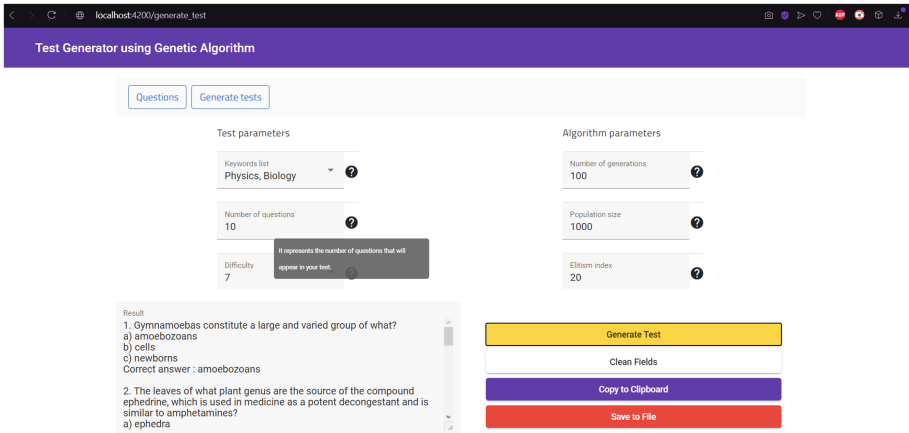


Fig. 2. Generator interface

4 Conclusion and Results

A more and more common belief, which is supported by this project, is that the world needs to embrace the possibilities provided by the automation of processes. Even though, until now, this idea was more often seen in business projects, the pandemic showed the need for a sustainable and flexible educational system that can evolve depending on the situation.

The current implementation of this paper is represented by a standalone web application, but the system could be easily integrated into any other e-learning system or used as a library for further development.

For the fundamental operations of the genetic algorithm: mutation and crossing, other methods can be chosen, such as those presented in [6] and [9]. A comparative study related to these variants we want to carry out in the next period. The novelty of this paper is the use of keywords and the degree of difficulty for the test and its questions integrated into an application that uses genetic algorithms.

References

1. SciQ Dataset - Allen Institute for AI. <https://doi.org/10.18653/v1>
2. Popescu, D.A., Bold, N., Domsa, O.: Generating assessment tests with restrictions using genetic algorithms. In: 2016 12th IEEE International Conference on Control and Automation (ICCA), pp. 696–700. ICCA (2016)
3. Popescu, D.A., Bold, N., Domsa, O.: A generator of sequences of hierarchical tests which contain specified keywords. In: 2016 IEEE 11th International Symposium on Applied Computational Intelligence and Informatics (SACI), pp. 255–260. SACI (2016)
4. Popescu, D.A., Bold, N., Nijloveanu, D.: A method based on genetic algorithms for generating assessment tests used for learning. *Journal* **54**, 53–60 (2016)
5. Kostadinova, I., Rasheva-Yordanova, K., Garvanova, M.: Analysis of algorithms for generating test questions in E-testing systems. In: Proceedings of EDULEARN19 Conference, pp. 1–3 (2019)
6. Mitchell, M.: *An Introduction to Genetic Algorithms*. MIT Press (1996)
7. Guang, C., et al.: A implementation of an automatic examination paper generation system. *Math. Comput. Model.* **51**(11–12), 1339–1342 (2010)
8. Thigale, P., Nadkar, S., Kokle, B., Bhole, N., Ghodichor, F.S.: Automatic question paper generator system by keyword based shuffling algorithm using randomization technique. In: IJARCCCE, vol. 10, no. 5, May 2021. <https://doi.org/10.17148/IJARCCCE.2021.105118>
9. Li, Y., Li, S., Li, X.: Test paper generating method based on genetic algorithm. *AASRI, Procedia* **1**, 549–553 (2012)
10. Chen, X.-D and Wang, H.-Y., Exam-paper generating algorithm based on improved genetic algorithm. *J. Harbin Inst. Technol.* **37**, 1174–1176+1248 (2005)
11. Mehboob, U., Qadir, J., Ali, S., Vasilakos, A.: Genetic algorithms in wireless networking: techniques, applications, and issues. *Soft Comput.* **20**(6), 2467–2501 (2016). <https://doi.org/10.1007/s00500-016-2070-9>



Double-Layer Controller for Detecting Learners' Erroneous Knowledge in Database Programming

Christos Troussas^(✉) , Akriki Krouska , and Cleo Sgouropoulou 

Department of Informatics and Computer Engineering, University of West Attica, 12243 Egaleo, Greece

{ctrouss, akrouska, csgouro}@uniwa.gr

Abstract. Adaptive learning systems employ educational techniques and use computer algorithms to orchestrate the interaction with the learner. One example of their activities is the error detection and diagnosis. Error diagnosis serves for identifying the learners' mistakes, their nature and the reason for happening. This module is important since it can help learners advance their knowledge. In view of this compelling need, this paper presents a double-layer controller for detecting learners' erroneous knowledge in the tutoring of database programming, and specifically the Structured Query Language. The controller can reason about two main error categories, i.e. syntax and logic errors, and holds syntax and logical check operators. For the error diagnosis process, the syntax check operator incorporates a string-matching similarity technique, while the logical check operator incorporates the Start End Mid algorithm and the Sørensen–Dice coefficient. The educational software for the database programming, that incorporates the double-layer controller, was evaluated and the results showed that the presented mechanism has a significant effect on learners' performance.

Keywords: Adaptive content · Error diagnosis · Misconception diagnosis

1 Introduction

With the development of Information and Communication Technologies, people have witnessed many improvements in different fields of their daily life. One example is the field of education. Specifically, education has been enriched with custom software to offer its benefits to a larger pool of people. To this end, adaptive learning systems have prevailed in this field, since they are developed to dynamically tailor to the type or level of course material based on an individual learner's knowledge and capabilities, in ways that boost this/her performance with both automatic and manual interventions [1]. Adaptive learning systems employ educational techniques and use computer algorithms as well as artificial intelligence to orchestrate the interaction with the learner [2, 3]. The advantages of such systems include the delivery of personalized material and/or learning activities to learners towards addressing their specific needs and preferences.

The characteristics of adaptive learning systems include: automated processes of learners' assessment, addressing the different knowledge levels of learners, delivery of automated and detailed feedback to learners, regulating course content degree of difficulty, personalized domain knowledge delivery to students and error detection and diagnosis [4, 5].

Error diagnosis can be an important module in adaptive learning systems that serves for diagnosing the learners' mistakes and can reason about the nature and the reason that they happened [6]. In computer programming, two are the main errors, i.e. syntax and logic errors. A syntax error is a mistake in the grammar of a programming language, influencing the presence or sequence of tokens or characters in a program. A logical error is a mistake in the program which makes it operate mistakenly, but it can run without problems, delivering a non-expected result.

Analyzing the related literature, there is evidence that the field of error diagnosis has not been adequately explored mainly in the tutoring of computer programming, not to mention database programming. The researches in this field employ error diagnosis for identifying either syntax or logic errors, by employing different smart techniques, which have already been used variously in the literature, such as fuzzy logic, periodical advice delivery about program's behavior, concept maps, string similarities, Levenshtein distance [7–17]. It needs to be noted that a recent study points out that there is a fertile ground to further research the field of diagnosis learners' errors in computer programming [7]. Another recent review underlines that the advances in error diagnosis specifically for database programming, i.e. SQL, are limited [18].

The novelty of this research lies in proposing two different novel algorithms for syntax and logic error diagnosis, which are incorporated in a double-layer controller. The double-layer controller serves for the algorithms to operate smoothly and synergistically, bringing best results.

In view of the above, this paper presents two novel algorithmic techniques for diagnosis learners' erroneous knowledge in the tutoring of database programming. More specifically, the system incorporates a syntax error control operator, which uses a string-matching similarity technique for detecting the corresponding category mistakes in students' answers, and a logical error control operator, which combines the Start End Mid algorithm and the Sørensen–Dice coefficient for identifying logical mistakes. The presented system was evaluated using pre/post nonequivalent groups design in order its effectiveness on student performance to be compared with a conventional learning system, which has no error detection mechanism.

2 Double-Layer Controller Architecture

During the process of students' assessment, when they submit a response, this response passes inside the double-layer controller. Firstly, it goes through the syntax check operator. It is analyzed and if syntax errors are detected, they are presented to the user without checking logical errors. In case the double-layer controller does not record syntax errors, then the answer is proceeded to the logical error control operator for further analysis; if logical errors are identified, they are presented to the user.

2.1 Syntax Error Control Operator

The syntax check operator is based on the user's response, which is the SQL language. Once the query is given by the user to execute, it is checked for errors; in that case, the answer is further analyzed and the reserved words that define the structure of the query are identified. For each keyword, its structure is checked according to the SQL documentation which specifically defines its structure and what is acceptable. Incorrectly given structures are collected and returned to the user. The algorithm of syntax error control operator is as follows:

```

syntaxErrors ← empty;
errorType ← "SYNTAX"
if language == "MySQL" then
    reservedKeywords ← initKeywordList();
    if wrongAnswer.startsWith( "SELECT" ) then
        for reservedKeyword:reservedKeywords do
            //separating wrong answer based on reserved keyword
            position;
            finalText ← keywordPartition(wrongAnswer,
reservedKeyword);
            reason ← getReasonFormat(finalText);
            syntaxErrors ← add(errorEntity(reservedKeyword, reason,
errorType));
        end
    else
        keyword ← "";
        if wrongAnswer.startsWith( "INSERT" ) then
            keyword ← "INSERT";
        else if wrongAnswer.startsWith( "UPDATE" ) then
            keyword ← "UPDATE";
        else if wrongAnswer.startsWith( "CREATE" ) then
            keyword ← "CREATE";
        end
        keywordStatus ← checkFormat(wrongAnswer, keyword);
        if keywordStatus == TRUE then
            reason ← getReasonFormat(keyword);
            syntaxErrors ← add(errorEntity(keyword, reason,
errorType));
        end
    end
return syntaxErrors

```

2.2 Logical Error Control Operator

It needs to be noted that the handling of logical errors is not as clear as the handling of syntax errors. Thus, in the present work, multiple techniques have been implemented that work in combination so that the detection of logical errors is effective.

The first technique that is implemented is the comparison of the two answers, namely the user's answer and the correct answer. The two answers are broken into pieces and checked for their similarity. At the end of this process, these results are collected and the logical error control operator proceeds with the next technique.

The second technique, which is implemented in combination, is a variation of the "Start End Mid" algorithm [19]. With this technique, the user's query as well as the correct query are executed, and both results are retrieved and compared. Each part of the user's query that is different from the correct query is presented in a list, so that the user can be presented with his/her mistakes in a clear way.

The third and final technique used to determine logical errors is to the Sørensen-Dice coefficient [20, 21]. This algorithm belongs to the k-shingling family against which computes the shingles, i.e. n-character sequences, for each coefficient of comparison. The algorithm accepts the two variables where, as a result, their similarity and distance are produced.

The similarity is proportionate to the size of intersection and inversely proportional to the user's query (given answer) and the correct query (correct answer).

$$\text{Similarity} = \frac{2 * \text{sizeofintersection}}{\text{user'squery} + \text{correctquery}}$$

The intersection is defined as the number of occurrences of the keys of the two answers, calculating the k-shingles for each answer. The implementation was based on the implementation of Esko Ukkonen [22] which implements an algorithm for extracting distinctive k-shingles which are essentially the k characters that appear sequentially in a document. In our case, k is set to 3 as it matches the number of answers. Variables are the user's query (given answer) and the correct query (correct answer). The similarity takes values from 0 to 1, with the closest to 0 indicating that the two answers have less similarities and respectively the closest to 1 indicating that the two answers are more similar. 0 indicates that they are completely different and have nothing in common while 1 indicates that they are identical.

This coefficient is used in combination with the previous two techniques for finding logical errors and through performed empirical tests with users' answers and correct answers, the coefficient used for the validity of the answers should have a value greater than 0.7.

When the coefficient is calculated less than or equal to 0.7, then the logical errors calculated by the previous techniques remain and are presented to the user. When it is greater than 0.7, the logical errors that have been calculated in conjunction with the execution results are deleted as it is observed that the user, even with some logical errors, eventually has the appropriate knowledge. This rate is subject to change if deemed necessary mainly through the continuous use of the platform and the monitoring of the exported results.

Furthermore, in the Sørensen – Dice algorithm, the distance calculated from the difference of the unit with the similarity, calculated as mentioned above, is also in the fraction. It is not used directly in the handling and calculation of logical errors, but is stored for further analysis.

$$Distance = 1 - Similarity$$

The algorithm of logical error control operator is as follows:

```

logicErrors ← empty;
errorType ← “LOGICAL”
if language == “MySQL” then
    logicErrors ← compareStrings(correctAnswer, wrongAnswer);
    //executing correct and wrong answers;
    wrongValues ← getValuesFromColumns(wrongAnswer);
    correctValues ← getValuesFromColumns(correctAnswer);
    similarity ← sorensenDice.similarity(wrongAnswer, correctAnswer);
    if wrongValues != correctValues & similarity <=0.7 then
        return logicErrors;
    end
    logicErrors ← empty;
    return logicErrors;
end

```

3 Evaluation Results and Discussion

The participants involved in the evaluation process were 60 undergraduate students, in a public university, at its Department of Informatics. The students were in the second year of their studies, having an average age of 19.58 years. The evaluators separated the students in the experimental group and the control one. Both groups consisted of 30 students, each having similar characteristics, concerning demographic data and levels of knowledge, skills, and abilities. The evaluation process lasted one academic semester; and during that period, the experimental group used the presented system for learning the SQL language, while the control group used a conventional system, having the same user interface and content but without error diagnosis. The reason why the conventional system used was to investigate the potential of our system in comparison to traditional ones.

In order to assess the effectiveness of the presented system on student performance, a pre/post nonequivalent groups design was used. As such, both the experimental and control groups were given a pretest before the use of the corresponding systems, for assessing their prior knowledge on course subject. At the end of the course, they were given a posttest for evaluating the knowledge acquired. This evaluation design was chosen in order to investigate not only whether participants who used the presented system improved their performance, but also whether it was improved more than that

of participants who used the conventional system. Hence, the potential of the developed system is estimated.

The results of paired t-test between the pretest and posttest grades for each group are illustrated in Table 1. Regarding the experimental group, the mean of pretests was 57.77%, while this of posttests was 77.13%, having a difference of 19.37 units. Moreover, the t-test results showed a significant improvement of students' performance, since t Stat value was -20.65 and P-value was $6.86E-19$, which is lower than 0.05. On the other hand, the mean of pretests of the control group was 57.03%, while this of posttests was 71.13%, having a difference of 14.10 units. The t-test on control group pretest/posttest grades revealed a significant improvement on learning outcomes, since $t = -17.29$ and $P = 8.18E-17 < 0.05$; however, this improvement is in lower extent than that of experimental group. These findings are in line with the hypothesis test related to pretest grades of the two groups (Table 2) and this of posttest grades of the two groups (Table 3). In particular, the P-value of pretest grades of the experimental and the control group is 0.68, higher than the alpha 0.05; as such, there is no significant difference in previous knowledge of students of both groups. To the contrary, the hypothesis test related to posttest grades of the two groups showed a significant difference in learning outcomes of students, having P-value of 0.01 and smaller than alpha 0.05. All the above illustrates the superiority of the presented system towards the conventional one, and in essence, the effectiveness of the double-layer controller for detecting students' erroneous knowledge in database programming and helping students improve their performance.

Table 1. Pretest/posttest testing results

	Experimental group	Control group
Pretest mean	57.77	57.03
Posttest mean	77.13	71.13
Observations	30	30
Mean difference	19.37	14.10
Standard deviation	5.14	4.47
Confidence level (95%)	1.92	1.67
Pearson correlation	0.87	0.84
t Stat	-20.65	-17.29
P-value	$6.86E-19$	$8.18E-17$

Table 2. Hypothesis test on pretest results

	Experimental group	Control group
Mean	57.77	57.03
Variance	51.01	42.72
Observations	30	30
Pooled variance	48.87	
Hypothesized mean Difference	0	
df	58	
t Stat	0.41	
P-value	0.68	

Table 3. Hypothesis test on posttest results

	Experimental group	Control group
Mean	77.13	71.13
Variance	100.19	66.33
Observations	30	30
Pooled variance	83.26	
Hypothesized mean Difference	0	
df	58	
t Stat	2.55	
P-value	0.01	

4 Conclusions

This paper presents a novel double-layer controller for diagnosing learners' mistakes in SQL tutoring. This controller is activated when the learner tries to compile the query. The answer firstly goes through the syntax check operator. It is analyzed and if syntax errors are detected, they are presented to the user without checking logical errors. In case the double-layer controller does not record syntax errors, then the answer is proceeded to the logical error control operator and it is analyzed; if logical errors are identified, they are presented to the user.

The syntax control operator employs a string-matching technique to detect possible syntax mistakes. The logic control operator incorporates the Start End Mid algorithm and the Sørensen–Dice coefficient to perform logical errors diagnosis. The software

for the tutoring of SQL programming, incorporating the double-layer controller, was evaluated and the results showed that the presented mechanism has a significant effect on improving student performance.

Future research steps include a more detailed evaluation in the diagnosis of corresponding errors in other programming languages. Moreover, the exploration of the employment of other algorithmic approaches for the error diagnosis is in our next plans.

References

1. Rüdian, S., Pinkwart, N.: Generating adaptive and personalized language learning online courses in Moodle with individual learning paths using templates. In: 2021 IEEE International Conference on Advanced Learning Technologies (ICALT), pp. 53–55 (2021). <https://doi.org/10.1109/ICALT52272.2021.00024>
2. Troussas, C., Krouska, A., Alepis, E., Virvou, M.: Intelligent and adaptive tutoring through a social network for higher education. *New Rev. Hypermedia Multimedia* **26**(3–4), 138–167 (2020). <https://doi.org/10.1080/13614568.2021.1908436>
3. Chiu, T.K.F., Meng, H., Chai, C.-S., King, I., Wong, S., Yam, Y.: Creation and evaluation of a pretertiary Artificial Intelligence (AI) curriculum. *IEEE Trans. Educ.* **65**(1), 30–39 (2022). <https://doi.org/10.1109/TE.2021.3085878>
4. Huamani, G.T., Inga, P.M.T.: WIP Adaptive evaluation for a systems theory course according to the learning context. In: 2021 IEEE World Conference on Engineering Education (EDUNINE), pp. 1–4 (2021). <https://doi.org/10.1109/EDUNINE51952.2021.9429166>
5. Troussas, C., Krouska, A., Sgouropoulou, C.: A novel teaching strategy through adaptive learning activities for computer programming. *IEEE Trans. Educ.* **64**(2), 103–109 (2021). <https://doi.org/10.1109/TE.2020.3012744>
6. Xu, S., Chee, Y.S.: Transformation-based diagnosis of student programs for programming tutoring systems. *IEEE Trans. Software Eng.* **29**(4), 360–384 (2003). <https://doi.org/10.1109/TSE.2003.1191799>
7. Henley, A.Z., Ball, J., Klein, B., Rutter, A., Lee, D.: An inquisitive code editor for addressing novice programmers' misconceptions of program behavior. In: Proceedings of the 2021 IEEE/ACM 43rd International Conference on Software Engineering: Software Engineering Education and Training (ICSE-SEET), Madrid, Spain, 25–28 May 2021, pp. 165–170 (2021)
8. Lai, A.F., Wu, T.T., Lee, G.Y., Lai, H.Y.: Developing a web-based simulation-based learning system for enhancing concepts of linked-list structures in data structures curriculum. In: Proceedings of the 2015 3rd International Conference on Artificial Intelligence, Modelling and Simulation (AIMS), Kota Kinabalu, Malaysia, 2–4 December 2015, pp. 185–188 (2015)
9. Chang, J.-C.; Li, S.-C.; Chang, A.; Chang, M.: A SCORM/IMS compliance online test and diagnosis system. In: Proceedings of the 2006 7th International Conference on Information Technology Based Higher Education and Training, Ultimo, Australia, 10–13 July 2006, pp. 343–352 (2006)
10. Barker, S., Douglas, P.: An intelligent tutoring system for program semantics. In: Proceedings of the International Conference on Information Technology: Coding and Computing (ITCC 2005), Las Vegas, NV, USA, 4–6 April 2005, vol. 1, pp. 482–487
11. Khalife, J.: Threshold for the introduction of programming: providing learners with a simple computer model. In: Proceedings of the 28th International Conference on Information Technology Interfaces, Cavtat, Croatia, 19–22 June 2006, pp. 71–76 (2006)
12. Troussas, C., Krouska, A., Virvou, M.: Injecting intelligence into learning management systems: the case of adaptive grain-size instruction. In: 2019 10th International Conference on Information, Intelligence, Systems and Applications (IISA), pp. 1–6 (2019). <https://doi.org/10.1109/IISA.2019.8900779>

13. Krugel, J., et al.: Automated measurement of competencies and generation of feedback in object-oriented programming courses. In: Proceedings of the 2020 IEEE Global Engineering Education Conference (EDUCON), Porto, Portugal, 27–30 April 2020, pp. 329–338 (2020)
14. Troussas, C., Krouska, A., Virvou, M.: NLP-based error analysis and dynamic motivation techniques in mobile learning. In: 2019 10th International Conference on Information, Intelligence, Systems and Applications (IISA), pp. 1–8 (2019). <https://doi.org/10.1109/IISA.2019.8900729>
15. Leal-Flores, A.J., Gonzalez-Guerra, L.H.: Teamwork with an automatic tutoring environment as learning strategy in programming courses. In: 2021 IEEE Global Engineering Education Conference (EDUCON), pp. 131–135 (2021). <https://doi.org/10.1109/EDUCON46332.2021.9454102>
16. Algaaribeh, S.M., Dousay, T.A., Jeffery, C.L.: Integrated learning development environment for learning and teaching C/C++ language to novice programmers. In: 2020 IEEE Frontiers in Education Conference (FIE), pp. 1–5 (2020). <https://doi.org/10.1109/FIE44824.2020.9273887>
17. Thurner, V.: Fostering the comprehension of the object-oriented programming paradigm by a virtual lab exercise. In: 2019 5th Experiment International Conference (exp.at 2019), pp. 137–142 (2019). <https://doi.org/10.1109/EXPAT.2019.8876484>
18. Paladines, J., Ramirez, J.: A systematic literature review of intelligent tutoring systems with dialogue in natural language. *IEEE Access* **8**, 164246–164267 (2020). <https://doi.org/10.1109/ACCESS.2020.3021383>
19. Ardiansah, J.T., Wibawa, A.P., Widiyaningtyas, T., Yasuhisa, O.: SQL logic error detection using start end mid algorithm. *Knowl. Eng. Data Sci. (KEDS)* **1**(1), 33–38 (2018). <https://doi.org/10.17977/um018v1i12017p33-38>, pISSN 2597–4602
20. Sørensen, T.: A method of establishing groups of equal amplitude in plant sociology based on similarity of species and its application to analyses of the vegetation on Danish commons. *Kongelige Danske Videnskabernes Selskab.* **5**(4), 1–34 (1948)
21. Dice, L.R.: Measures of the amount of ecologic association between species. *Ecology* **26**(3), 297–302 (1945). <https://doi.org/10.2307/1932409>. JSTOR1932409
22. Ukkonen, E.: Approximate string-matching with q-grams and maximal matches. *Theoret. Comput. Sci.* **92**(1), 191–211 (1992)



Identifying Metacognitive Processes Using Trace Data in an Open-Ended Problem-Solving Learning Environment

Rumana Pathan^(✉) , Daevesh Singh , Sahana Murthy, and Ramkumar Rajendran 

Indian Institute of Technology Bombay, Mumbai, India
rumana_pathan@iitb.ac.in

Abstract. Learning with open-ended learning environments (OELE) requires employing several metacognitive phases (e.g., planning and activation, monitoring) and processes (e.g., target goal setting, selection and adaptation of cognitive strategies). But, since such metacognitive phases and processes are challenging to execute by novice learners, it is essential to provide personalised feedback. Trace data is widely used to analyse metacognitive phases and processes, yet no existing research automatically identifies such phases/ processes using prediction models. This paper demonstrates the automatic prediction of four phases and five metacognitive processes using trace and think-aloud data of four learners interacting with a problem-solving OELE (MEtLE). We found that the Random Forest model trained using features extracted from learner interactions helps predict with a classification accuracy of up to 0.84, and Cohen's kappa value that signifies fair to substantial agreement. We have used one-vs-all for multiclass classification with 10-fold cross-validation. Also, we applied SMOTE algorithm to upsample minority class instances to improve prediction models.

Keywords: Metacognitive processes · Trace data · Think-aloud data · Prediction model · Open-ended learning environment

1 Introduction

To effectively learn in an open-ended learning environment (OELE), novice learners must employ and regulate several metacognitive phases (e.g., planning and activation) and processes (e.g., analyse the learning context, set learning goals, decide and use learning strategies, assess them, and monitor emerging understanding) [1]. Although crucial, novice learners do not regulate metacognitive phases/processes from time to time [2]. Since failed regulation can stop the learners from accomplishing the learning goals [3], there is a need to identify novice learners' metacognitive phases/processes in an OELE and provide scaffolds or adaptive feedback.

There are several ways to identify learners' metacognitive phases/processes, such as self-report questionnaires, think-aloud protocols, classroom observations, etc. [4]. Such measures elucidate metacognitive phases/processes and help us understand the nature of self-regulated learning (SRL). However, there are a few drawbacks; for instance, novice

learners are poor reporters and calibrators of their learning activities [5], thus questioning the validity of self-reported measures. Similarly, thinking aloud requires learners to report what they think consciously and demands a time-consuming manual analysis. On the other hand, several studies have used computer-generated trace data, also known as event logs, to identify and measure metacognitive phases/processes [6]. Moreover, trace data of learners enables real-time identification of metacognitive phases/processes, which is paramount in providing adaptive scaffolds to learners.

The computer-generated traces facilitate the implementation of various learning analytics techniques, such as sequential pattern mining, process mining, clustering of similar learning behaviours, etc., that allow us to understand the dynamic behaviour of SRL [7]. Using such analysis techniques, several researchers have visualised frequencies of metacognitive processes used by learners, identified patterns and correlations between metacognitive processes, and clustered learners based on their metacognitive behaviour [7–10]. Some current research work identifies metacognitive processes of learners in real-time by identifying sample events or defining a library of phases/processes generated via hypothesis and data-driven techniques (such as pattern mining process mining) [3]. While these techniques capture learners' course of actions and associate meaningful inferences, it does not validate/confirm if the learner is actually displaying the metacognitive phase/process. For example, in the existing approach, it is assumed that when a learner uses a planning tool, he is undergoing processes related to target goal setting (planning). But using the planning tool can also be an action associated with monitoring learners' goals. Hence, we need a machine learning (ML) approach that identifies the metacognitive phase/processes based on the combination/sequence of action(s). Such a technique will be robust and provide valuable information for decision making. Moreover, no existing research employs trace data and think-aloud data to identify metacognitive processes using ML models.

Hence, the goal of our paper is to automatically identify metacognitive phases and processes displayed by learners in a problem-solving OELE. To accomplish this goal, we collected computer-generated trace data and think-aloud data of four learners interacting with a problem-solving OELE, viz. MEttLE. MEttLE (Modelling-based estimation learning environment) is a web-based learning environment for engineering estimation problem solving [11]. The following sections will describe the OELE context and the learner interactions captured (Sect. 2), followed by a literature synthesis of existing work and research identifying metacognitive phases and/or processes using trace data (Sect. 3). Further, we describe the research goal and procedure for identifying metacognitive phases and processes (Sect. 4). Finally, we report results and discussion (Sect. 5), followed by a conclusion and future work (Sect. 6).

2 OELE Context: MEttLE

MEttLE is an OELE that supports the estimation problem solving of novice learners [11]. In this learning environment, learners are given a task, such as estimating the power required by the motor of a racing car, with certain specifications (e.g., wheel diameter, track distance to be covered, and car weight). The intertwining cognitive and metacognitive tasks designed in MEttLE support learners during problem-solving by

providing metacognitive prompts. It is also equipped with metacognitive prompts and expert guidance to help learners plan, monitor, and reflect on their choices.

The estimation problem-solving process in METtLE is structured into five tasks, i.e., functional, qualitative, and quantitative modelling, followed by calculation and evaluation. The functional model stimulates learners to detail how the system works, identify various entities involved, and identify their connections. Similarly, learners are triggered to determine relationships between the multiple entities involved in the qualitative modelling sub-goal, which can be incorporated into an equation in the quantitative modelling sub-goal to estimate the power required. Throughout the modelling phase, learners resolve inefficiencies of the system in their models and make necessary assumptions and approximations to simplify the analysis. To get an answer, learners substitute realistic values in the calculation sub-goal, followed by evaluation. Since METtLE is open-ended, learners can perform the five tasks in any order and revise them.

METtLE is a complex problem-solving environment and is equipped with a plethora of tools and resources such as a simulator, calculator, info center, scribble pad, causal map builder, equation builder, and a problem map. It also includes hints, expert guidance, and metacognitive prompts in specific sub-goals to support learners.

3 Literature Synthesis

SRL is an extraordinary umbrella that considers several aspects that influence learning (e.g., self-efficacy, volition, cognitive strategies) in a comprehensive and holistic approach [4]. Many frameworks and models of SRL explain the various metacognitive phases, processes and contextual factors that affect learning. For instance, Zimmerman and Schunk [12] proposed the cyclic model of SRL, emphasising three phases, i.e., forethought & planning, monitoring, and self-reflection. Similarly, Pintrich's framework [13] operationalises SRL into four phases (i.e., planning & activation, monitoring, control & regulation, and reaction & reflection) across four areas (i.e., regulation of cognition, motivation/affect, behaviour, and context). Winne & Hadwin's [14] model extends the work of Pintrich and others by outlining five different facets of tasks (i.e., conditions, operations, products, evaluations, and standards) that can take place in the four phases (i.e., task definition, goals and plans, studying tactics, and adaptations). There are various models of SRL that predicate slightly different views on how learners self-regulate. Still, the most empirically supported models in the literature are Pintrich, Winne and Hadwin, and Zimmerman [15].

Several existing assessment methods measure SRL, such as thinking aloud protocols, classroom observations, online trace data analysis, and self-reporting [6]. Panadero and colleagues [6] characterise the historical development of SRL measurements as three waves: a) through self-report lenses, b) use of online measures, and c) new conceptualisation of SRL measurement using intervention and assessment. Several studies have examined how metacognitive processes influence learning [16, 17] using self-report data such as MSLQ questionnaires. However, Winne & Noel [5] have found that most students are poor reporters and calibrators of their learning activities, questioning data from self-reports. Therefore, a more objective way of identifying and measuring student regulation is using online measures such as think-aloud data [18]. In think-aloud protocols, learners are asked to verbalise what they did at each point in the problem-solving

process. Such verbalisations can elucidate metacognitive phases/processes and help us understand the dynamic nature of SRL. Recently, trace data (or log data) has been used to measure SRL in several research studies [1, 3, 5, 15, 19, 20]. Traces capture learner actions on the fly (in real-time) along with the context.

Several learning analytics methods, such as relationship mining, clustering, and discovery with machine learning models, are used to assess and interpret trace data to identify metacognitive phases and processes [7, 22]. For example, Cloude et al. [8] used frequencies and means to interpret and visualise metacognitive processes. Similarly, Li S. et al. [10] performed agglomerative clustering to identify students' overall SRL profiles in engineering design. Likewise, Saint et al. [9] use stochastic process mining to visualise the difference between two groups of students and their use of regulatory processes. Although applying the mining techniques mentioned above provides empirical measures to understand how learners' regulatory processes and phases unfold over time, it is crucial to triangulate/validate log data findings with the ground truth. We did not come across existing research studies that have triangulated their log data findings with the ground truth, i.e., concurrent think-aloud data.

Several research studies measure metacognitive processes with the aforementioned methods, for example, Metatutor [1] and Learn-B [15]. However, the tasks associated with most learning environments involve reading and assimilating text [21]. None of the environments supports tasks that involve solving complex problems, which requires tools such as simulators, causal map builders, etc. Hence, from the existing studies that use trace data to identify metacognitive processes, we recognise the below gaps:

- No current study identifies metacognitive phases and/or processes from trace data using ML models.
- Existing studies do not triangulate the performance of models with ground truth data (e.g., think-aloud data).
- Existing studies that identify metacognitive processes focus only on reading and assimilating tasks.

Hence, there is a need to develop an ML model to identify metacognitive phases and processes in a problem-solving OELE.

4 Research Methodology

This paper's broad research goal is to automatically identify metacognitive phases and processes displayed by learners in a problem-solving environment using computer-generated traces and think-aloud data. In this section, we will be describing the study design and the data analysis procedure. The data analysis procedure delineates the processing performed on think-aloud and trace data, followed by the process to map the two modalities, and finally, the methodology to address the research goal.

4.1 Study Design

Five learners (4 male, one female) who had completed at least one year in Engineering participated in the study voluntarily. The study proposal had been reviewed and approved

by the Institute Ethics Committee of IIT Bombay. The study was conducted in a lab set-up; wherein individual learners interacted with the open-ended learning environment (METtLE) for 60–90 min. In the introduction phase, learners read the informed consent form, post which they were explained that they would be solving an estimation-related problem in electrical engineering. A time duration of 90 min was allotted to each learner to solve the estimation related problem in METtLE. All the participants were asked to do concurrent think-aloud; however, due to some technical glitches, the data of one learner was not considered for analysis. In concurrent think-aloud, learners verbalised their thoughts concurrently while solving an estimation-based problem on METtLE. If the learners were quiet, we instigated them to think aloud; however, the need arose rarely. Learners solved a multiplication activity as a warm-up since thinking aloud is not natural, especially in the presence of others.

4.2 Data Analysis

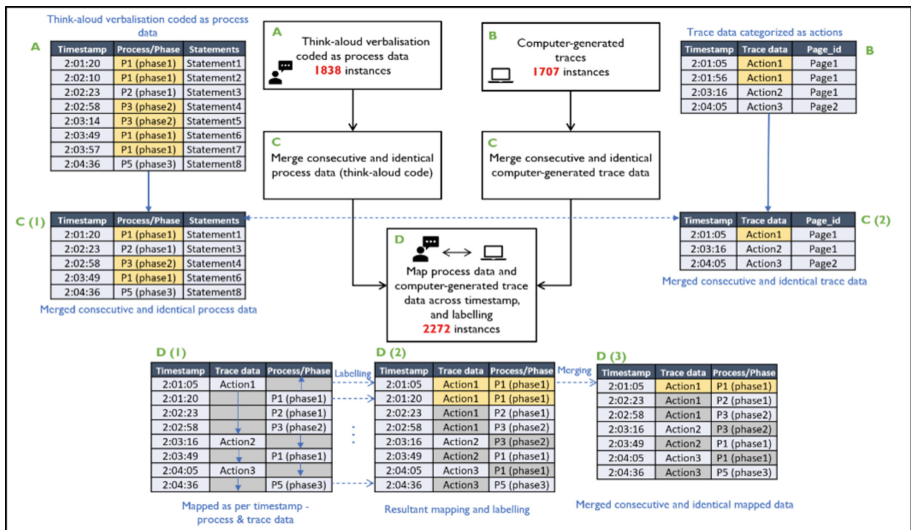


Fig. 1. Process of analysing and mapping process/phase data and computer-generated trace data

The data sources collected in this study include learners' think-aloud verbalisations, the timestamp, and automatically generated computer traces of learners' interaction. The procedure of analysing think-aloud and computer-generated trace data is summarised in Fig. 1. As seen in the figure, the analysis can be categorised into four steps, i.e., a) Analyse think-aloud data by manually coding the verbalisation into metacognitive processes and phases, b) Analyse the trace data by transforming the raw traces into meaningful learner actions, c) Merge consecutive and identical information, by merging the similar consecutive coded processes/phases and trace data, and d) Map the data sources, by mapping the coded processes/phases and trace data as per timestamp, and

labelling the resultant mapping. In the following passages, we will be detailing the steps mentioned above.

A. Coding of Think-Aloud Data

We coded the think-aloud data from four learners manually, using a mechanism to identify metacognitive processes in a problem-solving OELE, as detailed in [22]. The coding mechanism consists of four phases (i.e., planning & activation, monitoring, control & regulation, and reaction & reflection) and 17 metacognitive processes (e.g., target goal setting, feeling of knowing, etc.) based on Pintrich's framework of SRL [13] and the design of a problem-solving OELE (MEtLE) [11]. The coding mechanism is focused on discussing the area of regulation of cognition, i.e., metacognition. In this area, 'planning and activation' (Phase1) involves processes involving planning, setting goals and activating relevant prior knowledge for a given task. Likewise, 'monitoring' (Phase2) encapsulates various monitoring processes representing metacognitive awareness, such as a feeling of knowing. 'Control and regulation' (Phase3) involve efforts and control to select and adapt various cognitive strategies. Finally, 'reaction and reflection' (Phase4) covers different reactions and reflections of learners on the task. We coded 1838 phrases manually using the coding mechanism and the timestamp, and the rest (151) were marked 'not on task' (NT).

Out of the total phrases coded, 79% belonged to Phase3, 9% to Phase2, 8% to Phase1, and only 4% to Phase4. Five processes, i.e., a judgment of learning and comprehension monitoring, selection and adaptation of control strategies, model building techniques, estimation reasoning, and gathering context-specific knowledge, occurred more frequently (>100 instances) than others. The process judgement of learning and comprehension monitoring' (P1) is identified when learners assess how well they have learned certain information. Similarly, the 'selection and adaptation of control strategies' (P2) process is identified when learners select and use various cognitive strategies for learning, reasoning, and problem-solving. On the other hand, the use of tools made available in the OELE, e.g., the use of variable manipulation simulator, can be called 'model building techniques' (P3). In the same way, 'estimation reasoning' (P4) and 'gather context-specific knowledge' (P5) are identified when learners use question prompts/hints/expert guidance to do estimation reasoning and reads/gathers context-specific knowledge. Figure 2 displays the frequency distribution of the processes P1-P5, other processes, and the NT verbalisations.

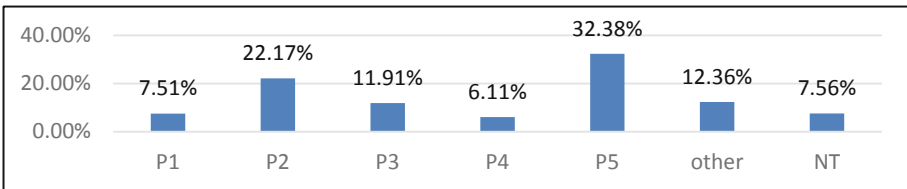


Fig. 2. Distribution of frequency of metacognitive processes

B. Preprocessing of Trace Data

We captured the automatically generated computer traces of learner interaction using a data logging mechanism described in [23] and processed the raw traces into 51 meaningful actions. These actions can be further classified into six broad categories:

- understanding the problem context (e.g., read estimation introduction, read problem statement)
- select and develop a modelling subgoal (e.g., choose a functional model subgoal, and create a hypothesis using fictive motion words from word bag)
- evaluate the model (e.g., use metacognitive prompts and expert guidance to evaluate the functional model created)
- plan and set goals (e.g., use metacognitive prompts to reflect and plan next steps)
- use tools (e.g., vary the parameters on the simulator and observe changes in the graph, and add nodes and links in the causal map builder)
- seek expert advice (e.g., voluntarily click on expert guidance and hints provided in the system)

C. Merging Consecutive and Identical Information

We identified consecutive and identical occurring information in the think-aloud and trace data in this step. For example, in Fig. 1, table C(1), the process/phase data P1(phase1) occurs consecutively at timestamp (2:01:20) and timestamp (2:02:10). Therefore, we merge the two rows by keeping the timestamp value of the first process data. The timestamp value of the next occurring process (i.e., P2 at 2:02:23) ensures that the time spent in performing the two previously merged processes remains intact. Similarly, we merged the consecutive and identical occurring trace data in table C(2) of Fig. 1.

D. Mapping Two Data Sources

We require labelled trace data for classification to automatically identify metacognitive processes and phases. Therefore, we append an extra layer of information to label the trace data, i.e., process/phase data (the coded think-aloud verbalisations). To map the two data sources, i) we first align them across a synchronised timeline, ii) label the resultant mapping, and iii) remove consecutive and identical occurring mapped pairs.

In Fig. 1, table D(1) displays the aligned process and trace data using the synchronised timestamp information. Since the timestamp of trace and process/phase data is captured at different times, it is important to map the two sources correctly. For example, the first and second actions, i.e., Action1 and Action2, occur at timestamp 2:01:05 and 2:03:16, respectively. In between the two activities, three process/phase data, i.e., P1 (phase1), P2 (phase1), and P3 (phase2), occur with timestamps 2:01:20, 2:02:23, and 2:02:58, respectively. The difference signifies that while displaying processes P1-P3, the learner's context was Action1. Hence, the corresponding rows of the process data are labelled as Action1 (Refer to Fig. 2, table D(2)). Similarly, when the trace data Action2 was recorded, the learner displayed process/phase data P3 (phase2); hence the corresponding row across Action2 is labelled as P3 (phase2). Finally, we removed the consecutive and identical occurring pairs of mapped trace and process data, as shown in table D(3) of the figure.

To summarise, we collected four learners’ think-aloud and trace data. We merged the two modalities across a synchronised timeline and used the mapped data for further analysis. The mapping resulted in 2272 instances of process and phase data mapped to trace data.

4.3 Methodology

We extracted features to train machine learning classifiers to automatically identify the five metacognitive processes and four phases (i.e., process/ phase data) from the trace data (i.e., processed learner actions). We computed the features along three dimensions. That is, a) all the learner actions within the last 5 min of interaction, b) time spent in each action, and c) the sequence of actions in which it occurred. For example, Table 1 consists of an example excerpt of learner interaction and the mapped think-aloud processes/phases of a learner. To predict instance_id 6, all the learner interactions 5 min before 0:06:54 were captured (i.e., all interactions between 0:01:54–0:06:54). Thus, to predict instance_id 6, feature1 includes information from the time window of instance_id 2 to 6. Similarly, feature2 includes the time spent on each action in seconds over the last 5 min. And finally, feature3 includes the sequence in which the action was performed.

Table 1. Excerpt of trace data and mapped process/phase data for feature extraction

Instance_id	Time	Trace data	Process/Phase data
1	0:01:44	Action1	P8 (phase4)
2	0:03:39	Action1	P1 (phase1)
3	0:04:02	Action1	P7 (phase3)
4	0:06:20	Action1	P1 (phase1)
5	0:06:29	Action1	P7 (phase3)
6	0:06:54	Action2	P7 (phase3)
7	0:07:30	Action2	P8 (phase4)

Since we predict four phases and five processes, one way to implement such multi-class classification is by using the ‘one-vs-all’ technique. Given a classification problem with N possible solutions, a one-vs-all solution consists of N separate classifiers, i.e., one binary classifier for each possible outcome [24]. Hence, we created four classifiers for phases and five classifiers for processes and tested them using 10-fold cross-validation [25].

Phase3 (79%) occurs more frequently than Phase4 (4%); similarly, processes such as P7 (32.4%) occur more frequently than other processes such as P8 (1.8%). Since all the categories are not approximately equally represented, the dataset is imbalanced. Thus, there are very few examples of the minority class for a model to learn the decision boundary effectively. We address this issue by upsampling the minority class using the synthetic minority oversampling technique (SMOTE) [26]. We partitioned the data into training (90%) and testing (10%) datasets and upsampled both the datasets using the python library ‘imblearn.over_sampling.SMOTE’.

Finally, we created the classifiers using the machine learning models proven effective for small datasets [27] in Orange¹, which is an open-source machine learning and data visualisation tool, i.e., Logistic regression (LR), Naive Bayes (NB), Support Vector Machine (SVM), and Random Forest (RF). The details of the parameter and hyperparameter values for the different models are 1) LR: Lasso regression with a value of regularisation constant = 1, 2) NB: Preprocessing to remove empty columns and discretising numeric values, 3) SVM: RBF kernel with the default gamma constant, and 4) RF: 10 trees with pre-pruning and depth of individual trees to 3.

To analyse the performance of different machine learning models, we used the following performance metric [28]:

- Classification accuracy (CA): is the ratio of the number of correct classifications to the total number of classifications.
- F score (F1): is the harmonic mean of recall and precision.
- Kappa (κ): measures agreement between the actual and the predicted labels by considering the by-chance prediction. κ is calculated using Eq. (1), where P_0 is the overall accuracy of the model and P_e is the measure of the agreement between the model predictions and the actual class values as if happening by chance. κ value between 0–0.2 indicates poor/no agreement, 0.21–0.39 indicates fair agreement, 0.4–0.59 indicates moderate agreement, 0.6–0.79 indicates substantial agreement, and 0.8–1 indicates strong/ almost perfect agreement [29].

$$\kappa = \frac{P_0 - P_e}{1 - P_e} \quad (1)$$

5 Results and Discussion

This section describes the results of the prediction models developed using the synchronised and upsampled data at the phase and process level. We first describe the results of phase-level classification of data synchronised with trace data by discussing 1) the performance of machine learning models on multiclass classification, 2) the performance of one-vs-all classification, and 3) the performance of one-vs-all classification after upsampling. Similarly, we then discuss the results of the process-level classification of data. The best-performed classifier metrics are highlighted in bold for each phase/process.

5.1 Multiclass Classification of Phase-Level Data

Table 2 summarises the model prediction results of the five classes, i.e., four phases (Phase1–4) and NT verbalisations signifying ‘No phase’. We employed 10-fold cross-validation using the four classifiers. Although LR outperformed other models in terms of CA (0.634) and F1 (0.514), the κ value (0.024) indicates poor agreement between the classifier and ground truth.

¹ <https://orangedatamining.com/>

Table 2. The evaluative metric of performance of ML models on multiclass classification

ML Model	CA	F1	κ
LR	0.634	0.514	0.024
NB	0.412	0.441	0.062
SVM	0.575	0.500	0.019
RF	0.541	0.483	-0.008

5.2 One-Vs-All Classification of Phase-Level Data

One way to implement multiclass classification is using the one-vs-all technique. We first create five classifiers for the phases (i.e., Phase1-Phase4, no phase) using the four ML models. Table 3 reports the model prediction results. NB model outperformed other models for the phases (i.e., 1, 2, 4, and no phase) in terms of kappa value. E.g., the CA, F1, and kappa values for Phase4 using the NB model are 0.79, 0.84, and 0.1, respectively. However, the kappa value <0.2 indicates poor agreement between the classifier and the actual value.

Table 3. Performance of ML models on one-vs-all classification

Phase	LR			NB			SVM			RF		
	CA	F1	κ	CA	F1	κ	CA	F1	κ	CA	F1	κ
Phase1	0.89	0.84	-0.01	0.77	0.79	0.01	0.88	0.84	0	0.88	0.84	-0.02
Phase2	0.89	0.84	0.01	0.81	0.82	0.09	0.87	0.83	-0.02	0.86	0.83	0.04
Phase3	0.63	0.58	0.08	0.58	0.58	0.08	0.56	0.56	0.02	0.57	0.56	0.01
Phase4	0.95	0.93	0	0.79	0.84	0.10	0.95	0.93	0	0.94	0.93	0.01
No_phase	0.90	0.86	0.01	0.77	0.80	0.02	0.89	0.85	-0.03	0.88	0.85	-0.02

5.3 One-Vs-All Classification of Phase-Level Data After Upsampling

Since the different phases are not approximately equally represented, we upsample the data using SMOTE algorithm. We categorised the data into training (90%) and testing sets (10%) and upsampled both the datasets using SMOTE algorithm. Table 4 reports the model prediction results of the one-vs-all classification after the upsampling of the train and test dataset. The RF model outperformed other ML models for phase1, phase4, and no_phase. For e.g., phase1 is classified with CA (0.84), F1 (0.84), and κ (0.6). The κ value signifies a substantial agreement between the classifier and the actual data. Similarly, SVM and LR models outperformed other ML models for phase2 and phase3. The κ value for phase2, phase4, and no_phase illustrate moderate agreement, whereas the κ value for

phase 3 indicates minimal/fair agreement between the classifier and actual data. Phase 3 (i.e., Control and regulation) occurs more frequently than others (i.e., 79%) and is associated with multiple learner actions. Therefore, the low κ value (0.20) is because the process is difficult to distinguish using trace data.

There is also an improvement in the κ values before and after upsampling the data. Increasing instances of minority classes using SMOTE technique is helping the model learn the decision boundary effectively, thus enhancing the agreement between the classifiers and actual data.

Table 4. Performance of ML models on one-vs-all classification after upsampling (phase)

Phase	LR			NB			SVM			RF		
	CA	F1	κ	CA	F1	κ	CA	F1	κ	CA	F1	κ
Phase1	0.70	0.70	0.41	0.84	0.84	0.69	0.67	0.67	0.33	0.84	0.84	0.60
Phase2	0.55	0.55	0.10	0.57	0.54	0.15	0.74	0.73	0.47	0.69	0.68	0.23
Phase3	0.60	0.60	0.20	0.58	0.58	0.17	0.44	0.44	-	0.56	0.56	0.13
Phase4	0.59	0.58	0.18	0.74	0.73	0.48	0.67	0.67	0.35	0.81	0.80	0.48
No_phase	0.55	0.54	0.11	0.59	0.52	0.18	0.51	0.49	0.03	0.74	0.73	0.41

5.4 One-Vs-All Classification of Process-Level Data After Upsampling

We performed multiclass and one-vs-all classification of the five processes (P1-P5) using the four ML models (i.e., LR, NB, SVM, and RF). We found CA and F1 less than 0.5, and the κ value signified poor agreement between the classifier and the actual data. Therefore, we categorised the data into training (90%) and testing (10%) datasets, upsampled both sets and created five one-vs-all classifiers using the ML models. Table 5 reports the results of the model prediction. RF outperformed other models on evaluative metrics CA, F1, and κ for processes P1, P3, and P4. For e.g., process P3 (i.e., model building) is predicted with CA (0.79), F1 (0.77), and κ (0.58).

Table 5. Performance of ML models on one-vs-all classification after upsampling (process)

Process	LR			NB			SVM			RF		
	CA	F1	κ	CA	F1	κ	CA	F1	κ	CA	F1	κ
P1	0.46	0.44	-0.09	0.61	0.53	0.21	0.56	0.54	0.11	0.60	0.60	0.21
P2	0.43	0.41	-0.15	0.63	0.62	0.25	0.65	0.65	0.31	0.46	0.45	-0.70
P3	0.70	0.69	-0.39	0.74	0.73	0.47	0.65	0.65	0.29	0.79	0.79	0.58
P4	0.65	0.64	0.29	0.78	0.77	0.56	0.66	0.65	0.31	0.78	0.77	0.56
P5	0.56	0.56	0.12	0.63	0.63	0.27	0.53	0.52	0.05	0.47	0.46	-0.07

The Cohen's kappa value of process P3 and P4 (i.e., 0.58 and 0.56) indicates a moderate agreement between the classifier and actual value. Similarly, the κ value of processes P1(0.21), P2 (0.31), and P5 (0.27) signifies a fair agreement between the classifier and actual values. The kappa value of these classifiers is lower than the others because the processes P1, P2, and P5 occur more frequently than other processes and hence are associated with many learner interactions, making them difficult to differentiate using trace data. Also, the processes that signify P2 and P5 are selecting cognitive strategies and gathering context-specific knowledge, respectively, which are associated with the most frequent phase reported in our study, i.e., Control and Regulation (79%).

6 Conclusion

The article aimed to identify metacognitive processes displayed by learners in a problem-solving OELE. We mapped the coded think-aloud and trace data across a synchronised timeline to develop the prediction model and extracted features such as actions performed, time spent, and sequence. We can predict four phases (e.g., planning and activation) and five metacognitive processes (e.g., judgment of learning and comprehension monitoring) with a classification accuracy of up to 0.84 and a κ value indicating fair to substantial agreement. We found that the RF model outperformed other models in predicting most phases and processes.

Our work holds a significant role in demonstrating the automatic identification of metacognitive processes using trace data in problem-solving OELE. A methodological contribution of our paper is to illustrate the mapping of trace and think-aloud data across a synchronised timeline to examine the metacognitive phases/processes displayed by a learner. Thus, we have utilised the unobtrusive nature of trace data to capture learner information and triangulate with think-aloud data, which is a rich and nuanced way to capture metacognitive phases and processes. Moreover, since our machine learning model can predict metacognitive phase and processes unobtrusively, the model can be used to identify learners' difficulties. Identification of such challenges enables personalisation and adaptive scaffolds to support novice learners during problem-solving in an OELE.

Although we can predict metacognitive phases and processes with a good kappa and classification accuracy, this study has limitations. For instance, the data is collected from a small N (four), making it difficult to generalise this model to a larger N. Therefore, we limit the paper's scope to developing an ML model to identify metacognitive phases/processes in MEttLE only and not generalising it to other systems. Similarly, there are 17 metacognitive processes defined in the coding mechanism used; however, we have identified only five processes because the frequency of other processes is very low. This is again a consequence of the small N reported in our study. Hence, our future goal is to conduct this study with a significant N and use advanced classifiers such as neural networks to improve the classification accuracy and Cohen's kappa. We also plan to identify the combination of actions that could signify clear discrimination of the metacognitive processes/phases. Such discriminants would be beneficial for identifying metacognitive phases and processes automatically using trace data only.

Acknowledgement. The authors would like to thank the participants for their participation in the study. The authors acknowledge the Research Scholars of the Indian Institute of Technology (IIT) Bombay and the Indian Ministry of Education for their support in conducting this research.

References

1. Azevedo, R., Moos, D.C., Johnson, A.M., Chauncey, A.D.: Measuring cognitive and metacognitive regulatory processes during hypermedia learning: Issues and challenges. *Educ. Psychol.* **45**(4), 210–223 (2010)
2. Zimmerman, B.J.: A social cognitive view of self-regulated academic learning. *J. Educ. Psychol.* **81**(3), 329 (1989)
3. Munshi, A., Biswas, G.: Personalization in OELs: developing a data-driven framework to model and scaffold SRL processes. In: Isotani, S., Millán, E., Ogan, A., Hastings, P., McLaren, B., Luckin, R. (eds.) *AIED 2019. LNCS (LNAI)*, vol. 11626, pp. 354–358. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-23207-8_65
4. Panadero, E.: A review of self-regulated learning: Six models and four directions for research. *Front. Psychol.* **8**, 422 (2017)
5. Winne, P.H., Jamieson-Noel, D.: Exploring students' calibration of self reports about study tactics and achievement. *Contemporary Educ. Psychol.* **27**(4), 551–572 (2002)
6. Panadero, E., Klug, J., Järvelä, S.: Third wave of measurement in the self-regulated learning field: When measurement and intervention come hand in hand. *Scand. J. Educ. Res.* **60**(6), 723–735 (2016)
7. Viberg, O., Khalil, M., Baars, M.: Self-regulated learning and learning analytics in online learning environments: a review of empirical research. In: *Proceedings of the Tenth International Conference on Learning Analytics & Knowledge*, pp. 524–533, March 2020
8. Cloude, E.B., Taub, M., Lester, J., Azevedo, R.: The role of achievement goal orientation on metacognitive process use in game-based learning. In: Isotani, S., Millán, E., Ogan, A., Hastings, P., McLaren, B., Luckin, R. (eds.) *AIED 2019. LNCS (LNAI)*, vol. 11626, pp. 36–40. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-23207-8_7
9. Saint, J., Gasevic, D., Pardo, A.: Detecting learning strategies through process mining. In: PammerSchindler, V., PérezSanagustín, M., Drachsler, H., Elferink, R., Scheffel, M. (eds.) *EC-TEL 2018. LNCS*, vol. 11082, pp. 385–398. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-98572-5_29
10. Li, S., Du, H., Xing, W., Zheng, J., Chen, G., Xie, C.: Examining temporal dynamics of self-regulated learning behaviors in STEM learning: a network approach. *Comput. Educ.* **158**, 103987 (2020)
11. Kothiyal, A., Murthy, S.: METTLE: a modelling-based learning environment for undergraduate engineering estimation problem solving. *Res. Pract. Technol. Enhanc. Learn.* **13**(1), 1–28 (2018). <https://doi.org/10.1186/s41039-018-0083-y>
12. Zimmerman, B.J., Schunk, D.H. (eds.): *Self-regulated Learning and Academic Achievement: Theoretical Perspectives*. Routledge, London (2001)
13. Pintrich, P. R.: The role of goal orientation in self-regulated learning. In: *Handbook of Self-regulation*, pp. 451–502. Academic Press (2000)
14. Winne, P. H., Hadwin, A.E.: *Studying as Self-Regulated Learning*, pp. 291–318. Routledge, London (1998)
15. Siadaty, M., Gasevic, D., Hatala, M.: Trace-based micro-analytic measurement of self-regulated learning processes. *J. Learn. Anal.* **3**(1), 183–214 (2016)

16. Bergin, S., Reilly, R., Traynor, D.: Examining the role of self-regulated learning on introductory programming performance. In: Proceedings of the First International Workshop on Computing Education Research, pp. 81–86, October 2005
17. Zheng, L.: The effectiveness of self-regulated learning scaffolds on academic performance in computer-based learning environments: a meta-analysis. *Asia Pac. Educ. Rev.* **17**(2), 187–202 (2016). <https://doi.org/10.1007/s12564-016-9426-9>
18. Greene, J.A., Azevedo, R.: A macro-level analysis of SRL processes and their relations to the acquisition of a sophisticated mental model of a complex system. *Contemp. Educ. Psychol.* **34**(1), 18–29 (2009)
19. Siadaty, M., et al.: Learn-B: a social analytics-enabled tool for self-regulated workplace learning. In: Proceedings of the 2Nd International Conference on Learning Analytics and Knowledge, pp. 115–119, April 2012
20. Taub, M., Mudrick, N.V., Azevedo, R., Millar, G.C., Rowe, J., Lester, J.: Using multi-channel data with multi-level modeling to assess in-game performance during gameplay with Crystal Island. *Comput. Hum. Behav.* **76**, 641–655 (2017)
21. Pathan, R., Rajendran, R., Murthy, S.: A literature review of modelling SRL using trace data, 19 January 2022. <https://doi.org/10.35542/osf.io/j326y>
22. Pathan, R., Murthy, S., Rajendran, R.: A coding mechanism for analysis of SRL processes in an open-ended learning environment. In: 29th International Conference on Computers in Education Conference, ICCE 2021. Asia-Pacific Society for Computers in Education (2021)
23. Pathan, R., Shaikh, U., Rajendran, R.: Capturing learner interaction in computer-based learning environment: design and application. In: 2019 IEEE Tenth International Conference on Technology for Education (T4E) (pp. 146–153). IEEE, December 2019
24. Rifkin, R., Klautau, A.: In defense of one-vs-all classification. *J. Mach. Learn. Res.* **5**, 101–141 (2004)
25. Refaeilzadeh, P., Tang, L., Liu, H.: Cross-validation. *Encyclopedia Database Syst.* **5**, 532–538 (2009)
26. Chawla, N.V., Bowyer, K.W., Hall, L.O., Kegelmeyer, W.P.: SMOTE: synthetic minority over-sampling technique. *J. Artif. Intell. Res.* **16**, 321–357 (2002)
27. Sharma, A., Paliwal, K.K.: Linear discriminant analysis for the small sample size problem: an overview. *Int. J. Mach. Learn. Cybernec* **6**, 443–454 (2015). <https://doi.org/10.1007/s13042-013-0226-9>
28. Powers, D. M.: Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation. arXiv preprint [arXiv:2010.16061](https://arxiv.org/abs/2010.16061) (2020)
29. McHugh, M.L.: Interrater reliability: the kappa statistic. *Biochemia Medica* **22**(3), 276–282 (2012)



Intervention Prediction in MOOCs Based on Learners' Comments: A Temporal Multi-input Approach Using Deep Learning and Transformer Models

Laila Alrajhi^(✉), Ahmed Alamri, and Alexandra I. Cristea

Computer Science, Durham University, Durham, UK

{laila.m.alrajhi, ahmed.s.alamri,
alexandra.i.cristea}@durham.ac.uk

Abstract. High learner dropout rates in MOOC-based education contexts have encouraged researchers to explore and propose different intervention models. In discussion forums, intervention is critical, not only to identify *comments* that require replies but also to consider *learners* who may require intervention in the form of staff support. There is a lack of research on the role of intervention based on learner comments to prevent learner dropout in MOOC-based settings. To fill this research gap, we propose an intervention model that detects when staff intervention is required to prevent learner dropout using a dataset from FutureLearn. Our proposed model was based on learners' comments history by integrating the most-recent sequence of comments written by learners to identify if an intervention was necessary to prevent dropout. We aimed to find both the proper classifier and the number of comments representing the appropriate most recent sequence of comments. We developed several intervention models by utilising two forms of supervised multi-input machine learning (ML) classification models (deep learning and transformer). For the transformer model, specifically, we propose the siamese and dual temporal multi-input, which we term the multi-siamese BERT and multiple BERT. We further experimented with clustering learners based on their respective number of comments to analyse if grouping as a pre-processing step improved the results. The results show that, whilst multi-input for deep learning can be useful, a better overall effect is achieved by using the transformer model, which has better performance in detecting learners who require intervention. Contrary to our expectations, however, clustering before prediction can have negative consequences on prediction outcomes, especially in the underrepresented class.

Keywords: MOOCs · Intelligent intervention system · Multi-input model

1 Introduction

During the COVID pandemic and lockdown, most educational institutions around the world turned to online study [1]. Platforms such as MOOCs became increasingly attractive for a large number of institutions and learners to allow them to continue their

studies [2]. Nevertheless, dropout rates on MOOC-based courses can reach 90% [3], which remains, even during the pandemic [4], one of the most long-standing problems of such learning environments [5]. In recent years, many researchers have proposed several solutions to curb dropout rates [6, 7], among them, constructing intervention models to determine learner needs based on identifying urgent comments posted to discussion forums [8, 9]. Interaction with an instructor is considered one of the most important indicators for avoiding dropout in MOOC learners [10]. However, in terms of identifying if intervention is required based on the comments posted in discussion forums as asynchronous communication platforms between learners and instructors [11], due to the huge numbers of comments, instructors cannot effectively monitor, track, identify, and respond to all comments that may require intervention. Therefore, many researchers have attempted to create models to identify comments posted by learners who might need intervention [12, 13]. However, we consider that it might also be helpful to consider the sequence of learners' textual comments (Sect. 3.2) to reduce dropout rates and improve the quality of the interventions offered.

This study aimed to develop a model to identify learners who require intervention by an instructor based on the sequence of learner comments to predict and mitigate learner dropout on MOOC-based courses. As the absence of interaction and feedback by instructors on discussion forums has been associated with increased dropout rates [10, 14], our objective was to propose an intelligent intervention model. We formalised this challenge as a text classification problem by developing and employing a supervised binary classification model with multiple text inputs based on learner comments. The input consists of the most recent comments of the learner (as further defined in Sect. 3.2) and the output is the predicted dropout. We applied and trained two recent popular types of classifiers: deep learning [15] and transformer [16], and examined various numbers of inputs for prediction. Therefore, we investigated the following research question (RQ):

RQ. Can we predict learners who may drop out and identify their need for intervention from their most recent comments?

To the best of our knowledge, this is the first study to attempt to identify MOOC learners who may need instructor intervention by using their historical online forum comments as data. The other contribution of this work is that we use a multi-input approach for siamese and dual BERT with binary text classification, with the resulting integrated networks being termed multi-siamese BERT and multiple BERT, respectively.

2 Related Work

The issue of intervention to help learners in MOOC environments is an interesting area for many research communities [17, 18, 19] and an important research direction. In prior literature, instructor intervention has been studied from two perspectives: (i) comments on discussion forums, and (ii) learners.

The use of comments on discussion forums for intervention prediction has received a wide research focus; researchers have attempted to establish different intervention models as a text classification task [8, 12, 20, 13], or used comment features as an input of the classifier [21, 22].

From a learner perspective, prevalent studies have addressed intervention and dropout rates using input characteristics or clickstream data, such as predicting dropouts per week, based on the weekly history of the learner [23]. Also, [19] created a similar weekly prediction mechanism by applying a deep learning approach.

In contrast, there is limited research on intervention based on the comments of learners who are likely to drop out [24]. This is due to the low percentage of learners who enroll on a MOOC course *and* write comments (only around 5–10% [25]). For example, [26] showed that out of 55,013 and 10,190 learners who had registered and enrolled on courses, only 750 and 519 engaged with discussion forums by posting comments, respectively. Among the few pieces of research on this topic, [27] used NLP tools to predict learners who completed a MOOC course with an accuracy of 67.8%. Other researchers combined clickstream data with discussion forum data. For example, [28] predicted learner completion by employing clickstream data and language in a discussion forum with a 78% accuracy rate.

Furthermore, using sentiment analysis gathered from learners' comments, [29] predicted attrition based on different features including sentiment analysis using a neural network and achieved 72.1% accuracy. Another researcher [30] using the same method predicted dropout rates based on sentiment analysis and clickstream data. Also, [31] found a significant correlation between sentiment and attrition.

As previously stated, this study aimed to develop an intelligent intervention system to reduce learner dropout in MOOC courses. The proposed model is a novel approach that predicts learner dropout (need for intervention) based on learner comments history as a multi-input text classification task to improve instructor intervention and reduce dropout rates.

3 Methodology

3.1 Dataset

The dataset for this research consisted of a real MOOC forum on a Big Data course, Run 2, offered in 2013 and hosted by Warwick University on the FutureLearn platform [32]. This dataset was selected because it contains a high percentage of learners who dropped out due to the difficulty of the topic and some comments did not attract instructor intervention. We collected a total of 5786 comments posted to the discussion forum during the first 5 weeks of the 9-week course, amounting to approximately 50% of the course. Comments from the first half of the course were collected because it is better to intervene at an early stage, before dropout [5]. Exploring the data, it included about 871 active learners, who were defined as those who participate in discussion forums and write at least one text comment (commenters) [33] from 11281 enrolled learners and 4683 accessed learners. Enrolled learners are those who registered while accessed learners are those who both enrolled *and* accessed the course at least once during the first 5 weeks [34]. The number of comments written by active learners varied from 1–209. To create a corpus for all commenters, we collected the history of their comments, as their most recent comments during the first 5 weeks. Then, we defined learners needing intervention as those who dropped out after week 5. To define dropout, we followed the approach of [34] on their weekly prediction of dropout: they supposed that learners

are considered to have dropped out if, in the following week, they did not access 80% of the available topics. Therefore, for each learner, dropout was defined as accessing less than 80% of the available topics in week 6, therefore, the dropout rate was 65.9% (574 learners needed instructor intervention) while 34% of learners (297) completed the course.

3.2 Intervention Model

To identify the learners' need for instructor intervention, we propose the general architecture of our prediction model. We implement this model based on the Python library. The input of this model is the most recent sequence of learner comments while the output is the prediction of if a learner needs instructor intervention (dropout) or not.

For the number of inputs, i.e. the most recent sequence of learner comments, we assume that the learner writes multiple comments and that the number of such comments is an unknown value and may differ from one learner to another. Therefore, we need to investigate the optimal (i.e. minimal) number of historical comments that can help to predict dropout. As an initial experiment, we examined an incremental number of comments ranging from 3–7; but, as mentioned, the total number of comments ranged from 1–209.

Then, we clustered commenters into three groups (as we identified that the optimal number of clusters was 3 using the silhouette method), based on the number of comments written, using the Fisher Jenk algorithm [35], as shown below in Table 1. Next, we focused on group 1, as it contains the highest number of learners (797 commenters) and is thus the most representative of the average number of comments a learner writes. Of these learners, 557 (69.8%) dropped out and 240 (30%) completed the course. After that, we repeated the same experiments for the best intervention models using group 1 (797 commenters) with the mean input rounded up from 3.66 to 4 and excluded the other two groups. Please note that group 1 also had the smallest standard deviation (Std).

Table 1. Statistics of each cluster group.

Group	Count	Mean	Std	Minimum	Maximum
1	797	3.66	3.43	1	16
2	65	28.89	12.24	17	62
3	9	108	43.40	71	209

We developed our prediction models based on two main types of algorithms: deep learning and transformer. The reason for using these models is because they represent the cutting-edge in NLP and eliminate the need for specific feature engineering because they can extract features. We will illustrate the two types in the following sub-sections.

Deep Learning. We applied two cutting-edge deep learning algorithms: convolutional neural networks (CNN) [36] and recurrent neural networks (RNN) [37]. For RNN, we

used long short-term memory (LSTM) [38], bidirectional LSTM [39], gated recurrent units (GRU) [40] and bidirectional GRU [41]. We split the data randomly into training data and testing data (80% and 20%, respectively, equivalent to 696 and 175 learners, respectively). Then, we split the training data into training data and validation data (80% and 20%, respectively, equivalent to 556 and 140 samples, respectively). Each input was treated as a sub-model before these sub-models were concatenated to build the main model. Lastly, we trained the model using the Adam optimizer (batch size = 64; epochs = 20). The prediction for the output of the final/output layer followed [8] where if the probability value is larger than 0.5, it is deemed positive (1). The outcomes of (0) indicate a potential dropout and urgent intervention is required, and an outcome of (1) represents a completer and no intervention is required.

The general architectures are the same for all models. As a preprocessing step to prepare the data for the input, we built a dictionary for each input that contains unique vocabulary words. To specify the length of the word sequences, we followed [8], constraining the length of each input to 200 words; we also explored and found that most comments were ≤ 200 words, which means just 1.3% of comments were affected by truncation. The shortest sequence was padded by 0 and comments > 200 words were trimmed. The next layer after the input layer is the embedding layer. This layer obtains dense vector representations for words, which we use and fine-tune during training, starting with a pre-trained word embedding 'word2vec' [42] (Word2vec GoogleNews-vectors-negative300). We used 'word2vec' because [8] found that word2vec outperforms GloVe on urgent post (comment) classification tasks. Then the following layers are different, according to the different networks (CNN and RNN).

CNN. The general architecture is shown in Fig. 1. In the convolutional layers, for each input, we applied three Conv1D with 128 units and different kernel sizes (3, 4 and 5) following [8]. These layers go through a rectified linear unit (ReLU) activation, followed by a max-pooling layer to further compress features. Then we concatenate the output from each input. Next, we concatenate all the outputs for all the inputs. This is passed to the dense layer with 64 neurons and ReLU activation. Then, a dropout layer is employed to avoid overfitting [43] as a regularisation technique. Finally, the output layer has 1 unit with a sigmoid activation function because it performs a binary classification task.

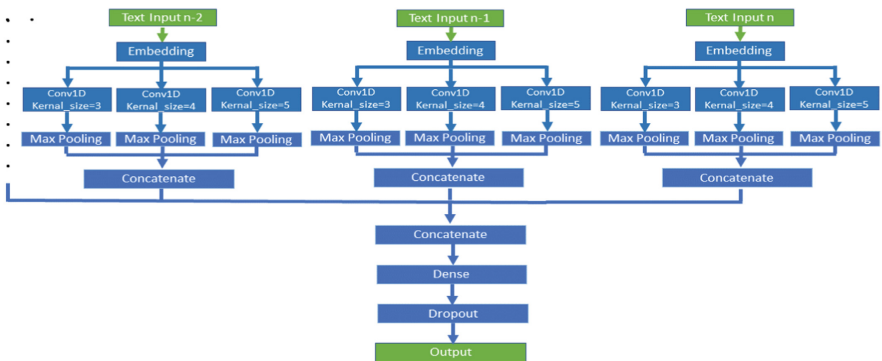


Fig. 1. The general architecture of CNN with multi-input.

RNN. These different networks share the same architecture (see Fig. 2), but the difference between LSTM and GRU is that GRU has fewer parameters. For the bidirectional LSTM and GRU, we train two layers by adding another hidden layer to reverse to the first layer. Thus, as the next layer after the embedding layer, we have an RNN layer with LSTM or GRU or their bidirectional LSTM and GRU with 128 units. Afterwards, we concatenate the output for each input. Then we add a dense layer and dropout layer as for the CNN. Finally, we move to the output layer with the sigmoid as an activation function to obtain the classification.

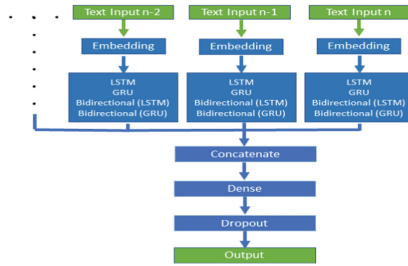


Fig. 2. The general architecture of RNN with multi-input.

Transformer. We developed two different models and built upon the siamese and dual transformers BERT networks to enable the insertion of more than one input into the BERT model [44]. We were inspired to use these two techniques by Marco Cerliani’s code on GitHub, which we consequently modified and to which we added more than two inputs (3–7 inputs); additionally, we converted these two multiclass classification models into two binary classification models: multi-siamese BERT and multiple BERT.

The structure of these models is presented in Fig. 3. We convert each text input to transformer inputs and set the maximum length = 202; to compare with the deep learning model, we add two more for special tokens ([CLS] and [SEP]). Then, we utilise BERT base (number of transformer layers = 12, total parameters = 110M), as the training time is less than for the BERT large. We used the same training and testing data as in the deep learning models. Then we train our models using the Adam optimizer, batch size = 6 and epochs = 3. The same for the deep learning model: we calculate the prediction; where if the value is larger than 0.5, it is supposed positive (1). The (0) denotes a potential dropout and needs urgent intervention and (1) denotes a completer and no intervention is required.

Multi-siamese BERT. In this model, the different text input passes to the same transformer. Then, the output is compressed with a global average pooling. After that, we concatenate them and pass them to the dense, dropout, and output layers.

Multiple BERT. In this model, each input passes to different transformers and reduced with average pooling; then we concatenate all the outputs of the global average pooling; after that, as in the multi-siamese BERT, the output is passed to the dense, dropout, and output layers.

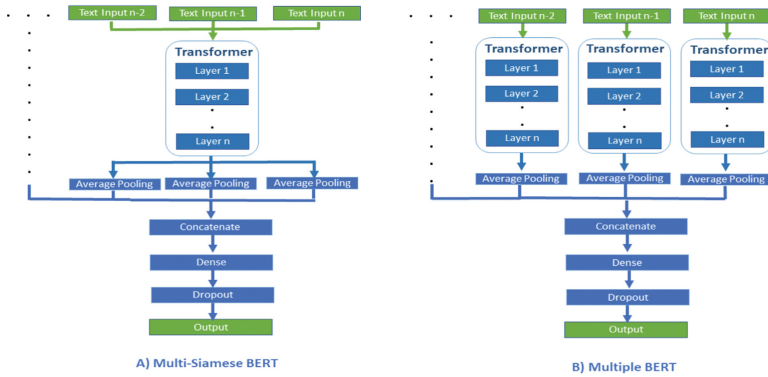


Fig. 3. The general architecture of a) multi-siamese BERT and b) multiple BERT.

4 Results

The experimental results of our multi-input model predictions to address the RQ are presented in Table 2. In addition to accuracy (Acc.), precision (P), recall (R) and F1-score (F1) metrics are also used to comprehensively assess the performance (in percentages) of the different models. In general, the results reveal that LSTM and bi-LSTM achieve high general accuracy but perform badly on the minority class (1), even for the best number of inputs (in italics). GRU also performs badly on the minority class. The remaining models, CNN, Bi-GRU, multi-siamese BERT, and multiple BERT are all more balanced in their optimum models (with the optimal number of inputs). For these best models, multi-siamese BERT and multiple BERT perform comparably and outperform CNN and bi-GRU.

Table 2. The performance of the different multi-input models with different inputs (all learners).

Type		Input	Acc.	0			1		
				P	R	F1	P	R	F1
Deep Learning	CNN	3	0.66	0.76	0.76	0.76	0.44	0.43	0.44
		4	<i>0.69</i>	<i>0.77</i>	<i>0.80</i>	<i>0.78</i>	<i>0.49</i>	<i>0.43</i>	<i>0.46</i>
		5	0.67	0.70	0.93	0.80	0.31	0.08	0.12
		6	0.66	0.73	0.81	0.77	0.42	0.32	0.37
		7	0.64	0.74	0.75	0.74	0.40	0.38	0.39
	LSTM	3	0.70	0.70	0.99	0.82	0.50	0.02	0.04
		4	<i>0.70</i>	<i>0.70</i>	1.00	0.82	1.00	0.02	<i>0.04</i>
		5	0.70	0.70	1.00	0.82	1.00	0.02	0.04
		6	0.70	0.70	1.00	0.82	1.00	0.02	0.04
		7	0.70	0.70	1.00	0.82	1.00	0.02	0.04

(continued)

Table 2. (continued)

Type		Input	Acc.	0			1			
				P	R	F1	P	R	F1	
Transformer	Bi-LSTM	3	0.65	0.74	0.78	0.76	0.41	0.36	0.38	
		4	0.73	0.74	0.93	0.82	0.61	0.26	0.37	
		5	0.67	0.74	0.80	0.77	0.44	0.36	0.40	
		6	0.71	0.75	0.88	0.81	0.53	0.32	0.40	
		7	0.65	0.72	0.81	0.76	0.39	0.28	0.33	
	GRU	3	0.70	0.70	1.00	0.82	0.00	0.00	0.00	
		4	0.70	0.70	1.00	0.82	0.00	0.00	0.00	
		5	0.70	0.70	1.00	0.82	1.00	0.02	0.04	
		6	0.70	0.70	0.99	0.82	0.50	0.02	0.04	
		7	0.70	0.70	1.00	0.82	1.00	0.02	0.04	
	Bi-GRU	3	0.67	0.74	0.80	0.77	0.44	0.36	0.40	
		4	0.63	0.73	0.75	0.74	0.39	0.38	0.38	
		5	0.67	0.76	0.77	0.76	0.45	0.43	0.44	
		6	0.63	0.76	0.69	0.72	0.42	0.51	0.46	
		7	0.69	0.77	0.79	0.78	0.49	0.47	0.48	
	Transformer	Multi-siamese BERT	3	0.71	0.81	0.76	0.78	0.52	0.58	0.55
			4	0.63	0.85	0.58	0.69	0.44	0.75	0.56
			5	0.69	0.85	0.68	0.75	0.49	0.72	0.58
			6	0.65	0.85	0.60	0.70	0.45	0.75	0.56
7			0.65	0.83	0.61	0.71	0.45	0.72	0.55	
Multiple BERT		3	0.67	0.79	0.73	0.76	0.47	0.55	0.50	
		4	0.67	0.83	0.66	0.74	0.47	0.68	0.55	
		5	0.65	0.88	0.58	0.70	0.46	0.81	0.59	
		6	0.71	0.81	0.75	0.78	0.52	0.60	0.56	
		7	0.67	0.77	0.75	0.76	0.46	0.49	0.48	

Thus, we further analysed how the best performing algorithms (transformers) performed for the groups identified, and if our grouping can increase the performance in the given group of focus (group 1).

We can see from Table 3 that the performance of group 1 in multiple BERT outperforms all commenters in multiple BERT in some metrics: (.69%) accuracy, (.92%) recall and (.80%) F1-score for dropout and need intervention (0) but performs badly on the minority class (1) compared to all commenters. Therefore, it provided negative values on prediction outcomes, contrary to our expectations, especially in class (1).

Table 3. The comparison between the performance of different multi-input transformer models with 4 inputs (all learners and group 1).

Type	Group	Acc.	0			1		
			P	R	F1	P	R	F1
Multi-siamese BERT	All	0.63	0.85	0.58	0.69	0.44	0.75	0.56
	Group 1	0.68	0.70	0.91	0.79	0.59	0.24	0.34
Multiple BERT	All	0.67	0.83	0.66	0.74	0.47	0.68	0.55
	Group 1	0.69	0.70	0.92	0.80	0.64	0.25	0.36

5 Conclusion

Although MOOCs offer many learning benefits, they suffer from unacceptable dropout rates. This paper attempted to predict dropout from learners' most recent comments by building ML models including deep learning and transformer with multi-input to enable instructors to intervene more effectively. We developed transformer models based on siamese and dual BERT to insert more than one input for the transformer models. The multi-input consists of the most recent learner comments. We additionally examined the number of inputs needed to predict when intervention was required.

The results indicate that the intervention model can predict dropout and the need for intervention with more accuracy and better detects at-risk learners with the transformer models. However, contrary to our expectations, grouping learners before prediction might harm prediction outcomes, particularly in the minority class. In the future, we plan to replicate this research with other courses and different numbers of comments to further explore the generalisability of these findings. Moreover, we will add clickstream data as additional features.

References




1. Soni, V.D.: Global impact of e-learning during COVID 19 (2020). SSRN 3630073
2. Aljarrah, A.A., Ababneh, M.A.-K., Cavus, N.: The role of massive open online courses during the COVID-19 era: challenges and perspective. *New Trends Issues Proc. Human. Soc. Sci.* **7**(3), 142–152 (2020)
3. Rivard, R.: Measuring the MOOC dropout rate. *Inside High. Ed.* **8**, 2013 (2013)
4. Dang, A., Khanra, S., Kagzi, M.: Barriers towards the continued usage of massive open online courses: a case study in India. *Int. J. Manag. Educ.* **20**(1), 100562 (2022)
5. Cristea, A.I., et al.: Earliest predictor of dropout in MOOCs: a longitudinal study of FutureLearn courses. In: Association for Information Systems (2018)
6. Dalipi, F., Imran, A.S., Kastrati, Z.: MOOC dropout prediction using machine learning techniques: review and research challenges. In: 2018 IEEE Global Engineering Education Conference (EDUCON). IEEE (2018)
7. Goopio, J., Cheung, C.: The MOOC dropout phenomenon and retention strategies. *J. Teach. Travel Tour.* **21**(2), 177–197 (2021)

8. Guo, S.X., et al.: Attention-based character-word hybrid neural networks with semantic and structural information for identifying of urgent posts in MOOC discussion forums. *IEEE Access* **7**, 120522–120532 (2019)
9. Alrajhi, L., Alharbi, K., Cristea, A.I.: A multidimensional deep learner model of urgent instructor intervention need in MOOC forum posts. In: Kumar, V., Troussas, C. (eds.) *ITS 2020*. LNCS, vol. 12149, pp. 226–236. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-49663-0_27
10. Hone, K.S., El Said, G.R.: Exploring the factors affecting MOOC retention: a survey study. *Comput. Educ.* **98**, 157–168 (2016)
11. Ramesh, A., et al.: Understanding MOOC discussion forums using seeded LDA. In: *Proceedings of the Ninth Workshop on Innovative Use of NLP for Building Educational Applications (2014)*
12. Sun, X., et al.: Identification of urgent posts in MOOC discussion forums using an improved RCNN. In: *2019 IEEE World Conference on Engineering Education (EDUNINE)*. IEEE (2019)
13. Khodeir, N.A.: Bi-GRU urgent classification for MOOC discussion forums based on BERT. *IEEE Access* **9**, 58243–58255 (2021)
14. Wei, X., et al.: A convolution-LSTM-based deep neural network for cross-domain MOOC forum post classification. *Information* **8**(3), 92 (2017)
15. Young, T., et al.: Recent trends in deep learning based natural language processing. *IEEE Comput. Intell. Mag.* **13**(3), 55–75 (2018)
16. Vaswani, A., et al.: Attention is all you need. In: *Advances in Neural Information Processing Systems (2017)*
17. Whitehill, J., et al.: Beyond prediction: first steps toward automatic intervention in MOOC student stopout (2015). SSRN 2611750
18. Cobos, R., Ruiz-Garcia, J.C.: Improving learner engagement in MOOCs using a learning intervention system: a research study in engineering education. *Comput. Appl. Eng. Educ.* **29**(4), 733–749 (2021)
19. Xing, W., Du, D.: Dropout prediction in MOOCs: using deep learning for personalized intervention. *J. Educ. Comput. Res.* 0735633118757015 (2018)
20. Almatrafi, O., Johri, A., Rangwala, H.: Needle in a haystack: identifying learner posts that require urgent response in MOOC discussion forums. *Comput. Educ.* **118**, 1–9 (2018)
21. Chaturvedi, S., Goldwasser, D., Daumé III, H.: Predicting instructor’s intervention in MOOC forums. In: *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics: Long Papers*, vol. 1 (2014)
22. Chandrasekaran, M.K., et al.: Learning instructor intervention from MOOC forums: early results and issues. arXiv preprint [arXiv:1504.07206](https://arxiv.org/abs/1504.07206) (2015)
23. Kloft, M., et al.: Predicting MOOC dropout over weeks using machine learning methods. In: *Proceedings of the EMNLP 2014 Workshop on Analysis of Large Scale Social Interaction in MOOCs (2014)*
24. Prenkaj, B., et al.: A survey of machine learning approaches for student dropout prediction in online courses. *ACM Comput. Surv. (CSUR)* **53**(3), 1–34 (2020)
25. Rose, C., Siemens, G.: Shared task on prediction of dropout over time in massively open online courses. In: *Proceedings of the EMNLP 2014 Workshop on Analysis of Large Scale Social Interaction in MOOCs (2014)*
26. Gitinabard, N., et al.: Your actions or your associates? Predicting certification and dropout in MOOCs with behavioral and social features. arXiv preprint [arXiv:1809.00052](https://arxiv.org/abs/1809.00052) (2018)
27. Crossley, S., et al.: Language to completion: success in an educational data mining massive open online class. *Int. Educ. Data Min. Soc.* (2015)

28. Crossley, S., et al.: Combining click-stream data with NLP tools to better understand MOOC completion. In: Proceedings of the Sixth International Conference on Learning Analytics and Knowledge. ACM (2016)
29. Chaplot, D.S., Rhim, E., Kim, J.: Predicting student attrition in MOOCs using sentiment analysis and neural networks. In: AIED Workshops (2015)
30. Mrhar, K., Douimi, O., Abik, M.: A dropout predictor system in MOOCs based on neural networks. *J. Autom. Mob. Robot. Intell. Syst.* 72–80 (2021)
31. Wen, M., Yang, D., Rose, C.: Sentiment analysis in MOOC discussion forums: what does it tell us? In: Educational Data Mining 2014. Citeseer (2014)
32. FutureLearn. <https://www.futurelearn.com>
33. Alrajhi, L., Alamri, A., Pereira, F.D., Cristea, A.I.: Urgency analysis of learners' comments: an automated intervention priority model for MOOC. In: Cristea, A.I., Troussas, C. (eds.) ITS 2021. LNCS, vol. 12677, pp. 148–160. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-80421-3_18
34. Alamri, A., Sun, Z., Cristea, A.I., Stewart, C., Pereira, F.D.: MOOC next week dropout prediction: weekly assessing time and learning patterns. In: Cristea, A.I., Troussas, C. (eds.) ITS 2021. LNCS, vol. 12677, pp. 119–130. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-80421-3_15
35. North, M.A.: A method for implementing a statistically significant number of data classes in the Jenks algorithm. In: 2009 Sixth International Conference on Fuzzy Systems and Knowledge Discovery. IEEE (2009)
36. Kim, Y.: Convolutional neural networks for sentence classification. arXiv preprint [arXiv:1408.5882](https://arxiv.org/abs/1408.5882) (2014)
37. Elman, J.L.: Finding structure in time. *Cogn. Sci.* **14**(2), 179–211 (1990)
38. Gers, F.A., Schmidhuber, J., Cummins, F.: Learning to forget: continual prediction with LSTM. *Neural Comput.* **12**(10), 2451–2471 (2000)
39. Graves, A., Schmidhuber, J.: Framewise phoneme classification with bidirectional LSTM and other neural network architectures. *Neural Netw.* **18**(5–6), 602–610 (2005)
40. Cho, K., et al.: Learning phrase representations using RNN encoder-decoder for statistical machine translation. arXiv preprint [arXiv:1406.1078](https://arxiv.org/abs/1406.1078) (2014)
41. Chung, J., et al.: Empirical evaluation of gated recurrent neural networks on sequence modeling. arXiv preprint [arXiv:1412.3555](https://arxiv.org/abs/1412.3555) (2014)
42. Mikolov, T., Le, Q.V., Sutskever, I.: Exploiting similarities among languages for machine translation. arXiv preprint [arXiv:1309.4168](https://arxiv.org/abs/1309.4168) (2013)
43. Otter, D.W., Medina, J.R., Kalita, J.K.: A survey of the usages of deep learning for natural language processing. *IEEE Trans. Neural Netw. Learn. Syst.* **32**(2), 604–624 (2020)
44. Devlin, J., et al.: Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint [arXiv:1810.04805](https://arxiv.org/abs/1810.04805) (2018)



Not Another Hardcoded Solution to the Student Dropout Prediction Problem: A Novel Approach Using Genetic Algorithms for Feature Selection

Yixin Cheng¹(✉) , Bernardo Pereira Nunes¹ , and Rubén Manrique² 

¹ The Australian National University, Canberra, Australia

Yixin.Cheng@anu.edu.au, Bernardo.Nunes@anu.edu.au

² Universidad de Los Andes, Bogotá, Colombia

rf.manrique@uniandes.edu.co

Abstract. Preventing student dropout is a challenge for higher education institutions (HEIs) that have worsened with COVID-19 and online classes. Despite several research attempts to understand and reduce dropout rates in HEIs, the solutions found in the literature are often hardcoded, making reuse difficult and therefore slowing progress in the area. In an effort to advance the area, this paper introduces a novel portable approach based on genetic algorithms to automatically select the optimal subset of features for dropout prediction in HEIs. Our approach is validated on a dataset containing approx. 248k student records from a Brazilian university. The results show that the proposed approach significantly increases the accuracy in dropout prediction, outperforming previous work in the literature. Our contributions in this paper are four-fold: the implementation of a (i) novel efficient and accurate automatic feature selector that does not require expert knowledge; (ii) an adaptive deep learning model for dropout prediction in sequential data sets; (iii) a portable solution that can be applied to other data sets/degrees; and, (iv) an analysis and discussion of the performance of feature selection and predictive models for dropout prediction.

Keywords: Student dropout prediction · Automatic feature selection · Genetic algorithm · Long short-term memory

1 Introduction

The high dropout rate is an issue that hits the world of higher education hard. It is a latent problem that significantly impacts students, Higher Education Institutions (HEIs) and governments. According to [25], one-fifth of first-year undergraduates drop out of their degrees across Australia each year. In Brazil, the situation is no better and it is estimated that only 62.4% of university enrolments succeed in obtaining an undergraduate degree [24]. More recently, due to

the pandemic, we have witnessed a shift from in-person to online classes and while this change has shown several benefits, retention rates in online teaching are lower than in face-to-face teaching [7, 10].

Given the importance of predicting student dropout, several studies have been proposed in an attempt to increase the accuracy of existing methods. Some studies focused on the selection of predictive features. For example, Manrique et al. [18] extracted time series feature sets from a traditional student dataset to predict student dropout; Mujica et al. [6] used non-standard features such as affective and cognitive; Liu et al. [17] proposed a weighted feature extraction approach based on early student interactions and behaviour; Nagrecha et al. [20] and Ai et al. [1] used student clickstream data; Yang et al. [32] explored social links within discussion forums; and, Hasbun et al. [9] used extracurricular activities. Other studies, on the other hand, focused on different predictive models as, for instance, decision trees [20, 27, 29], random forest (RF) [3, 16, 20, 29], time series forest (TSF) [8], convolutional neural networks (CNNs) [3, 34], long short-term memory (LSTM) [4], or even combined models such as in [29, 33] with CNNs and recurrent neural networks (RNNs) or in [31] with CNNs, LSTM and support vector machine (SVM).

Despite the contributions of previous works, several challenges still remain such as the high computational cost and complexity of automatically generating optimal subsets of features or, if not automatic, the bias on the manual selection of features; the difficulty in eliminating independent and redundant features; the lack of portability of the features to other contexts; the generation of high quality datasets to generate accurate models [14, 20, 27]; and find the best feature representation for dropout prediction [31].

This paper addresses the generation of subsets of features and the dropout prediction problem by combining two strategies. The first strategy focuses on the automatic selection of relevant features using an evolutionary approach and the second strategy uses a sequential deep learning model for dropout prediction. The outcomes of our combined approach contribute to the field by (1) outperforming the previous original approach presented in [18]; (2) reducing the feature selection search space, resulting in better quality datasets for dropout prediction; (3) proposing a method to automatically select an optimal set of features; and, (4) providing an analysis of performance of features selection and prediction models. The code for this project is available at <http://github.com/SocialMachineLab/StudentDropoutPrediction>.

2 Methodology

Figure 1 depicts our dropout prediction approach step-by-step. Briefly, the first step obtains and preprocesses the data using data wrangling and machine learning (ML) techniques; the second step is responsible for the feature selection from the preprocessed data; and, finally, the last step trains and tests a deep sequential model based on Long Short-Term Memory and Fully Connected (FC) neural layers. Each step is explained in details below.

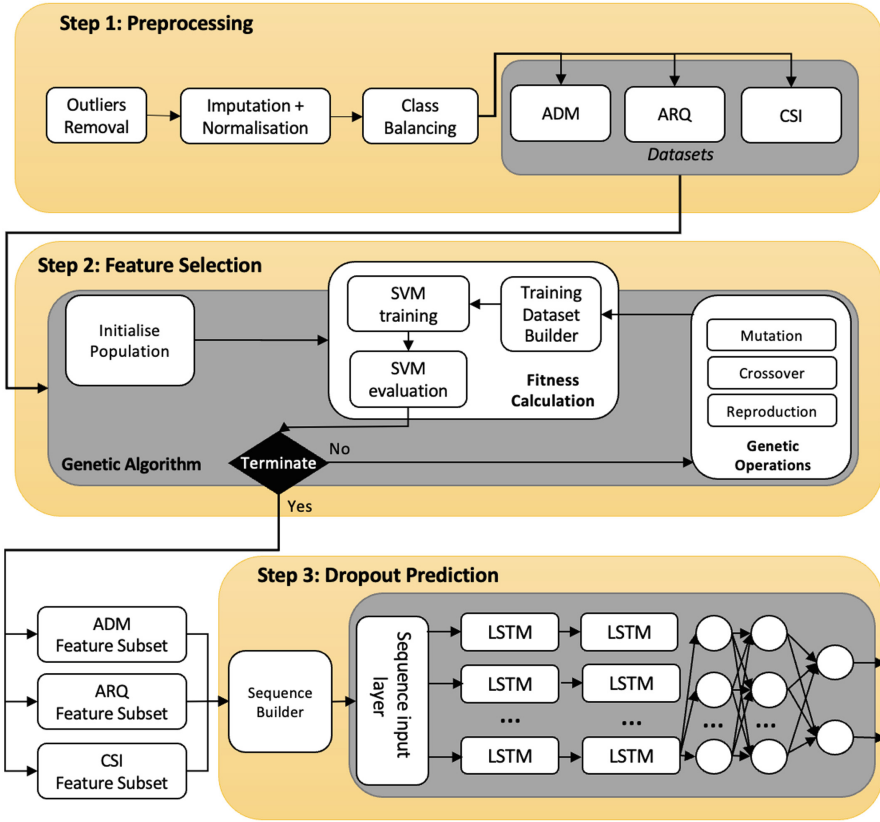


Fig. 1. Dropout prediction workflow.

2.1 Data Preprocessing

The dataset used in this paper has been used in previous research [18] and is provided by a Brazilian university. It contains 248,730 academic records of 5,582 students enrolled in six distinct degrees from 2001 to 2009. Table 1 shows the most relevant dataset attributes (features) in Portuguese with the corresponding English translation.

Most of the data is related to the status of a student in a given course and degree. The “semestre” attribute is related to the semester a student took a course. The attribute “sit_vinculo_atual” indicates that there are 12 status of enrolment where three of them (“desligado”, “matricula em abandono” and “jubilado”) represent the dropout status. Marks are scaled from 0 to 10 (inclusive) where 10 is the highest mark. The dataset is anonymised and the identifiers do not allow re-identification. Students’ identifiers are encoded with dummy alphanumeric code such as “aluno1010” (“student1010”).

To preprocess the dataset, we split it into three parts, each corresponding to a different degree: CSI (“Information Systems”), ADM (“Management”), and ARQ (“Architecture”). For each of the three resulting datasets, duplicate data was removed, and some inconsistencies and outliers were resolved. Categorical data was converted to numerical data using the LabelEncoder tool¹. Missing data was imputed using the random forest algorithm as it does not assume normality or require specification of parametric models. In addition, random forest outperforms other imputation algorithms as shown in [11].

For the dropout status attribute (“sit_vinculo_atual”), the instances were replaced by 0 (dropout) and 1 (enrolled). With the exception of attributes “sit_vinculo_atual” and “semestre”, all the input data were normalised using the traditional z-score method.

The last preprocessing step consists of oversampling to compensate for the imbalanced nature of this type of data. For this, we used the combination of Synthetic Minority Oversampling Technique (SMOTE) and Edited Nearest Neighbour (ENN) to synthesise the number of instances in the minority class (dropout). SMOTE considers the minorities to generate and balance the dataset with new instances based on real instances, ENN focuses on the majorities to eliminate instances on the boundary of the majority class whose predictions calculated by the k-nearest neighbours (KNN) differ from those of other majority class instances.

Table 1. Dataset attributes.

Original	EN translation	Original	EN translation
cod_curso	course code	mat_ano	enrolment year
nome_curso	course name	mat_sem	enrolment semester
cod_hab	degree code	periodo	term
nome_hab	degree name	ano	year
cod_enfase	emphasis code	semestre	semester (s.)
ano_curriculo	curriculum year	grupos	groups
cod_curriculo	curriculum code	disciplina	course
matricula	student identifier	semestre_recom	recommended s
no_creditos	no. of credits	semestre_do_aluno	student semester
grau	grades	turma	class
sit_final	final status (pass/fail)	sit_vinculo _actual	current status
tentativas	no. of attempts in a course	nome_disciplina	course name
nome_professor	professor name	identificador	identifier

2.2 Feature Selection

After data is preprocessed, the feature selection step is carried out using the SVM-based Genetic Algorithm (SVM-GA) to filter out irrelevant features.

¹ <https://scikit-learn.org/>.

Genetic algorithms are commonly used to generate solutions to difficult optimisation and search problems based on simulation of evolution and the natural selection of species. Although applying GA to feature selection is proven to improve model performance [2, 13, 15, 22], it has not been explored in student dropout studies.

The general procedure is displayed in Step 2 of Fig. 1. First, a random population of individuals is created. Each generated individual is binary encoded with each bit representing a feature of the input dataset. In this paper, an individual is represented by a 32-dimensional binary vector [*cod_curso* : 0, *grau* : 1, ..., *matricula* : 1] where each dimension determines the features considered in the calculation of the fitness of that individual.

The “natural” selection of individuals is based on an SVM classification model, which has demonstrated good performance as a fitness metric in feature selection in previous works [28]. The evolution process is similar to the roulette wheel where individuals with higher fitness scores have greater chance to be selected to generate offspring for the next generations. Individuals are created based on the following operators: crossover (new individuals produced by mating two existing individuals), reproduction (duplicated) and mutation (features are randomly turned on/off). The genetic algorithm ends when a predefined maximum number of generations is reached. Finally, the individual with the highest score is selected as input to the next step, that is, the feature set with the highest fitness score is used as input for our proposed dropout prediction approach.

2.3 Dropout Prediction via Neural Network

This paper formulates the problem of dropout prediction as to the use of students’ academic data to predict whether these students will drop out in a following semester. As student academic information is essentially a temporally ordered sequence of courses and grades, we opted for a neural network model capable of encoding the sequential nature of the data. The general architecture of the network is presented in Fig. 2, which includes two LSTM layers and three fully connected layers.

The LSTM is the main component of the neural network architecture. LSTM is a special type of recurrent neural network (RNNs) and unlike traditional feedforward neural networks, it has a feedback connection and gate schema that power LSTM to learn and memorise the information over time sequences. They can basically ‘remember’ previous states and use this information to decide what will be next. This feature makes them suitable for handling temporal data [5].

The input layer represents the student data. We use 32 sequential points of time per student representing the information across 8 semesters and 4 courses each semester. Students who dropped out in the midway or have not completed all the semesters, post-padding time series sequences are performed to ensure the total time step is the same. In each time step, the student’s course information is encoding using the set of features previously selected by our SVM-GA strategy.

After, the input information for each time step is sent to a pair of LSTM layers. These layers are responsible for extracting patterns from the features,

their sequential behaviour and encoding them in what is called the “hidden state” of the LSTM. Details of its calculation can be found in [5]. The final output hidden state is regarded as the input of a 3-layer fully connected neural network to predict the dropout.

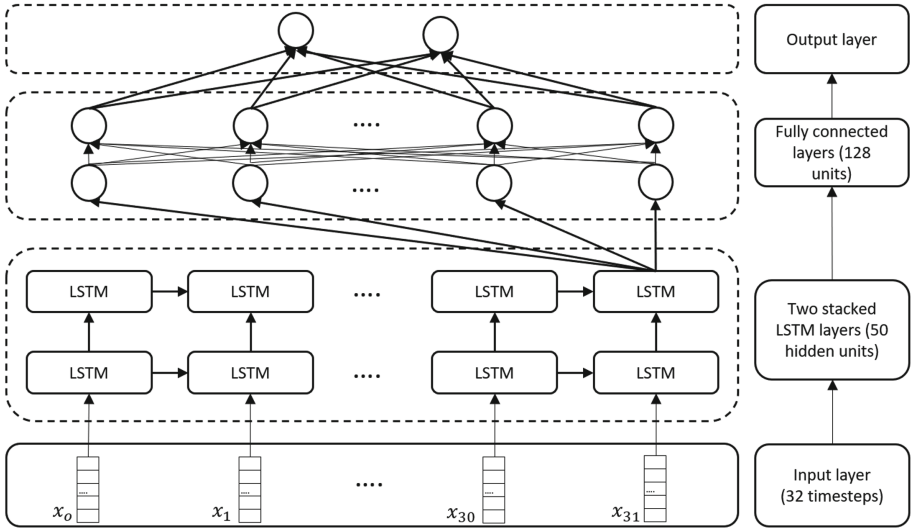


Fig. 2. Neural network architecture for dropout prediction.

The most appropriate parameters (i.e., the weights) of the neural network architecture are found through a training process using a subset of the data and backpropagation algorithm [5]. Furthermore, the hyperparameters of the model are the values of the input configurations. These are specified before the training process, and unlike model parameters, hyperparameters are selected by the designer. To find a good hyperparameter configuration, we performed a grid search on a limited set of values and selected those with which we obtained the best dropout prediction accuracy at the output. Table 2 presents the process of hyperparameter tuning and the values for each parameter. The selected values are displayed in bold².

3 Experiments and Discussion

To evaluate our approach, we used the ADM, ARQ, and CSI datasets described in Sect. 2.1 and the traditional evaluation metrics for binary classification problems: (A)ccuracy, (P)recision, (R)ecall, and F1 score as explained in [12].

Our experiments are divided into two parts. The first focuses on the feature selection method using the genetic strategy proposed in Sect. 2.2 and the second

² This information is important for the reproducibility of the results.

Table 2. Hyperparameters grid values (the best values found are in bold).

Hyperparameter	Value
No. of LSTM	1, 2 ,3,4
No. hidden size	30,40, 50 ,60,70
Dropout rate	0.2,0.4,0.5,0.6, 0.7 ,0.75,0.8
Batch First	True , False
No. layer of FC	2 ,3,4,5
No. hidden size	16,32,64, 128
Activation Function	ReLU (only in output layer), Sigmoid , Tanh
Epoch	30,50,60,70,80, 100 ,110,120
Optimiser	SGD, Adam
Learning rates	0.001 ,0.01,0.1
Loss function	MSE , Cross-Entropy, BCE

part focuses on the dropout prediction method using neural networks proposed in Sect. 2.3.

3.1 Experiment 1: Feature Selection Using SVM and Genetic Algorithms

By performing a sensitivity analysis on the genetic algorithm (as in [26, 30]), the population size, crossover rate, mutation, and the total number of generations for the genetic algorithm were set to 1,000, 0.75, 0.002, and 100, respectively. After running the genetic algorithm, the best individuals were taken as the set of features to be used for each of the three datasets (i.e., ARQ, ADM, and CSI). In terms of the F1 score obtained by the best individuals in the fitness evaluation using the SVM classifier, we have that the ARQ dataset obtained the best performance (95.03%) followed by the ADM (93.83%) and CSI (89.36%) datasets.

Table 3 shows the features that were not selected for each dataset. Five features were identified as irrelevant to the datasets while a few others were found to be irrelevant to specific datasets only. This is an interesting finding, as many studies [9, 18, 23] use the same set of features to predict student dropout in different degrees.

As in [19, 21], we used the eXtreme Gradient Boosting (XGBoost) and SHapley Additive exPlanation (SHAP) to evaluate feature selector’s decisions and obtain the top 10 most decisive features for each dataset (i.e., ADM, ARQ, and CSI datasets). As shown in Fig. 3, 4 and 5, the bar charts on the left side list the features ranked by mean SHAP values while the beeswarm plots on the right side show each feature instance (a single dot in the feature row) and corresponding SHAP and feature values, which provide an information-dense summary of impact of the top features on the model’s output. Based on the figures, we can

Table 3. Results of the dropped features.

Dataset	Dropped features	Common dropped features
ADM	mat_ano, disciplina, ano, matricula, no_credits	nome_professor, diff, identificador, cod_curriculo, turma
ARQ	ano_curriculo	
CSI	ano_curriculo, mat_sem, periodo, grupos, semestre, grau	

verify the following findings: (i) although it is the same prediction problem over three degrees, the relevant attributes are different between datasets and separating datasets by degrees is necessary for performing feature selection; and, (ii) the selector successfully identified and dropped the features with less contribution while keeping the highly relevant ones.

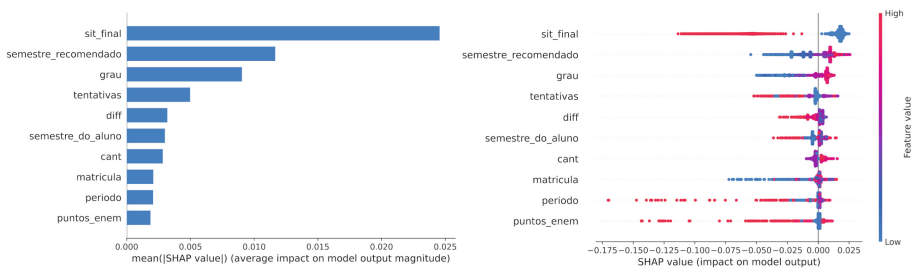


Fig. 3. Feature importance on ADM (bar and beeswarm).

It is important to note that the relevancy of each feature is determined by the quality of its instances. For example, in the case of the ‘nome_professor’ (name of the lecturer) feature the instances do not vary across semesters for the same course, therefore not being a good feature for predictive purposes. This feature, however, may be relevant for other universities where there are multiple lecturers for the same course. Another example is the specific feature ‘grau’ (grade), not selected in the CSI dataset. This feature is often considered relevant to dropout prediction approaches, however, for some reason to be investigated (possibly due to low variance or bias or evolutionary randomness), the grade feature was not selected in this particular dataset.

According to [29], the feature extraction and selection process is often an inefficient and inconsistent manual process. Automatic feature selection strategies are a great contribution to improve the early dropout prediction. Finally, in addition to helping to automatically select the features for dropout prediction in different datasets, the use of GA also reduces the feature space resulting in a more efficient process for subsequent models of dropout prediction.

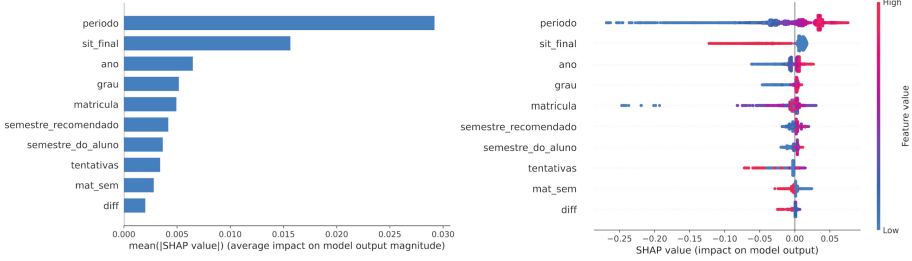


Fig. 4. Feature importance on ARQ (bar and beeswarm).

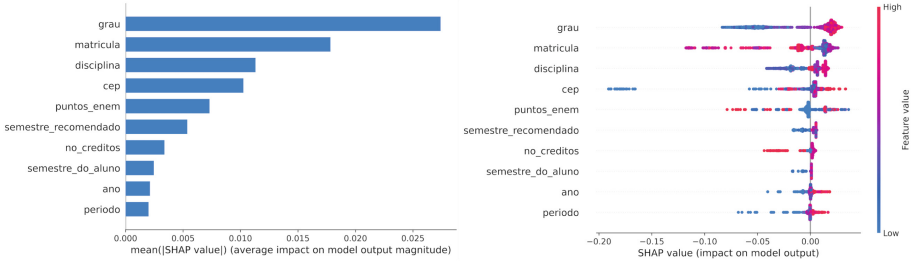


Fig. 5. Feature importance on CSI (bar and beeswarm).

3.2 Experiment 2: Students’ Dropout Prediction

The results obtained with the neural network described in Sect. 2.3 are presented in this section. As mentioned before, the representation of each time step uses the features selected by the SVM-GA strategy. The performance of the model on the test dataset validates the suitability of the proposed model (see Table 4).

Table 4. Average accuracy after 10, 100, 200 iterations, and the top accuracy for the ADM, ARQ, and CSI datasets.

Dataset	10 iter.(%)	100 iter.(%)	200 iter.(%)	Top. acc(%)
ADM	88.76	88.24	90.17	92.83
ARQ	94.84	95.76	95.19	97.65
CSI	74.98	75.84	76.31	81.52

Our models perform well in ADM and ARQ, whose best accuracy reaches 92.83% and 97.65%, respectively. Notably, the accuracy of ARQ improves the result in Manrique’s previous work (95.2%) by 2.45% [18]. This result is important to the extent that Manrique et al. carried out a manual process of selecting and building features that require deep domain knowledge. Therefore, we

hypothesise that the automatic feature selection process is able to outperform models with features manually selected by experts.

To validate this hypothesis, we must consider the contribution of the neural network architecture, which in our proposal considers temporal aspects via the LSTM layers. Therefore, we also conducted an experiment regarding the performance of model without feature selection (i.e., we employed all features after the preprocessing step (see Fig. 1)). As results are shown in Table 5, the largest difference between using or not GA appears in the ARQ dataset (8.9%). This also indicates the effectiveness of GA in terms of the contribution to the dropout prediction.

Table 5. Model comparison with and without feature selection step.

Dataset	Accuracy(%) without GA (GA_{A1})	Accuracy(%) with GA (GA_{A2})	$(GA_{A1} - GA_{A2})(\%)$
ADM	85.42	92.87	7.45
ARQ	88.75	97.65	8.9
CSI	75.64	81.52	5.88

The resulting model training history in terms of accuracy and loss in the testing dataset for ADM is shown in Figs. 6–7. They reveal the evolution of the predictive capacity of the model, whose end of the abscissa is an epoch. We can observe that the model in ADM always converges after 35–40 epochs and maintain stability. The same occurs to the ARQ dataset after 32–38 epochs (see Figs. 8–9). On the other hand, CSI presents unstable results probably because of the small number of student instances (5x less than ARQ and ADM), however, even for small datasets, the accuracy is still over 81% (see Figs. 10–11).

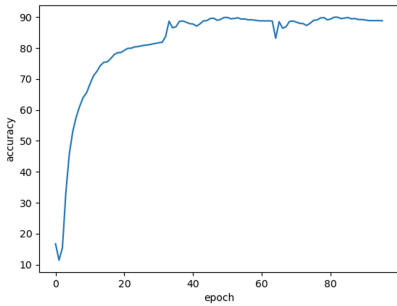


Fig. 6. Accuracy of LSTM+FC on ADM.

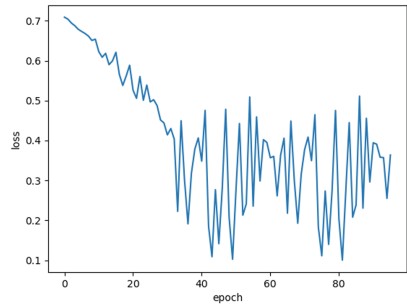


Fig. 7. Loss of LSTM+FC on ADM.

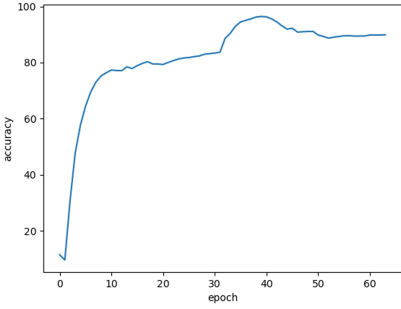


Fig. 8. Accuracy of LSTM+FC on ARQ.

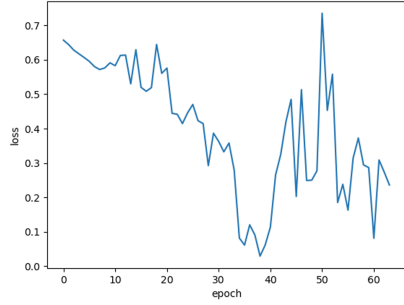


Fig. 9. Loss of LSTM+FC on ARQ.

Fluctuations in training loss over epochs can be observed in Fig. 7, 9, and 11. The small batch size is one of the latent reasons as the models will become more sensitive to mislabeled samples with a smaller batch size, which results in an update of an obvious increase of loss. However, due to the nature of our study, the batch size is automatically determined by the size of dataset and fixed points of time (32), which cannot be adjusted manually.

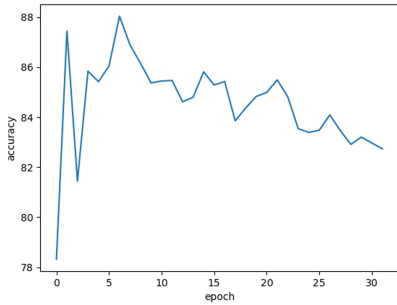


Fig. 10. Accuracy of LSTM+FC on CSI.

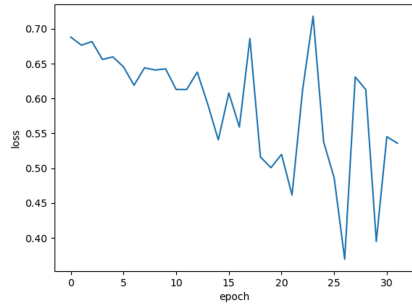


Fig. 11. Loss of LSTM+FC on CSI.

4 Conclusion and Future Works

In this study, owing to the importance of dropout rates in Higher Education Institutions, and the negative impact in lowering students' retainment during the pandemic, we built a new deep learning-based approach to dropout prediction using SVM and GA for feature selection. Our approach started with data preprocessing that included outlier removal, imputation and oversampling. SVM-GA is then applied to select the most relevant features for a dataset/degree

and, based on the selected features, LSTM is used for the final dropout prediction. The proposed approach outperformed prior work [18] achieving 97.65% of accuracy in the ARQ dataset. The contributions of this work lie in the ability of our SVM-based evolutionary algorithm to efficiently and accurately obtain subsets of features from any dataset without the need for an expert and the strong adaptability of LSTM in predicting dropout in sequential datasets.

As future work, we intend to extend the current approach to help with curriculum analysis by understanding the impact of each course on the student dropout decision and implementing a course enrolment recommendation tool to maximise student success while increasing retention rates. With regards to our method, instead of using fixed time steps in LSTM, which may result in sparse matrix for certain inputs, we intend to implement dynamic ones as this will make full use of the dataset avoid padding zeros. We also intend to generate more balanced datasets - a very difficult task as per its nature the datasets are imbalanced. However, this would help us measure the abilities and limitations of the proposed model. Finally, develop pre-trained models for dropout prediction reducing training costs and enabling other researchers to fine tune the model with their own data.

References

1. Ai, D., Zhang, T., Yu, G., Shao, X.: A dropout prediction framework combined with ensemble feature selection. In: Proceedings of the 2020 8th International Conference on Information and Education Technology, pp. 179–185. ICIET 2020, ACM, NY, USA (2020). <https://doi.org/10.1145/3395245.3396432>
2. Babatunde, O., Armstrong, L., Leng, J., Diepeveen, D.: A genetic algorithm-based feature selection. *Int. J. Electron. Commun. Comput. Eng.* **5**, 889–905 (2014)
3. Baranyi, M., Nagy, M., Molontay, R.: Interpretable deep learning for university dropout prediction. In: Proceedings of the 21st Annual Conference on Information Technology Education, pp. 13–19. SIGITE 2020, ACM, NY, USA (2020). <https://doi.org/10.1145/3368308.3415382>
4. Cai, L., Zhang, G.: Prediction of MOOCs dropout based on WCLSRT model. In: IEEE Conference on Advanced Information Technology, Electronic and Automation Control Conference (IAEAC), vol. 5, pp. 780–784 (2021). <https://doi.org/10.1109/IAEAC50856.2021.9390886>
5. Chollet, F.: Deep Learning with Python. Manning, November 2017
6. Diaz-Mujica, A., Pérez, M., Bernardo, A., Cervero, A., González-Pienda, J.: Affective and cognitive variables involved in structural prediction of university dropout. *Psicothema* **31**, 429–436 (2019). <https://doi.org/10.7334/psicothema2019.124>
7. Garratt-Reed, D., Roberts, L.D., Heritage, B.: Grades, student satisfaction and retention in online and face-to-face introductory psychology units: a test of equivalency theory. *Front. Psychol.* **7** (2016). <https://doi.org/10.3389/fpsyg.2016.00673>
8. Haiyang, L., Wang, Z., Benachour, P., Tubman, P.: A time series classification method for behaviour-based dropout prediction. In: IEEE 18th International Conference on Advanced Learning Technologies (ICALT), pp. 191–195 (2018). <https://doi.org/10.1109/ICALT.2018.00052>


9. Hasbun, T., Araya, A., Villalon, J.: Extracurricular activities as dropout prediction factors in higher education using decision trees. In: IEEE 16th International Conference on Advanced Learning Technologies (ICALT), pp. 242–244 (2016). <https://doi.org/10.1109/ICALT.2016.66>
10. Herbert, M.A.: Staying the course: a study in online student satisfaction and retention. *Online J. Distance Learn. Adm.* **9**(4), 300–317 (2006)
11. Hong, S., Lynn, H.S.: Accuracy of random-forest-based imputation of missing data in the presence of non-normality, non-linearity, and interaction. *BMC Med. Res. Methodol.* **20** (2020). Article No. 199. <https://bmcmmedresmethodol.biomedcentral.com/articles/10.1186/s12874-020-01080-1#citeas>
12. Hossin, M., Sulaiman, M.N.: A review on evaluation metrics for data classification evaluations. *Int. J. Data Min. Knowl. Manag. Process* **5**(2), 1 (2015)
13. Huang, J., Cai, Y., Xu, X.: A hybrid genetic algorithm for feature selection wrapper based on mutual information. *Pattern Recogn. Lett.* **28**(13), 1825–1844 (2007). <https://doi.org/10.1016/j.patrec.2007.05.011>
14. Kostopoulos, G., Kotsiantis, S., Ragos, O., Grapsa, T.N.: Early dropout prediction in distance higher education using active learning. In: 2017 8th International Conference on Information, Intelligence, Systems Applications (IISA), pp. 1–6 (2017). <https://doi.org/10.1109/IISA.2017.8316424>
15. Leardi, R.: Application of genetic algorithm-pls for feature selection in spectral data sets. *J. Chem. - J. Chemometr* **14**, 643–655 (2000). [https://doi.org/10.1002/1099-128X\(200009/12\)14:5/63.0.CO;2-E](https://doi.org/10.1002/1099-128X(200009/12)14:5/63.0.CO;2-E)
16. Limsathitwong, K., Tiwatthanont, K., Yatsungnoen, T.: Dropout prediction system to reduce discontinue study rate of information technology students. In: 5th International Conference on Business and Industrial Research (ICBIR), pp. 110–114 (2018). <https://doi.org/10.1109/ICBIR.2018.8391176>
17. Liu, K., Tatinati, S., Khong, A.W.H.: A weighted feature extraction technique based on temporal accumulation of learner behavior features for early prediction of dropouts. In: IEEE International Conference on Teaching, Assessment, and Learning for Engineering (TALE), pp. 295–302 (2020). <https://doi.org/10.1109/TALE48869.2020.9368317>
18. Manrique, R., Nunes, B.P., Marino, O., Casanova, M.A., Nurmikko-Fuller, T.: An analysis of student representation, representative features and classification algorithms to predict degree dropout. In: Proceedings of the 9th International Conference on Learning Analytics & Knowledge, pp. 401–410. LAK19, ACM, NY, USA (2019). <https://doi.org/10.1145/3303772.3303800>
19. Marcilio, W.E., Eler, D.M.: From explanations to feature selection: assessing shap values as feature selection mechanism. In: 2020 33rd SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI), pp. 340–347 (2020)
20. Nagrecha, S., Dillon, J.Z., Chawla, N.V.: Mooc dropout prediction: lessons learned from making pipelines interpretable. In: 26th International Conference on World Wide Web Companion, pp. 351–359. WWW 2017 Companion, Republic and Canton of Geneva, CHE (2017). <https://doi.org/10.1145/3041021.3054162>
21. Parsa, A.B., Movahedi, A., Taghipour, H., Derrible, S., Mohammadian, A.: Toward safer highways, application of xgboost and shap for real-time accident detection and feature analysis. *Accid. Anal. Prev.* **136**, 105405 (2019). <https://doi.org/10.1016/j.aap.2019.105405>
22. Pereira Nunes, B., Mera, A., Casanova, M.A., Fetahu, B., P. Paes Leme, L.A., Dietze, S.: Complex matching of RDF datatype properties. In: Decker, H., Lhotská, L., Link, S., Basl, J., Tjoa, A.M. (eds.) DEXA 2013. LNCS, vol. 8055, pp. 195–208. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-40285-2_18

23. Rovira, S., Puertas, E., Igual, L.: Data-driven system to predict academic grades and dropout. PLOS ONE **12**, e0171207 (2017). <https://doi.org/10.1371/journal.pone.0171207>
24. Sales, A.R.P., Balby, L., Cajueiro, A.: Exploiting academic records for predicting student drop out: a case study in Brazilian higher education. J. Inf. Data Manag. **7**, 166–180 (2016)
25. Shipley, B., Ian, W.: Here comes the drop: university drop out rates and increasing student retention through education, January 2019
26. Srinivas, C., Reddy, B.R., Ramji, K., Naveen, R.: Sensitivity analysis to determine the parameters of genetic algorithm for machine layout. Procedia Mater. Sci. **6**, 866–876 (2014). <https://doi.org/10.1016/j.mspro.2014.07.104>
27. Sukhbaatar, O., Ogata, K., Usagawa, T.: Mining educational data to predict academic dropouts: a case study in blended learning course. In: TENCON 2018–2018 IEEE Region 10 Conference, pp. 2205–2208 (2018). <https://doi.org/10.1109/TENCON.2018.8650138>
28. Tao, Z., Huiling, L., Wenwen, W., Xia, Y.: Ga-SVM based feature selection and parameter optimization in hospitalization expense modeling. Appl. Soft Comput. **75**, 323–332 (2019). <https://doi.org/10.1016/j.asoc.2018.11.001>
29. Wang, W., Yu, H., Miao, C.: Deep model for dropout prediction in MOOCs. In: International Conference on Crowd Science and Engineering, pp. 26–32. ICCSE 2017, ACM, New York (2017). <https://doi.org/10.1145/3126973.3126990>
30. Whitcombe, J., Cropp, R., Braddock, R., Agranovski, I.: The use of sensitivity analysis and genetic algorithms for the management of catalyst emissions from oil refineries. Math. Comput. Model. **44**(5), 430–438 (2006). <https://doi.org/10.1016/j.mcm.2006.01.003>
31. Wu, N., Zhang, L., Gao, Y., Zhang, M., Sun, X., Feng, J.: Clms-net: Dropout prediction in MOOCs with deep learning. In: Proceedings of the ACM Turing Celebration Conference - China, ACM TURC 2019, ACM, NY, USA (2019). <https://doi.org/10.1145/3321408.3322848>
32. Yang, D., Sinha, T., Adamson, D., Rose, C.P.: Turn on, tune in, drop out: anticipating student dropouts. In: in Massive Open Online Courses, in NIPS Data-Driven Education Workshop (2013)
33. Zhang, Y., Chang, L., Liu, T.: MOOCs dropout prediction based on hybrid deep neural network. In: 2020 International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC), pp. 197–203 (2020). <https://doi.org/10.1109/CyberC49757.2020.00039>
34. Zheng, Y., Gao, Z., Wang, Y., Fu, Q.: MOOC dropout prediction using FWTS-CNN model based on fused feature weighting and time series. IEEE Access **8**, 225324–225335 (2020). <https://doi.org/10.1109/ACCESS.2020.3045157>

Tutoring and Learning Systems: New Approaches, Frameworks, and Theories



Equitable Access to Intelligent Tutoring Systems Through Paper-Digital Integration

Nirmal Patel¹(✉) , Mithilesh Thakkar¹, Bansri Rabadiya², Darshan Patel¹, Shrey Malvi¹, Aditya Sharma¹, and Derek Lomas³

¹ Playpower Labs, Gandhinagar, India
{nirmal,mithilesh.thakkar,darshan.patel,shrey.malvi,
aditya.sharma}@playpowerlabs.com

² Teach for India, Ahmedabad, India

bansri.rabadiya2020@teachforindia.org

³ Delft University of Technology, Delft, Netherlands
j.d.lomas@tudelft.nl

Abstract. Intelligent Tutoring Systems (ITS) can only respond adaptively to the digital learning activities of the students. If students are learning offline without any digital devices, they have little or no means to receive personalized learning materials with the help of intelligent systems. This paper proposes a Paper-Digital Integration System that can provide offline learners equitable access to ITS capabilities by looking at their work on paper and giving personalized printable feedback. We analyzed data from a paper algebra assessment of $N = 17$ students and found mistakes that may generalize and help us offer adaptive paper-based recommendations to students. Our analysis showed us some specific algebra mistakes that may help in providing intelligent feedback.

Keywords: Paper-digital integration · Offline intelligent tutor · Equitable access · Offline learning

1 Introduction

A UNICEF survey in November 2020 found that nearly two-thirds of the school-aged children in the world do not have access to the internet at home [1]. Data show that a significant number of students across the world do not have access to any digital devices. The 2021 Annual Status of Education Report of India showed that over 30% of the students in rural India might not have access to a smartphone at home [2] (India has 250M+ K-12 learners [9]). Paper is a pervasive medium for learning for students with low access to digital learning facilities. Intelligent tutoring systems cannot respond to student learning on paper as what happens on paper is barely accessible to digital systems. The more time students spend learning on paper, the less opportunity intelligent tutoring systems have to support and nurture them. To increase the impact of intelligent tutors on the

students, we need to create a data-feedback loop between offline learning and intelligent systems. Once the digital systems have data about offline learning on paper, we can provide students online or offline adaptive feedback in response to their paper learning (Fig. 1).

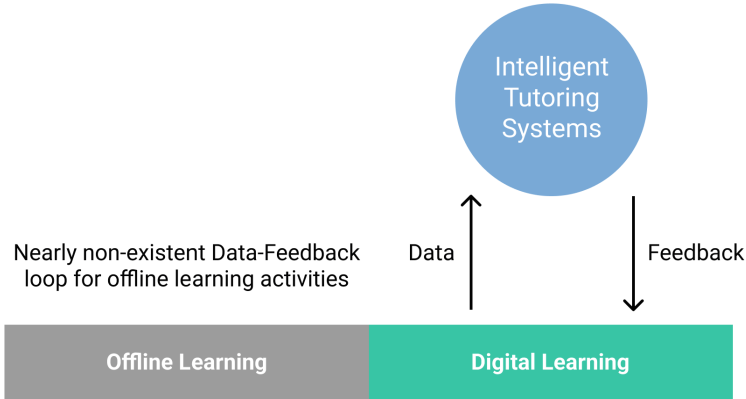


Fig. 1. Intelligent tutors are only equipped to support learning in the digital domain. Observational assessments are probably the only source of student data from the Offline Learning environment.

Paper is probably the most significant data source for offline learning in classrooms. Observational assessments also provide data for offline learning, but worksheets, workbooks, and notebooks are likely to contain a much higher quantity of offline learning data. The enormous amount of educational data on paper can enrich learning science. A workshop held at the 2020 NIPS Conference on Machine Learning for Education discussed ‘ImageNets for Education’, structured datasets that contain images of student work [6]. These structured datasets can help us better understand formative learning at scale. For STEM subjects, large-scale ImageNet for Education datasets can reveal distinct problem-solving patterns and help us identify common misconceptions or knowledge gaps. Large-scale image data also enable new possibilities for building intelligent tutors that can provide adaptive feedback to students by comparing their work with historical examples that have been reviewed by teachers or subject matter experts.

Writing is less likely to be abandoned in favor of digital typing for classroom learning - probably because writing improves outcomes. A recent meta-analysis showed that writing is an effective way to learn science, social studies, and mathematics [7]. A study from 2014 showed that when college students took written notes, they were better at answering conceptual questions [11]. Writing is also the preferred modality for solving math problems. Anthony et al. found that students preferred writing-based input for interacting with intelligent tutors [4]. In a small classroom study, Hinkley et al. observed that when students received digital devices for math learning, they still attempted to interact with those devices

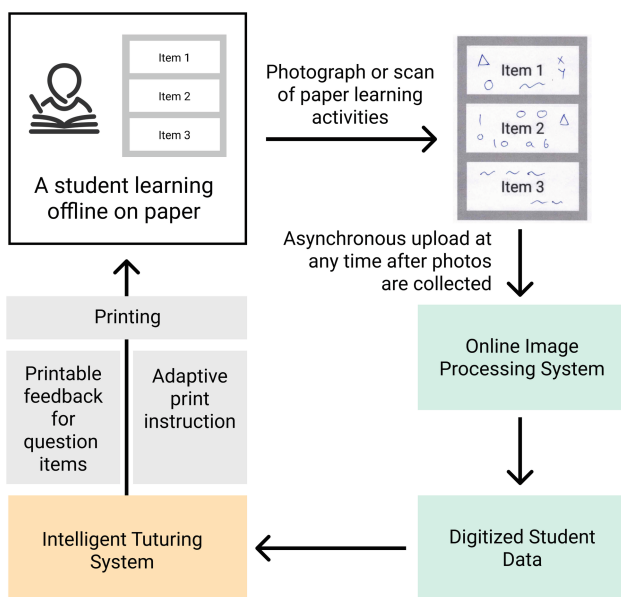


Fig. 2. Proposed paper-digital integration system where ITS can support students who are learning offline.

as they do with paper [8]. Writing on digital devices with pens is affordable to only a tiny portion of students, and it immediately poses equity challenges as different devices have different processing capabilities. For example, it is easier to solve multi-step math problems on a large-screen tablet than on mobile screens. Paper-digital integration can help us preserve the benefits of writing while keeping digital systems informed about student learning.

The inability to receive intelligent feedback for offline learning puts marginalized students at a disadvantage for receiving the benefits of Intelligent Tutoring Systems (ITS). We propose a Paper-Digital Integration System to make ITS accessible to students learning offline on paper.

The rest of the paper is structured as follows: Sect. 2 outlines our proposed paper-digital integration system. In Sect. 3, we describe how our work contributes to the field of ITS, and how it relates to prior work on digitizing and analyzing handwritten data. Section 4 describes the algebra assessment we used for our data analysis. Sections 5 and 6 contain image data and their analysis. Later sections discuss the need and potential impact of paper-digital integration.

2 Proposed Paper-Digital Integration System

To support offline students, an ITS needs to get learning data from offline settings. Paper is one of the most common sources of data in the offline learning

scenarios. We can collect images of offline student work on paper, and extract data from the images using AI models. The image collection can happen through online or offline smartphone devices or other digital devices that have camera. We can also collect photos through scanners. We can upload these images to the cloud asynchronously (i.e., immediately or when the internet becomes available). Computer vision programs can extract question-level data from the photos of student work and link them with student data models in the ITS. Handwriting recognition of item response photographs can give us step-by-step problem-solving processes of the students. Based on the student response, the ITS can produce feedback in a printable document form that we can deliver to offline students by printing it where the internet is available, followed by physical delivery. The feedback can contain corrective remarks on students' mistakes in the photographs, personalized instruction based on student performance, and other paper-based recommendations. The generated print materials can go through the same paper photographing/scanning process and help us provide further individualized feedback to the students. Figure 2 on page 6 shows an illustration of the described system.

For our proposed system to work, the ITS must reliably collect student learning data from page photographs. To provide students automated individualized feedback for their paper learning activities, we can use the results of the handwriting detection and match them with historical data to find potential automated responses. If the new work does not match any known pattern or historical instances, it can be sent for human review.

3 Related Work

Our proposed system aims to connect paper and digital to enable equitable access to the ITS. The unique part of our system is that it can operate in offline mode by transporting print learning materials. This is different from most of the prior works that only operate in the digital domain. Our proposed system can work in the areas that do not have any digital devices. We believe that this is a novel aspect of our system which may not be directly related to any prior work.

Some work has been done to create systems that can give intelligent feedback to student learning on paper. A recent study presented Homework Helper, a system to provide feedback on addition problem-solving [5]. To build the system, the authors collected a training dataset of how students solve multi-digit addition problems on paper and used that data to give new students feedback about their specific types of addition mistakes. Researchers have also described systems to cluster handwritten mathematical equations [13]. Clustering handwritten work can help us identify generalizable problem-solving patterns that can directly inform pedagogical practices.

Several systems have been created to incorporate online handwriting input into intelligent tutoring systems. Anthony et al. conducted a series of studies to develop handwriting input interfaces for cognitive tutors [3]. A recent study found no significant difference in student outcomes when comparing the input

modality in a digital intelligent tutor. The researchers in this study compared online handwriting input with the typing-based input [10].

4 Research Questions

We attempted to build a proof-of-concept program that mimicked our system's data collection and feedback generation parts. We collected paper assessment data of $N=17$ students from a Grade 7 classroom in Ahmadabad, India. We analyzed the scans of the classroom assessment to answer two key research questions:

1. **RQ1:** Can we reliably digitize handwritten question responses on paper (using handwriting recognition)?
2. **RQ2:** Are we able to see any generalizable patterns in the data that can be used to give students automated individualized feedback?

The handwritten algebra test we used for our analysis consisted of 12 questions. There were 5 one-point questions, 3 two-point questions, 2 three-point questions, and 2 four-point questions. Some questions gave students an option to answer either part A or part B. One question asked the students to mention two things they learned in the ongoing topic. We decided to analyze 4 questions that only asked students to solve specific algebra problems that had steps. The test had four pages, and each question had space below it for the response. One student ended up consuming all of the provided space and wrote their answer on a separate sheet of paper. The four problems that we analyzed are written below.

- Q6 - Subtract $3x^2 - 5x + 7$ from $5x^2 - 7x + 9$
- Q7 - Multiply: $(2y + 5)(2y + 5)$
- Q9 - Add $8x^2 + 7xy - 6y^2$, $4x^2 - 3xy + 2y^2$, and $-4x^2 + xy - y^2$
- Q12 - Multiply $(x^2 + 2y)$ by $(x^3 - 2xy + y^3)$ and find the value of the product for $x = 1$.

5 Handwriting Data

Once the students finished the test, we collected the pages and scanned them using a feed scanner. We scanned 64 pages in total, out of which 4 pages had unacceptable skews produced by the machine. After collecting scans of each page, we manually cropped responses to the selected questions for each student. Afterward, we arranged the data so that all responses to one question were within one folder. This data arrangement immediately allowed us to look at the variance in the item response patterns of the students. Next, we took the images of individual item responses and manually segmented them into lines. Open-source line segmentation algorithms are available through software packages like OpenCV, but they need to be tuned to work on real-world datasets. We did the line segmentation step manually. Once the response data were at the line level,



Fig. 3. $N = 17$ handwritten responses to the question “Multiply: $(2y + 5)(2y + 5)$ ”. The “lines” folder contained the individual lines of each response.

we used a commercially available math handwriting recognition service *Mathpix*. This service provides an API that can take images of equations and return the recognized math script in various formats, including LaTeX and ASCII Math. We used ASCII Math output for our analysis as that was something that we could read and sort easily. Figure 3 shows our collected responses for the question “Multiply: $(2y + 5)(2y + 5)$ ”.

6 Analysis

The Mathpix API for handwritten equation recognition returned a detection confidence metric that allowed us to determine whether the digital handwriting in ASCII Math should be considered for analysis or not. The distribution of the detection confidence in Fig. 4 on Page 7 shows that majority of the values had high confidence. Specifically, 62% of the equations had detection confidence of over 90%. We analyzed the confidence values for their face validity and found them reliable, although a bigger and systematic analysis would be required for usage at scale. We also found that equations with scratches had low confidence values, even though most of the symbol recognitions were accurate. Based on our observations, we concluded that the commercially available handwriting recognition capability was likely reliable for a majority of the students, but a systematic analysis is required before using it at scale (**RQ1**).

To answer RQ2, we looked at the step-level data of student responses. We first attempted to cluster the student responses by using string edit distance. We produced one string of response for each student by concatenating the individual steps and then calculated the pairwise distance matrix between the students for each question. Using hierarchical clustering over the distance matrices for the four unique questions did not yield interesting clusters, even though some

students had arrived at the same answer in the test. We found the edit distance metric problematic because if one student had written a short response that had the same final answer as another student who wrote a more extended response, the edit distance remained large.

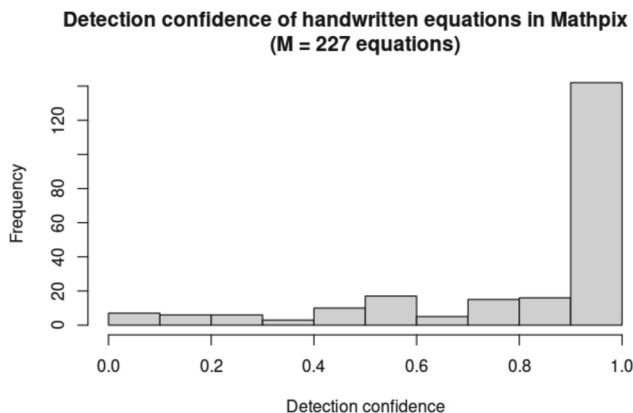


Fig. 4. Distribution of handwritten equation recognition confidence using Mathpix.

Given that we had a small sample, we decided to tabulate and sort the final steps of the students and see whether any students had given similar answers or had shown similar errors. By sorting the data, we quickly found that for Q6, three students had added the constants of the quadratic terms instead of subtracting them (i.e., students did $5+3$ instead of $5-2$). For Q7, we found that five students used an incorrect process for bracket expansion and multiplied the first two numbers, and added/multiplied the last two numbers. Three of these students answered the $(2y+5)(2y+5)$ as $(4y+10)$, while one answered it as $(4y+25)$ and another as $(4y-25)$ (this can be verified through Fig. 3). Q9 had much variance in the answers, and no students had a similar solution. Still, we saw that some students had mistakenly added the constants of the quadratic terms without considering their signs. This ‘misconception’ led to five students making a mistake in calculating the first term of the answer (which should be $8x^2$, but two students got $16x^2$ while two got $-16x^2$). For Q12, we found that three students had given the same accurate answer while others had given various types of responses, none of them being true. In summary, we recognized at least one generalizable misconception about bracket opening in Q7 and common mistakes in Q6 and Q8 about adding/subtracting the constants of the quadratic terms. These findings provided a partial answer to **RQ2** and led us to hypothesize that on a larger sample of data, we may see similar errors again and find new generalizable error patterns.

7 Discussion

Our proposed system produced an ‘ImageNet for Education’ dataset, where we had item responses in the form of handwriting images. We built this dataset for four questions a small number of students. On a larger scale, such a dataset can help us build problem-solving process models that capture the step-by-step process of solving problems typically presented in STEM disciplines. These models can contain aggregated information about how students respond to individual question items. For mathematics and related subjects like physics, where students typically solve problems step-by-step, we may be able to build models that can accurately predict the student’s next step given the previous steps. Such models can provide real-time feedback to students solving problems on screens using digital pens.

Nearly all intelligent tutoring systems available today operate in the digital domain. While technology and digital devices may be the future, we need to consider the realities of the present and create ITS that can deliver benefits to all students. If intelligent tutoring systems can be nearly as effective as human tutors [12], then expanding their reach through Paper-Digital Integration is likely to increase the overall impact of ITS research.

8 Conclusion and Future Work

Our study shows how Paper-Digital Integration can extend the benefits of ITS to offline learners. Education technology should be designed to address the needs of all learners. We plan to collect a bigger dataset in the future that contains question items from various topics and subjects and devise algorithms that can give data-driven automated feedback on paper learning. By looking at many samples of handwritten item responses to questions, we may be able to devise a taxonomy of common types of mistakes that students make. Such a taxonomy can help us design targeted interventions that can address specific student needs. ITS that automatically give targeted intervention resources on known misconceptions should be created and evaluated to understand the potential impact of learner data on paper.




References

1. Two thirds of the world’s school-age children have no internet access at home, new UNICEF-ITU report says (2020). <https://www.unicef.org/press-releases/two-thirds-worlds-school-age-children-have-no-internet-access-home-new-unicef-itu>
2. India’s annual status of education report (rural) 2021, p. 28 (2021). <http://img.aseercentre.org/docs/aser2021forweb.pdf>
3. Anthony, L., Mankoff, J., Mitchell, T., Gross, M.: Developing handwriting-based intelligent tutors to enhance mathematics learning. Unpublished doctoral dissertation, Carnegie Mellon University, USA (2008)
4. Anthony, L., Yang, J., Koedinger, K.R.: A paradigm for handwriting-based intelligent tutors. *Int. J. Hum. Comput. Stud.* **70**(11), 866–887 (2012)

5. Davis, S.R., DeCapito, C., Nelson, E., Sharma, K., Hand, E.M.: Homework helper: providing valuable feedback on math mistakes. In: International Symposium on Visual Computing, pp. 533–544. Springer (2020)
6. Garg, K., Heffernan, N., Meyers, K.: Workshop: advances and opportunities: machine learning for education (2022). https://nips.cc/virtual/2020/public/workshop_16104.html
7. Graham, S., Kiuahara, S.A., MacKay, M.: The effects of writing on learning in science, social studies, and mathematics: a meta-analysis. *Rev. Educ. Res.* **90**(2), 179–226 (2020)
8. Hinkley, W., Heffernan, N., Lee Bouygues, H.: The benefits of using pencil and paper in math (2020)
9. Jain, P., Dhawan, A., Ishwaran, G., Jhingan, A.: Private sector’s contribution to k-12 education in India (2014)
10. de MORAIS, F., JAQUES, P.A.: Does handwriting impact learning on math tutoring systems? *Inform. Educ.* (2021)
11. Mueller, P.A., Oppenheimer, D.M.: The pen is mightier than the keyboard: advantages of longhand over laptop note taking. *Psychol. Sci.* **25**(6), 1159–1168 (2014)
12. VanLehn, K.: The relative effectiveness of human tutoring, intelligent tutoring systems, and other tutoring systems. *Educ. Psychol.* **46**(4), 197–221 (2011)
13. Vu, T.M., Phan, K.M., Ung, H.Q., Nguyen, C.T., Nakagawa, M.: Clustering of handwritten mathematical expressions for computer-assisted marking. *IEICE Trans. Inf. Syst.* **104**(2), 275–284 (2021)



“See the Image in Different Contexts”: Using Reverse Image Search to Support the Identification of Fake News in Instagram-Like Social Media

Farbod Aprin¹ (✉) , Irene-Angelica Chounta² , and H. Ulrich Hoppe¹ 

¹ RIAS – Rhein-Ruhr-Institut für angewandte Systeminnovation e.V., Bürgerstr. 15, 47057 Duisburg, Germany

{fa,uh}@rias-institute.eu

² Department of Computer Science and Applied Cognitive Science, University of Duisburg-Essen, 47057 Duisburg, Germany

irene-angelica.chounta@uni-due.de

Abstract. Social media are an integral part of the daily lives of today’s young generation. In addition to the positive impact on learning through these channels, there are also risks related to toxic content like “fake news” on various social media. Fake news aims to change opinions based on disinformation or misinformation supporting conspiracy theories, e.g., related to the pandemic. Fake news creators use various multimedia artifacts, including images taken from serious and valid news sources, to attract the audience’s attention. Tracking images in different contexts can give social media users important clues to distinguish fake news from credible information. We report on the development of a web-based learning environment that includes a “virtual learning companion” to help learners improve their understanding, awareness, and critical thinking concerning such social media threats. The learning environment mimics Instagram and includes toxic and non-toxic content in a controlled way. The companion is implemented as a browser plugin that communicates with students via chat. The companion poses knowledge activation questions and answers according to an underlying script. The companion offers other sources with the same image identified through Reverse Image Search (RIS). The goal is to help learners find the same image in different contexts with different textual descriptions and keywords. For this purpose, we added basic NLP mechanisms to extract keywords from these contexts, including keywords that signal persuasiveness. Currently, we evaluate the impact of this tool and the provided support in distinguishing fake or credible news.

Keywords: AI-based learning support systems · Learning companion systems (LCS) · Intelligent tutoring system (ITS) · Recommendation · Awareness tool · Misinformation · Fake news · Social media · Chatbot · xAPI · Gamification · COVID-19

1 Introduction

Besides positive and beneficial aspects of social media, like facilitating information sharing and co-creativity, there are also threats and risks that specifically affect young users. Fake news, impersonation and conspiracy theories that can impact users’ views about events and news are such threats. This kind of misinformation is typically presented as authentic and credible in a persuasive way. For example, news posts on social media are usually presented as a combination of text and multimedia (especially images). The role of social media in the news ecosystem was highlighted in August 2017 in a report of the Center and Knight Foundation stating that two-thirds of U.S. adults obtain 67 percent of their daily news from social media, and 20 percent do so frequently [1]. Most U.S. Americans who see fake news reports are bound to believe in it [2].

The Virtual Learning Companion (VLC) assists learners in making judgments via a chatbot interface during the process of fact-checking. A specific type of support relies on “Reverse Image Search” (RIS) to retrieve and provide access to other instances of the same image in different contexts. Additionally, the VLC delivers keywords and phrases using natural language processing (NLP) from the other source websites. The companion also asks knowledge activation questions and motivates the learners to engage in fact-checking and to remain skeptical about the social media posts.

The following section provides relevant background from different areas. The first aspect is general human-oriented approaches to identifying and fighting fake news in social media. Such human-oriented approaches are to be distinguished from current AI-based techniques for the automatic detection of fake news. Given the educational orientation of our project, human-oriented approaches are particularly relevant. We also comment on systems supporting learners/users in fake news detection and the creation of awareness. Finally, we elaborate on the roots of our approach in the ITS tradition of learning companion systems. The specificity of our own approach lies in the provision of comparative contextual information using images as cues in combination with Reverse Image Search. We see this as a kind of learning from various contexts, which is a new approach to addressing fake news from an educational perspective.

2 Background

2.1 Human-Oriented Approaches to Dealing with Fake News

Victoria L. Rubin defines a conceptual model for fake news in the form of a triangle with the vertices Susceptible Host, Virulent Pathogen, and the Conducive Environment [3]. According to the model, fake news spreads if and only if the three causal factors occur simultaneously. Three interventions suggested interrupting the interaction of the mentioned factors: Automation to disarm the virulent pathogen, education to defuse the susceptible host, and regulation to make the conducive environment safe.

Automated, AI-based approaches can detect toxic features and add labels to the information items. One example of this approach is the LiT.RL News Verification Browser, a search tool for newsreaders, reporters, editors, or information experts. The tool examines the language used in digital news websites to determine whether it is clickbait (to 94% accuracy on a test set of 5670 texts for an offline experiment), satirical news, or fake

news. LiT.RL visualizes the results by highlighting the content in color-coded categories. The classification provided by the LiT.RL is not perfect and is not always adequate for public usage. Multimedia formats are unsupported [4].

Automated news validation systems based on NLP techniques can be helpful to assist content creators in quickly verifying common characteristics of misinformation. Additionally, such systems could also help teachers to teach critical content assessment skills or help information experts to reduce information overload for news clients by filtering out and flagging suspicious narratives [5].

The labeling strategy that is currently used on social media is sensitive for any keyword that could potentially be fake news (e.g., COVID-19 or vaccine). On the other hand, it is cumbersome to track and detect fake content. For example, deep fake image manipulation makes detecting fake content hard for the machine and for humans [6].

Another strategy to defuse the susceptible host and to help learner develop their own mechanisms to recognize and counter such influences is to educate [3]. This approach can be divided into two main strategies: increase the awareness and improve the learner's prejudice. There are adventitious techniques to "improve prejudices and awareness". For example, by using games and techniques that show the learner how to seek the truth and credible information from different contexts by using libraries and search engines. Moreover, to improve the learner's prejudice, we can deliver a series of games to show how the fake news content creators make the content on social media and what kinds of content are usually categorized as misinformation.

To improve the learner's critical thinking rather than working on their bias, we can equip learners with a "fact-checking awareness tool". Such tools are designed to motivate learners to practice fact-checking. According to the American Library Association (ALA), accessing accurate information from different contexts, without censorship and filtering, is the best way to counter misinformation and media manipulation [7].

The last point in the triangle is "Regulatory". Legislative work should be methodically proactive and robust as the global Internet society aims to eradicate these pathogenic "Fakes" in the conducive digital environment by preventing them from reaching and "contaminating" susceptible hosts. The EU commission warned that social media companies would face new regulations unless they urgently attack fake news¹. Some programs were set up under the lead of the EU and international organizations to tackle disinformation related to the pandemic situation in June 2020². The inventor of the WWW, Tim Berners-Lee, made this statement in a recent Web Foundation conference: "*The administrations must acclimate rules and regulations to the digital age. They must guarantee that markets remain competitive, innovative, and open. And they have a responsibility to protect people's rights and freedoms on the Internet*"³.

¹ Tech firms could face new EU regulations over fake news, <https://bit.ly/3BqBBLw>, Jan 2022.

² Fighting disinformation, bit.ly/EUDisinfoCOVID19, Jan 2022.

³ 30 years on, what's next #ForTheWeb, <https://bit.ly/3LzacLS>, Jan 2022.

2.2 Awareness Tools

There are games and tools to increase learner awareness rather than filtering for possible fake news in social media. *Harmony Square*⁴ and *Bad News*⁵ games are designed to teach the learner to understand the six common misinformation techniques involved in spreading disinformation by playing the role of Twitter Editor. These games expose players to fake news tactics, and they win by publishing headlines that attract the most followers. After learning each technique, they reward the player with misinformation badges. The result of an empirical study for *Bad News* indicates an enhancement in learner’s ability to spot misinformation techniques compared to a gamified control group, boosts people’s confidence in their judgment, and can also confer psychological resistance to common online misinformation strategies in different countries with various background cultures [8].

A serious mini-game platform and information booklet have already been prepared and distributed to help teenagers critically reflect on digital advertising by *Media Awareness Network (MNet) (Current name Media Smarts*⁶)[9]. MNet has provided several free games for teen learners, some of which are specifically designed to address the issue of bias and prejudice with lack of information and to promote critical thinking skills. Media Smarts mentioned in their agenda that their goal is to encourage teens to examine the information and seek alternative viewpoints to raise their awareness [10]. In the *Reality Check*⁷ game, an awareness game of Media Smarts, students learn how to find evidence, such as where a story initially came from and how to check it against other sources. They also learn how to use tools, such as fact-checking websites and reverse image searches in the narrative form. *Reality Check* game and our approach have a common strategy of focusing on teaching fact-checking techniques and motivating learners to fact-check any dubious news to reduce their biases. Meanwhile, *Bad News* and *Harmony* games inoculate learners and improve prejudging about the information.

2.3 Learning Companion Systems

Learning Companion Systems (LCS) is a subset of Intelligent Tutoring Systems (ITSs). They are characterized by providing personalized support and adaptive feedback through an explicit agent or companion [11]. The companion, guides the learner gradually and commonly adopts a non-authoritative role. The agent environment interface may incorporate multimedia, interactive buttons, menus, text, voice, animation, diagrams, virtual reality, or other interactive techniques.

LCS usually includes machine learning and natural language processing (NLP) techniques to facilitate communication between the LCS and the learners. Logging and tracking the student’s interactions with an LCS is used in student modeling. One scenario for the LCS would be to ask the learner to describe the logic behind their answers as a

⁴ Harmony Square: <https://harmonysquare.game/en>, Jan 2022.

⁵ Bad-news-game: <https://www.sdmlab.psychol.cam.ac.uk/research/bad-news-game>, Jan 2022.

⁶ Canada’s center for digital and media literacy <https://mediasmarts.ca/> 22-01-2022.

⁷ Reality Check: The Game <https://mediasmarts.ca/digital-media-literacy/educational-games/reality-check-game>, 12-02-2021.

reflective question about the controversy of probable fake news. Each step during their task might lead to more robust learning, in line with the work on self-explanation at the time. Learners might develop many answers and articulate the reasons behind their responses that distill their understanding (self-regulated learning strategies) [12].

Based on the definitions of adaptive systems, adaptability is one of the most important issues for a tutoring system that includes LCS. The system should adapt to the learner's responses and actions and be flexible depending on the context and previous responses of the learner.

The agent response can be implemented in the conversation as feedback to the learner's text input or provide an appropriate visual response to the learner's interaction in various ways, such as offering an educational video as a recommendation [13]. Modeling students as a foundation for adaptive feedback in LCS tutorial dialogs can immensely increase learning progress for students with low and high prior knowledge [14]. An LCS can involve numerous roles in an instructional context. For example, the role of a leader who suggests new ideas for learners or a critic who challenges learners' suggestions [15].

Concerning the question of how knowledgeable the learning companion agent should be to reach the learner's expectation and motivate the student to continue collaborating with the agent, *Hietala* and *Niemirepo* pointed out that the learners lose their motivation if they use a knowledgeable and robust companion all the time. Notably, in the beginning, a companion that makes mistakes like humans is more effective, still, for a challenging assignment or dealing with a new issue [16].

Our Virtual Learning Companion (VLC) supports features such as role-playing for learners, providing adaptive feedback based on previous learner interactions and responses. Plus, providing educational recommendations and analysis artifacts based on RIS links judging environmental images and asking knowledge activation questions. In addition, receiving input and displaying information are among the core functions of the VLC system.

3 Approach

3.1 System Components and Architecture

We have chosen the PixelFed open-source framework for social media for a controlled social network environment. This environment allows for providing content in a controlled way and facilitates basic social media interactions. PixelFed is similar to Instagram: it is suitable for images, captions, and comments. The VLC is developed as a Chrome browser plugin that interacts with PixelFed artifacts. It moderates through chat with questions and recommendations while the student engages with the environment's artifacts. Learners will interact with the environment guided by the tasks they obtain from the companion. As illustrated in technical architecture in (cf. Figure 1), the companion system comprises two sections: first, a frontend (the Chrome extension that interacts with the PixelFed environment); second, the backend microservices with a middleware (NodeJS express⁸) that mediates the frontend communication through REST API.

⁸ NodeJS is a JavaScript runtime built on Chrome's V8 JavaScript engine, 12-11-2021.

As part of our middleware, we have implemented an API for connecting to a WIT.AI⁹ module in the backend. According to the trained model, this module allows for extending the chatbot interaction with an AI-powered module that detects the user’s intent for each free text conversation. However, this feature was not used in the current trial for the chatbot conversation besides the underlined script. We create and expand the metadata for each image on PixelFed environment and store it in the document-oriented database (MongoDB¹⁰). We use Learning Locker as Learning Record Store (LRS) to log the action records in the standard Experience API (xAPI) structure.

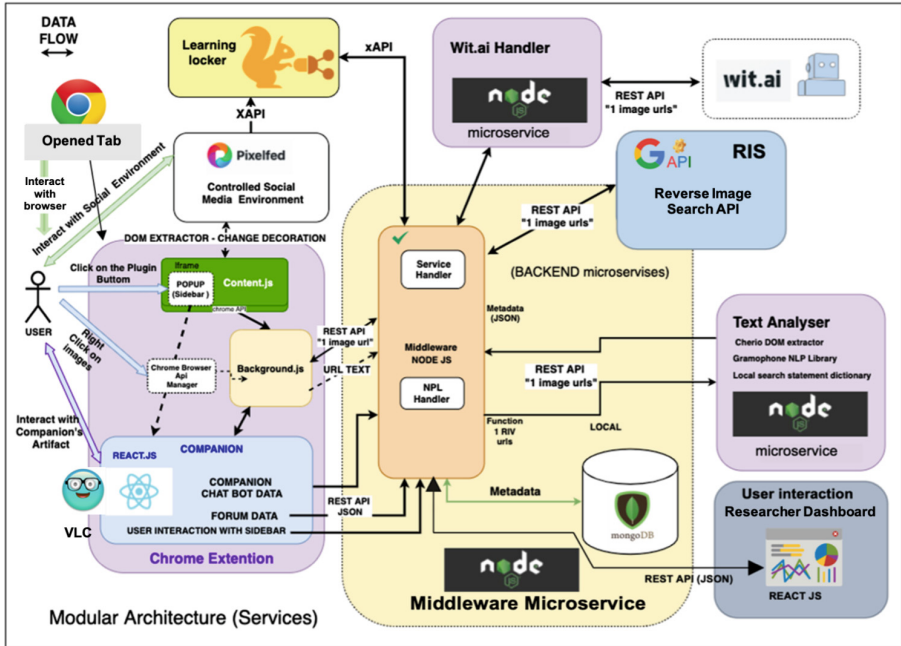


Fig. 1. The technical architecture and dataflow diagram for the companion system with service links (APIs)

A researcher dashboard module (see Fig. 1) has been implemented to analyze and visualize the user interactions.

⁹ WIT-AI is a tool to Build Natural Language Experiences <https://wit.ai>, 12-11-2021.

¹⁰ MongoDB, document-based, distributed database built for modern application, 12-11-2021.

3.2 Reverse Image Search Module (RIS)

In the front-end, the learner can select an image on the social media interface; then the companion system (as a chrome extension) sends the image URL to the middle-ware microservice. Associated with the extracted semantic keywords, it searches in the extracted text for the bold sentences corroding to the predefined dictionary, such as the sentences that contain words like “Claim”, “Fake,” etc. In the text analyzer microservice, we utilize the Cheerio library¹¹ to accomplish web scraping in JavaScript (Dom Tree extractor).

Moreover, to analyze the scrapped text of each RIS website, the Gramophone¹² module was used to extract the keywords based on term frequency and plus the filtering algorithm that we implemented on this service. The Gramophone can be configured to extract arbitrary length phrases (n-grams) of any length and not just keywords.

4 Scenario for Trials

The learning environment uses a Chrome browser to access PixelFed with a prepared collection of fake/fact news images and captions. The actual companion (VLC) is implemented as an extension or plug-in of the Chrome browser.

Learners gain access to the common PixelFed environment by clicking on the provided unique, anonymous user-token link. Then, they are presented with brief instructions on how to open the Chrome extension. The companion image appears in the sidebar and guides them to select an image on PixelFed to activate the chatbot. After right-clicking on an image and selecting the VLC button, the companion automatically starts communicating with a chatbot-style conversation (e.g., “Hey, you made it!”) and asks some demographic questions (gender, age...), following up with knowledge activation questions that the learner has to answer as free text, e.g., ask the learner to express their opinion about the selected image in PixelFed. In the next step, the companion asks learner whether the Post on PixelFed Fake News or Fact is, and the learner has to vote/classify for only one of five categories (Fake, Probably Fake, Not sure, Probably Fact, Fact). Then, the learner must answer reflective questions to explain the reasons behind their answer.

The companion applies to stimulate reflection questions (e.g., “How sure are you?”). Next, the companion unlocks a “Recommended” tab that includes links from the Reverse Image Search (RIS). The same image is displayed in a different context and on websites in the extension’s sidebar (cf. Fig. 2). Learners can visit these connections and compare each RIS link’s keywords, metadata, bold phrases based on a predefined dictionary, and brief descriptions of each RIS link for the selected post. In this step, the companion motivates, and guides learners use an external tool to freely search on the Internet or perform and use reverse images tool themselves to get even more context on the topic.

¹¹ <https://zetcode.com/javascript/cheerio/>, Cheerio module web scraping in JavaScript 02-2022.

¹² <https://github.com/bxjx/gramophone>, Gramophone module keyword extractor 02-2022.

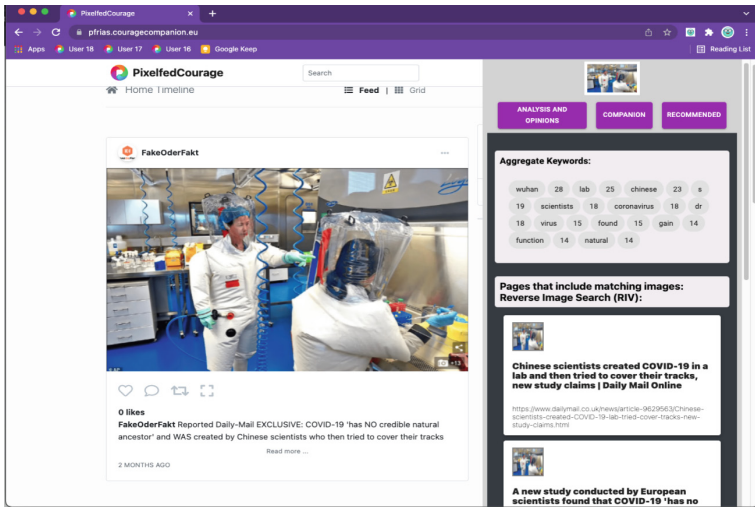


Fig. 2. The PixelFed environment (left side) with an image and the caption and the VLC add-on with contextual information (right side)

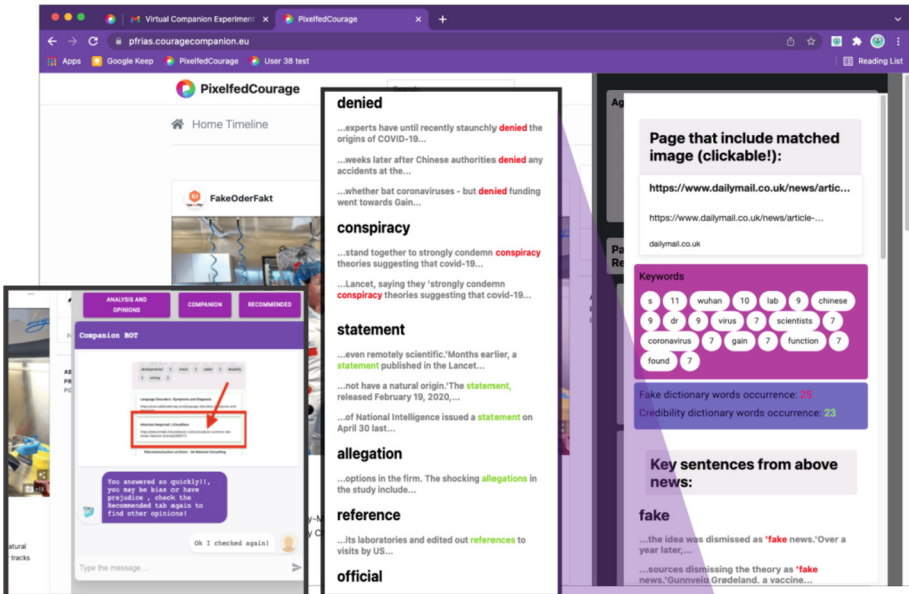


Fig. 3. Learner interactions with the VLC: left – advice to check generated metadata more closely - after clicking on an RIS link, the corresponding keyword analysis is displayed (right); center: learner scrolls down through “key sentences” with highlighted phrases.

Figure 3 illustrates how learners can see each one of the retrieved RIS links in more detail. Learners can review the title and click on each link to see the bold sentences with the words highlighted and compare them to the caption on PixelFed. For example, learners will be able to see if the image is manipulated, the text is distorted, indicates misinformation, or is based on credible news from the well-known channels.

For researchers, it is interesting to know if learners change their minds after interacting with the additional artifacts provided by RIS or the recommended learning instructions. It is important to understand in which direction learners think and if their judgment changes after receiving feedback and interacting with the learning guide from the companion. Therefore, we log all meaningful learner interactions, timestamps, and any related websites that the learner visited during the conversation with the learning guide in the standard xAPI format during the test in LRS.

After interacting with the metadata for each image and returning to the chatbot conversation, the companion can provide a warning or feedback as a pedagogical intervention. Suppose learners quickly assess or respond without checking the recommended RIS. In this case, they receive a notification in the chat, e.g., “Be careful not to be biased!”. After browsing the RIS links, the companion then provides new (adapted) feedback and asks if they want to change their idea and vote again.

After the revision chatbot asks them to explain and justify their vote change, they have this conversation for all three established images in the PixelFed. In the end, the companion unlocks the “Analysis Tab and Collaboration” to check others’ votes as a bar chart.

The system also stores in the database learner’s responses, the companion’s chat reactions, the learner interactions with environmental artifacts and the tab that learner opened in the browser during the experiment in its internal database. We are considering applying for the following school trials; for three selected images for the PixelFed environment related to pandemic situations. The images may contain fake or credible news and conspiracy theories that are used to teach critical thinking as a skill or to raise learner’s awareness. Also, teach them how to use the RIS tool to find the same image in different contexts to make better judgments from the signals. At the end the companion gives some feedback from experts and their judgements over the provided posts and asks the learner if they were aware of this RIS method to detect fake news.

5 User Evaluation

In this first trial with the Chrome extension and PixelFed environment, forty-five invitations were sent to volunteer test users from the University of Duisburg-Essen to participate in the companion system. The invitations are accompanied by an anonymous, randomly selected user token with the instructional video and files. Twelve participants, aged 19–65, from different cultural backgrounds, actively participated in the study. The purpose of this preliminary, informal study was to test the practicability of the system (as a mainly technical test). Given this orientation, our user population was based on personal contacts, not striving for representativeness.

Test users were presented with three images in the control environment (PixelFed) that contained a fake, facts, or a controversial item about COVID-19. As mentioned in the example scenario, participants were instructed to interact with the companion (Chrome extension) and finish the conversation for all three images. The results show that providing metadata from retrieved RIS links and instructing the companion motivate and help users to revise their initial judgments in a positive direction. After RIS and the companion instructions, one in four participants changed their judgment at least once for the provided social media post (6/24 votes).

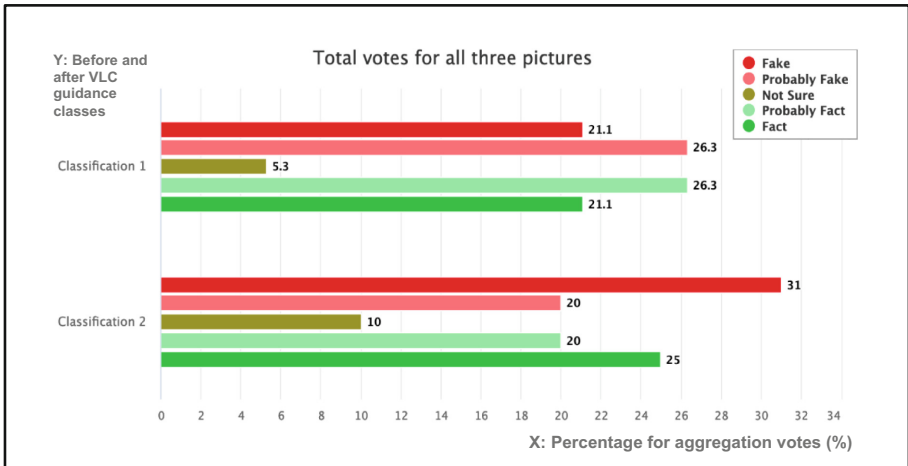


Fig. 4. Distribution of the votes for all three images before and after the guidance by the companion (classification 1, 2)

The above chart (Fig. 4), shows all votes aggregated over the three images by labels on a five-point scale. Classification 1 represents the votes before seeing and possibly taking into account the suggestions based on RIS and keyword extraction. Classification 2 corresponds to the judgements after provision of this information. One observation is that the deterministic classifications as a clear “Fake” or “Fact” have increased, but this is also true for the neutral category “Not Sure”. The previously known expert judgements for the three examples were indeed one “Fake”, one “Fact” and one “Not Sure”. In addition, we have checked the individual votes and changes (per anonymous user-id): Five user judgements were changed after having received the additional information. Using a numerical scale between -2 and $+2$ (i.e., -2 for “Fake” and $+1$ for “Probably Fact”), we could characterize these changes as increasing or decreasing the difference of the individual judgements to the expert judgement per item. Most of the changes (namely 3) led to decreasing the difference. Indeed, all these cases of “positive” changes resulted in the final judgement matching the expert judgement. There were 7 such matches in the initial classification 1 and 10 in the final classification 2.

The item with the least number of coincidences of user judgements with expert ratings was the item that had been classified as “Not Sure” by the experts. There was only one initial match (which was not changed) and additional one in a final judgement

(change from “Probably Fake” to “Not sure”). In the dialog with a companion, this user commented: “There was some evidence that this is probably not fake - I think so.” Another commented that the additional information was helpful to rethink the judgement more accurately. This user changed had the vote from “Probably Fake” to “Fake” on the item that was characterized as “Fake” in the expert judgment. The user explained this as follows: “Because the image’s caption is completely different from other resources. It seems that the image for this post is randomly picked from the web”.

6 Conclusion and Outlook

We have introduced a learning environment that consists of a virtual learning companion hooked up on an Instagram-like social media platform. The companion is implemented as a browser plugin that communicates as a chatbot on top of the social network. This companion interacts with learners via chat and triggers the user’s action with the environment artifacts. The companion exploits multimedia content (images) to provide adaptive feedback to learners and intervene with a pedagogical reflection.

This tool aims to motivate learners to sharpen their critical thinking and ask them to review other information items using similar (or the same) images in different contexts, rather than improving their prejudice (misinformation inoculation) in combating fake news. The companion application is equipped with a reverse image search tool that lists the most relevant websites with similar images to achieve this goal. Basic NLP techniques are applied to the textual content of the alternative sources to extract important keywords based on term frequency and filtering sentences according to the predefined dictionary. The metadata (keywords and phrases) extracted from the alternative sources are displayed in the application’s sidebar to help the learner in making their judgments.

An informal test with a small ad-hoc user group has confirmed the usability and practicability of the scenario. The test results indicate that delivering contextual metadata from retrieved RIS links and instruction from the companion motivates learners to revising and improving their initial judgments.

In the on-going and future development of the described learning scenario, we plan to include machine learning support in different ways: (1) Enhancing the chat interaction using WIT.AI for adaptive recognition of intents, and (2) using trained classifiers in the analysis of textual content around images to improve the identification and extraction of relevant keywords and phrases. Finally, we intend to apply the companion also to open social media platforms such as Instagram. This would not make a technical difference for the VLC as a browser plug-in, but it would require pedagogical control mechanisms to ensure responsible handling of information access. Additionally, we intend to enable social interaction between peers and of peers with moderators. By interacting with the same metadata for the shared content with images, the peers and moderators will be able to share their opinions about controversial content and present their different reasons and backgrounds.

Acknowledgement. This work was partially funded by the Volkswagen Stiftung in the line “AI and Future of Societies”.

References

1. Shearer, E., Gottfried, J.: News use across social media platforms. Pew Research Center (2017). <https://www.pewresearch.org/journalism/2017/09/07/news-use-across-social-media-platforms-2017/>
2. Silverman (BuzzFeed News Media Editor), Singer-Vine, Jeremy (Reporter), B.N.: Most Americans who see fake news believe it, new survey says. *BuzzFeed News* **6**(2) (2016)
3. Rubin, V.L.: Disinformation and misinformation triangle: a conceptual model for “fake news” epidemic, causal factors, and interventions. *J. Doc.* **75**, 1013–1034 (2019). <https://doi.org/10.1108/JD-12-2018-0209/FULL/XML>
4. Rubin, V., Brogly, C., Conroy, N., Chen, Y., Cornwell, S.E., Asubiaro, T. v.: litrl/litrl_code: Litrl Browser Experimental 0.14.0.0 Public (2019). <https://doi.org/10.5281/ZENODO.2588566>
5. Chen, Y., Conroy, N.J., Rubin, V.L.: News in an online world: the need for an “automatic crap detector.” *Proc. Assoc. Inf. Sci. Technol.* **52**, 1–4 (2015). <https://doi.org/10.1002/PRA2.2015.145052010081>
6. Nguyen, T.T., et al.: Deep learning for deepfakes creation and detection: a survey. <https://arxiv.org/pdf/1909.11573.pdf>. Accessed 15 Dec 2022
7. McDonald, J.D., Levine-Clark, M.: Encyclopedia of library and information sciences. *Am. Libr. Assoc.* 67–84 (2017). <https://doi.org/10.1081/E-ELIS4-11>
8. Basol, M., Roozenbeek, J., van der Linden, S.: Good news about bad news: gamified inoculation boosts confidence and cognitive immunity against fake news. *J. Cogn.* **3**, 1–9 (2020). <https://doi.org/10.5334/JOC.91/METRICS/>
9. de Jans, S., Hudders, L., Herrewijn, L., van Geit, K., Cauberghe, V.: Serious games going beyond the call of duty: Impact of an advertising literacy mini-game platform on adolescents’ motivational outcomes through user experiences and learning outcomes. *Cyberpsychology (Brno)* **13** (2019). <https://doi.org/10.5817/CP2019-2-3>
10. Titley, G., Keen, E., Földi, L.: Starting points for combating hate speech online. <https://edoc.coe.int/en/fundamental-freedoms/6478-starting-points-for-combating-hate-speech-online.html>. Accessed 28 Feb 2022
11. Chou, C.Y., Chan, T.W., Lin, C.J.: Redefining the learning companion: The past, present, and future of educational agents. *Comput. Educ.* **40**, 255–269 (2003). [https://doi.org/10.1016/S0360-1315\(02\)00130-6](https://doi.org/10.1016/S0360-1315(02)00130-6)
12. Chi, M.T.H., Bassok, M., Lewis, M.W., Reimann, P., Glaser, R.: Self-explanations: how students study and use examples in learning to solve problems. *Cogn. Sci.* **13**, 145–182 (1989). [https://doi.org/10.1016/0364-0213\(89\)90002-5](https://doi.org/10.1016/0364-0213(89)90002-5)
13. Aleven, V., Beal, C.R., Graesser, A.C.: Introduction to the special issue on advanced learning technologies. *J. Educ. Psychol.* **105**, 929–931 (2013). <https://doi.org/10.1037/A0034155>
14. Katz, S., Albacete, P., Chounta, I.-A., Jordan, P., McLaren, B.M., Zapata-Rivera, D.: Linking dialogue with student modelling to create an adaptive tutoring system for conceptual physics. *Int. J. Artif. Intell. Educ.* **31**(3), 397–445 (2021)
15. Goodman, B., Linton, F., Gaimari, R.: Encouraging student reflection and articulation using a learning companion: a commentary. *Int. J. Artif. Intell. Educ.* **26**(1), 474–488 (2015). <https://doi.org/10.1007/s40593-015-0041-4>
16. Hietala, P., Niemirepo, T.: The competence of learning companion agents. *Int. J. Artif. Intell. Educ.* **9**, 178–192 (1998)



A Theory Based Adaptive Pedagogical Agent in a Reading App for Primary Students - A User Study

Anna Riedmann^(✉), Philipp Schaper, Benedikt Jakob, and Birgit Lugin

Human-Computer-Interaction, University of Wuerzburg, Wuerzburg, Germany
anna.riedmann@uni-wuerzburg.de

Abstract. The Covid19 pandemic situation stressed the importance of appropriate digital applications that foster knowledge acquisition and can be used independently. This requires empirically validated applications, adapted to and evaluated within their target group. We therefore developed a theory based mobile app for training reading skills including two games and a pedagogical agent. The app is based on a validated analogue reading intervention for second-graders, which has already been partly digitized, aiming to address the gap of theory based educational applications for children. We used the ARCS model to modify the application and especially the agent's behavior and compared this extended version to a control version in a user study with primary school students, investigating effects on motivation and performance. The children were motivated in both conditions and performed significantly better in one of the two games in the ARCS modeled version. The app and the pedagogical agent were well perceived and could be successfully used in the home study setting without external aid by all children, which supports the applicability for applied settings.

Keywords: Mobile learning · Primary education · Pedagogical agent

1 Introduction

In recent years, digital learning aids have served as an additional tool to intensify the content learned at school [42, 47]. Especially in times of the Covid19 pandemic, which often results in alternating and distance learning, the transfer of knowledge is not an easy undertaking for all parties involved, whether students, parents or teachers. Educational mobile apps can serve as a useful alternative to traditional instructional methods and might assist learners when forced to practice at home, facilitating learning in an anywhere-anytime approach [5]. These apps can also provide a useful tool for adaptive interventions, for example in the field of reading development (e.g. [13, 34]). Especially with a view to a meaningful

Funded by the German Federal Ministry of Education and Research (BMBF) under the grant agreement MobiLe (03VP07080).

extension and supplementation of regular school lessons, validated and methodologically correct approaches are required [10]. Because knowledge acquisition often requires the ability to read [27], it is critical to address problems concerning reading development as early as possible, which might otherwise persist into adulthood [19]. To support children with reading deficits, adequate educational mobile apps might allow for interventions outside of the classroom in line with the current shift to digital learning [30]. Although mobile apps can benefit the acquisition of knowledge of young learners, there is a lack of methodologically rigorously conceptualized applications for children [10] following design approaches based on established learning theories [12,47]. To address such a theory based perspective in a digital reading intervention we conceptualized a pedagogical agent (PA) according to the taxonomies of the ARCS model [16], an instructional model consisting of four categories: Attention (A), Relevance (R), Confidence (C), and Satisfaction (S); ARCS. We integrated the PA into a subset of an evaluated prototype of a digitized intervention [33,36] based on an empirically validated analogue reading intervention for German second-graders [26]. We conducted a user study to evaluate the application within the target group, comparing it against a control version.

2 Theoretical Background and Related Work

2.1 Theory on Motivation in Learning

For adult as well as young learners, motivation, comprising all aspects of human intentions and activities [35], is critical to complete tasks and directly related to the quality of learning [3]. Therefore, it is important to identify central aspects that affect learners' motivation. For this purpose, Keller [17] developed the ARCS model of motivation to investigate why people actually learn something [16,17]. The author initially defined four basic categories as fundamental components to create an exciting, meaningful, and pleasantly challenging learning experience: attention, relevance, confidence, and satisfaction [16].

Each of the four components must be met for people to become and remain motivated [17]. Several areas of psychological research are summarized in those categories [17], and the model is often applied as a problem-solving strategy and has already been studied in more than 50 different countries around the world, where the four main categories have been reliably transferred to different environments [40]. The ARCS model serves as a systematic approach to identify learners' motivational demands and it suggests appropriate improvements in the given context, thereby enhancing students' motivation and learning performance [18].

Attention is a fundamental element of motivation and a prerequisite for learning [17]. Initially, getting attention might be easy, however, maintaining the learner's attention can be a difficult task [17]. The overall goal is to develop a balance between relaxation (boredom or indifference) versus tension (hyperactivity or anxiety) [17].

Students' individual sense of *relevance* depends on whether the task at hand is perceived as personally important to the learner, or whether a desired goal is associated with the learning content [16]. Instructional activities can be presented as relevant by linking current problems with already acquired knowledge and highlight the prospective benefit of the learning content [17].

Confidence describes the perceived likelihood of success and the extent to which learners feel in control of success [16]. To reduce the students' fear of failure, it is important to foster learners' confidence in their own performance by highlighting that success usually results from appropriate effort [17]. In the learning process, confidence can be supported by dynamic difficulty adjustment of the learning environment and constructive feedback addressing the learner's performance specifically [17].

Satisfaction ensures that the desire to continue learning is maintained [22]. Satisfaction can be defined as all states that are conducive to people feeling good while engaged in a particular activity [17]. To support the learners' satisfaction, rewards in the learning environment should be planned thoroughly, performance should be acknowledged immediately and negative influences, such as threats, should be avoided [17].

Table 1 displays all four components of the ARCS model and the corresponding strategies as proposed by Keller [17]. The application of individual subcomponents for the current study is further examined in Sect. 4.

Table 1. Teaching strategies according to ARCS model as proposed by Keller [17].

Attention	Relevance
Incongruity, Conflict (A1)	Experience (R1)
Concreteness (A2)	Present Worth (R2)
Variability (A3)	Future Usefulness (R3)
Humor (A4)	Need Matching (R4)
Inquiry (A5)	Modeling (R5)
Participation (A6)	Choice (R6)
Confidence	Satisfaction
Learning Requirements (C1)	Natural Consequences (S1)
Difficulty (C2)	Unexpected Rewards (S2)
Expectations (C3)	Positive Outcomes (S3)
Attributions (C4)	Negative Influences (S4)
Self-Confidence (C5)	Scheduling (S5)

2.2 Educational Mobile Apps

A review by Haßler et al. [10] indicates that mobile apps appear to be suitable for knowledge acquisition of young learners. The need for their application and technology-enhanced learning in general has become increasingly important in times of the Covid19 pandemic [30]. Educational mobile applications have already been used to introduce a variety of topics to different target groups, e.g. language acquisition for children [44, 45] and science for elementary students [41], indicating positive effects [37, 41, 42, 45]. The integration of mobile applications in the classroom offers motivational potential and might increase learners' motivation [42], especially when compared with traditional instruction methods, such as teaching in the classroom without technological support [41]. Mobile applications have also been implemented according to ARCS taxonomies [17], resulting in beneficial effects on motivation and learning [20, 46, 48].

2.3 Learning to Read with Technology Support

The ability to read is of critical importance in a literate society. Texts transfer social knowledge and allow interpersonal communication [1]. However, about four to five percent of German children face difficulties during the reading development phase [6]. In this case, early intervention is critical, because reading deficits of young readers can persist into adulthood [19]. Technological tools might improve reading performance and support the learners' motivation (e.g. [9, 28]). Trabandt [43] also emphasizes the quantifiable potential of using new media such as tablets to increase children's learning success. In a review by Jamshidifarsani et al. [13], the authors recommend utilizing digital applications and their associated benefits to facilitate reading acquisition.

Previous work has already successfully tested the benefits of reading apps for the reading development of primary school children. For example, Redcay et al. [31] demonstrated a positive effect on reading accuracy and fluency for second-graders using a learning app. In another study, Lenhard et al. [21] found that students from a higher grade level also performed better in terms of reading comprehension with digital support than the comparison group. Further, a study by Görden et al. [9] indicated that a digital reading intervention independently carried out by primary school children can benefit their reading skills. Therefore it can be assumed that digital reading interventions might be useful to support early reading development.

2.4 Pedagogical Agents

A pedagogical agent can be considered an autonomous, virtual character in a learning environment, supporting the learning process [23]. Such a computer-based system in the role of a teacher, tutor or learner communicates and interacts with learners in everyday language and might embody a distinct personality and adapt its instructional strategies to the behavior of the learner [23,29]. Main tasks comprise the presentation of learning content and interacting with learners [23,29]. For this purpose, PAs use various interaction possibilities, such as expressing emotions using facial expressions or gestures, providing nonverbal feedback, guiding attention and adapting to the learner during interaction [15]. A PA can be considered intelligent when integrating adaptive features, often powered by artificial intelligence (AI) [2].

The results of a meta-analysis by Schroeder et al. [38] suggest that applications involving a PA might be superior to systems that lack such an agent, and that the PA might influence learning in a positive way. Additionally, PAs also hold motivational potential [24] and can result in a positive perception of the learning experience by the learners [29].

3 Contribution

As noted previously, there is a lack of methodological rigorous approaches investigating educational mobile applications using randomized control trials. Additionally, these approaches are often not based on established learning theories [12,47]. We address this in our research by using a theory based approach which we evaluate within the target group. We applied the taxonomies described by Keller [17] to the behavior of a PA in order to enhance learner motivation and performance within an educational mobile application prototype by Schaper et al. [36]. Subsequently, we conducted a user study to compare the ARCS model adjusted version (ARCS model condition) with a control group. Our contribution is twofold: 1) We followed a systematic, theory based approach to develop an educational mobile application with a PA and 2) we evaluated the application within the target group of primary students comparing it to a control version.

Based on the predictions for adjustments based on the ARCS model [17] we expect higher motivation in the ARCS model condition (H1a). Additionally, we hypothesize higher attention (H1b), higher relevance (H1c), higher confidence (H1d) and higher satisfaction (H1e) in the ARCS model condition.

We expect participants in the ARCS model condition to perform better in the two implemented games (H2a and H2b) relative to participants in the control version, assuming that motivation directly relates to the quality of learning [3].

4 Learning Environment

We implemented two separate learning environments sharing the same basic structure, with one serving as the control group and the other being modeled according to the ARCS model [17] (ARCS model condition). The developed educational mobile applications are in terms of content based on an analogue reading intervention for second-graders [26], aiming for increased reading accuracy and speed during the word recognition process [26]. The intervention has already been partially digitized in the form of a prototype and includes a PA, which primarily has a supporting role in the prototype [33,36], but whose behavior is not based on a motivational model.

We chose two games of the digitized intervention based on their ability to cover relevant aspects of the analogue training and adapted them according to the taxonomy of strategies of the ARCS model [17] for the ARCS model condition. For this purpose, we refer to the systematic model for motivational design of instruction proposed by Keller [18] comprising of ten steps, applied in the subsequent section: (1) Obtain course information, (2) obtain audience information, (3) analyze the audience, (4) analyze existing materials, (5) list objectives and assessments, (6) list potential tactics, (7) select and design tactics, (8) integrate with instruction, (9) select and develop materials, and (10) evaluate and revise. We followed this design approach to implement the ARCS model as effectively as possible into the app. Additionally, we aimed to avoid the integration of unnecessary motivational tactics that could be more irritating than helpful [18]. The focus was on the behavioral characteristics of the PA, but other components of the application have also been adapted to promote motivation. The control condition includes the same educational games integrated in the same order. The following section elaborates the concept of the ARCS model condition, referencing the teaching strategies listed in Table 1.

4.1 Application and Target Group

Obtaining Course Information (1) and Analyzing the Target Group (2, 3). The application targets second-graders who are usually between seven and nine years old. Children read at least 72% of the words presented to them without error by the end of first grade. However, the development of word reading fluency varies and might increase as reading development progresses [19], requiring early intervention. This particular form of course should specifically address the deficient reading abilities [25]. A digital reading intervention can be used regardless of time and location providing a different approach to promote reading abilities. Further, Schaper et al. [36] demonstrated that the analogue intervention can be adapted as a digital version children enjoy engaging with.

Analyzing Existing Materials (4). For the application, two educational games of the digital adaption [36] of the analogue reading intervention [26] were chosen for implementation. These two games cover central aspects of the analogue intervention, such as hyphenation and vowel identification and thus address

reading deficits both with regard to assuring phoneme-grapheme associations and the acquisition of syllable rhythm and syllable structure.

When opening the app for the first time, a splash screen appears which in turn leads, once it is clicked, to an introductory video of the PA Uli, the owl. The gender-neutral name Uli was chosen to prevent associations to a certain gender. The owl introduces itself and briefly explains the course of the training.

Afterwards, users work on the two games, each introduced by a corresponding tutorial video explaining the core features of the task. In the first game called ‘Bee Flight’ students have to correctly recognize syllables of individual words by drawing syllable arcs under the displayed word using the device’s touch input (see Fig. 1, left). ‘Sailor Game’ is the second selected game and comprises the task of identifying vowels of a displayed word by touching them (see Fig. 1, right). Both games include 15 word items each. The learning environment ends with a short video in which the PA positively acknowledges the user’s performance.

By addressing different aspects of reading ability, the games aim to intensify skills that have already been learned in class (R1). Furthermore, the PA briefly explains the task in a tutorial setting by working on an example item. This allows the learners to grasp the task in a low-risk environment and is intended to increase the children’s confidence in themselves (C5). In addition, students receive immediate feedback after completing a word item. In order to motivate the primary school children and to prevent negative emotions, such as frustration, positive feedback is always given, be it praise when an item is successfully completed or encouraging words when it is not (S3).

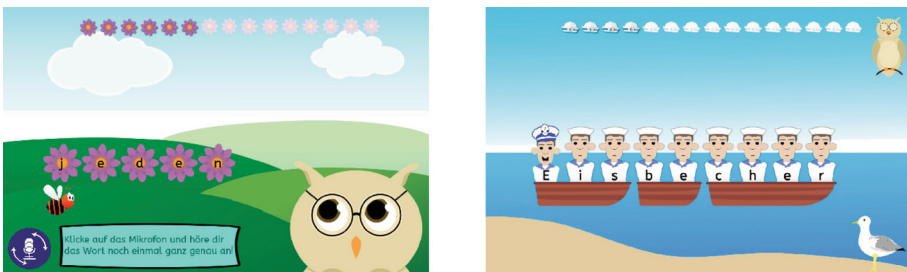


Fig. 1. Both games of the application with PA showing different emotional expressions and progress bar on top of the screen (Left: ‘Bee Flight’ with Uli, the owl, recommending to listen to the word that was incorrectly solved. Right: ‘Sailor Game’ with agent showing one of several varying facial expressions due to correctly solved task.).

4.2 Modeling of Agent Behavior and App Design According to ARCS Model

Listing Objectives, Assessments and Potential Strategies (5, 6). For the ARCS model version of the app, we adapted the initial app prototype [33,36] with a focus on the PA using the ARCS model [17] to create a motivational

design which aims to motivate learners and also increase their learning performance [7]. The resulting PA integrates adaptive behavior patterns based on a comprehensive decision algorithm and does not resort to AI methods, making it a borderline case for the definition of intelligent as defined by Clarebout et al. [2]. However, the PA can be considered intelligent due to its adaptive functionalities, refraining from the use of AI methods to model its behavior as traceable as possible according to the ARCS teaching strategies.

Selecting and Designing Strategies and Materials (7, 9). The agent introduces itself in an introductory video and explains the task with the aim of creating the impression that the subsequent tasks will be solved together to initiate social context (see Self-Determination Theory, [4]) (R4). On finishing the training, the agent praises the user's effort (S1).

In addition to Uli, the owl, acting as the main PA, the players are also introduced to Barbara, the bee, and Max, the seagull, by means of an introductory video (R4). These videos are part of a plot wrapping the app's content and aim to stress the importance of engaging with these characters by playing the educational games (R2) as part of the storyline of the app. The additional integration of humorous elements into the videos, e.g. funny character behavior, provides learners with incentives to develop an interest in the tasks (A4, A6). At the end of the games, the characters also express their gratitude and acknowledge the player's effort (C4) to increase the level of satisfaction (S2).

To keep students' motivation as high as possible, Keller [17] suggests that the difficulty of individual tasks should be adjusted to the users' abilities (C2) to avoid children feeling under-challenged nor over-challenged [49]. Thus, the different word items were organized into three different levels of difficulty (cf. [26]). Learners start at the first level and progress through the levels after successfully completing several items in the respective game. Accordingly, the level of difficulty decreases if the user makes multiple mistakes. For the control version, the difficulty remains at the second level.

The player receives brief feedback immediately after processing each action. To sustain the learner's motivation, feedback is always phrased in a positive way, i.e. either praising or encouraging (S3). While the feedback in the control version only varies in terms of auditory content, in the ARCS model condition, the PA's non-verbal presentation, e.g. facial expression, also changes in relation to the given feedback to increase the attention of the player (A3; see Fig. 1). Keller [17] also stresses the importance of directly supporting the learner by providing informative and helpful feedback (S3). Hence, all word items are categorized in different groups depending on which reading peculiarities the respective word item possesses (e.g. special characters that are common in German, such as β). That way, users receive word-specific help from the PA.

After completing both games, pupils are presented with their individual game statistics, displaying information about their top performances in child-friendly design (fastest correctly completed word item; largest series of correct items). The statistics should assist the students in assessing their individual performance

(C5). The PA and the characters of the app shown next to the statistics are supposed to convey the feeling of social inclusion to the players (R4).

Integrate with Instruction (8) and Evaluate (10). The training in both the ARCS model and control version has been implemented as an Android app in Android Studio [8]. In the following sections, the app modeled according to the ARCS taxonomies (see Sect. 4) is compared against the control version and evaluated on primary students.

5 User Study

5.1 Participants

29 primary school children with written consent by their legal guardians participated. Four had to be excluded due to incomplete data, resulting in a final sample of $N = 25$ participants with a mean age of 7.84 ($SD = 0.47$). For the analysis of the need satisfaction questionnaire specifically, additional nine children had to be excluded due to missing data only in this specific questionnaire. Participants were randomly allocated to the ARCS model condition ($n = 13$) and the control condition ($n = 12$).

5.2 Measures

Motivation. For all questionnaires used, we adapted the formulation of the items to fit the target group in consultation with their teacher of blinded school.

We assessed the instructional materials using the Reduced Instructional Materials Motivation Survey (RIMMS) [22]. The questionnaire's subscales correspond to the four categories of the ARCS model, each comprising three items measured on a five-point Likert scale: *Attention*, e.g. "I liked that there were different tasks." ($\alpha = .49$), *Relevance*, e.g. "It helps me if I can read well." ($\alpha = .42$), *Confidence*, e.g. "I was sure from the beginning that I would manage the tasks." ($\alpha = -.02$), and *Satisfaction*, e.g. "I had a lot of fun playing." ($\alpha = .48$). An overall score across all scales measures motivation [22] ($\alpha = .74$).

We also covered the current achievement motivation of learners with a focus on their motivational and expectation-related attitudes towards the learning environment for a more comprehensive insight into the motivation of the participants during learning and performance situations. We used the Questionnaire on Current Motivation (QCM) [32] to assess the current achievement motivation of the participants. It includes four subscales (18 items in total, seven-point Likert scale): (1) *Anxiety*, e.g. "I was a little afraid of these tasks." ($\alpha = .48$), (2) *Interest*, e.g. "I like this kind of riddles and puzzles." ($\alpha = .21$), (3) *Probability of Success*, e.g. "I think I was able to solve the tasks well." ($\alpha = -.03$), and (4) *Challenge*, e.g. "I had to put a lot of effort into the tasks." ($\alpha = .02$).

Performance and Additional Measures. We assessed the children’s performance in both games using the number of correct answers given. We also assessed need satisfaction for competence ($\alpha = -.57$), autonomy ($\alpha = -.21$) and relatedness ($\alpha = .69$) with an adapted version of the questionnaire by Sheldon et al. [39], including two items per subscale on a five-point Likert scale. Finally, children were presented with five questions regarding their attitude towards the PA, e.g. “I liked Uli, the owl.”, and six statements concerning usability features of the application, e.g. “Operating the application works.”.

5.3 Procedure

Due to the Covid19 pandemic, we conducted the evaluation of the application as a home study. Before the start of the experiment, parents were offered to participate in an information event about the planned study. Additionally, they were informed about the study by means of an information letter and gave written consent. The participating children randomly received a tablet for one day with the pre-installed application based on condition, as well as the questionnaires which were all distributed by their teacher. They worked on the learning app by themselves during the same day at home. Afterwards, they were instructed to fill out the questionnaire. Working on the app and the subsequent questionnaire lasted about 20 min. On the next day, the tablets and the questionnaire were collected in a sealed envelope. Throughout the survey, the investigator was available by phone and e-mail to answer potential questions, which, however, was not used by parents.

6 Results

All analyses were conducted with JASP [14] and alpha was set at .05. Due to the small sample size, we used Mann-Whitney U tests for group comparisons. Table 2 displays all descriptive values and statistical tests.

Motivation. For H1a, we found no significantly higher motivation in the ARCS model condition. With a view to the subscales, there was also no significantly higher attention (H1b), relevance (H1c), confidence (H1d) or satisfaction (H1e) in the ARCS model version.

For the current achievement motivation of the learners there were no significant differences between both conditions, neither for anxiety, interest, probability of success or challenge.

Table 2. Questionnaire means with SD in parentheses. Results of U tests.

	Scale	ARCS model	Control	<i>U</i>	<i>p</i>
Motivation (overall score)		4.52 (0.58)	4.49 (0.33)	99.00	.256
Attention	1–7	4.62 (0.58)	4.61 (0.51)	81.00	.885
Relevance	1–7	4.56 (0.76)	4.56 (0.48)	92.00	.437
Confidence	1–7	4.49 (0.54)	4.44 (0.59)	81.50	.866
Satisfaction	1–7	4.41 (0.66)	4.36 (0.78)	79.00	.978
Anxiety	1–7	4.06 (1.02)	3.77 (0.98)	91.00	.494
Interest	1–7	6.37 (0.74)	6.02 (0.62)	102.50	.185
Probability of success	1–7	5.85 (0.83)	6.21 (0.72)	59.50	.322
Challenge	1–7	5.50 (0.78)	5.54 (0.80)	73.50	.817
Number of correct answers in ‘Bee Flight’		31.62 (6.38)	29.33 (4.64)	88.50	.584
Number of correct answers in ‘Sailor Game’		55.69 (6.03)	43.67 (3.94)	144.00	<.001
Competence (<i>n</i> = 16)	1–5	3.65 (0.67)	3.58 (0.38)	30.00	1
Autonomy (<i>n</i> = 16)	1–5	4.10 (0.88)	3.92 (0.80)	36.00	.537
Relatedness (<i>n</i> = 16)	1–5	4.50 (0.82)	3.75 (1.17)	45.50	.072
I learned something	1–5	4.15 (0.99)	4.00 (1.48)	78.50	1
I knew what I had to do	1–5	4.39 (1.04)	4.67 (0.65)	70.50	.630
Operating the application works	1–5	4.77 (0.60)	4.25 (0.87)	105.00	.080
While playing, the time passed quickly	1–5	4.62 (0.65)	4.17 (1.12)	92.50	.373
I liked the way the game looks	1–5	4.39 (0.65)	4.33 (1.07)	71.00	.693
If I make a mistake in front of the class, I feel insecure.	1–5	3.35 (1.21)	2.92 (1.17)	91.00	.475
I liked Uli, the owl	1–5	4.85 (0.38)	4.75 (0.45)	85.50	.583
Uli, the owl, helped me to learn	1–5	4.00 (0.91)	4.00 (1.04)	76.50	.954
It was good that Uli, the owl, was there	1–5	4.46 (0.66)	4.50 (0.67)	75.00	.878
I think Uli, the owl, is smart	1–5	4.46 (0.78)	4.58 (0.79)	69.50	.597
I didn’t like making mistakes in front of Uli, the owl.	1–5	3.46 (1.66)	2.00 (1.41)	116.00	.033

Performance. Participants showed no significantly better performance in the game ‘Bee Flight’ in the ARCS model version compared to the control version (H2a). However, we found significantly better performance of participants playing ‘Sailor Game’ in the ARCS model condition (H2b).

Additional Measures. We found no significantly higher competence, autonomy and relatedness in the ARCS model version. The questions assessing the attitude towards the PA and usability features of the application serve as exploratory measures and should be interpreted carefully (for all values, see Table 2).

7 Discussion

In terms of motivation, learners did not demonstrate significantly higher values in the ARCS model version when compared to the control condition, thus, we reject H1a. The manipulation of the behavior of the PA and overall app structure according to the taxonomy of strategies of the ARCS model [17] did not result in significantly higher values for attention, relevance, confidence or satisfaction in the ARCS model version, rejecting H1b-e. The results for additionally assessed achievement motivation reflect these findings.

Participants did not perform significantly better in the game ‘Bee Flight’ in the ARCS model version, thus rejecting H2a. However, learners working on the app in the ARCS model version performed significantly better in ‘Sailor Game’ than those who used the control version. We therefore accept H2b.

Regarding additional measures, learners did not demonstrate significantly higher competence, autonomy or relatedness in the ARCS model version. Interestingly, especially relatedness is descriptively higher in the condition with the ARCS modeled behavior for the PA. This corresponds to the slightly higher descriptive values for children in the ARCS model version stating that they liked the PA, and more explicitly to the clearly higher descriptive values for the subjective feeling of discomfort of the children when making mistakes in front of Uli, the owl. It seems that the children did indeed perceive the agent as somehow relevant, yet the PA seemed to fail building a safe learning environment for children allowing for mistakes. Additionally, the agent’s opinion seemed important to the children and they potentially aimed for impressing it, resulting in unpleasantness in the case of failure. However, probably due to missing data for this adapted version of the need satisfaction questionnaire [39] we found no indication on differences in relatedness.

With a view to additional measured subjective impressions regarding the PA, there seemed to be no clear differences between both conditions. However, it is noticeable that all mean values are at the upper end of the scale, except the question about making mistakes in front of the PA, indicating that the PA was well perceived in both conditions. However, all statements regarding the app’s usability features and the participants’ attitude towards the PA were only intended to capture the general impression of the participants.

Considering the motivational outcomes of our educational mobile application, our results did not demonstrate any notable differences between both conditions, indicating that both versions of the application might have motivated the children to the same extent. This might be due to the structure and concept of the learning app version used in the control condition which already has motivational potential and integrates several components promoting motivation (e.g. reinforcing feedback). Further, this version also integrates a PA, with less complex behavioral patterns, yet still as a potential motivational component of the application. The low dropout rate (four children), even within the home study setting, also indicates that the app was able to motivate the children sufficiently to work on it in both versions, despite other school tasks.

However, our learning environment in the ARCS model version seems to benefit the performance of learners. This might be due to the informative and helpful feedback on specific word items provided by the PA and might have helped the children to improve over the course of the learning environment. It further seems that working on the app is an effective opportunity to practice reading skills in addition to regular homework which in turn might have resulted in better performance values.

7.1 Limitations and Future Work

It was not possible for us to conduct the experiment in a controlled classroom setting, which led to the home study setting described above and made standardization more difficult. Children worked on the learning application on their own within one day with no option to observe their behavior while interacting with the app by an instructor. Further, the potential influence of parents on the training process cannot be excluded, even though they have been instructed to this effect. However, based on internal app statistics, we were able to determine that all children had completed the learning environment in one run, as they were instructed beforehand. The above average motivation also indicates that the training can be carried out in a home setting. Most importantly, the successful deployment of the application in a home study setting indicates that children are able to use and navigate through the app on their own. Still, the lack of knowledge about the environment in which children worked on the educational application and potential distractions might be one of the major limitations of this study. The study setting, and size and complexity of the questionnaire might also be the reason why numerous entries in the questionnaire were missing, requiring to exclude children from the results which in turn might have contributed to the poor reliability scores.

Additionally, the intervention lasted only about 20 min, which could have led to that expected effects could not become apparent at all in combination with an overall small sample size. Thus, future work should replicate this study with an extended time period for working on the app, including a within-subjects design, multiple sessions and a larger sample size. Further, as noted in a literature review by Heidig et al. [11], extending the study design by including a control group without a PA might yield valuable indications on whether a PA-supplemented app generally exceeds an app without a PA.

8 Conclusion

In this study, we investigated the effect of an educational mobile application featuring a PA on motivation and performance of second-graders. The app and the agent's behavior have been modeled according to the ARCS model [17]. We conducted a user study and compared the ARCS modeled app version to a control version. We did not find significant differences regarding motivation, the children were relatively motivated in both conditions. Yet, the children performed

significantly better in one of the two games in the experimental version. The PA was well perceived and overall, the ARCS modeled app version seems to benefit children's performance while maintaining a high level of motivation.

References

1. Becker-Mrotzek, M., Lindauer, T., Pfof, M., Weis, M., Strohmaier, A., Reiss, K.: Lesekompetenz heute: eine schlüsselqualifikation im wandel. In: Reiss, K., Weis, M., Klieme, E., Köller, O. (eds.) PISA 2018, pp. 21–46. Waxmann Verlag GmbH (2019)
2. Clarebout, G., Heidig, S.: Pedagogical agents. In: Seel, N.M. (ed.) Encyclopedia of the sciences of learning, pp. 2567–2571. Springer reference, Springer, New York (2012). https://doi.org/10.1007/978-1-4419-1428-6_942
3. Deci, E.L., Ryan, R.M.: Intrinsic Motivation and Self-Determination in Human Behavior. Perspectives in social psychology, Plenum Press, New York (1985). <https://doi.org/10.1007/978-1-4899-2271-7>
4. Deci, E.L., Ryan, R.M.: The what and why of goal pursuits: human needs and the self-determination of behavior. *Psychol. Inq.* **11**(4), 227–268 (2000). https://doi.org/10.1207/S15327965PLI1104_01
5. Drigas, A., Angelidakis, P.: Mobile applications within education: an overview of application paradigms in specific categories. *Int. J. Interact. Mob. Technol. (iJIM)* **11**(4), 17 (2017). <https://doi.org/10.3991/ijim.v11i4.6589>
6. Fischbach, A., et al.: Prävalenz von lernschwächen und lernstörungen: Zur bedeutung der diagnosekriterien. *Lernen und Lernstörungen* **2**(2), 65–76 (2013). <https://doi.org/10.1024/2235-0977/a000035>
7. Goksu, I., Islam Bolat, Y.: Does the arcs motivational model affect students' achievement and motivation? a meta-analysis. *Rev. Educ.* **9**(1), 27–52 (2020). <https://doi.org/10.1002/rev3.3231>
8. Google LLC: Android studio (2021). <https://developer.android.com/studio>
9. Görgen, R., Huemer, S., Schulte-Körne, G., Moll, K.: Evaluation of a digital game-based reading training for German children with reading disorder. *Comput. Educ.* **150**, 103834 (2020). <https://doi.org/10.1016/j.compedu.2020.103834>
10. Haßler, B., Major, L., Hennessy, S.: Tablet use in schools: a critical review of the evidence for learning outcomes. *J. Comput. Assist. Learn.* **32**(2), 139–156 (2016). <https://doi.org/10.1111/jcal.12123>
11. Heidig, S., Clarebout, G.: Do pedagogical agents make a difference to student motivation and learning? *Educ. Res. Rev.* **6**(1), 27–54 (2011). <https://doi.org/10.1016/j.edurev.2010.07.004>
12. Herodotou, C.: Young children and tablets: a systematic review of effects on learning and development. *J. Comput. Assist. Learn.* **34**(1), 1–9 (2018). <https://doi.org/10.1111/jcal.12220>
13. Jamshidifarsani, H., Garbaya, S., Lim, T., Blazevic, P., Ritchie, J.M.: Technology-based reading intervention programs for elementary grades: an analytical review. *Comput. Educ.* **128**, 427–451 (2019). <https://doi.org/10.1016/j.compedu.2018.10.003>
14. JASP Team: Jasp (2021). <https://jasp-stats.org/>
15. Johnson, W.L., Lester, J.C.: Pedagogical agents: back to the future. *AI Mag.* **39**(2), 33–44 (2018). <https://doi.org/10.1609/aimag.v39i2.2793>



16. Keller, J.: Motivational design of instruction. In: Reigeluth, C.M. (ed.) *Instructional Design Theories and Models*, pp. 383–434. Taylor and Francis, Hoboken (1983)
17. Keller, J.M.: Development and use of the arcs model of instructional design. *J. Instr. Dev.* **10**(3), 2–10 (1987). <https://doi.org/10.1007/bf02905780>
18. Keller, J.M., Suzuki, K.: Learner motivation and e-learning design: a multinationalally validated process. *J. Educ. Media* **29**(3), 229–239 (2004). <https://doi.org/10.1080/1358165042000283084>
19. Landerl, K., Wimmer, H.: Development of word reading fluency and spelling in a consistent orthography: an 8-year follow-up. *J. Educ. Psychol.* **100**(1), 150–161 (2008). <https://doi.org/10.1037/0022-0663.100.1.150>
20. Laurens Arredondo, L.A., Valdés Riquelme, H.: M-learning adapted to the arcs model of motivation and applied to a kinematics course. *Comput. Appl. Eng. Educ.* (2021). <https://doi.org/10.1002/cae.22443>
21. Lenhard, W., Baier, H., Endlich, D., Schneider, W., Hoffmann, J.: Rethinking strategy instruction: direct reading strategy instruction versus computer-based guided practice. *J. Res. Reading* **36**(2), 223–240 (2013). <https://doi.org/10.1111/j.1467-9817.2011.01505.x>
22. Loorbach, N., Peters, O., Karreman, J., Steehouder, M.: Validation of the instructional materials motivation survey (IMMS) in a self-directed instructional setting aimed at working with technology. *Brit. J. Educ. Technol.* **46**(1), 204–218 (2015). <https://doi.org/10.1111/bjet.12138>
23. Martha, A.S.D., Santoso, H.: The design and impact of the pedagogical agent: A systematic literature review. *J. Educ. Online* **16**(1) (2019). <https://doi.org/10.9743/JEO.2019.16.1.8>
24. Moreno, R.: Multimedia learning with animated pedagogical agents. In: Mayer, R.E. (ed.) *The Cambridge handbook of multimedia learning*, pp. 507–524. Cambridge University Press, Cambridge (2005). <https://doi.org/10.1017/CBO9780511816819.032>
25. Müller, B., Otterbein-Gutsche, G., Richter, T.: Praxis psychologischer beratung (und intervention): Leseförderung mit silben und sprachsystematik. konzeption eines trainingsprogramms. *Psychologie in Erziehung und Unterricht* **65**(1), 52 (2018). <https://doi.org/10.2378/peu2018.art03d>
26. Müller, B., Richter, T., Otterbein-Gutsche, G.: Lesen mit Willy Wortbär: Ein silbenbasiertes Training zur Förderung der Worterkennung beim Lesen: [Reading with Willy Wortbär: A syllable-based intervention to foster visual word recognition]. Hogrefe, Göttingen, Germany (2020)
27. Nagler, T., Lindberg, S., Hasselhorn, M.: Leseentwicklung im grundschulalter. kognitive grundlagen und risikofaktoren. *Lernen und Lernstörungen* **7**(1), 33–44 (2017). <https://doi.org/10.1024/2235-0977/a000185>
28. Omheni, N., Kacem, A.H.: “i-Read”: a collaborative learning environment to support students with low reading abilities. In: Micarelli, A., Stamper, J., Panourgia, K. (eds.) *ITS 2016. LNCS*, vol. 9684, pp. 221–226. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-39583-8_21
29. Pérez-Marín, D., Pascual-Nieto, I.: An exploratory study on how children interact with pedagogic conversational agents. *Behav. Inf. Technol.* **32**(9), 955–964 (2013). <https://doi.org/10.1080/0144929X.2012.687774>
30. Pokhrel, S., Chhetri, R.: A literature review on impact of covid-19 pandemic on teaching and learning. *High. Educ. Future* **8**(1), 133–141 (2021). <https://doi.org/10.1177/2347631120983481>

31. Redcay, J.D., Preston, S.M.: Improving second grade student's reading fluency and comprehension using teacher-guided ipad app instruction. *Interact. Technol. Smart Educ.* **13**(3), 218–228 (2016). <https://doi.org/10.1108/itse-12-2015-0035>
32. Rheinberg, F., Vollmeyer, R., Burns, B.D.: Fam: Ein fragebogen zur erfassung aktueller motivation in lern- und leistungssituationen. *Diagnostica* **47**(2), 57–66 (2001). <https://doi.org/10.1026//0012-1924.47.2.57>
33. Riedmann, A., et al.: Iteratively digitizing an analogue syllable-based reading intervention. *Interact. Comput.* (2022). (in press) <https://doi.org/10.1093/iwc/iwac005>
34. Ronimus, M., Eklund, K., Pesu, L., Lyytinen, H.: Supporting struggling readers with digital game-based learning. *Educ. Technol. Res. Dev.* **67**(3), 639–663 (2019). <https://doi.org/10.1007/S11423-019-09658-3>
35. Ryan, R.M., Deci, E.L.: Self-determination theory and the facilitation of intrinsic motivation, social development, and well-being. *Am. Psychol.* **55**(1), 68–78 (2000). <https://doi.org/10.1037/0003-066X.55.1.68>
36. Schaper, P., et al.: Towards a digital syllable-based reading intervention: an interview study with second- graders. In: *British HCI Conference 2021 (BCS-HCI 2021)* (2021). <https://doi.org/10.14236/ewic/HCI2021.29>
37. Schoppek, W., Tulis, M.: Enhancing arithmetic and word-problem solving skills efficiently by individualized computer-assisted practice. *J. Educ. Res.* **103**(4), 239–252 (2010). <https://doi.org/10.1080/00220670903382962>
38. Schroeder, N.L., Adesope, O.O., Gilbert, R.B.: How effective are pedagogical agents for learning? a meta-analytic review. *J. Educ. Comput. Res.* **49**(1), 1–39 (2013). <https://doi.org/10.2190/EC.49.1.a>
39. Sheldon, K.M., Filak, V.: Manipulating autonomy, competence, and relatedness support in a game-learning context: new evidence that all three needs matter. *Br. J. Soc. Psychol.* **47**(Pt 2), 267–283 (2008). <https://doi.org/10.1348/014466607X238797>
40. Simsek, A.: Interview with john m. keller on motivational design of instruction. *Contemp. Educ. Technol.* **5**(1) (2014). <https://doi.org/10.30935/cedtech/6117>
41. Su, C.H., Cheng, C.H.: A mobile gamification learning system for improving the learning motivation and achievements. *J. Comput. Assist. Learn.* **31**(3), 268–286 (2015). <https://doi.org/10.1111/jcal.12088>
42. Tillmann, A., Bremer, C.: Einsatz von tablets in grundschulen. In: *Tablets in Schule und Unterricht*, pp. 241–276. Springer, Wiesbaden (2017). https://doi.org/10.1007/978-3-658-13809-7_11
43. Trabant, S.: Tablets in kindertagesstätten. *MedienPädagogik: Zeitschrift für Theorie und Praxis der Medienbildung*, pp. 1–15 (2019). <https://doi.org/10.21240/mpaed/00/2019.02.26.x>
44. Veenhof, G., Sandberg, J., Maris, M.: ZooQuest: a mobile game-based learning application for fifth graders. In: Cerri, S.A., Clancey, W.J., Papadourakis, G., Panourgia, K. (eds.) *ITS 2012. LNCS*, vol. 7315, pp. 687–688. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-30950-2_122
45. Walter-Laager, C., Brandenburg, K., Tinguely, L., Schwarz, J., Pfiffner, M.R., Moschner, B.: Media-assisted language learning for young children: effects of a word-learning app on the vocabulary acquisition of two-year-olds. *Br. J. Educ. Technol.* **48**(4), 1062–1072 (2017). <https://doi.org/10.1111/bjet.12472>
46. Wu, T.T.: Improving the effectiveness of English vocabulary review by integrating arcs with mobile game-based learning. *J. Comput. Assist. Learn.* **34**(3), 315–323 (2018). <https://doi.org/10.1111/jcal.12244>
47. Zhang, L., Nouri, J.: A systematic review of learning and teaching with tablets. In: *14th International Conference of Mobile Learning*, pp. 79–88 (2018)

48. Zhang, W.: Design a civil engineering micro-lecture platform based on the arcs model perspective. *Int. J. Emerg. Technol. Learn. (IJET)* **12**(01), 107 (2017). <https://doi.org/10.3991/ijet.v12i01.6487>
49. Zohaib, M.: Dynamic difficulty adjustment (DDA) in computer games: a review. *Adv. Hum. Comput. Interact.* **2018**, 1–12 (2018). <https://doi.org/10.1155/2018/5681652>



Intelligent Tutor for Designing Function Interface in a Programming Language

Dmitrii Litovkin¹, Anton Anikin^{1,2}(✉) , Kirill Kulyukin¹ ,
and Oleg Sychev¹ 

¹ Volgograd State Technical University, Volgograd, Russia
anton.anikin@vstu.ru

² Software Engineering School, Volgograd, Russia
info@seschool.ru

<https://vstu.ru>, <https://seschool.ru>

Abstract. Intelligent tutoring systems play an essential role in learning. In programming learning, the specificity of the learning process is related to creating code in a programming language and developing appropriate skills. One of the basic skills in code development is designing functions and their interfaces in a programming language. For these skills mastering using ITS, it is important to detect the student's mistakes early and provide formative explanatory feedback for the student to help them find and fix the errors. In this paper, we propose the intelligent approach to explanatory feedback generation for the task of function prototype creation training. We developed an approach to automatic teaching function design, a formal model of the subject domain based on OWL ontology and Jena rules to detect errors in the students' answers using software reasoners, and intelligent tutor based on the developed formal model.

Keywords: Intelligent tutoring systems · Programming learning · Ontology · Reasoning · Jena rules

1 Introduction

To master programming, it is necessary to experience not only theory but also coding, that can be achieved by creating many programs. However, it is not easy for teacher to prepare many programming tasks and corresponding coding problems. In programming learning, active learning is effective if the learner actively and interactively works on the subject. Active learning includes project-based learning and collaborative/cooperative learning [18].

On the other hand, feedback is one of the most powerful ways to enhance learning and increases student performance. It is an essential component of any intelligent tutoring system (ITS) and of scaffolding for learning [2, 9, 14]. ITS is a computer system that aims to provide immediate and customized instruction

The reported study was funded by RFBR, project number 20-07-00764.

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2022
S. Crossley and E. Popescu (Eds.): ITS 2022, LNCS 13284, pp. 293–302, 2022.
https://doi.org/10.1007/978-3-031-09680-8_27

or feedback to learners, usually without requiring intervention from a human teacher [14]. According to [15] an expert system is called an Intelligent Tutoring System if it simulates the individual learning process of a student and teacher by improving understanding and perception.

In this paper, we propose to automate the explanatory feedback generation for the task of function prototype creation training that is an important task in programming learning. We developed an approach to automating teaching function design, a formal model of the subject domain based on OWL ontology and Jena¹ rules that are able to detect errors in the students' answers using software reasoners, and intelligent tutor based on the developed formal model.

2 Related Work

The important task in ITS for programming learning is to provide the student with programming tasks and formative feedback for effective programming skills mastering. Some ITS for programming learning allow generating custom tasks using domain models and predefined templates for this domain [7], but most of them are based on predefined tasks only [1, 4, 13, 19]. The feedback in the different ITS provides pass/fail results, errors, and the last correct line in the code [19] or more complex task-oriented predefined hints [4] and hints generated using some rules and context [5, 12]. Most of the ITS use hardcoded domain models (like [4, 7]), but few systems (e.g., [12, 17]) use formal models to represent the domain laws and rules.

Solvelets [8] is a programming tutor for imperative programming in C++. It covers algorithm formulation, program development, and coding. The design steps corresponds to real programming practice. The student receives formative feedback at each stage. There are 3 types of questions: selection, location and coding. If the questions are answered incorrectly, the student receives appropriate hints (e.g., about incorrect data type used).

Some tools [3, 6] are focused on visualization of task, algorithm, code elements to help the student better understand these aspects of the task solving and to provide the feedback in the visual form as well. So, UUhistle [16] is a Python code visualizer with main idea to visualize important parts of the computer's memory (stack and stack frames) in a novice-friendly way. Unlike debuggers, UUhistle visualizes the actions of a line of code in details rather than treating a line of code as an atomic operation. UUhistle only parses syntactically correct code, as dynamic parsing occurs only at runtime.

To get feedback in the process of code writing (including solving educational problems), compilers messages, built-in development environment tools, and built-in debugging tools can be used. Such feedback allows to correct errors before the completion of work on the code, but is not always informative enough, does not always accurately indicate the cause of the error and how to locate and fix it, which in many cases makes it difficult for students with insufficient experience to effectively use this type of feedback.

¹ <https://jena.apache.org>.

Formal testing methods allow checking the correctness of the created program code with minimal feedback. Testing tools are well developed, but with unsuccessful testing, the benefit for the student is minimal (he does not know what the error is and how to fix it, feedback is limited to a set of tests on which the program does not work, further error correction depends on already acquired debugging skills and knowledge of the programming language and development principles).

Static code analysis and related tools allow detection errors of various kinds (including those that are not critical and not always detected during testing (depending on the level of code coverage by tests), often allow to provide informative feedback that allows to localize and fix the error². Tools that focus on static code analysis for early error detection are important in professional software development and in programming learning as well, but they does not takes into account the task formulation and domain and works with program code only.

For the last several years, there is a growing interest in applying machine learning models to tasks that process source code and the natural language elements, such as comment generation, code generation, method naming, and code completion [11]. So, e.g., in [10] a neural machine translation (NMT) based approach and Global Transformer-based Neural Model for method name suggestion is used, which considers the local context, the project-specific context, and the documentation of the method.

None of the considered tools and approaches to the best of our knowledge can provide sufficient informative feedback for the task of function prototype creation training for the early stages of program developing. The cause of this problem is that the grading of a function prototype requires not only usage of programming-language rules but also the task context in the form of specific subject-domain knowledge which is not available in existing systems.

3 Approach to Teaching Designing Function Interface

The key skill in procedural programming is isolating a subroutine as a part of the code (algorithm) and defining its interface (e.g., header or prototype). In the following examples, we will use the task of designing a function (subroutine implementation) in the C programming language.

Designing function interface requires performing the following tasks:

1. Identifying **data elements** that should be passed to and/or returned from the function. A data element is a set of data describing an entity in the target domain (see the example below).
2. Determining **directions** of the identified data elements: input, output, or changed (both input and output).
3. Determining the method of storing each data element in memory (e.g., a date can be implemented either as three separate variables or as a structure with three fields).

² <https://pvs-studio.com/en/>.

4. Designing meaningful function name (identifier).
5. Determining the function arguments and their order. For each argument the following characteristics should be defined using a programming language: name, data type, direction. In some programming languages, an argument also can have a default value.
6. Determining the function's return value and its data type.
7. Writing the subroutine interface using the target programming language.

We have developed an approach and tutoring application (intended for integration in a bigger intelligent tutoring system) that teaches designing function interface in six stages. It is aimed at developing of all the skills listed above except designing the function name (4) and determining the argument order, their names, and default values in (5). The process of designing a function in the developed application goes as follows.

Create a prototype of function `distance_between_dates` that:

Calculates the count of days between the birth date and **the first** date at school

You must select each data element from **the** text and click on the button.

The phrase you selected is too short. It describes the data element partly

Check data element

Data element

Count of days

Date of birth

Fig. 1. Stage 1: Identifying data elements

Stage 1. Identifying Data Elements. The student should identify the data passed to or returned from the function in the textual description of the function (a similar approach was used in the Solvelets intelligent tutor [8]). Given: subroutine's goal in natural language as sequence of tokens $TSL = \{tlex_k\}$. Required: the set of data elements $E = \{e_i\}$ that should be passed to and/or returned from the function. Each data element e_i is selected as sequence of tokens $ESL(e_i) = \{ellex_k\}$. The intersection of all $ESL(e_i)$ is empty. An example of the first stage³ is provided in Fig. 1. A student should repeatedly select the subsequence of tokens $stud_ESL$ corresponding to any of the required elements

³ The fullscreen figures for all the stages are available at: <https://github.com/Kirill34/PrototypeCreatingTutor>.

$E = \{e_i\}$. The student receive error messages for selecting too long or too short sequence of tokens, or selecting the sequence of tokens that do not belong to any data element. When all the data elements from $E = \{e_i\}$ are identified, the student moves to the next stage.

Stage 2. Determining the Directions of Data Elements. Given: set of data elements $E = \{e_i\}$. Every e_i has direction $dir(e_i)$: input, output or changed. Student must select direction for every data element e_i .

This stage is implemented as a matching question. If the student chooses a wrong data direction, a hint message about their error is shown. After all directions are determined correctly, the student moves to the next stage.

Stage 3. Choosing Implementation of Data Elements. Given: set of data elements $E = \{e_i\}$. Each e_i has a set of possible implementations (methods of storing the data element in memory) $ECDT(e_i)$. Student must choose one of possible implementation which will be used in the function prototype for e_i . Implementation can be number, character, string, collection of elements (e.g., arrays, lists, maps), object of structures, sequence of components c_j where c_j has name $n(c_j)$ and datatype $cdt(c_j)$. E.g., a date can be implemented either as three separate integer components or as an object with three properties.

You can see an example of this stage in Fig. 2. The student gets error messages if they select too many or too few components for the implementation for some data elements (in case several components were used to represent a single data element, e.g., two numbers to represent a date instead of three numbers), or in case datatype mismatch . The student can move to the next stage after building a consistent set of implementations $RE(e_i) = \{c_j\}$.

Data element	Data direction	Data implementation
Count of days between dates	Output data	A number
Date of birth	Input data	2 scalars: day (number), month (number) <div style="background-color: #f8d7da; padding: 5px; margin-top: 5px;">A property "year" is not implemented.</div>
First date at school	Input data	... <div style="border: 1px solid gray; padding: 5px; margin-top: 5px;"> ... An object with 3 properties: day (number), month (number), year (number) 3 scalars: day (number), month (number), year (number) 2 scalars: day (number), month (number) A number A collection of 3 elements of type "number" A collection of N elements of type "number" A collection of N elements of type "number", size of collection (number) </div>

Fig. 2. Stage 3. Choosing implementation of data elements

Stage 4. Choosing the Method of Passing Data. Given: set of data elements $E = \{e_i\}$, their directions $dir(e_i)$, and implementations $RE(e_i) = \{c_j\}$. The student must choose the method of passing data $tm(c_j)$ for every c_j : it can be a function argument (parameter), return value, etc. The possible errors on this stage is choosing a return value for input data elements, having more than one return value from a function, or choosing an argument with a wrong direction. The application generates names $n(c_j)$ for all chosen function arguments.

Data element	Data component	Method of passing data	Datatype of C-language
Count of days between dates		Return value	int
Date of birth	Number of day	Input parameter Parameter name: date_of_birth_day	int * <div style="border: 1px solid red; background-color: #ffe6e6; padding: 5px; margin-top: 5px;">Do you want to change the "date_of_birth_day" using a pointer?</div>
	Number of month	Input parameter Parameter name: date_of_birth_month	int ... int int * float char struct Date
	Number of year	Input parameter Parameter name: date_of_birth_year	

Fig. 3. Stage 5. Choosing data types

Stage 5. Choosing Programming Language Data Types. Given: set of data elements $E = \{e_i\}$, their implementations $RE(e_i) = \{c_j\}$ and the method of passing data $tm(c_j)$ for each c_j . The student must choose data type (in programming language) $ldt(c_j)$ for every c_j . The example of the stage 5 in the form of test is provided on Fig. 3. Student should select datatype of C-language (int, char, int *, etc.) $ldt(c_j)$. If $ldt(e_i)$ is not acceptable for c_j , then student gets error message. The move to the next exercise is possible if student choose acceptable data types for all the data components from $RE = \{c_j\}$.

Stage 6. Constructing the Function Prototype. Given: set of tokens $LL = \{llex_i\}$ of the C language that are contained in the prototype (and, possible, some distractors), implementations $RE(e_i) = \{c_j\}$, the name $n(c_j)$, the method of passing data $tm(c_j)$ and data type $ldt(c_j)$ for every c_j . The student must create a sequence of tokens $STUD_CODE$ from $LL = \{llex_i\}$ that forms the function prototype. The example of the stage 6 is shown in Fig. 4. The student must add tokens from the panel to the prototype, receiving feedback for every wrong (typically, out of sequence) token they add.

Create a prototype of function `distance_between_dates`

```


/*!
\ param [in] date_of_birth_day Date of birth (Number of day) Type: int
\ param [in] date_of_birth_month Date of birth (Number of month) Type: int
\ param [in] date_of_birth_year Date of birth (Number of year) Type: int


```

`int distance_between_dates (date_of_birth_day`

Is "date_of_birth_day" a datatype of function's argument?

first_date_at_school_year () ; , bool float

long int char void *

Remove the last lexeme

Fig. 4. Stage 6. Constructing the function prototype

4 Tutor Implementation and Evaluation

To implement the scenario presented above, the ontological model of the task being solved is used. This model is set by the teacher for each specific task. It is presented in OWL2 language and includes:

- formulation of the task as a sequence of tokens $TSL = \{tlex_k\}$ that is a sentence in natural language;
- the set of data elements $E = \{e_i\}$ that are used in the task;
- for each data element e_i , its reference implementation $ddt(e_i)$ (method of storing the data element in memory) is specified;
- for each element e_i , a set of possible implementations $ECDT(e_i)$ is specified.

An OWL2 ontology that contains 20 tasks for creating a function prototype in C language was implemented. Based on this model, stages 1–6 from the scenario above were implemented. Additionally, the tutor creates another ontological model (in the OWL2 language), based on the student's answer, which includes:

- for each data element e_i , its implementation, specified as a set of components $RE(e_i) = \{c_j\}$;
- the method of passing data $tm(c_j)$ for each data component c_j ;
- the data type $ldt(c_j)$, which will be used for each c_j component in the C language.

This ontological model is used to solve and grade stages 4–6 of the scenario.

To detect errors in the student’s answers, production rules (in the Jena-rules format) are used. They do not depend on the formulation of a particular problem to be solved, receiving the task model as input data. The rules that grade stages 1 and 2 from the scenario are independent of the programming language and implemented fully, while the rules that evaluate stages 3–6 are specific to the programming language and partially implemented for the C language in the current tutor prototype.

The efficiency and helpfulness of the current implementation of the tutor prototype were evaluated in course “Programming Basics” of Volgograd State Technical University. The experiment involved 22 first-year Software Engineering students. They were given a theoretical basis for creating function prototypes in the C language. The tutor was used to develop and consolidate relevant skills. The students were given five tasks each, which they first solved manually and then using the intelligent tutor. Of the 110 attempts results when solving tasks manually, 16.4% turned out to be correct, which is explained by the lack of practical experience in this domain. When using the developed intelligent tutor, 98.1% of the attempts were correct. The average time for a successful solution of the task was 03:35 and 03:21, respectively. The high amount of successful attempts was achieved because of using of hints generated by the tutor. On average, 4.9 hints were generated in each attempt. A survey of students was conducted using the Likert scale. The following percentage of students “strongly agreed” and “agreed” with the statements below:

- hints helped to solve the task: 90.9%;
- hints helped to understand the causes of mistakes and avoid similar mistakes in the future: 95.5%;
- using the tutor takes a lot of time: 4.5%;
- it is advisable to use the tutor to solve similar tasks: 95.5%.

The developed tutor showed the its helpfulness and efficiency in training skills of creating a function prototype.

5 Conclusions and Future Work

An intelligent programming tutor prototype that provides immediate and customized formative feedback to learners without requiring intervention from a teacher has been implemented. It consist of a server and a web interface. The server component includes an OWL2 ontology that contains 20 tasks for creating a function prototype, Jena rules for evaluating student solutions, and the reasoner from Apache Jena Framework for reasoning based on the production rules and the OWL ontology. Jena rules are implemented in full for stages 1 and 2, and partially (for the specific programming language) for stages 3–6. The rules determine the kinds of errors the student makes which can be used to select the next task depending on the learning problems the particular student has. We

have shown that automatic model-based tutor for designing function interface based on the formal model of the subject domain is possible.

The future work includes implementation of all rules for stages 3–6 to detect student errors, and implementation of adaptive selection of learning tasks according to the errors that the student makes.

References

1. Brusilovsky, P., Su, H.-D.: Adaptive visualization component of a distributed web-based adaptive educational system. In: Cerri, S.A., Gouardères, G., Paraguaçu, F. (eds.) ITS 2002. LNCS, vol. 2363, pp. 229–238. Springer, Heidelberg (2002). https://doi.org/10.1007/3-540-47987-2_27
2. Cavalcanti, A.P., et al.: Automatic feedback in online learning environments: a systematic literature review. *Comput. Educ. Artif. Intell.* **2**, 100027 (2021). <https://doi.org/10.1016/j.caeai.2021.100027>
3. Denisov, M., Anikin, A., Sychev, O.: Dynamic flowcharts for enhancing learners' understanding of the control flow during programming learning. In: Basu, A., Stapleton, G., Linker, S., Legg, C., Manalo, E., Viana, P. (eds.) Diagrams 2021. LNCS (LNAI), vol. 12909, pp. 408–411. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-86062-2_42
4. Fabric, G.V.F., Mitrovic, A., Neshatian, K.: Adaptive problem selection in a mobile python tutor. In: Adjunct Publication of the 26th Conference on User Modeling, Adaptation and Personalization. ACM (2018). <https://doi.org/10.1145/3213586.3225235>
5. Jeurings, J., Gerdes, A., Heeren, B.: A programming tutor for haskell. In: Zsóck, V., Horváth, Z., Plasmeijer, R. (eds.) CEFP 2011. LNCS, vol. 7241, pp. 1–45. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-32096-5_1
6. Kim, T., Kim, S., Ryu, D.: CodingTM: development task visualization for SW code comprehension. In: 2021 Working Conference on Software Visualization (VIS-SOFT), IEEE, September 2021. <https://doi.org/10.1109/vissoft52517.2021.00012>
7. Kumar, A.N.: Generation of problems, answers, grade, and feedback—case study of a fully automated tutor. *J. Educ. Resour. Comput.* **5**(3), 3 (2005). <https://doi.org/10.1145/1163405.1163408>
8. Kumar, A.N.: An epistemic model-based tutor for imperative programming. In: Roll, I., McNamara, D., Sosnovsky, S., Luckin, R., Dimitrova, V. (eds.) AIED 2021. LNCS (LNAI), vol. 12749, pp. 213–218. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-78270-2_38
9. Le, N.T.: A classification of adaptive feedback in educational systems for programming. *Systems* **4**(2), 22 (2016). <https://doi.org/10.3390/systems4020022>
10. Liu, F., Li, G., Fu, Z., Lu, S., Hao, Y., Jin, Z.: Learning to recommend method names with global context (2022). <https://doi.org/10.48550/arXiv.2201.10705>
11. Nie, P., Zhang, J., Li, J.J., Mooney, R.J., Gligoric, M.: Evaluation methodologies for code learning tasks, August 2021. <https://arxiv.org/pdf/2108.09619.pdf>
12. O'Rourke, E., Butler, E., Díaz Tolentino, A., Popović, Z.: Automatic generation of problems and explanations for an intelligent algebra tutor. In: Isotani, S., Millán, E., Ogan, A., Hastings, P., McLaren, B., Luckin, R. (eds.) AIED 2019. LNCS (LNAI), vol. 11625, pp. 383–395. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-23204-7_32

13. Papadakis, S., Kalogiannakis, M., Zaranis, N.: Developing fundamental programming concepts and computational thinking with Scratch Jr in preschool education: a case study. *Int. J. Mob. Learn. Organ.* **10**(3), 187 (2016). <https://doi.org/10.1504/ijmlo.2016.077867>
14. Psofka, J., Mutter, S.: *Intelligent Tutoring Systems: Lessons Learned*. Lawrence Erlbaum Associates, Mahwah (1988)
15. Rathore, A.S., Arjaria, S.: *Intelligent Tutoring System*, pp. 121–144 (01 2020). <https://doi.org/10.4018/978-1-7998-0010-1.ch006>
16. Sirkiä, T.: Recognizing programming misconceptions. An analysis of the data collected from the UUhistle program simulation tool. Master's thesis, Aalto University. School of Science (2012). http://www.uuhistle.org/publications/sirkia_masters_thesis.pdf
17. Sychev, O., Anikin, A., Penskoy, N., Denisov, M., Prokudin, A.: CompPrehension - model-based intelligent tutoring system on comprehension level. In: Cristea, A.I., Troussas, C. (eds.) *ITS 2021. LNCS*, vol. 12677, pp. 52–59. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-80421-3_6
18. Uehara, M.: Programming learning by creating problems. In: *2020 Eighth International Symposium on Computing and Networking Workshops (CANDARW)*, IEEE, November 2020. <https://doi.org/10.1109/candarw51189.2020.00059>
19. Yoo, J., Pettey, C., Seo, S., Yoo, S.: Teaching programming concepts using algorithm tutor. In: *EdMedia+ Innovate Learning*, pp. 3549–3559 (2010)



Effects of Guidance on Learning About Ill-defined Problems

Sungeun An^{1(✉)}, Emily Weigel², and Ashok K. Goel¹

¹ School of Interactive Computing, Georgia Institute of Technology,
Atlanta, GA 30308, USA
sungeun.an@gatech.edu

² School of Biological Sciences, Georgia Institute of Technology,
Atlanta, GA 30332, USA

Abstract. We present a study that examines the effects of guidance on learning about addressing ill-defined problems in undergraduate biology education. Two groups of college students used an online laboratory named VERA to learn about ill-defined ecological phenomena. While one group received guidance, such as giving the learners a specific problem and instruction on problem-solving methods, the other group received minimal guidance. The results indicate that, while performance in a problem-solving task was not different between groups receiving more vs. minimal guidance, the group that received minimal guidance adopted a more exploratory strategy and generated more interesting models of the given phenomena in a problem-solving task.

Keywords: Self-directed learning · Guided learning · Modeling and simulation · Ill-defined problems · Online laboratory

1 Introduction

Interactive Learning Environments (ILEs) enables learners to explore ill-defined problems such as modeling complex systems [3, 4]. Many studies have found that while guided learning in ILEs leads to more efficient learning [6, 10, 15], it can limit critical thinking and creative problem solving [7, 11]. In contrast, while self-directed exploration can be inefficient [15], it often results in more varied and interesting solutions [18].

However, many studies exploring the effects of guidance in ILEs have focused on K-12 education with well-defined goals, assessments, and outcomes [6, 10, 14, 17, 19]. The precise role of guidance in the context of ill-defined problem-solving is not yet well understood. For example, Chen (2007) found that problem-solving prompts did not have a positive effect on solving ill-structured problems [5]. This contradicted Ge & Land's (2003) study indicating that prompts had significantly improved students' problem-solving [8]. Koedinger & Alevan (2007) called balancing the amount of guidance the "assistance dilemma" to achieve the

optimal learning [16]. To solve the assistance dilemma, it is necessary to understand the trade-offs in providing guidance for learning on ill-defined problems as well.

We present a study on the use of an ILE called VERA (for Virtual Experimentation Research Assistant) for learning about ill-defined problems in undergraduate biology education. VERA is a web-based online laboratory that enables users to construct conceptual models of ecological systems and run interactive agent-based simulations of these models [1, 2] (vera.cc.gatech.edu). This allows learners to explore multiple hypotheses about ecological phenomena and perform “what if” experiments to either explain an ecological phenomenon or attempt to predict the outcomes of changes to an ecological system. Our goal in this study was to answer two research questions. *RQ1: Does more guidance indeed make learning more efficient in ill-defined problem-solving contexts? RQ2: Does minimal guidance indeed result in more varied and interesting models?*

To answer these questions, we conducted an experiment to analyze the relative effects of guidance on the learning process and outcomes. Two groups of students engaged in two different but related learning activities: one with guidance such as giving the learners a specific problem and instruction on problem-solving methods, and the other with minimal guidance. After the learning activities, a new problem-solving task was given to students to examine which learning context enabled the students to solve the task effectively and creatively.

2 A Brief Description of the VERA Online Laboratory

In VERA, learners build conceptual models of complex phenomena, evaluate them through simulation, and revise the models as needed. VERA enables learners to construct Component-Mechanism-Phenomena (CMP) models [13] of ecological phenomena that originate from our earlier work on Structure-Behavior-Function models of complex systems [9]. A CMP model specifies the biotic and abiotic components participating in an ecological phenomenon, the relationships among the components, and the processes that arise through the interactions among the components. VERA automatically translates a CMP model into an agent-based simulation on the NetLogo platform [12] (ccl.northwestern.edu/netlogo). Using VERA system involves three high-level activities: (1) model construction (e.g., adding biotic/abiotic components and relationships), (2) parameterization (e.g. selecting a set of simulation parameters for each component/relationship), and (3) execution of the agent-based simulation. Parameterization is further divided into four categories: (1) *t-parameters* (trivial parameter) that directly manipulates an initial population (e.g., initial population, minimum population), (2) *b-parameters* related to the biological properties (e.g., lifespan, reproductive rate), (3) *a-parameters* related to the abiotic substances (e.g., amount, growth rate), and (4) *i-parameters* that adjust the relationship between two components (e.g., interaction probability, consumption rate).

3 Experimental Design

The goal of this study was to compare the relative effects of more guidance and minimal guidance. Figure 1 illustrates a schematic diagram of the study. Independent variable is the amount of guidance in Phase 1 (problem representation) and Phase 2 (problem solving). Dependent variables are outcomes (performance, efficiency, model quality) in Phase 3.

3.1 Participants

The participants were college students in an undergraduate biology class at a large public R1 institution located in the southeastern US. This class teaches scientific methods and skills that address population ecology. Students in the class used VERA in the accompanying laboratory sections. The duration of this class was 2 h and 45 min. Our study consisted of six lab sections (2 hr 45 min each) with 16 students for each section ($N = 96$). The students were assigned to either Group A (guided) or Group B (self-exploration) based on their sections.

3.2 Procedure

As shown in Fig. 1, the students in three lab sections were assigned to Group A ($N = 48$); The students in the remaining three sections were assigned to Group B ($N = 48$). As students arrived in the classroom, two teaching assistants played a pre-recorded introduction video giving a high-level description of VERA and how to use the system (5 min). All students were then asked to complete a consent form and indicate their major and year. The modeling task consisted of three phases: problem representation (Phase 1), problem solving (Phase 2), and a problem-solving task (Phase 3). In Phase 1 and 2, Group A explored a given problem with guidance in using VERA (e.g., representing a problem, solving a problem, as described in detail below). Meanwhile, Group B explored a problem of their choice with minimal guidance. In Phase 3, Group A and Group B were given the same problem to solve.

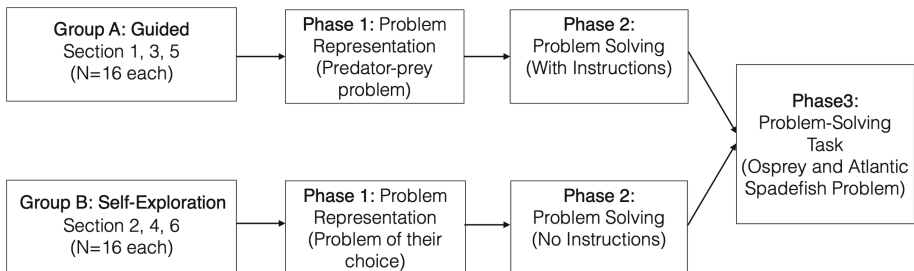


Fig. 1. Schematic diagram of experimental design.

3.3 Group A: Guided Learning

The students in Group A were given some heuristics guidance during Phase 1 and Phase 2 ($N = 48$). In Phase 1, the students were asked to create a model that represents a common interaction in ecology: a predator-prey relationship in which the predator consumes the prey and the prey gradually decreases. In Phase 2, the students were asked to adjust their models by reducing the predator population so that the prey can survive longer. In this way, the students were given a problem representation as well as instructions on how to solve it. The instruction stated “there are many ways to reduce the population of the predators. One way is to directly reduce the predator’s initial population. Another way is to indirectly reduce the predator’s population by adjusting various parameter values of the predator.”

3.4 Group B: Self-Exploration

The students in Group B were given minimum guidance during Phase 1 and Phase 2 ($N = 48$). In Phase 1, the students were asked to create a model that represents any problem in ecology which includes an abnormal increase or decrease in the population of a target species. In Phase 2, the students were asked to adjust their models to mitigate the problem in their previous model by adding/deleting components and/or adjusting various parameter values. Since the students in Group B received more general instructions, they had to define a problem as well as explore various hypothetical solutions to solve the problem.

3.5 Problem-Solving Task

In Phase 3 (Problem-Solving Task), the students both in Group A and Group B were given a new problem to solve. The pre-built model that represents a food web between Osprey (*Pandion haliaetus*) and Atlantic Spadefish (*Chaetodipterus faber*) was given to both groups of students (see 2 (A)). Their goal was to adjust this model to create a stable ecosystem where both species can survive. The students were asked to write a list of initial thoughts of how they would change the model. Then, they tested their hypothetical solutions in VERA and compared them in terms of effectiveness, cost, practicality, etc.

4 Data Analysis

4.1 Outcome Measures

The students’ outcomes during Phase 3 (problem-solving task) were measured using the students’ models, their log data, and survey answers on Qualtrics based on the following metrics.

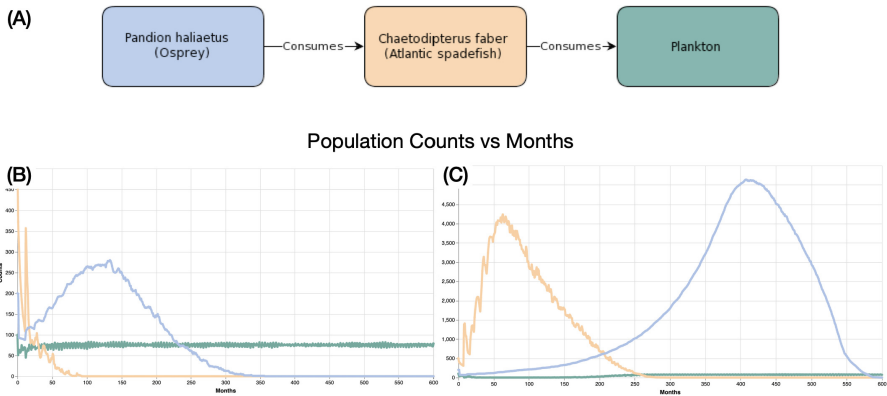


Fig. 2. (A) The original model used in the task. (B) The simulation output of the original model. (C) The simulation output of the successfully adjusted model (Both osprey and atlanta spadefish survived longer than the original model.)

- **O1 Success in making models according to specification:** The simulation outcomes between the original model and the students' adjusted models were compared. The score is 1 (correct) when both species (Osprey and Atlantic Spadefish) survived longer than the original model (for example, Fig. 2 (C)); 0.5 (partially correct) when only one species survived longer; and 0 (incorrect) when none of the species survived longer.
- **O2 The number of hypothesis:** The number of hypothetical solutions and corresponding models generated by the students were measured.
- **O3 Time taken to making models according to specification:** The time taken in the problem-solving task (Phase 3) was measured based on the time the students submitted their answers on Qualtrics.
- **O4 Model complexity (depth):** Model complexity (the total number of components and relationships that exist within a conceptual model) was measured. If a student doesn't add or delete any components (nodes) and relationships (edges), the model complexity remains 5 as the original model (e.g., see Fig. 2).
- **O5 Model variability (breadth):** Model variability measures how many distinct simulation parameter categories were used. It is calculated by the percentage of the parameter categories changed within a conceptual model. Since there was no abiotic substance in the original model, *t-parameters*, *b-parameters*, and *i-parameters* were used.

4.2 Results

A total of 79 students consented to the study (A = 39; B = 40): Participants included 53 female, 23 male, 3 prefer not to say; 19–23 years of age, mean 20.2 years. Group A created a total of 81 models for the task; Group B created 84 models. A Shapiro-Wilk test was performed in each outcome and showed a

significant departure from normality. Based on this outcome, a non-parametric test, Mann-Whitney U test, as well as descriptive statistics, were used to summarize the results.

Performance and Efficiency (O1, O2, O3). Figure 3 shows comparative outcomes *O1* (Task score), *O2* (The number of hypotheses), and *O3* (Completion Time) between the two groups. The mean of the task score was higher for the guided group ($M = 0.91, SD = 0.17$) and the self-exploration group ($M = 0.84, SD = 0.20$) with the same median and the interquartile range ($Mdn = 1.0, IQR = 0.25$). The number of hypotheses generated during the problem-solving task was also higher for the guided group ($M = 2.20, SD = 0.57$) than the exploration group ($M = 2.25, SD = 0.58$) with the same median and the interquartile range ($Mdn = 2.0, IQR = 0.0$). On the other hand, the mean and the median of completion time was lower for the guided group ($M = 842.55, SD = 511.19, Mdn = 708.34, IQR = 355.56$) than the exploration group ($M = 957.49, SD = 565.41, Mdn = 832.20, IQR = 422.67$).

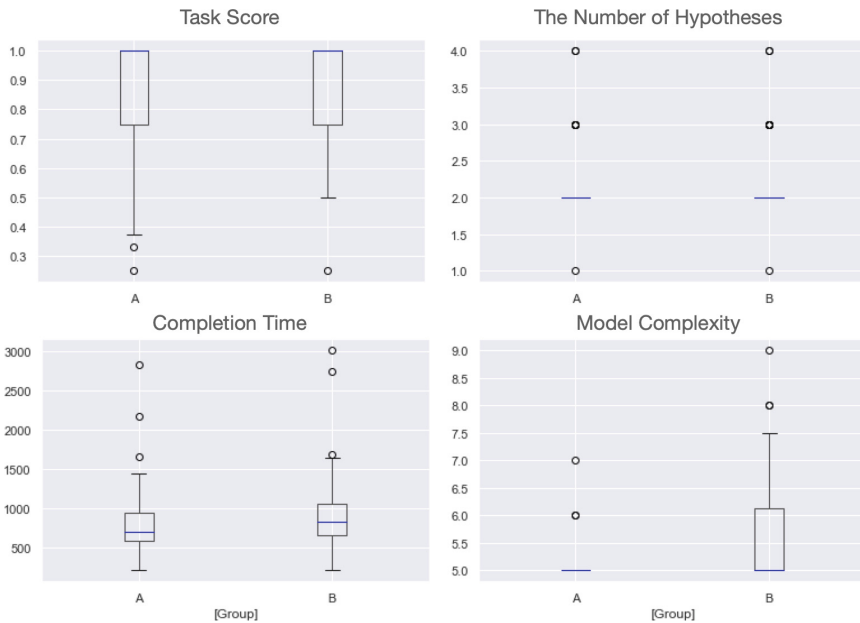


Fig. 3. Comparative outcomes: (O1) Task score. (O2) The number of hypotheses. (O3) Completion time. (O4) Model complexity.

Model Quality (O4, O5). We measured model quality using two proxies: *model complexity* and *model variability*. Figure 3 shows comparative model complexity between the two groups. The mean of the measured model complexity was

higher in the exploration group B ($M = 8.02, SD = 1.45, Mdn = 5.0, IQR = 1.25$) than the guided group A ($M = 7.09, SD = 0.74, Mdn = 5.0, IQR = 0.0$). This difference was statistically significant as determined by the Mann-Whitney U test ($U = 503.0, p < .0005$).

Figure 4 shows comparative model variability between the two groups. The most frequently used parameter for the guided group was the t-parameter of the predator (Osprey) ($M = 23.32, SD = 21.76$), followed by b-parameter of the prey (Atlantic Spadefish) ($M = 21.07, SD = 25.65$) and b-parameter of the predator ($M = 15.74, SD = 23.92$). On the contrary, the most frequently used parameter for the self-exploration group was i-parameter, the interaction parameter, ($M = 18.08, SD = 17.82$), followed by t-parameter of the predator ($M = 17.67, SD = 24.56$) and b-parameter of the prey ($M = 17.44, SD = 24.02$).

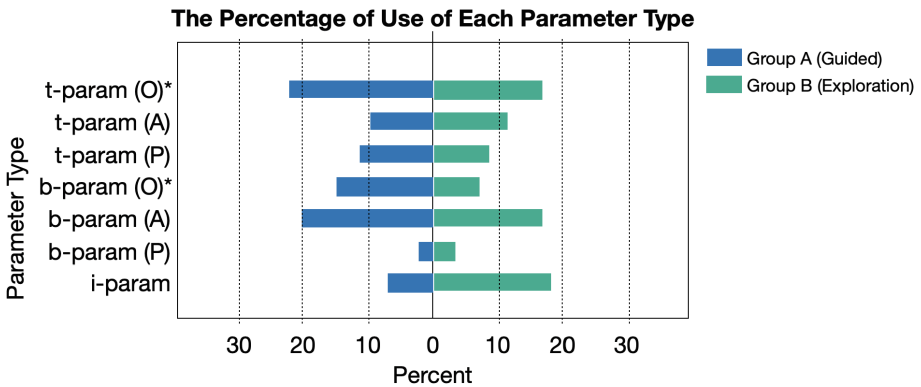


Fig. 4. The percentage of use of each parameter type. *Parameters for osprey (Predator). O:Osprey. A:Atlantic spadefish. P:Plankton.

5 Discussion

Our data concur with earlier findings in well-defined problem contexts (e.g., [6]), specifically that giving more information or assistance leads to higher accuracy and efficiency of learning [6, 10]. In our study, the students who were given guidance in solving an ill-defined problem solved the second ill-defined problem in Phase 3 slightly more efficiently in terms of completion time (O3), but the difference was not statistically significant. Irrespective of guidance, there does not seem to be much difference in the ability to understand the problem, devise hypothetical solutions, and test and validate one’s own hypotheses.

However, our data adds to what is known about guidance in relation to the complexity of models (O4). While the students in the guided group typically made small changes by adjusting parameter values, some students in the self-exploration group also added more components to the model. Given that the

performance between the two groups was similar (e.g., task score, the number of hypotheses, and completion time), the way they approached the problem showed interesting differences. In addition to manipulating simulation parameters, adding new components and relationships as we observed in our data likely represents more constructivist learning and discovery learning described in other studies [6, 15, 18, 20]. Many studies have shown that minimally guided learning can foster students' creative thinking by having them independently explore broader issues [18, 21]. Similarly in our study, the students solved the task by adding new components beyond manipulating existing variables.

The students in the guided group focused on adjusting the parameters of the components, especially the parameters of the predator (see Fig. 4). This may be due to the fact that the students were asked to adjust their models by reducing the predator population in Phase 2. On the contrary, the students in the self-exploration group approached the problem in more varied ways. In addition to adding new components and relationships, they commonly used *i*-parameters such as interaction probability and consumption rate, which were not frequently used by the guided group. This suggests that providing a specific example or instruction on problem-solving methods has the potential to affect students' learning as it might restrict what learners try to do in learning.

6 Conclusion

In this paper, we presented a study in which two groups of college students used the online laboratory called VERA to learn about ecological phenomena: while one group received more guidance (e.g., heuristics), the other group received minimal guidance (e.g., process constraints). The data indicates that the group that received minimal guidance adopted a more exploratory strategy on the problem-solving task and generated more varied and complex models of the given phenomena. However, the group that received more guidance did not show significant benefits in efficiency and accuracy, which was unexpected. These preliminary results invite further research on the proper level of guidance in learning about ill-defined problems in undergraduate science education.

Acknowledgement. This research was supported by US NSF grant #1636848. We thank members of the VERA project, especially Spencer Rugaber, Luke Eglington, and Stephen Buckley. We also thank the TAs and the students in the biology lab. This research was conducted in accordance with IRB protocol #H18258.

References

1. An, S., Bates, R., Hammock, J., Rugaber, S., Weigel, E., Goel, A.: Scientific modeling using large scale knowledge. In: Bittencourt, I.L., Cukurova, M., Muldner, K., Luckin, R., Millán, E. (eds.) AIED 2020. LNCS (LNAI), vol. 12164, pp. 20–24. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-52240-7_4

2. An, S., Broniec, W., Rugaber, S., Weigel, E., Hammock, J., Goel, A.: Recognizing novice learner's modeling behaviors. In: Cristea, A.I., Troussas, C. (eds.) ITS 2021. LNCS, vol. 12677, pp. 189–200. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-80421-3_21
3. Brown, A.L., Campione, J.C.: Interactive learning environments and the teaching of science and mathematics. *Toward Sci. Pract. Sci. Educ.* 111–139 (1990)
4. Brown, A.L., Campione, J.C., Webber, L.S., McGilly, K.: Interactive Learning Environments: A New Look at Assessment and Instruction. In: Gifford, B.R., O'Connor, M.C. (eds.) *Changing Assessments. Evaluation in Education and Human Services*, vol. 30, pp. 121–211. Springer, Dordrecht (1992). https://doi.org/10.1007/978-94-011-2968-8_5
5. Chen, C.H., Bradshaw, A.C.: The effect of web-based question prompts on scaffolding knowledge integration and ill-structured problem solving. *J. Res. Technol. Educ.* **39**(4), 359–375 (2007)
6. Cohen, M.T.: The effect of direct instruction versus discovery learning on the understanding of science lessons by second grade students (2008)
7. Dabbagh, N.: Scaffolding: an important teacher competency in online learning. *TechTrends* **47**(2), 39 (2003)
8. Ge, X., Land, S.M.: Scaffolding students' problem-solving processes in an ill-structured task using question prompts and peer interactions. *Educ. Technol. Res. Dev.* **51**(1), 21–38 (2003)
9. Goel, A.K., Rugaber, S., Vattam, S.: Structure, behavior, and function of complex systems: the structure, behavior, and function modeling language. *Ai Edam* **23**(1), 23–35 (2009)
10. González-Calero, J.A., Arnau, D., Puig, L., Arevalillo-Herráez, M.: Intensive scaffolding in an intelligent tutoring system for the learning of algebraic word problem solving. *Br. J. Educ. Technol.* **46**(6), 1189–1200 (2015)
11. van Joolingen, W., de Jong, T.: Model-based diagnosis for regulative support in inquiry learning. In: Azevedo, R., Aleven, V. (eds.) *International Handbook of Metacognition and Learning Technologies*. SIHE, vol. 28, pp. 589–600. Springer, New York (2013). https://doi.org/10.1007/978-1-4419-5546-3_38
12. Joyner, D.A., Goel, A.K., Papin, N.M.: Mila-s: generation of agent-based simulations from conceptual models of complex systems. In: *Proceedings of the 19th International Conference on Intelligent User Interfaces*, pp. 289–298 (2014)
13. Joyner, D.A., Goel, A.K., Rugaber, S., Hmelo-Silver, C., Jordan, R.: Evolution of an integrated technology for supporting learning about complex systems. In: *2011 IEEE 11th International Conference on Advanced Learning Technologies*, pp. 257–259. IEEE (2011)
14. Kim, M.C., Hannafin, M.J.: Scaffolding 6th graders' problem solving in technology-enhanced science classrooms: a qualitative case study. *Instr. Sci.* **39**(3), 255–282 (2011)
15. Klahr, D., Nigam, M.: The equivalence of learning paths in early science instruction: effects of direct instruction and discovery learning. *Psychol. Sci.* **15**(10), 661–667 (2004)
16. Koedinger, K.R., Aleven, V.: Exploring the assistance dilemma in experiments with cognitive tutors. *Educ. Psychol. Rev.* **19**(3), 239–264 (2007)
17. Lazonder, A.W., Harmsen, R.: Meta-analysis of inquiry-based learning: effects of guidance. *Rev. Educ. Res.* **86**(3), 681–718 (2016)
18. Rahman, M.H.: Using discovery learning to encourage creative thinking. *Int. J. Soc. Sci. Educ. Stud.* **4**(2), 98 (2017)

19. Rittle-Johnson, B., Koedinger, K.R.: Designing knowledge scaffolds to support mathematical problem solving. *Cogn. Instr.* **23**(3), 313–349 (2005)
20. Schunk, D.H.: *Learning Theories an Educational Perspective*. 6th edn. Pearson, London (2012)
21. Veermans, K., de Jong, T., van Joolingen, W.R.: Promoting self-directed learning in simulation-based discovery learning environments through intelligent support. *Interact. Learn. Environ.* **8**(3), 229–255 (2000)



MEMORABLE: A Multi-playEr custoMisable seriOus Game fRAmework for cyBer-security LEarning

Jingyun Wang^(✉) , Ryan Hodgson , and Alexandra I. Cristea 

Durham University, Durham DH1 3LE, UK
jingyun.wang@durham.ac.uk

Abstract. In this paper, we propose an educational game framework allowing instructors to customise the game’s learning content in the context of cyber-security, with the aim of ensuring learners are engaged with educational games. This can further support them continuing to acquire useful cyber-security knowledge, while playing with their peers. Based on this framework, a prototype digital game called “Cyberpoly” was implemented and evaluated. This game allows unfamiliar or potentially unappealing, ‘dry’ learning contents (on cyber-attacks and incidents) to be placed in a context similar to a “monopoly” game board, encouraging multiple players to take an active role when landing on various cyber-attack or incident squares. The learner can not only answer the questions generated by the game, but also actively send cyber-attacks to other players, when landing on another’s land. Learner data was collected from 30 undergraduate participants, with results suggesting that most found it engaging and felt motivated to learn. Further to this, we also obtained feedback from two academic professors, to discuss not only game-play but also game element management.

Keywords: Customisable game content · Cyber-security knowledge · Multi-player · Educational games

1 Introduction

With the proliferation of online conferences and business meetings during the Coronavirus pandemic, cyber-security knowledge has become an essential necessity for users of technology. It is reported that more data was compromised in 2020 than in the previous 15 years combined [1]. It is found that 90% of security incidents and breaches were caused by human errors [15], and a fundamental aspect of the issue is the lack of awareness and knowledge on cyber-security of the end users. On the other hand, it is challenging to teach cyber-defense knowledge in an accessible and enjoyable way, especially to those without a technical background. One approach to address this issue is through educational digital games [16]. Combining the enjoyable game features with a serious educational objective may be beneficial in motivating a user to learn about a subject [14],

particularly if they have no prior knowledge of the domain. Additionally, this may assist in the retention of any learned knowledge [12]. This is especially promising for highly-technical subjects, such as cyber-security, where users may not find the subject interesting, or may be unfamiliar with the subject.

To support cyber-security learners in an engaging way, we propose a *customisable multi-player educational game framework for cyber-security*. The most prevalent motivation is to provide a serious game element bank, serving as a template, which can be reused by various games, with the aim of supporting cyber-security education. All of the educational contents within the serious game element bank are organised and maintained by instructors, in order to support serious games to focus on conveying key cyber-knowledge to learners and help them to develop their skills and understanding. Furthermore, a prototype digital game was implemented and evaluated based upon this framework. Based on the feedback from experiment participants, we seek to evaluate if this educational game may enhance learning motivation due to its competition environment.

The main contributions of this research are: **(1) A fine-granularity approach for game element placement.** Placing game elements in a meaningful context is essential for serious game design [9]. The educational contents chosen to be incorporated into our game framework are various existing cyber-incidents, cyber-attacks, which exploit the different vulnerabilities, and the laws related to cyber-security, as well as the countermeasures that can be applied, to prevent cyber-attacks. With this game element placement. Intelligent Tutoring functions could be easily implemented to identify the knowledge status of learner and provide adaptive learning path based on their behaviour in the game. **(2) Customisable game elements.** In previous educational games in cyber-security [2–6], the content being taught is built into games and fixed, which limits the scope and flexibility. Since the cyber-security topics are continuously expanding and changing, being able to create new content and adjust existing content in educational games can keep them relevant for a large number of players. We proposed to support the instructors to manage the contents used in the educational games, which can ensure they have up to date contents in response to the emerging cyber-attacks, cyber incidents, countermeasures and laws. Moreover, by making use of contents in these three areas, the instructors are expected to easily create/edit questions. Those questions, together with contents related to the three areas consisting of the “serious game element bank” (defined in this research) should be able to be further reused by multiple games. **(3) Gameplay.** The gameplay of most previous educational games on cyber-security [2–6] consisted simply of individually answering different categories/topics/difficulty level of questions delivered automatically, suggesting that more enjoyable gameplay could be obtained by interaction with peers, to hold the player’s interest. In our prototype multi-player game “Cyberpoly”, the player can actively send a cyber-attack, using one of the attack cards to others, when landing in their land, and the landlord needs to respond to the attack. This scenario not only increases the interaction between players but also simulates reality in which the players need to respond to any received cyber-attack.

2 Related Work

Digital educational games, which offer the players the opportunity to take roles, think and act the way they cannot experience in reality, has been proven to be a powerful tool to enable knowledge acquisition, by providing engaging and motivational contexts [9]. Instead of focusing on providing entertainment, the main purpose of educational games is to incorporate pedagogical theories and educational content into game elements. Therefore, the pedagogical theory behind the game design is the most essential issue for the educational effectiveness [9].

Several educational games have been developed for supporting cyber-security learning. CyberSprinters [3] was developed by the National Cyber Security Centre (NCSC) as an educational resources toolkit, requiring learners aged 7–11 to answer questions while guiding an avatar through a virtual world. Keep Tradition Secure [4] by Texas A&M University required players to simply answer questions, as they made their way around campus. Targeted Attack [5] by Trend Micro was a video-based single mode simulation game based on the format of the old “Choose Your Own Adventure” books. The player acted as the Chief information officer of a global organisation, on the verge of making the first release of a biometrically authenticated mobile payment application, to deal with various cyber-security situation related to such a framework. Cybersecurity Lab [2] by PBS allocated players the role of the chief technology officer of a social network company and required them to answer questions to gain coins for strengthening their cyber-defenses. The UK National Crime Agency, together with Cyber Security Challenge UK, have provided a selection of interactive games [6], which require players to complete various tasks in different scenarios. However, the fixed educational content in such games leads to inflexibility, due to limiting to a given user type. Therefore, in this work, we propose an educational game framework which may enable teachers to organise teaching materials for serious games aiming to support various cyber-security learners.

Moreover, existing serious games targeted at cyber-security learning only support learners to practice alone (single player mode), with none providing a multiplayer mode [18] with a collaborative learning environment. The primary objective of collaborative learning [10] is to allow learners to interact in ways such that certain learning mechanisms are triggered. In a cooperative setting, players may combine complementary skills, knowledge or resources, when completing in-game tasks; in contrast, competitive play styles encourages players to achieve better results based upon specified performance evaluation metrics, such as score, or time goals [17]. Pirates Treasure Hunt [8], targeted towards European elementary school children aged 7–10 years, was presented to stimulate the interest and attention on other cultures and lifestyles. Multi-players were tasked with using their acquired knowledge to spot non-European objects within the 2D game environment. VocaMomo, proposed by [11], intended to teach English vocabulary, with stakeholders being students and their parents. The game applied the basic rules of Monopoly and Scrabble strategy game, where players take turns rolling dice, with the aim at each turn to find a correctly spelled word, through dragging and dropping alphabet tiles. The use of popular games as a base for the

serious game is a common pattern. Similarly, [13] proposed an multi-player game to educate children about nutrition and health using a Bingo process, where the player picked a number from the table, with each number representing a question on nutrition. Players were rewarded in experience points and the team with the maximum points won the quiz. Therefore, a multi-player serious game environment, which combines the merits of collaborative learning, is designed and implemented for cyber-security learning in this paper.

3 MEMORABLE: A Customisable Serious Game Framework

For this work, we demonstrate the implementation of an educational game framework, which allows instructors to customise the game's learning content (serious game element) in the context of cyber-security: cyber-incidents, attacks, laws/countermeasures, and questions created conveniently based on the first three types of content (as shown in Fig. 1). This framework design ensures that the educational game is usable by a variety of audiences through enabling customisation, thus allowing for game content to be tailored to different countries, age groups, etc., contrasting to existing educational games with fixed content limited to a single demographic. Additionally, emphasis is placed upon the requirement for the modification of learning material to be as simple and intuitive as possible. The educational aspect of the game incorporates multiple choice questions, aiming for teachers to be able to easily create a large quantity of questions and answers, for use in the game. Therefore, we ensured that system design would account for the capability of instructors to modify basic building blocks of two categories of the multiple-choice questions related to cyber-security laws/countermeasures, attacks, and incidents, so that these components could be reused and combined in different ways.

3.1 Game Element Management Panels

Figure 1(a) shows the cyber elements management panel, in which the instructors can add, edit and delete 3 types of elements (cyber-attacks, cyber-incidents, and laws and countermeasures). The system also provides a question management panel for organizing two types of questions. Event questions involving an incident (i.e. a story about a cyber-security event that has occurred) and multiple answer options, which can either be about laws/countermeasures, or attacks. Attack card questions, involving knowledge about a specific cyber-attack are demonstrated in Fig. 2(b). When creating questions and corresponding answers, the instructor can choose the content organised in the three cyber-element lists from a drop-down menu, so that they may easily create questions and edit choice options, as shown in Fig. 1(b). This framework supports two types of users: Instructors, with the ability to input and modify the learning materials of the game, and students, who may play with peers in any corresponding educational games built on those materials. Multiple game sessions can take place at the same

time, ensuring a large number of learners can play educational games concurrently and keep track of their progress in all the games. Furthermore, a learner’s playing record is saved within a database, allowing the instructor to analyse the learning status of each learner, and adjust learning contents or provide individual instruction.

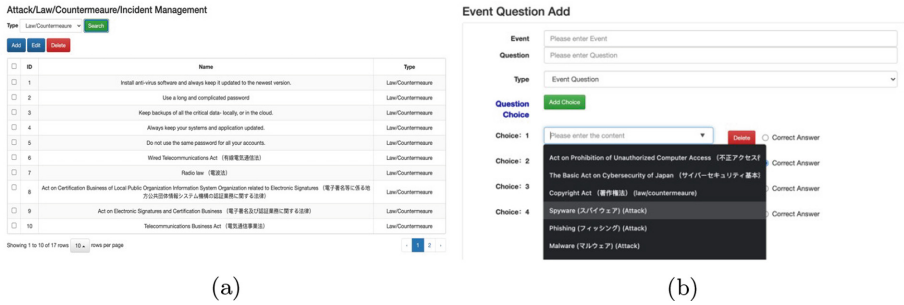
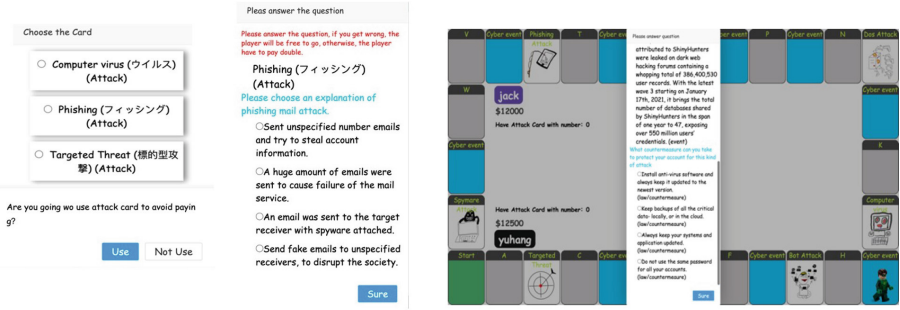


Fig. 1. Learning content management interfaces.

3.2 A Prototype Multi-player Game: “Cyberpoly”

A prototype digital game called “Cyberpoly” was implemented based on this architecture. When a player logs in the game lobby, they are presented with a room of tables, ensuring players may take part in a game with other players on a table. Each table allows 4 players maximum, and a minimum of 2 once all players present have agreed to start. Figure 2 depicts the main game-play interface of two players. Rules of play are the following: 1. The player moves the specified numbers of grid spaces in turns, based upon a dice roll. 2. Upon landing on an area, a player may choose to buy the area, if it does not belong to anyone already. 3. The player may choose to use an attack card to avoid paying (event 1, as shown in Fig. 2(a-1)) or to pay the corresponding fee when landing at the other player’s area. If the landlord correctly answers the received question (as shown in Fig. 2(a-2)), the attacker will need to pay double, otherwise, the attacker does not need to pay anything. 4. A player landing on the blue area will trigger a cyber-security incident (event 2, as shown in Fig. 2(b)). and cyber-attack cards will be rewarded, if they correctly answer the question appearing. 5. Upon landing on an area with a cyber-attack image, a question relevant to the attack will be asked (event 3, the question is similar to Fig. 2(a-2) but with different punishment and reward). If the player correctly answers this question, they are rewarded with in-game money, if they fail to answer correctly, the player will lose money. 6. Each player starts the game with the same amount of in-game money. The players to lose all their money will lose the game.

The formulation of this play structure is designed to ensure students take in cyber-security knowledge at a deeper level through competition, by stimulating the students’ learning interest. As noted, we divide cyber-security knowledge



(a-1) use a attack card (a-2) a question received by landlord (b) A cyber-security incident (Event 2) .
 (a) Sending an attack card (Event 1).

Fig. 2. The main interface of the prototype game “Cyberpoly”.

into cyber-attacks, incidents, and countermeasures, with the aim of ensuring that in-game events represent common attacks, incidents and countermeasures employed within the cyber-security domain. The educational features present within the game are implemented with the aim of providing educational value, which is relevant to the industry, with cyber-incidents in event 2 representing the cyber-incidents happening everyday in a real-world setting. Cyber-attack cards in event 1 allow a user to choose to attack any players landing upon an owned property. By this design, a player who is familiar with all cyber-security knowledge (in the game) should not be concerned regarding such an attack card. On the other hand, attackers should learn that there is an element of risk when starting a cyber-attack, both in the game and in reality. In order to stimulate a player’s memory, not only the punishment and reward mechanisms as shown in red sentences on Fig. 2(a-2), but also images representing specific attacks on the pre-marked board are formulated.

4 Results

4.1 Experimental Setup

An experiment was conducted on a set of 30 undergraduates and 2 academic professors teaching an ‘Introduction to Cyber-Security’ module in a Japanese university for 4 years. After learning the rules of play, students played “Cyberpoly” in pairs, while the two professors firstly were guided to add new questions through content management interfaces (as shown in Fig. 1) to the game and then play against each other. Students were required to respond to a questionnaire consisting of 16 questions on 3 dimensions: Game mode, game design and system usability [7]. Response options are a seven-point Likert scale (1–3: strongly to slightly disagree, 4: neutral, 5–7: slightly to strongly agree). Additionally, students were asked 3 open questions for comments and suggestions in addressing these 3 dimensions. Academic professors were required to evaluate the game from

a teacher's perspective, focusing not only on the content of the game, but also on the evaluation of the management system. Therefore, in addition to the first 3 questions related to game modes and a corresponding open question asking for suggestion addressing game modes (these 4 questions are the same as the first 4 questions in the questionnaire for students), a one hour in-depth interview was conducted, to collect more detailed feedback from the two professors.

4.2 Student and Teacher Feedback

The first question asked for feedback on whether a single player mode would be useful for memorising questions and gameplay. The students' average rating of this question is 4.6 (S.D. = 1.47). 4 student participants (2 strongly disagree, 1 disagree, 1 slightly disagree) mentioned that the single mode is similar with doing a exam, so it is not necessary; 16 student participants (2 strongly agree, 7 agree, and 7 slight agree) would still prefer a single player mode to practice alone first, prior to joining a multiplayer game mode, which is consistent with the opinion of two academic professors (both agree). In terms of the multiplayer game mode, the students' average ratings of question 1-2 (The ability for multiple players to play together in groups makes the game more enjoyable.) is 6.37 (S.D. = 0.66) while the professors' ratings both at 5; and the average rating of question 1-3 (The ability for multiple groups to play the game at the same time is useful.) is 5.7 (S.D. = 1.00) while the professors' ratings are both 5.

In terms of Educational element and game design feedback from students, the average rating of question 2-1 (The design of the game interface is attractive and appealing.) is 5.80 (S.D. = 0.79); the average rating of question 2-1 (The game-play is fun and motivates the player to keep playing.) is 5.67 (S.D. = 0.87); the average rating of question 2-3 (The system is useful because it can support learning some cyber security knowledge.) is 6.00 (S.D.=0.52). These suggest that most participants agreed that the game may assist in enabling motivation to learn cyber-security knowledge.

In terms of system usability feedback from students, the descriptive data is shown in Table 1. Among these 10 statements, questions 3-1, 3-3, 3-5, 3-7 are positive statements, which acknowledge the system usability, and the rest are negative statements. The average rating of the 5 positive statements is 5.83 (> 4 (the neutral point)); the average rating of the 5 negative statements is 2.63 (< than 4 (the neutral point)). This suggests that most of student participants are satisfied with the system usability.

Given that the current interface allows addition, removal and editing of cyber-security events and questions, the professors identified several areas after being interviewed, which could enhance future iterations of the interface. Firstly, they requested the introduction of an AI player, to ensure that games are playable, without a human opponent. Secondly, they requested the addition of a spectator mode, to allow players, to view other tables without joining. This allows those who are unfamiliar with gameplay or lack security knowledge to learn from the perspective of bystanders. Regarding the educational elements and basic

Table 1. The descriptive data of rating of the system usability.

Question (id)	Mean	S.D.
3-1. I think that I would like to use this system frequently	4.83	1.16
3-2. I found the system unnecessarily complex	2.97	1.22
3-3. I thought the system was easy to use	6.10	0.40
3-4. I think that I would need the support of a technical person to be able to use this system	2.40	0.92
3-5. I found the various functions in this system were well integrated	6.03	0.55
3-6. I thought there was too much inconsistency in this system	2.43	1.05
3-7. I would imagine that most people would learn to use this system very quickly	6.27	0.51
3-8. I found the system very cumbersome to use	2.07	0.68
3-9. I felt very confident using the system	5.93	0.51
3-10. I needed to learn a lot of things before I could get going with this system	3.30	1.19

design of the game, the professors believed that the game may promote cyber-security learning; however, more attractive elements could be brought in the game. Suggestions include the addition of features such as introduction of a mechanism that allow landlords to set up various in-game security devices to reinforce their lands. Regarding the design of the game, the professors agreed that it has a potential as a good platform for cyber-security learning, in particular, through the rule of attack cards (using a attack card whilst taking the risk of double payment).

5 Conclusion and Future Work

In this paper, we have proposed the MEMORABLE framework for multi-player customisable serious games in cyber-security and implemented an instance of this platform via the ‘Cyberpoly’ game, demonstrating how this approach increases the motivation of students. In summary, though triggering a variety of cyber-security incidents, using cyber-attack cards to attack other players, and responding to questions about cyber-countermeasures, the player of “Cyberpoly” would be able to learn knowledge about cyber-security in the game. By aiming to reach the in-game goal, players may achieve the real-life goal of acquiring new cyber-security knowledge. Further to this, the introduction of a management interface ensures that instructors may tailor the game to their module’s content or player knowledge level.

The feedback from users suggests that the game achieves the goal of stimulating learning interests and motivating players to learn the cyber-security content, through rewards and punishment. Whilst successful, the current game state is relatively primitive and lacking in important features, such as features enabling

interaction between players, promoting meta-knowledge about the learning process to the learner, and further flexibility, in terms of expansion of topics and content. Thus, we aim to implement communication functionality between learners in future iterations of the game. By discussing the tasks presented, learners may achieve further knowledge retention. Limitations to this work are presented through a focus upon the cyber-security field, however, the approach may be generalised to any educational domain through adapted placement of the game elements.

References

1. Cybersecurity investment 2020. <https://www.canalys.com/newsroom/cybersecurity-investment-2020>
2. Cybersecurity lab. <https://www.pbs.org/wgbh/nova/labs/lab/cyber/>
3. Cybersprinters: Game and activities. <https://www.ncsc.gov.uk/information/cybersprinters-game-and-activities>
4. Keep tradition secure. <https://keeptraditionsecure.tamu.edu/>
5. Targeted attack: The game - defend your data. Choose wisely. Succeed or fail. <http://targetedattacks.trendmicro.com/>
6. Uk National Crime Agency: Cybergames. <https://cybergamesuk.com/cybergames>
7. Brooke, J.: Sus: A quick and dirty usability scale. *Usabil. Eval. Ind.* **189** (1995)
8. Garzotto: Investigating the educational effectiveness of multiplayer online games for children. In: *Proceedings of the 6th International Conference on Interaction Design and Children*, pp. 29–36 (2007). <https://doi.org/10.1145/1297277.1297284>
9. Kordaki, M., Gousiou, A.: Digital card games in education: a ten year systematic review. *Comput. Educ.* **109**, 122–161 (2017). <https://doi.org/10.1016/j.compedu.2017.02.011>, <https://www.sciencedirect.com/science/article/pii/S036013151730043X>
10. Laal, M., Ghodsi, S.M.: Benefits of collaborative learning. *Proc. Soc. Behav. Sci.* **31**, 486–490 (2012). <https://doi.org/10.1016/j.sbspro.2011.12.091>, <https://www.sciencedirect.com/science/article/pii/S1877042811030205>, World Conference on Learning, Teaching and Administration - 2011
11. Lo, J.J., Kuo, T.Y.: A study of parent-child play in a multiplayer competitive educational game. In: *2013 IEEE 13th International Conference on Advanced Learning Technologies*, pp. 43–47 (2013)
12. Putz, L.M., Hofbauer, F., Treiblmaier, H.: Can gamification help to improve education? Findings from a longitudinal study. *Comput. Hum. Behav.* **110**, 106392 (2020). <https://doi.org/10.1016/j.chb.2020.106392>, <https://www.sciencedirect.com/science/article/pii/S074756322030145X>
13. Seah, E.T.W., Kaufman, D., Sauvé, L., Zhang, F.: Play, learn, connect. *J. Educ. Comput. Res.* **56**, 675–700 (2018)
14. Shi, L., Cristea, A.I.: Motivational gamification strategies rooted in self-determination theory for social adaptive e-learning. In: Micarelli, A., Stamper, J., Panourgia, K. (eds.) *ITS 2016. LNCS*, vol. 9684, pp. 294–300. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-39583-8_32
15. Storm: 90% of security incidents trace back to pebkac and id10t errors (2015). <https://www.computerworld.com/article/2910316/90-of-security-incidents-trace-back-to-pebkac-and-id10t-errors.html>

16. Toda, A.M., et al.: A taxonomy of game elements for gamification in educational contexts: proposal and evaluation. In: 2019 IEEE 19th International Conference on Advanced Learning Technologies (ICALT), vol. 2161, pp. 84–88. IEEE (2019)
17. Wendel, V., Gutjahr, M., Göbel, S., Steinmetz, R.: Designing collaborative multiplayer serious games: escape from Wilson island - a multiplayer 3D serious game for collaborative learning in teams. *Educ. Inf. Technol.* **18** (2013). <https://doi.org/10.1007/s10639-012-9244-6>
18. Wendel, V., Konert, J.: Multiplayer serious games. In: Dörner, R., Göbel, S., Effelsberg, W., Wiemeyer, J. (eds.) *Serious Games*, pp. 211–241. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-40612-1_8



Improving Program Matching to Automatically Repair Introductory Programs

Maheen Riaz Contractor^(✉) and Carlos R. Rivero^{ID}

Rochester Institute of Technology, Rochester, USA
mc1927@rit.edu, crr@cs.rit.edu

Abstract. Automated program repair is a promising approach to deliver feedback to novice learners at scale. CLARA is an effective repairer that uses a correct program to fix an incorrect program. CLARA suffers from two main issues: rigid matching and lack of support for typical constructs and tasks in introductory programming assignments. We present several modifications to CLARA to overcome these problems. We propose approximate graph matching based on semantic and topological information of the programs compared, and modify CLARA's abstract syntax tree processor and interpreter to support new constructs and tasks like reading from/writing to console. Our experiments show that, thanks to our modifications, we can apply CLARA to real-world programs. Also, our approximate graph matching allows us to repair many incorrect programs that are not repaired using rigid program matching.

Keywords: Program repair · Approximate graph matching

1 Introduction

The number of novice programming learners is on the rise [6]. It is not possible to manually assist these learners, making automated approaches appealing [11, 16]. Automated program repair consists of finding a way to repair errors that exist in software, potentially without human involvement [12]. We focus on test-based repair that performs the repair based on test cases provided as input. Current test-based approaches deliver the repairs found as feedback to learners [7, 13–17].

Among these approaches, we focus on CLARA [7] because of its following features: 1) It supports programs in Python 3.x, which is a popular choice among novice learners; 2) It relies on existing correct programs to repair incorrect programs; 3) It separates the program matching process from the program repair process; and 4) It is open source [3]. Even though these are desirable features, one of CLARA's main drawbacks is its rigid program matching [8]: correct and incorrect programs must have the same control statements (conditions and loops)

This material is based upon work supported by the National Science Foundation under Grant No. 1915404.

in the same order to match. This is a significant constraint in practice since such a correct program may not exist, or may not even be possible [9].

A previous approach has focused on mitigating this problem [8]. For each comparison between a correct and an incorrect programs, it applies program refactoring to the correct program until their control statements match. This approach has several drawbacks: 1) Combinations of transformations are applied to the correct program until it matches, if found, the incorrect program, which is computationally expensive; 2) The transformations must not change the logic of the correct program; otherwise, the correct program may not be correct anymore; 3) The set of transformations must be defined in advanced and, because of the previous problem, they are not trivial to define.

In this paper, we explore a different approach to improve CLARA’s matching flexibility based on approximate graph matching. We present CLARA’s graph matching algorithm and our proposed algorithm that considers semantic and topology information of the graph formed by the control statements of the programs (control flow graphs). Furthermore, the original source code of CLARA does not support many options that are mandatory in introductory programming assignments, such as reading from/writing to console, import statements, or calls to functions present in third-party libraries. Our experiments involving real-world correct and incorrect programs from three problems from Codeforces show that CLARA is able to effectively repair many of the incorrect programs evaluated using our approximate approach. For every problem, every incorrect program was compared against 50 correct programs sorted by date. On average, our approach repaired 32% more incorrect programs than rigid matching. Moreover, a maximum of one extra correct program was needed by our approximate approach to repair the additional incorrect programs.

The rest of this paper is as follows: Sect. 2 describes CLARA; Sect. 3 describes our modifications to CLARA’s AST processor and interpreter; Sect. 4 presents our approximate graph matching approach; Sect. 5 summarizes our experimental results. Section 6 presents our conclusion.

2 The Automated Program Repairer CLARA

The Automated Clustering and Program Repair tool, CLARA [7], was first introduced in 2016. It was created to provide feedback to students in introductory programming courses. CLARA is a tool that works on C++, Java, and Python. In this paper, we focus on Python. CLARA relies on abstract syntax trees and control flow graphs. On one hand, an abstract syntax tree (AST) is a tree representation of the source code containing nodes, where every node represents a language construct or operation. On the other hand, a control flow graph contains the information of control flow statements like conditions and loops, and encodes the order of evaluation of the program.

Figure 1 presents CLARA’s workflow. The AST processor takes each program as input, parses it and creates a model. Both correct and incorrect models, along with the test case, are provided as input to the repair task. During the

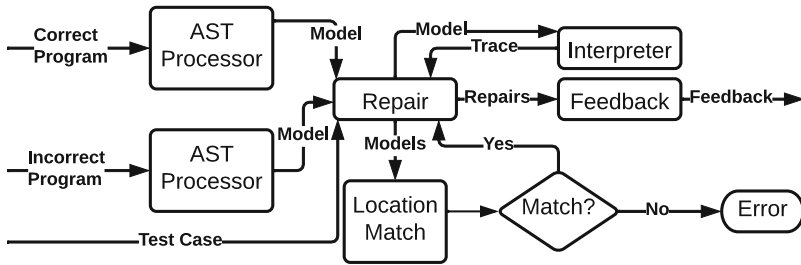


Fig. 1. CLARA's workflow for matching and repairing programs

repair process, CLARA provides each model to the interpreter, which returns its executed trace based on the input test case. CLARA then uses the repairs found to generate feedback that is delivered to the user.

Below, we present CLARA's model, and its matching and repair processes.

2.1 Model, Locations and Expressions

Before performing any program repair or matching, CLARA creates models for both the correct and incorrect programs. Each model is created based on the *ast* module of Python, which creates the AST of a program. CLARA traverses the AST and creates a model. The entire model consists of one program containing multiple functions. Each function consists of multiple locations. Locations contain expressions and are created based on branching control flow statements like if or while statements. Other control flow statements like function calls or sequencing of statements are not involved in location creation. If a function contains no branching control flow statement, a single location is created. Locations also contain information about which location to go to next, called transitions. There are two types of transitions, true and false. A true/false transition contains the next location to go to if the conditional expression inside the location evaluates to true/false. However, it is possible to have expressions inside a location that do not evaluate to a Boolean value, in which case, they will unconditionally go to a specific location. For example, transitioning back to the program body after the execution of the then or else branch inside an if condition. To help maintain consistency, these locations also contain true and false transitions, but its false transition always points to None, and the true transition always points to the next location.

CLARA contains three types of expressions: variables (*Var*), operators (*Op*), and constants (*Const*). A *Const* can be a string, byte, number, or a name constant. A *Var* represents variables, so it only supports strings. *Op* are computations involving any kind of operation, such as creating a list, set, or tuple, expressions involving comparisons, or conditions. Every *Op* contains two components: the name of the operation, and the arguments it has to operate on. Depending on the type of operation, an *Op* may contain different number of

arguments. For example, the operator *GetElement*(x, i) entails getting an element from a list, dictionary, or set that contains two arguments, the object x it needs to get the element from, and the element index i . *SetInit*, which creates a set, has multiple arguments as each argument is an element inside the set.

Control flow statements are also *Op* expressions. However, while processing those statements, CLARA makes changes to the model. As mentioned above, processing control flow statements results in the addition of extra locations to the model. The number of locations is different for each control flow statement. If the program contains an if condition, three or four locations will be added: one for the condition of the if statement, another for the expressions inside the then branch, one for the expressions after the if statement, and, finally, another for the expressions inside the else branch, if any. Loops always result in the addition of three locations: condition, expressions inside the loop, and after the loop.

CLARA restricts its models by requiring a variable to appear only once on the left side of an expression per location. It achieves this by inspecting all the declarations of a variable and nesting those declarations in its last use. Hence, during the repair and matching processes, where variable values have to be matched between programs, there is only one value per location. Also, every time CLARA uses a variable that has been defined earlier in an expression, rather than using the original variable, it converts it to a different version. CLARA uses this to determine whether a variable is being defined or used.

Since CLARA has a Python processor that creates its model, it also has an interpreter that helps execute this model in Python. The interpreter visits each expression in the model and recursively executes them. The interpreter contains functions for every type of operator

2.2 Single Function Matching and Repair

Program matching is performed between two programs expected to be correct and incorrect according to a test case. CLARA analyzes their structures and creates models from each program. Then, for every variable in one program, it finds a matching with a corresponding variable in the other program. This process is driven by variable tracing, involving comparing values of variables at each location. If the matched variables hold the same values at every point in the trace of the program, the two programs are determined as a match. Matching requires programs to have the same control flow. In practice, it is not easy to find programs that contain the same control flow since a specific task can be implemented in many different ways [10]. Therefore, it may be challenging or even impossible to find multiple correct programs that match.

Similar to program matching, the repair process starts by creating models from each of the programs and checking for a match in the control flow. Additionally, CLARA requires the programs to have at least one function in a program, and the same number of functions overall in the programs being compared. These functions must also have the same names and cannot be nested. These features determine the structure of the program. If there is a difference in the structure of the programs being repaired, CLARA throws an error (structure mismatch)

and does not run. If we know the control flow of a program, we can follow the execution of a variable and, hence, be able to pinpoint its value at any point in the program. Therefore, CLARA only repairs incorrect programs that have the same control flow as existing correct programs. In practice, however, it is not very common for programs to contain the same order of loops and conditions, limiting the amount of programs CLARA can repair.

Algorithm 1: CompareCFGs

in : u location in G_C ; $G_C = (U, E)$ control flow graph (correct program); v location in G_I ; $G_I = (V, F)$ control flow graph (incorrect program),
 $\phi : U \rightarrow V$ program matching
out: Whether G_C and G_I match

- 1 // If u is in ϕ , it must be mapped to v
- 2 **if** $u \in \text{dom } \phi$ **then**
- 3 | **return** $\phi(u) = v$
- 4 // If v is in ϕ , it must be mapped to u
- 5 **if** $v \in \text{ran } \phi$ **then**
- 6 | **return** $u = u'$ such that $\phi(u') = v$
- 7 // Map u to v
- 8 $\phi(u) := v$
- 9 // u' and v' are neighbors of u and v for true transitions
- 10 Let $u \xrightarrow{\text{True}} u', v \xrightarrow{\text{True}} v'$
- 11 // u'' and v'' are neighbors of u and v for false transitions
- 12 Let $u \xrightarrow{\text{False}} u'', v \xrightarrow{\text{False}} v''$
- 13 **return** $\text{CompareCFGs}(u', G_C, v', G_I, \phi) \wedge \text{CompareCFGs}(u'', G_C, v'', G_I, \phi)$

Since models contain multiple locations, they are matched based on their transitions as presented in Algorithm 1. It takes as input two control flow graphs G_C , derived from the correct program, and G_I , derived from the incorrect program, two locations in both graphs u and v , and a mapping ϕ . First, we make sure that u and v are mapped to the same locations if they are already present in ϕ , i.e., it is a bijective function. If this is not the case, we map u to v and proceed to compare its neighbors based on two recursive calls: one for the true transitions and another for the false transitions. In the initial call, both u and v are the initial locations of the programs. If the returned value is False, then there is a structure mismatch; otherwise, the programs match based on ϕ .

After building ϕ , during the repair process, CLARA traverses these locations and, for every variable x in location u of the correct program, CLARA searches for a match for a variable in the $\phi(u)$ location of the incorrect program. Due to the nature of CLARA's modeling, since a variable can only have one expression per location, it can only hold one value, reducing thus the complexity of comparing with other variables. During the repair process, a mapping of a variable only deals with its specific expression and the expression it is mapped to. While mapping a variable from the correct program, every variable in the incorrect program

is compared, whether or not it has already been mapped. Two variables match if their corresponding expressions evaluate to the same values using the same inputs, where the inputs denote the variables both the expressions depend on. This match is evaluated using a cost. Since both the correct expression and the incorrect expression depend on different variables, the cost of a match is denoted by the number of changes it takes to turn the incorrect program's expression into the correct program's expression, where each variable in the correct program's expression has been replaced by a variable from the incorrect program.

In the repair process, CLARA assumes that the number of variables in the correct program is the minimal number of variables needed to solve the program, so the number of variables in the incorrect program needs to exactly match with the number of variables in the correct program. Therefore, if the incorrect program contains extra variables, CLARA will suggest deleting the variables it cannot match. If the incorrect program contains fewer variables, CLARA will suggest creating new variables that were detected during the matching process. CLARA computes a cost for every location and relies on a linear programming solver to suggest variable matches such that the overall cost is minimized. Based on the matches and the cost, the final set of repairs are suggested, which can be of three types: variable additions, variable deletions, or variable changes.

3 Modifications to AST Processing and Interpreting

This section discusses modifications to CLARA's original source code to adapt it to repair programs in introductory programming assignments.

Most introductory programming assignments make use of printing to console statements to verify whether a program is correct or not. The verification of correctness determines whether a program needs to be repaired or not. Similar to other programming languages, certain computations in Python can be performed using variables or within the parentheses of a print statement, removing the need for variables. Hence, while evaluating the similarity between two programs or performing a repair, it is important to match the contents inside these print statements. CLARA's original source code did incorporate the parsing of print statements. However, with Python 3.x, the print operation switched from a statement to a function, which made print statements no longer supported. We updated the section of the Python AST processor that checks for function calls by checking if the print function was called and, accordingly, updated the model. Once the print function was correctly processed, comparison of the print operation during the matching and repair processes was automatically handled.

CLARA's original parser did not support Boolean values, only numbers and strings; therefore, statements containing False and True led to parsing errors. Since CLARA categorizes expressions into constants, variables and operations, we added another constant to the AST processor of type Boolean that can process the values False and True. It is also common in introductory programming assignments to require the use of external libraries, such as *math*, *re*, and *string*. These libraries provide functions, such as *sqrt*, *ceil*, *floor*, and *ascii*. Thus, for

CLARA to be able to execute these functions while repairing or matching, it is important to contain the functionality to parse and record the data inside these import statements. The original version of CLARA ignored all import statements, causing the program to occasionally crash during the repair and/or match processes as they could not be found while executing the function trace. Since CLARA has its internal version of an AST processor and interpreter for Python, after parsing and processing the import statements, we saved the statements in a manner such that a function can be successfully recognized during execution by the interpreter. These functions can appear by themselves, or with a library/alias attached, making the task more complex. As a result, we created a section in the AST processor for import processing and saved the imports and their data in a nested global dictionary, which was then provided to the interpreter to be accessed during execution. Furthermore, the original source code defined common functions not defined in the input program, such as *max*, *sum*, or *len*. CLARA's interpreter contains its own versions of these functions implementing their functionality. Since Python has a large collection of built-in functions, manual addition to the interpreter of every single function was prohibitive, causing CLARA to halt during the matching and repair processes if these functions are not present. We implemented an approach to circumvent the need for manual addition of every function by accessing the dictionary containing built-in Python functions and, then, looking them up during execution.

Python 3.x introduces a new way of variable assignment, which is based on list deconstruction or unpacking. For example, the following code excerpt respectively assigns the values 1, 2 and 3, to variables a, b and c: *a, b, c = [1, 2, 3]*. We updated the AST processor for recognizing and processing multiple assignments from a single statement. We separated the assignments with their corresponding expressions and added each assignment as a separate expression to the model. Without this modification, CLARA throws an error stating that multiple assignments are not possible, and halts the match and repair processes.

In introductory programming assignments, test cases are typically provided as input from console. CLARA's original source code does not support standard input. All inputs have to be provided via command line as function arguments. Since input arguments can be multiple lines long and are not always of a fixed length, this is not possible. Therefore, we updated CLARA to read all console inputs using an argument file and store them in an iterator that is accessible by the interpreter. We updated the interpreter to handle function calls to *input* separately. Every time the function is called, rather than actively executing the function, we return the next element from the iterator to simulate reading from console. However, while adding this feature, we encountered another problem. Since CLARA nests the expressions of variables while creating its model, it creates copies of the *input* function when there should only be one call. For example, the Python statement *a, b, c = input().split()* is split into three statements in CLARA's model: *a = GetElement(split(input()), 0)*, *b = GetElement(split(input()), 1)* and *c = GetElement(split(input()), 2)*; therefore, *input* is called three times instead of one. We updated the AST pro-

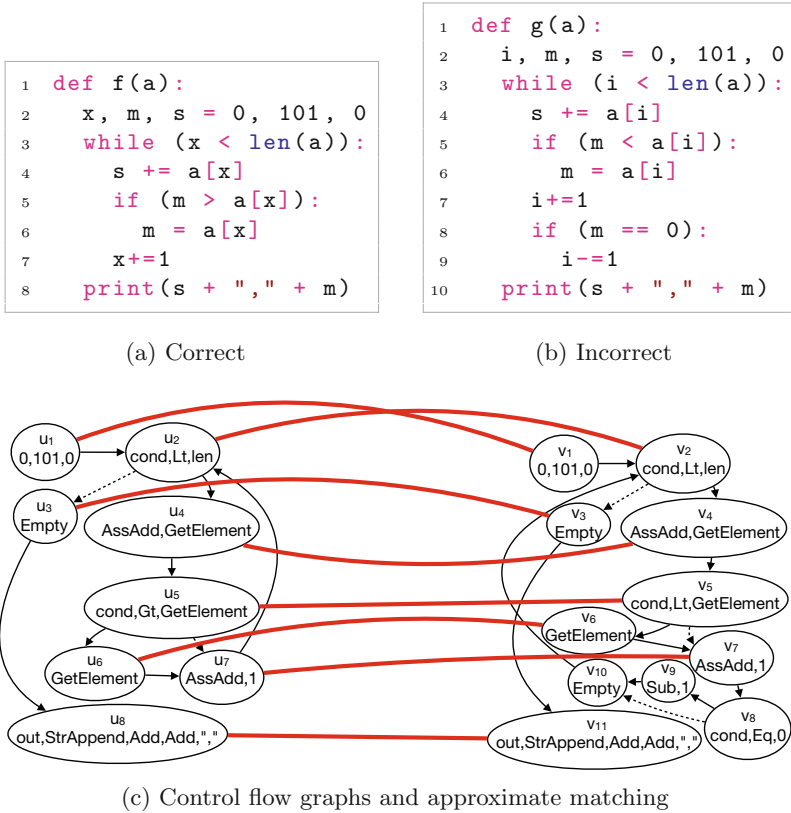


Fig. 2. Programs that aim to find the minimum in a and compute its summation

processor to create a new variable to call the *input* function, and for the expression calling the *input* function to use the new variable instead. Therefore, using the above example, the new model contains four statements as follows: $x = input()$, $a = GetElement(split(x), 0)$, $b = GetElement(split(x), 1)$ and $c = GetElement(split(x), 2)$.

4 Approximate Graph Matching

As mentioned earlier, one of CLARA’s main limitations is that it requires both the correct and the incorrect programs to have the same control flow. We propose an approximate graph matching algorithm that finds an approximate mapping between the control flow graphs of the correct and incorrect programs.

Figure 2 presents two programs we use to illustrate our approximate graph matching. These programs take in as input, an array containing integers between 0 and 100 and aim to find the minimum element as well as add all the elements in a . However, rather than finding the minimum element, the incorrect program

never changes the value of m , so the if condition is never executed. For each node in the control flow graph, we include a multiset of labels that contains all the semantic information we extract from the expressions used in the program. Each label is either a constant or an operation. Constants are useful to identify key values in the program, e.g., in Fig. 2, the maximum value in the input array is 100, so every number must be strictly less than 101. Operations allow us to discern the type of statement, e.g., addition or subtraction.

Figure 2c presents the control flow graphs derived from the programs mentioned above in which these multisets of labels are attached to each node in the graphs. The left side contains the correct graph and the right side contains the incorrect graph. We use $L(u)$ to denote the multiset of labels of node u . Dotted and solid edges represent false and true transitions, respectively. The node with *Empty* label represents a location in the model that contains no expressions.

We aim to find a matching such that there are no structure mismatches. Rather than modifying the correct program as proposed by Hu et al. [8], we decided to modify the incorrect program as those changes can be recorded and mentioned as feedback to the user. Note that using (uncontrolled) program refactoring can modify a program so thoroughly it no longer resembles its original form. Since the ultimate goal is to suggest repairs over the incorrect program, we decided to make changes only to it, i.e., the correct program remains unchanged. This matching considers both semantic (expressions) and topological (control flow) information of the graphs.

Algorithm 2 computes our approximate graph matching between the correct and the incorrect programs. We explore every possible permutation that maps all nodes in U to all nodes in V . We aim to find the permutation with the maximum similarity in terms of semantic and topological information. For each permutation ϕ , we inspect each pair (u, v) such that $\phi(u) = v$. We consider the Jaccard similarity between the labels of u and v ; then, we check if the neighbors of u and v , both false and true transitions, match with each other according to ϕ . All these similarities are combined and normalized into a single similarity. We keep the matching with the maximum similarity. Since the number of permutations can be large, we introduce an optimization such that the permutations explored first are those with higher label similarity. In our experiments, we impose a limit to the maximum number of permutations we explore. After applying our algorithm to the graphs shown in Fig. 2c, we obtain the approximate graph matching depicted in the figure, where the thick solid lines between nodes from the correct to the incorrect graphs represent the matching.

Finally, we exploit the matching found to recreate the input that CLARA’s repair process requires. Since the repair process takes the models of both programs as input, we use ϕ and the correct program’s model to recreate the incorrect program’s model. There are three possible options: 1) $|U| < |V|$, nodes need to be removed from the incorrect program model; 2) $|U| > |V|$, nodes need to be added to the model of the incorrect program; 3) $|U| = |V|$, there are no nodes to be removed or added, but the edges might need rearrangement. To recreate the model, we use the edges in the correct program. In our example, the new

Algorithm 2: ApproximateGraphMatching

```

in :  $G_C = (U, E)$  control flow graph (correct program);  $G_I = (V, F)$  control
      flow graph (incorrect program)
out:  $\phi_m : U \rightarrow V$  approximate program matching
1  $\phi_m := \{\}, s := 0$ 
2 // Get all permutations between  $U$  and  $V$ 
3 foreach  $\phi \in \text{Permutations}(U, V)$  do
4   // Initialize current global similarity
5    $c := 0$ 
6   foreach  $u$  and  $v$  such that  $v = \phi(u)$  do
7     // Label similarity between  $u$  and  $v$ 
8      $l := \text{Jaccard}(L(u), L(v))$ 
9     // Neighbors of  $u$  and  $v$ 
10    Let  $u \xrightarrow{\text{True}} u', v \xrightarrow{\text{True}} v'$ 
11    Let  $u \xrightarrow{\text{False}} u'', v \xrightarrow{\text{False}} v''$ 
12    // Edge similarity of neighbors
13     $n := 0.5$ 
14    if  $v' = \phi(u') \wedge v'' = \phi(u'')$  then
15      |  $n := 1$ 
16    else if  $v' \neq \phi(u') \wedge v'' \neq \phi(u'')$  then
17      |  $n := 0$ 
18    // Combine and normalize similarities
19     $c := c + (.5(l + n))/|\phi|$ 
20  if  $c > s$  then
21    | Update  $s := c$  and  $\phi_m := \phi$ 

```

incorrect program's model does not contain nodes v_8, v_9 and v_{10} . The edge from v_{10} to v_2 is replaced by a True edge from v_7 to v_2 .

5 Experimental Results

Our implementation is publicly available [1]. In our experiments, we focus on three programming problems 1560B [5], 1554A [2] and 997C [4] from the online programming website CodeForces. These problems were chosen as they fulfill the conditions of having at least one for/while loop, if condition, standard input, standard output, and had a sufficient number of user programs.

We consider a program to be invalid if it fails to pass a test case, i.e., syntax errors, timeouts and others are out of the scope of these experiments. The incorrect programs were further filtered based on the availability of test cases. Test cases longer than 50 lines are not entirely displayed by Codeforces, so we cannot rely on them. All incorrect programs that failed on such test cases were ignored. We consider a program to be repaired if, after the repairs suggested are applied, no other repairs are suggested for any of the other available test cases. Note that this is much restrictive than just considering the specific test case attempted.

Table 1. Results for problems 1560B, 1554A, and 977C using 50 correct programs sorted by date based on experimental settings *CLARA*, *AGM(L)*, *AGM(L+E)*. *CLARA* is original CLARA with the modifications described in Sect. 3; *AGM(L)* and *AGM(L+E)* are respectively our approximate graph matching approach using only labels and labels and edges.

Measurements (Problem 1560B)	<i>CLARA</i>	<i>AGM(L)</i>	<i>AGM(L+E)</i>
Control flow conflicts	9135	0	0
Comparisons repaired	373	1478	1461
Total comparisons	11150	9228	9304
Incorrect programs repaired	79	163	164
Total incorrect programs	223	223	223
Correct programs needed to minimize modification percentage	19	18	20
Minimum number of correct programs needed for all repairs	10	5	7
Measurements (Problem 1554A)	<i>CLARA</i>	<i>AGM(L)</i>	<i>AGM(L+E)</i>
Control flow conflicts	5688	0	0
Comparisons repaired	318	599	648
Total comparisons	6850	5472	6231
Incorrect programs repaired	39	81	77
Total incorrect programs	137	137	137
Correct programs needed to minimize modification percentage	13	17	20
Minimum number of correct programs needed for all repairs	6	7	7
Measurements (Problem 977C)	<i>CLARA</i>	<i>AGM(L)</i>	<i>AGM(L+E)</i>
Control flow conflicts	6668	0	0
Comparisons repaired	12	65	105
Total comparisons	8100	7482	7332
Incorrect programs repaired	10	46	65
Total incorrect programs	162	162	162
Correct programs needed to minimize modification percentage	3	4	5
Minimum number of correct programs needed for all repairs	3	3	4

For every problem, to repair all possible incorrect programs, we selected 50 correct programs sorted by date. This setting simulates an introductory programming assignment whose feedback can be automated without human intervention after we have observed a number of correct solutions. These 50 programs are all unique and provided by different users. Since the number of nodes

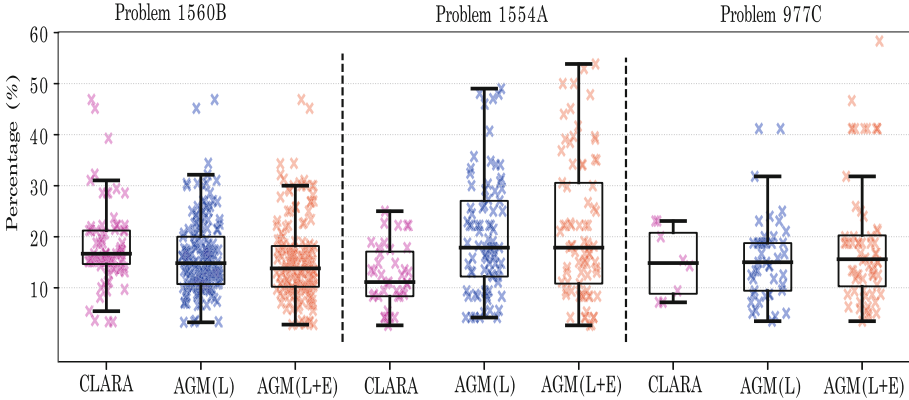


Fig. 3. Modification percentages of incorrect programs

in each control flow graph ranges from 5 to 25 in our experiments, we set a limit of 1000 in the number of permutations explored. The best approximate graph mapping was chosen among those permutations. Additionally, for each correct/incorrect comparison, we set a timeout of five minutes to provide feedback. We compare three approaches: 1) *CLARA*: original *CLARA* with the modifications described in Sect. 3; 2) *AGM(L)*: approximate graph matching (Algorithm 2) without using topological information (edges), only semantic information (labels); and 3) *AGM(L+E)*: approximate graph matching (Algorithm 2) using both topological (edges) and semantic (labels) information.

Table 1 presents our results. We observe that *CLARA* has a large number of structure mismatches for the three problems. Our approximate graph matching thus helps increase the number of valid comparisons. *AGM(L)* and *AGM(L+E)* were able to repair almost double the number of incorrect programs. Additionally, they required a maximum of one extra correct program to repair many more incorrect programs. For each repair, we also considered the percentage of the incorrect program modified. This percentage is based on the number of repairs suggested, as well as the total number of expressions in both the correct and incorrect program. Figure 3 presents box and whisker plots for the percentage of the incorrect program modified for each experiment on each problem. The median for each experiment is below 20%, which entails that, in many cases, a small percentage of the correct program was used to replace a part of the incorrect program.

6 Conclusions

Our modifications to the abstract syntax tree processor and interpreter, as well as our approximate graph matching algorithm has made *CLARA* more effective to deal with real-world programs. In future work, we will analyze optimizations to avoid exploring the whole search space of permutations during approximate

graph matching. Additionally, even though repairs can be delivered to users, novice learners will not be able to understand and apply them. We aim to post-process these repairs to be delivered to users in an effective way.

References

1. Approximate graph matching for CLARA. github.com/mcontractor/clara/
2. Cherry. codeforces.com/problemset/problem/1554/A
3. CLuster And RepAir tool for introductory programming assignments. github.com/iradicek/clara/
4. Less or equal. codeforces.com/problemset/problem/977/C
5. Who's opposite? codeforces.com/problemset/problem/1560/B
6. Camp, T., Zweben, S.H., Buell, D.A., Stout, J.: Booming enrollments: survey data. In: SIGCSE, pp. 398–399 (2016)
7. Gulwani, S., Radicek, I., Zuleger, F.: Automated clustering and program repair for introductory programming assignments. In: PLDI, pp. 465–480 (2018)
8. Hu, Y., Ahmed, U.Z., Mehtaev, S., Leong, B., Roychoudhury, A.: Re-factoring based program repair applied to programming assignments. In: ASE, pp. 388–398 (2019)
9. Marin, V.J., Contractor, M.R., Rivero, C.R.: Flexible program alignment to deliver data-driven feedback to novice programmers. In: ITS, pp. 247–258 (2021)
10. Marin, V.J., Pereira, T., Sridharan, S., Rivero, C.R.: Automated personalized feedback in introductory Java programming MOOCs. In: ICDE, pp. 1259–1270 (2017)
11. Marin, V.J., Rivero, C.R.: Clustering recurrent and semantically cohesive program statements in introductory programming assignments. In: CIKM, pp. 911–920 (2019)
12. Monperrus, M.: Automatic software repair: a bibliography. CSUR **51**(1), 1–24 (2018)
13. Piech, C., Huang, J., Nguyen, A., Phulsuksombati, M., Sahami, M., Guibas, L.J.: Learning program embeddings to propagate feedback on student code. In: ICML, pp. 1093–1102 (2015)
14. Pu, Y., Narasimhan, K., Solar-Lezama, A., Barzilay, R.: sk_p: a neural program corrector for MOOCs. In: OOPSLA Workshops, pp. 39–40 (2016)
15. Rolim, R., et al.: Learning syntactic program transformations from examples. In: ICSE, pp. 404–415 (2017)
16. Singh, R., Gulwani, S., Solar-Lezama, A.: Automated feedback generation for introductory programming assignments. In: PLDI, pp. 15–26 (2013)
17. Wang, K., Singh, R., Su, Z.: Search, align, and repair: data-driven feedback generation for introductory programming exercises. In: PLDI, pp. 481–495 (2018)



Gamification, User-Centered Design and Learning Objectives as the Basis for a Minigame-Based Cardiovascular Anatomy ITS

Reva Freedman^(✉), Virginia Naples, Ian Sullivan, Lucas Edwards,
and Dean LaBarbera

Northern Illinois University, DeKalb, IL 60115, USA
{rfreedman,vlnaples}@niu.edu

Abstract. We are building an ITS based on a three-dimensional matrix of minigames to teach human anatomy and physiology. The three dimensions are the bodily organ, the learning objective and the user's point of view, whether pedagogical or one of a variety of clinical tasks. The intended audience is advanced undergraduates majoring in the medical and allied health professions who are required to master this topic. The system is based on the principles of reusable gamification and design elements, user-centered design and explicit learning objectives. We illustrate the initial games we have developed and explain the role of the design principles in speedy and effective development. Our goal is to increase student success by aligning the system to the way students prefer to learn.

Keywords: Intelligent tutoring system · Anatomy education · Minigames

1 Introduction

Many students in the allied health professions find the required course in anatomy and physiology to be one of the most difficult in their program. Since we know that gaming is popular among traditional-age college students, especially on cell phones, we are studying whether a game-based ITS would be successful for this population of learners. Several recent survey articles in both medical and other types of education [2, 4, 9, 13] have shown ambiguous results. Even when experiments have shown successful learning, some experiments have not included a control group and many authors have not yet been able to identify the specific features of gamification that have led to success.

To study this question, we are developing an ITS for this population based on a set of minigames. Based on the second author's experience in personalizing teaching strategy for different students at different points in their educational journey, our system is designed as a three-dimensional matrix of minigames. The

horizontal axis identifies the organ system involved. Since the cardiovascular system is one of the most critical systems for success in anatomy and physiology, we have begun by identifying three points on this axis, the heart, the lung, and the cardiovascular system as a whole. The vertical axis identifies the learning objective of the minigame. The third axis identifies the point of view that the user takes in the game. In most of the early games we take a purely pedagogical view of anatomy and physiology, but we know that students are motivated by applied problems that relate to their proposed careers, such as diagnosis, treatment and patient followup. Therefore we have developed one game on cardiac arrhythmia [7] that takes a clinical perspective. Students can progress through the matrix at their own speed and following their own needs and interests.

Designing the system as a matrix of minigames allows us to provide content for different organs, with different learning objectives and different points of view, while providing an organized, practical approach to system development. To date we have developed about six games including two organs (heart and lung), three stages representing the learning objectives needed for basic anatomy and physiology, and two points of view (pedagogical and clinical). We are currently developing games for a third organ system (the cardiovascular system as a whole).

Some games also have multiple levels of difficulty. As the game gets progressively harder, time pressure and other game mechanics make the game more challenging, which aids student concentration and also makes the game more fun for the experienced player.

The games are instrumented to keep track of playing time and several measures of student learning. In the next phase of development, the games will send this information to a shared database so that we can analyze student behavior and guide students through the matrix. In addition to studying student success, we will attempt to identify which gamification features have contributed to student learning.

We are using Bloom's taxonomy [1] as a guide to the type of game play elements appropriate for the different kinds of learning that students of anatomy need to accomplish. Bloom's taxonomy provides an approach to categorizing learning tasks with respect to the level of abstraction required. Students have difficulty achieving the learning objectives identified at the higher levels of the taxonomy. These difficulties are symptomatic of cognitive overload [8]. Standard textbooks do not help students overcome these challenges, and even publisher-produced computer-based supplementary materials such as thePoint or Wiley-PLUS, do not provide scaffolding for more abstract levels of student learning. Our system will ensure that students have mastered the earlier stages of a topic before continuing on to the harder ones.

This paper describes the design of the system and the role of gamification, user-centered design and explicit learning objectives in producing an ecology of minigames that will help students in the paramedical professions fulfill their potential by helping them to pass a difficult core class.

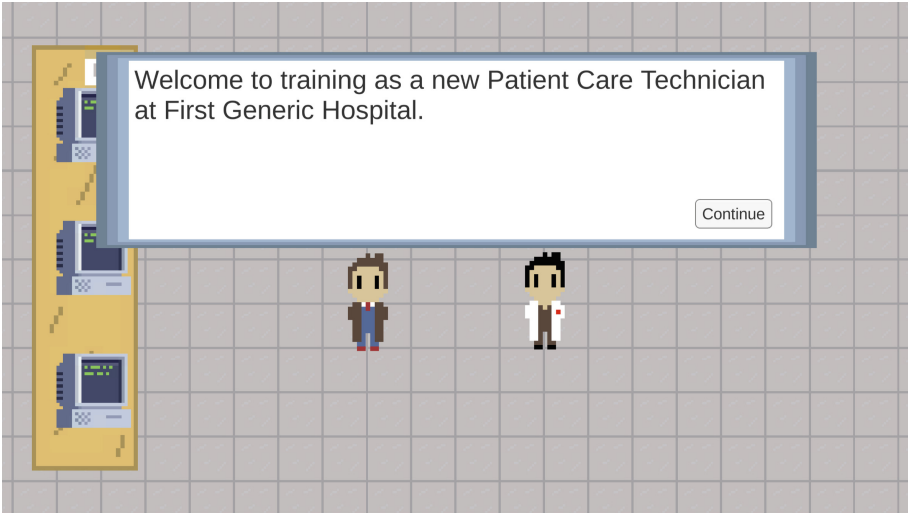


Fig. 1. Main menu/tutorial

2 Methodology

The games are implemented in C# using the Unity game engine [15]. In addition to allowing deployment on multiple platforms, including in particular cell phones and laptops, Unity allows easy access to high-level features such as graphics, animation, and an event loop.

Our development process is based on user-centered design, an iterative process that ensures that system authors remain focused on intended system users throughout all phases of design and implementation. [5] The term originated in the 1970s and was later adopted by Don Norman, a cognitive scientist and expert in usability engineering for everyday tools [11]. To accomplish this, user-centered teams need to be multidisciplinary, including domain experts and user representatives as well as system developers. In addition to a faculty member in Biological Sciences, all of our developers are themselves gamers.

Some core principles that have evolved from user-centered design include the idea that interfaces must be easy to learn, enjoyable to use, and have a consistent look and feel. These ideas are consistent with the goals of gamification.

User-centered design is always an iterative process since it is not possible to predict in advance how users will react to a design once they have the opportunity to try it [6, 12]. This concept is consistent with the academic research process.

The use of minigames reduces development time because it permits multiple team members to develop software in parallel. In addition, it reduces the complexity of individual components and provides common design elements suitable for reuse, not only reducing the time required to develop new components but also reducing the overhead in adding new developers to the team.

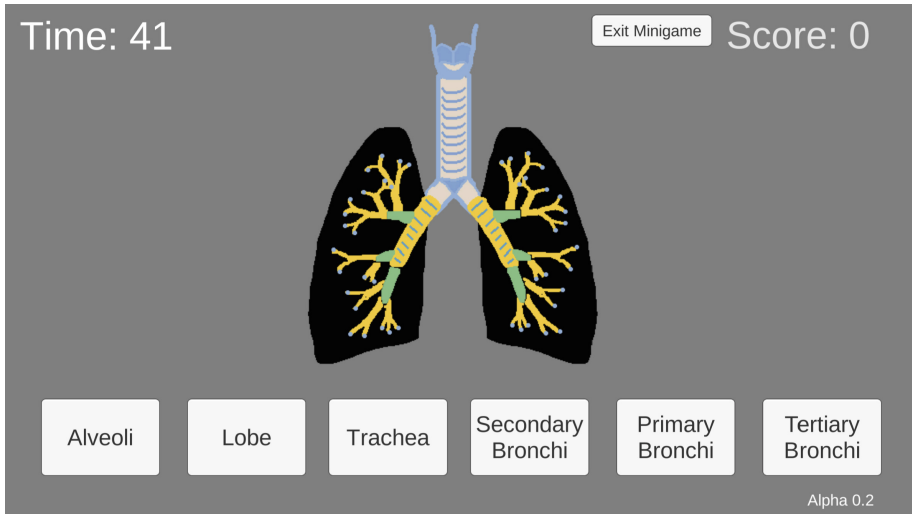


Fig. 2. Introduction to Lung Stage 1.

3 Game Descriptions

The initial view seen by the student (Fig. 1) serves both as main menu and as tutorial when needed. Since we would like the games to be as intuitive as possible, so that students spend their time playing and learning about the cardiovascular system rather than learning how to play, we have minimized the amount of required training. We do give the novice a head start by showing them how to move a character. Then they can choose which game they would like to play.

Lung Stage 1 (Fig. 2) is a simple matching game. Its goal is to teach the student the parts of the lung. The lung image starts out completely black. As the student drags each word to its correct location in the lung, that part changes to an iridescent pastel view. From largest to smallest, the student needs to label the lobes, the trachea, the primary, secondary and tertiary bronchi, and the alveoli. Lung Stage 2 starts to teach the physiology of the lung, which is continued in Lung Stage 3.

Figure 3 shows the stage 1 game for the heart. Our implementation of the anatomy and physiology of the heart differs from conventional textbooks [10, 14]. Standard anatomy textbooks make the student learn the names of the major vessels at the same time as they are trying to understand blood circulation to and from the heart. We reduce the cognitive load on the student by splitting these topics into two stages. In stage 1, students learn the most important conceptual ideas about blood circulation, i.e., how oxygenated and deoxygenated blood moves between the heart, the lungs and the rest of the body. Oxygenation happens in the lungs, but the heart is needed to pump that oxygenated blood to the rest of the body. An important concept to master is that the function of

arteries and veins between the lung and the heart is the opposite of their role in the rest of the body.

Then in stage 2, students connect this understanding with the names of the major vessels. The names of the vessels are not relevant until the student understands the big picture. Although standard textbooks do it the other way, we know that the standard textbook treatment is difficult for students to master.

Although our diagram is simpler than the one in standard textbooks, it is an accurate representation so that when students move on to more advanced learning material, whether in our system or elsewhere, they will not need to relearn material.

The white arrows in the center of the screen represent the various veins and arteries that exist within the body. The arrows are labeled by the names of the vessels and point in the direction of blood flow, either toward or away from the heart. The goal of the game is to color each arrow red or blue for oxygenated or deoxygenated blood, respectively.

When students color an arrow incorrectly, they receive feedback via a hint that helps them both play the game more successfully and also learn a key piece of the underlying content. For example, students who do not understand the role of the pulmonary veins receive the message “Remember that the pulmonary veins bring back oxygenated blood.” The messages are chosen so that if a student receives more than one of them, they form a coherent paragraph. Our eventual goal is to build an intelligent textbook that replaces the overly long and complex textbooks currently used for this course.

When all arrows are correctly colored, the game ends with a popup displaying the player’s time and providing a restart button that allows for further practice.

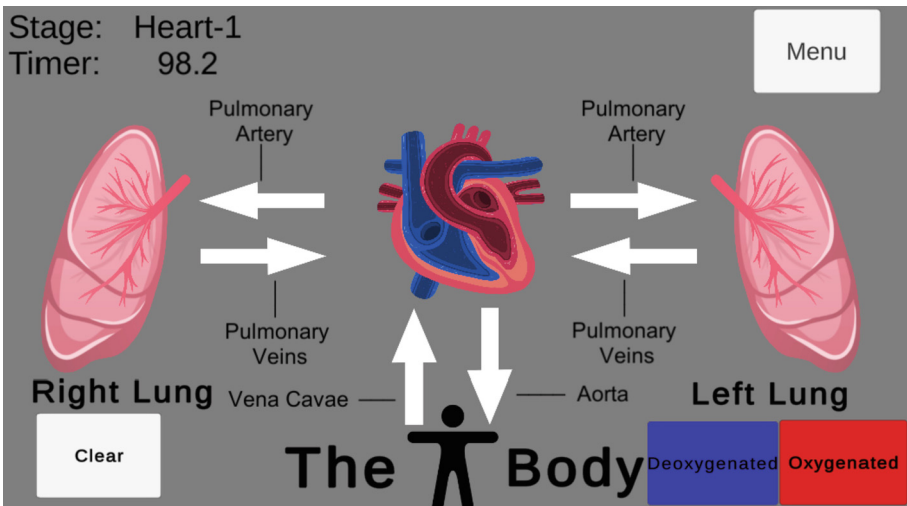


Fig. 3. Heart stage 1.

4 Discussion

As implementation has proceeded, we have found that the user-centered design process, the use of reusable design elements and gamification mechanics, and a design based on the levels of Bloom's taxonomy have worked well together to produce an integrated system.

We have attempted to reuse popular design elements wherever possible, both to reduce development cost and to provide a smoother experience for the student. The flow between screens and levels is consistent regardless of level or topic. Art and other screen design elements are repeated where possible. We have also attempted to insure the consistency of various design elements, such as element sizes and borders.

With regard to Bloom's taxonomy, the stage 1 game in each string of organ-based games involves matching or a similar remembering process. The second stage in each string incorporates aspects of level 2 and level 3 of Bloom's taxonomy. Similarly, we give immediate feedback in stage 1 and feedback at the end of a round of the game in later stages.

With regard to gamification mechanics, we have implemented feedback, points, levels and a timer. Students who prefer an untimed game can turn off the timer. We are currently implementing a leaderboard, and we are planning to implement badges. During design and programming, our developer/gamers have continually evaluated the games for intuitiveness, fun and playability.

5 Conclusions and Future Work

In this paper we have shown how the use of user-centered design, reusable gamification mechanics and design elements, and Bloom's taxonomy have enabled us to lay the foundations for an extensible system that will improve students' ability to learn the fundamentals of human anatomy and physiology, pass a required course in the topic, and make progress toward their degree and life goals.

There are many online anatomy tools available, including some for cell phones, e.g., [3], but we do not know of any that are based on a multidimensional ecosystem.

This project is especially relevant both during the Covid-19 pandemic and going forward because it allows students to study interactively even if they are not on campus. We believe that future development on cell phones is essential to make it convenient for this population of students to practice regularly.

By repeating the core design elements and game mechanics, we can potentiate the speed of game writing while keeping each minigame simple enough that student authors who participate can have a successful experience with a one-year commitment to the project.

References

1. Bloom, B.: Taxonomy of Educational Objectives. The Cognitive Domain. David McKay, New York, Handbook I (1956)

2. Donkin, R., Rasmussen, R.: Student perception and the effectiveness of Kahoot!: A scoping review in histology, anatomy, and medical education. *Anatomical Sci. Educ.* **14**(5), 572–585 (2021). <https://doi.org/10.1002/ase.2094>
3. Educational Technology and Mobile Learning: 11 free tools to teach human anatomy in 3D (2012). <https://www.educatorstechnology.com/2012/04/11-free-tools-to-teach-human-anatomy-in.html>
4. Hamari, J., Koivisto, J., Sarsa, H.: Does gamification work? – A literature review of empirical studies on gamification. In: Proceedings of the Forty-Seventh Annual Hawaii International Conference on System Sciences, pp. 3025–3034. IEEE, Piscataway (2014). <https://doi.org/10.1109/HICSS.2014.377>
5. Interaction Design Foundation: User-Centered Design (nd). <https://www.interaction-design.org/literature/topics/user-centered-design>
6. Jones, A.: 5 Principles of User-Centered Interface Design (2011). <https://www.sitepoint.com/5-principles-of-user-centered-interface-design/>
7. Kluga, B., et al.: Intelligent game-based mobile learning for contemporary anatomy education. *FASEB J.* **35**(S1) (2021). <https://doi.org/10.1096/fasebj.2021.35.S1.05299>
8. Mayer, R.E., Moreno, R.: Nine ways to reduce cognitive load in multimedia learning. *Educ. Psychol.* **38**(1), 43–52 (2008)
9. McCoy, L., Lewis, J.H., Dalton, D.: Gamification and multimedia for medical education: A landscape review. *J. Osteopathic Med.* **116**(1), 22–34 (2016). <https://doi.org/10.7556/jaoa.2016.003>
10. Moore, K., Agur, A., Dalley, A., II.: *Clinically Oriented Anatomy*, 8th edn. Wolters Kluwer, Philadelphia (2017)
11. Norman, D., Draper, S.: *User Centered System Design: New Perspectives on Human-Computer Interaction*. CRC Press, Boca Raton (1986)
12. Novoseltseva, E.: *User-Centered Design: An Introduction* (nd). <https://usabilitygeek.com/user-centered-design-introduction/>
13. Rutledge, C., et al.: Gamification in action: Theoretical and practical considerations for medical educators. *Acad. Med.* **93**(7), 1014–1020 (2018). <https://doi.org/10.1097/ACM.0000000000002183>
14. Tortora, G., Nielsen, M.: *Principles of Human Anatomy*, 14th edn. Wiley, Hoboken (2016)
15. Unity: (2021). <http://unity.com>



Investigating Clues for Estimating Near-Future Collaborative Work Execution State Based on Learners' Behavioural Data During Collaborative Learning

Yoshimasa Ohmoto^{1(✉)}, Shigen Shimojo², Junya Morita¹, and Yugo Hayashi²

¹ Shizuoka University, 3-5-1 Johoku, Naka-ku, Hamamatsu 432-8011, Japan
ohmoto-y@inf.shizuoka.ac.jp

² Ristumeikan University, 2-150 Iwakura-cho, Ibaraki, Osaka 567-8570, Japan

Abstract. Our final goal is to realize an Intelligent Tutoring System using cognitive tutor agent(s) who provide appropriate adaptive feedback according to the learner's state. We considered that estimating how learners would behave in the near future would be an important cue for providing appropriate support. In this study, we attempted to estimate whether learners will collaborate in the near future during task execution from their behavioral data. We conducted an experiment of cooperative learning between humans, and obtained data on facial expressions, gaze, and voice during the experiment. As a result, we suggest that the participant's state of execution of collaborative tasks in the near future can be inferred to some extent from multimodal information. On the other hand, it was not possible to successfully estimate whether or not participants would be actively collaborating in the near future. Extraction of features that can estimate the activity state of learners regardless of individual differences is a major task for the future.

Keywords: Near-future learner's state · Collaborative learning · Learning support

1 Introduction

Incorporating different perspectives plays an important role in promoting one's own understanding in learning. Cognitive science research has explored the successful process of collaboration between participants with different perspectives [7]. The implementation of cooperative learning to deepen understanding through such collaboration requires appropriate support by facilitators. To provide such interaction, Computer Support Cooperative Learning (CSCL) and Intelligent Tutoring System (ITS) have been proposed [5, 12].

In such systems, it is not so easy to provide appropriate support according to the state of the learner's involvement in the task activity and the mental state.

One of the challenges in CSCL is to determine whether the learner needs to be supported in the process of collaborative learning and to provide interventions in a dynamic manner. Recent research suggested that interactions based on learners' understanding and attitudes toward their tasks are important [8,9].

As for the state of the learner, indices of proficiency and understanding are intuitively considered. However, in collaborative learning, it is important for learners to actively engage in tasks and to share different perspectives with each other to gain new insights. The Interactive-Constructive-Active-Passive (ICAP) framework is a framework of human collaborative learning [4]. In ICAP framework, learners' activity states in collaborative learning are categorised as follows: Passive state, Active state, Constructive state, and Interactive state.

Such an estimation of the learner's internal state at a certain point in time is a useful clue to support cooperative learning. However, in human-agent interactions, humans often remain passive to the information given by agents [1,10]. When considering guiding the learner to a desired state, it is important not only to return feedback according to the learner's state at a certain point in time, but also to proactively predict what the learner is going to do and decide whether and what to intervene.

We considered that estimating how learners would behave in the near future would be an important cue for providing appropriate support. In this study, we attempted to estimate whether learners will collaborate in the near future during task execution from their behavioral data. We conducted an experiment of cooperative learning, and obtained data on facial expressions, gaze, and voice during the experiment. Then, we investigated clues to classify whether or not participants would carry out collaborative work in the near future.

2 Method

2.1 Participants

There were 18 study participants (female: 12, male: 6, average age: 19.00, $SD=0.94$), and all were Japanese students majoring in psychology. These participants participated in the experiment through a participant pool in exchange for course credit. Only freshmen and sophomore students majoring in psychology participated, and they were randomly grouped into dyads. The experimenter confirmed that students within a dyad had not interacted with one another previously.

2.2 Materials and Systems

Two personal computers and two monitors were used by participants, two video recorders filmed their conversations and facial expressions, and two Tobii systems recorded their eye gaze. During the experimental task, each participant was asked to individually make a concept map and then to collaboratively make a single concept map based on the individual maps. So, the participants used a tool we

built to make and share concept maps in real time. A monitor and video recorder were placed in front of each participant. They sat across from each other with a partition placed between them so they could not see each other. This is because we did not want them to communicate using facial expressions and gaze as cues in collaborative work, as we envisioned them collaborating with ITS.

2.3 Procedure

In the experiment, the task was to make inferences about a certain psychological phenomenon. The learning material was a causal attribution of success and failure based on attribution theory [15]. Before the task, participants read a text passage about causal attribution. In the story, the characters participated in school counselling with Michael Peter, who was discussing that he is worried about the new semester. The participants need to explain why the student (Michael Peter) was worried about the new semester based on the story by causal attribution. At the end of this phase, a comprehension test was performed (pre-test). Next, in the individual phase (10 min), the participants conducted the task [14] of applying the causal attribution of success and failure to the episode (10 min). In this study, dyads were asked to build concept maps about causal attribution and create them individually at that time. Finally, in the collaboration phase (15 min), the participants created a concept map collaboratively with reference to their individual maps. At the end of the experiment, another comprehension test was performed (post-test). In the pre-test and post-test, participants were asked to explain the given story based on attribution theory, and the content was evaluated by two coders (see [13]).

2.4 Independent Variables

Data Segmentation. During the experiment, facial features and gaze direction data were acquired at approximately 30 fps. The moving average was calculated by shifting the data by 5 s with 10-s window. We analyzed the data.

Facial Features. The facial movements were analyzed by OpenFace. This automatically calculated whether an Action Unit (AU [6]) appeared. The numerical value output by OpenFace indicated the strength of the AU and was obtained by a formula described in a “toolkit” [2]. There were 18 types of AUs observed among the participants as follows. AU01, AU02, AU04, AU05, AU06, AU07, AU09, AU10, AU12, AU14, AU15, AU17, AU20, AU23, AU25, AU26, AU28, and AU45 (see “<https://www.cs.cmu.edu/~face/facs.htm>” for details).

Gaze. Gaze was acquired by the Tobii system rather than OpenFace because we wanted to know where the participant was looking at the operating screen. A monitor was placed in front of each participant that showed three concept maps: their own concept map and the partner’s concept map that they had made in the individual phase and the collaborated concept map during the collaboration

phase. In the analysis, we used the location (own concept map, partner's concept map, or the collaborative concept map) and duration of participants' attention on this screen as independent variables (`gaze_self`, `gaze_other`, `gaze_c`).

Voice. The voices were captured by microphones placed in front of each participant. To reduce noise, only the parts of the speech uttered by each participant were cut out from the audio files. The cropped audio files were analyzed using Praat [3]. The features extracted in the analysis were fundamental frequency (F0), energy (Intensity), and formants (F1, F2). Within the moving average window, the z-score and SD of each feature in each participant were calculated, and these were used as independent variables (mean and SD in each time window).

2.5 Analysis

We tried to discriminate the user's near-future state by the machine learning based on the obtained data to find clues to implement a cognitive tutor agent that would provide appropriate intervention, inferring whether the learners were active. The dependent variable in this analysis was the state of the learner in the near future (active or passive). The average number of times the concept map was manipulated in 10s was 1.61 (SD = 2.65). The data consisted of 3204 cases (178 cases per participant). The relatively large SD suggests that there was time for intensive manipulation. We examined whether two states of the participant's activity can be inferred 10s after a certain point in the collaboration phase: 1) whether to manipulate the concept map more than once, and 2) whether to manipulate the concept map more than three times.

There were three types of operations on making a concept map: adding a node, creating a link, and labeling the link. We recorded the time these operations were performed in one-second increments. For the analysis, we counted the number of times the operation was performed in each time window.

Determining Whether to Manipulate the Concept Map More Than Once. We hypothesized that this state represented whether or not the participant would proceed with the collaboration in a concrete way. We used Support Vector Machine (SVM) for these discriminations (R 4.1.0 "kernlab" package).

The variables that did not contribute to discrimination from the 28 independent variables were removed by backward elimination. Finally, 19 variables remained (`F0_mean`, `Intensity_mean`, `F1_SD`, `F2_SD`, `F2_mean`, `AU04`, `AU05`, `AU06`, `AU07`, `AU10`, `AU12`, `AU15`, `AU17`, `AU20`, `AU25`, `AU26`, `AU45`, `gaze_self`, `gaze_other`). When leave-one-participant cross-validation was conducted using these variables, the accuracy was 0.72 and the F-measure was 0.71. Although this result is not high enough, it could be used as one of the clues to infer the learner's next activity.

Determining Whether to Manipulate the Concept Map More Than Three Times. We hypothesized that this state indicates whether the participants would be actively collaborating or not. We also used SVM for these discriminations.

The variables that did not contribute to discrimination were removed from the 28 independent variables by backward elimination. Finally, 18 variables remained (F0_SD, F0_mean, Intensity_SD, Intensity_mean, F1_SD, F1_mean, F2_SD, F2_mean, AU01, AU04, AU05, AU06, AU10, AU14, AU15, AU26, gaze_self, gaze_other). When leave-one-participant cross-validation was performed using these variables, the accuracy was 0.74 and the F-measure was 0.56. We think the reason for the low F-measure is presumably that the number of cases in which the concept map was manipulated more than three times in 10s was skewed at 718 out of 3204 cases. In the number of the cases per participant, the lowest was 16 out of 178 cases and the highest was 74 out of 178 cases, indicating a large dispersion. It is assumed that the F-measure was low because it included manipulation strategies with few examples.

Variable Selection Using Solution Path of Lasso Regression. Since it is difficult to interpret specific interactions of the users when there are many variables, we tried to reduce variables using the solution path of lasso regression (R 4.1.0 “glmnet” package). In order to reduce the number of variables as much as possible, we performed cross-validation using the cv.glmnet function and adopted the variables when the value of the regularization parameter λ was one standard error away from the minimum mean cross-validated error.

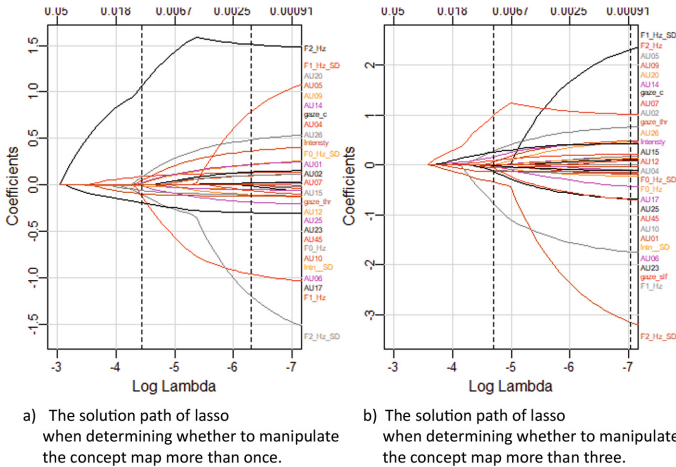


Fig. 1. The solution path of lasso.

Determining Whether to Manipulate the Concept Map More than Once. The solution path of lasso for this discrimination is shown in Fig. 1 a). The dotted line on the right is the value of λ that represents the minimum cross-validated error, and the dotted line on the left is λ that is one standard error away from it. In the condition, 12 variables remained (F0_SD, F1_SD, F1_mean, F2_SD, AU02, AU04, AU05, AU07, AU09, AU10, AU12, and AU15). When the SVM was run with these variables and discriminated, the accuracy was 0.69 and the F-measure was 0.68. Although the accuracy is slightly lower than backward elimination due to the reduction of more variables, the Gaze variables are no longer needed. This is an important finding for practical use because fewer variables to measure simplifies the system.

Determining Whether to Manipulate the Concept Map More Than Three Times. The solution path of lasso for this discrimination is shown in Fig. 1 b). The dotted line on the right is the value of λ that represents the minimum cross-validated error, and the dotted line on the left is λ that is one standard error away from it. In the condition, 14 variables remained (F0_mean, F1_SD, F1_mean, F2_SD, F2_mean, AU04, AU06, AU07, AU09, AU12, AU20, AU26, AU45 and gaze_c). When the SVM was run with these variables and discriminated, the accuracy was 0.67 and the F-measure was 0.51. This condition also reduced more variables than backward elimination, but had less effect on the correct answer rate and F-measure. However, since the F-measure is low, it is not possible to determine whether these trends are consistent or not. In addition, since the gaze variable was not eliminated, there are few implementation advantages.

3 Discussions and Conclusion

The main contribution of this study is that we suggest that the participant's state of execution of collaborative tasks in the near future can be inferred to some extent from multimodal information. The fact that it is possible to estimate about 70% of whether or not a concept map to be jointly created will be manipulated in the near future without considering the content of the utterance is a useful cue for cognitive tutoring agents to measure the timing of their interventions with learners. In addition, by using solution path of lasso, we were able to find a small number of variables, suggesting that it was also useful when trying to discriminate with other methods (SVM).

On the other hand, it was not possible to successfully estimate whether or not participants would be actively collaborating in the near future. This may not have been an appropriate indicator of active collaboration because participants did not often operate the concept map in a continuous manner. For example, since discussion is also an important component of collaborative work, a new index of collaboration activity could be created by appropriately weighting the frequency of concept map manipulation and utterances of the participants.

In a previous study, we attempted to estimate the Passive and other states of ICAP from multimodal information [11]. Comparing the variables used for

discrimination in that study with those used in the current study, we can see overlaps in AU05, AU07, AU15, AU17, and AU26, which are used for discriminating whether to manipulate the concept map more than once. Since we did not focus on voice variables in the previous study, the variables contributing to the discrimination may have changed due to the weighting of the variable selection. Since we used a different data set, we consider that there is an influence of individual differences. Extraction of features that can estimate the activity state of learners regardless of individual differences is a major task for the future.

References

1. Antoine, R., Brian, L., Dan, B., Maxine, E., et al.: Let's go public! taking a spoken dialog system to the real world. In: INTERSPEECH (2005)
2. Baltrušaitis, T., Robinson, P., Morency, L.P.: Openface: an open source facial behavior analysis toolkit. In: 2016 IEEE Winter Conference on Applications of Computer Vision (WACV), pp. 1–10. IEEE (2016)
3. Boersma, P.: Praat, a system for doing phonetics by computer. *Glott. Int.* **5**(9), 341–345 (2001)
4. Chi, M.T., Wylie, R.: The icap framework: linking cognitive engagement to active learning outcomes. *Educ. Psychol.* **49**(4), 219–243 (2014)
5. Dillenbourg, P., Fischer, F.: Computer-supported collaborative learning: the basics. *Zeitschrift für Berufs- und Wirtschaftspädagogik* **21**, 111–130 (2007)
6. Ekman, P., Friesen, W.V.: Measuring facial movement. *Environ. Psychol. Nonverbal Behavior* **1**(1), 56–75 (1976)
7. Hayashi, Y., Miwa, K., Morita, J.: A laboratory study on collaborative problem solving by taking different perspectives. *Cogn. Stud.* **14**(4), 604–619 (2007)
8. Hayashi, Y.: Multiple pedagogical conversational agents to support learner-learner collaborative learning: effects of splitting suggestion types. *Cogn. Syst. Res.* **54**, 246–257 (2019)
9. HAYASHI, Y.: Towards supporting collaborative learning with an intelligent tutoring system: predicting learning process by using gaze and verbal information. *Bull. Japanese Cogn. Sci. Soc.* **26**(3), 343–356 (2019)
10. Misu, T., Sugiura, K., Kawahara, T., Ohtake, K., Hori, C., Kashioka, H., Kawai, H., Nakamura, S.: Modeling spoken decision support dialogue and optimization of its dialogue strategy. *ACM Trans. Speech Lang. Process. (TSLP)* **7**(3), 1–18 (2011)
11. Ohmoto, Y., Shimojo, S., Morita, J., Hayashi, Y.: Investigating clues for estimating icap states based on learners' behavioural data during collaborative learning. In: International Conference on Intelligent Tutoring Systems, pp. 224–231. Springer (2021)
12. Rummel, N., et al.: New challenges in cscl: towards adaptive script support (2008)
13. Shimojo, S., Hayashi, Y.: How shared concept mapping facilitates explanation activities in collaborative learning: an experimental investigation into learning performance in the context of different perspectives. In: Proceedings of the 27th International Conference on Computers in Education (ICCE2019), pp. 172–177 (2019)
14. Weinberger, A., Fischer, F.: A framework to analyze argumentative knowledge construction in computer-supported collaborative learning. *Comput. Educ.* **46**(1), 71–95 (2006)
15. Weiner, B.: An attributional theory of achievement motivation and emotion. *Psychol. Rev.* **92**(4), 548 (1985)



Selfit v2 – Challenges Encountered in Building a Psychomotor Intelligent Tutoring System

Laurentiu-Marian Neagu^{1,3}(✉), Eric Rigaud², Vincent Guarnieri², Mihai Dascalu³,
and Sébastien Travadel²

¹ École Nationale Supérieure des Mines de Paris, 1 Rue Claude Daunesse, Sophia Antipolis, France

laurentiu.neagu@upb.ro

² Mines Paris - PSL, Paris, France

{eric.rigaud,vincent.guarnieri,
sebastien.travadel}@minesparis.psl.eu

³ University Politehnica of Bucharest, 313 Splaiul Independentei, 060042 Bucharest, Romania
mihai.dascalu@upb.ro

Abstract. Psychomotor skills development is fundamental for performing safely and efficiently daily life, professional, and leisure movements. Psychomotor learning and upkeep are critical for preventing the negative consequences of a sedentary or inactive lifestyle. Our underlying assumption is that the development of a self-paced, learner-oriented, highly adaptable, and interactive training environment dedicated to psychomotor skills provides valuable support both to persons already performing sports training and to less motivated persons not practicing physical activities. The current study presents an overview of the second version of the Selfit system, an Intelligent Tutoring System for psychomotor skills, focused on strength development. The development of the ITS raised several challenges related to knowledge modeling, assessment, self-improving tutoring, and communication with the learner in open environments. As such, this study describes the implementation of Selfit v2, while considering its main components and improvements, together with insights on how the main challenges were addressed and a preliminary user test with 7 trainees who experimented with the first version.

Keywords: Selfit ITS · Psychomotor skills · Strength development · ITS Evaluation

1 Introduction

Nowadays, designing digital and intelligent solutions dedicated to training psychomotor skills for children, adults, and the elderly are increasingly necessary. The rising percentage of the population having a sedentary or inactive lifestyle induces an augmentation, among other negative factors, of cardiovascular diseases incidence and mortality [1]. Psychomotor skills are generally developed by sports teachers in schools, together with sports trainers at amateur and professional levels, as well as in specific sports organizations. Digital technologies enable the design of innovative solutions to facilitate remote

exchanges between teachers, trainers, and students. Moreover, they also create the potential for the design of efficient and user adaptive digital solutions to support psychomotor development. For the psychomotor field, domain-specific learning technologies raises a set of challenges, which will be addressed further.

A study by Neagu, Rigaud, Travadel, Dascalu and Rughinis [2] reviewed publications describing Intelligent Tutoring Systems focusing on the development of psychomotor skills, and identified a few related systems. As such, the opportunity arose to build an ITS System for psychomotor development – Selfit [3] that supports different needs. The first one considers general individual fitness. Individuals follow a generic psychomotor training program for general motivation such as aesthetics, fitness, and general health by practicing at home, at the gym, or outside with or without any specific material. A second need targets a specific fitness activity, such as a sport for leisure or competition, for which individuals follow a specific psychomotor program. A third need reflects a specific collective fitness requirement. An organization uses the ITS to promote health and safety or prevent or cure specific pathologies or diseases such as the negative consequences of a sedentary lifestyle.

The current study introduces an overarching presentation of the new version of the Selfit ITS which introduces improvements in three directions: knowledge representation through the OntoStrength ontology [4], anatomical adaption [5], as well an improved user interface integrated into the mobile application.

The paper is structured as follows. The next section describes the second version of Selfit ITS with emphasis on knowledge representation, self-improving tutoring, and tutor-trainee interaction, followed by discussions on how the main challenges were addressed, conclusions, and future work directions.

2 Selfit Version 2 – An ITS for Psychomotor Strength Skills Development

The Selfit Intelligent Tutoring System is a smart ecosystem developed for training the adult strength skills, based on an ITS architecture. The system functionalities support strength profiling by considering strength fundamental movements with a performance scale from one to five: upper body (push horizontal, push vertical, pull horizontal, pull vertical) and lower body (hip dominant, knee dominant). It also supports strength workout prescription based on the initial evaluation - calibration challenges, his/her readiness to train - before training assessment, and the learning about the effects of past workouts – after training assessment.

The development of the Selfit system relies on the design and the integration of different modules. This section briefly illustrates the recent advancements of the ITS centered on the OntoStrength ontology, the anatomical adaptation tutoring system, and the user interface which was redesigned to accommodate new functionalities.

2.1 The OntoStrength Ontology

The OntoStrength ontology [4] aims to support the development of the Selfit ITS dedicated to enhancing psychomotor skills and associated bio-motor abilities, with a focus

on improving strength. Strength is defined as the maximal force or torque (rotational force) that a muscle or muscle group can generate or as the ability of the neuromuscular system to produce force against an external resistance [6].

Overall, OntoStrength models knowledge for training development program periodization, which includes Macro Cycle, Meso Cycle, Micro-Cycle, Workout, Warm-Up, Content, Cool Down, Exercise classes. The ontology also models knowledge on the user's psychomotor capacities (e.g., Muscle Contraction, Fundamental Movement, Hip Movement, Ankle Movement, Movement Skill). OntoStrength models a learner individualization signature, which is specialized in bio-motor skills: Flexibility, Endurance, Speed, Injury, and Strength Data storage, queries via SPARQL, and the views were implemented using the semantic graph database GraphDB (<https://graphdb.ontotext.com/>).

OntoStrength was developed using Ontology 101 Methodology (OD101) [7] by a multidisciplinary team, in Protégé software, having OWL as the formal representation language. The Team has expertise in physiology, biomechanical, muscular, and computer science fields. Besides the initial definition of the ontology performed by Neagu et al. [4], we now focus on its integration in Selfit v2.

The Selfit v2 *domain module* addresses the movement domain by first considering the diversity of human movements activities (daily life, leisure, or professional) and their description (muscular contraction, human body joints movements, or fundamental movements). Second, it includes the movement metabolic profile with its duration and intensity. The *tutoring module* required integration with the psychomotor development domain. This domain is related to the temporal organization of the development program by considering the objectives and content of the different training blocs (macro-cycle, micro-cycle, or workout). Moreover, the domain is also related to the specific development rules and constraints of the different sub-skills used by the development program. Thus, OntoStrength represents different abilities such as strength, endurance, or speed, all linked to associated development rules. The *student module* considers the evaluation domain of psychomotor skills. From this perspective, OntoStrength includes performance indicators and associated evaluation rules for each sub-skill considered by the development program. The ontology also addresses performance indicators used to monitor trainee responses to the training workout and to adjust the planned program to the reality of the effects of its application.

2.2 The Selfit v2 Anatomical Adaptation Tutoring System

The *Selfit* anatomical adaptation tutoring model supports the learning process by providing machine learning mechanisms to support the adaptation of the learning program to the trainee's characteristics and in particular the gender dimension.

The tutoring process supports the definition of strength training tasks following templates of training sequences based on the Anatomical Adaptation strength development process [6]. The template includes several sessions to train in the current week (associated micro-cycle), time to train for a session, and micro-cycle focus. A sub-list of templates used for generating micro-cycles in anatomical adaptation is presented in Table 1. The current Selfit version does not support adaptation for the higher micro-cycle levels, namely: mesocycle and macrocycle.

Table 1. Micro-cycle templates examples for anatomical adaptation.

Micro-cycle template name	# of trainings	Recommended trainee
Push/pull/lower/upper/lower	5	Men
Hip dominant/knee dominant/upper/lower/upper	5	Women
Upper/lower/full/full/full	5	Mixed

A multi-armed bandit algorithm [3] supports the definition of training workouts by adjusting the template content with inputs related to readiness to train, the effective realization of training tasks, and subjective assessment of tasks effects provided by the user. The sexual dimorphism dimension is also considered due to morphological, cognitive, and physiological differences between males and females. The design of training sessions relies on physiological temporal variables.

The delay between two sessions must be long enough to not induce overfatigue and not too long not to induce detraining. For men, the evolution of session loads follows temporal patterns with progressive development from one week to another. The easiest week is set every three weeks to facilitate learning assimilation – for example, one week with easy sessions, one week with medium sessions, one week with heavy sessions, followed again by easy sessions. For women, the evolution of session loads follows their menstrual cycles by using a specific template [5].

Selfit v2 implemented the Right Exercise at the Right Time (RiERiT) method, which has proven efficient for adaptation in the psychomotor field [8]. The tutor implements the multi-armed bandit algorithm for personalizing the training sequences in a session, based on session templates. A session template has a list of generic exercises, each of them with a targeted area, charge level, and rest time. Selfit proposes exercises most likely to increase the average competence level across all psychomotor components using previous trainee performance.

2.3 The Selfit v2 Graphical User Interface

The Graphical User Interface was developed as a Progressive Web Application [9], and provides functionalities for user authentication, learner calibration, providing training instructions to users, videos to guide the exercise execution, and monitoring user feedback (see Fig. 1).

Feedback is an essential dimension of psychomotor development and consequently to Selfit efficiency. Before starting a session, Selfit queries trainees to self-evaluate their fatigue level, motivation to train, sleep quality, and stress level. While training, Selfit asks trainees to self-evaluate the difficulty of each exercise at its end. The perceived difficulty is assessed based on the number of repetitions per set. After the training session, the system assesses trainee perception in terms of session difficulty.

Selfit does not track and does not store any personal, identifiable data of users; user profiles consist of a nickname, password, and a security question managed by the *Authentication* subcomponent. The *Calibration* subcomponent supports the definition of learning motivation by selecting physical qualities to be developed. Then, Selfit supports

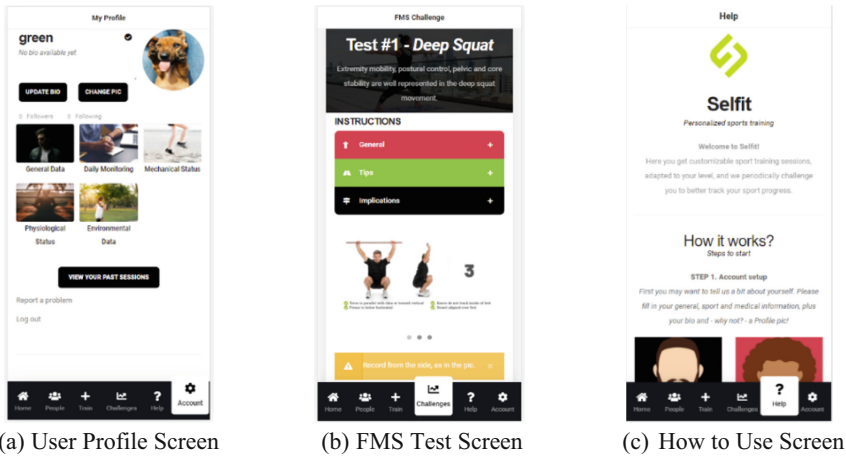


Fig. 1. Selffit graphical user interface

the trainee's initial and intermediary evaluations, as can be seen in Fig. 2. Learner evaluation is performed through self-assessment as users input the results on calibration challenges, which are the number of repetitions executes. Calibration sessions should be performed regularly, on the four of the six essential movements categories, to adjust the user's progress while training.

Through Selffit, users can self-evaluate their fatigue level, motivation to train, sleep quality, and stress level before starting a training session. Trainees could input manually if they failed an exercise or whether they could perform additional repetitions and, if yes, how many. After each session, trainees can self-evaluate the session's difficulty.

The *Training session* subcomponent provides learners with a training sessions description, which includes a summary of warmup, content, and cooldown exercises. Training sessions are easily configurable; trainees can select train location (home or gym), with many session templates available, the training materials available (barbell, elastic band, machine, etc.), and complementary muscles to target while training (anterior shoulder, biceps, forearms, thighs, etc.).

While training, a video demonstrating the movement and details required to accomplish the correct load (i.e., number of sets, number of repetitions, rest between repetitions and between sets) is displayed for each exercise. Features to improve user experience have been implemented in the graphical user interface, such as auto mode, fast mode, body area progress chart, or resume a session. The auto mode feature is helping learners to move automatically between time-based exercises, without user input. The fast mode feature enables the user to significantly decrease the time of a session, which will group exercises two by two, using the superset training concept – both exercises are executed together, as one set. This feature is useful when the learner is running out of time while training.

In a recent literature review of authoring tools in ITS models [10], it was shown that the “Display learners’ statistics” feature was the most common – 12.12%, by far, in the ITSs built in the last 15 years. For this reason, trainee statistics are also shown in Selffit,

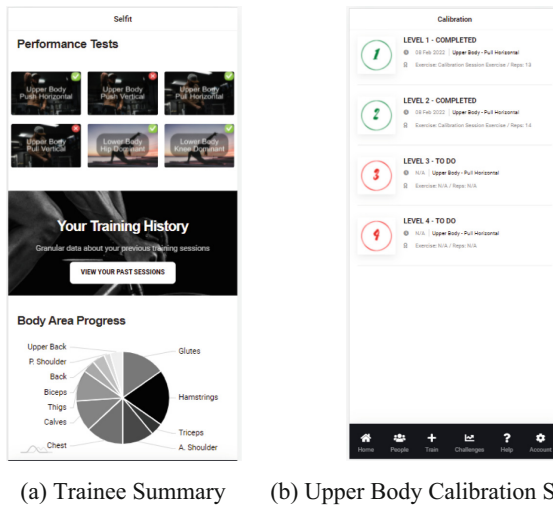


Fig. 2. Selfit interface – trainee calibration summary

an example can be seen in Fig. 2(a) – body area progress. Users can observe their full training history, per microcycle and session, and also their progress per trained body area.

3 Discussions on Encountered Challenges

The development of psychomotor skills is a complex problem that defines training workouts content by selecting performance factors and their associated acute program variables. This process considers global variables such as the activity profile, the user characteristics, the type and aims of the different periods (macrocycle, mesocycle, and micro-cycle), and contextual variables such as the content and effects of past workouts, his current fitness perception, the pain felt, or heartbeat. Then, trainees must understand the workout content, a movement, or a set of movements with indications about the number of repetitions and the rest between repetitions. Once the trainee achieved a workout, a quality evaluation involving the movement correct realization, the difference between the planned number of repetitions and the realized, direct effect on the human body (physiological measures or perceived evaluation of the effort) must be performed to support the control of the training program realization. Collected data supports the definition of the following workout, but it can be also used to revise the global variables structuring the periodization.

3.1 Challenges in Knowledge Representation

One of the most difficult challenges when building ITSs consists of creating a strong knowledge base from scratch [11]. As stated by Neagu et. al. [12], no previous ontology has been developed in the psychomotor field; existing works focus on recognizing sports

activities with technology or supporting decision-making based on data collected during sports competitions. As such, the effort of building psychomotor development knowledge required multidisciplinary expertise, and good coordination for defining concepts and relations in learning branches, such as physiology, biomechanical, muscular, or medical areas. Except for the GIFT project [13], the other existing work in psychomotor ITSs does not clearly outline how the knowledge is modeled. Cross-disciplinary teams of the GIFT NATO department developed and described the knowledge modeling for training marksmanship at the GIFT Symposium¹ every year, proposing also recently ontology-driven approaches for the modeling [13].

The development of OntoStrength was challenging because of the heterogeneity of domains to be integrated, each with its independent design. Moreover, different cultures and approaches co-exist in each domain, generating the necessity to organize concepts sometimes as complementary and sometimes as antagonists. Nevertheless, OntoStrength has a central role as it grounded the development of effective tutoring and the elements rendered in the graphical user interfaces.

3.2 Challenges in Self-improving Tutoring

The Selfit v2 tutoring module selects the optimal learning sequences to provide a good learning experience, based on the estimation of the trainee's competence levels and progression. Self-improving tutors are becoming more and more popular in ITSs for several reasons [14]: increased tutor flexibility, reduced long-term tutor cost, easiness to adapt to new student populations without human intervention, increased generality, and reasoning about uncertainty. We apply principles from cognition to the psychomotor field, starting from the study of Ropelato et al. [15] who created a virtual reality environment for learning how to drive, where learners received optimal sequences based on the Zone of Proximal Development and Empirical Success (ZPDES) algorithm.

The second challenge in designing our ITS was to develop a self-improving tutor capable to generate and adjust development tasks according to trainee characteristics. The first characteristic is gender. The morphological, cognitive, and physiological differences between males and females impact the content development of the training sessions, together with associated risks of injuries and psychological disorders. The second set of characteristics relates to trainee level on different performance skills. Selfit supports trainees' evaluation and considers their previous results while defining development objectives and content. The evaluation considers quantitative performance indicators (such as one Repetition Maximum – 1RM for strength development or the Maximum Aerobic Speed – MAS for endurance development), and qualitative indicators (such as technical quality indicators related to the abilities to correctly perform movements). The third set of characteristics reflects the training environment – as such, Selfit must enable user training in any location, at any time. Therefore, the tutoring system adapts to the characteristics of the trainee environment, such as the availability of materials (e.g., dumbbells, bodybuilding machines) to perform strength development tasks, or the availability of stalled tracks to perform speed and endurance development tasks.

¹ <https://gifttutoring.org/projects/gift/wiki/Overview>.

The fourth set of characteristics reflects user readiness to train. The realization of psychomotor training tasks assumes a certain level of trainee fitness and fatigue depending on the development schedule.

Nevertheless, numerous external factors might influence trainee fitness and fatigue, such as daily life, professional, or other psychomotor activities. The consequence is that trainee fitness and fatigue levels might be superior or inferior to the state assumed by the tutoring system. In addition, users might be injured and incapable to perform part of the psychomotor training tasks. Therefore, the tutoring system evaluates the readiness to train and, if necessary, adjusts the content of the training tasks by increasing or decreasing their difficulty. The fifth set of characteristics considers user responses to training tasks. The design of psychomotor training tasks assumed an immediate consequence to the trainee in terms of fatigue, medium-term consequences in terms of physiological improvement after a period of recovery, and long-term effects in synergy with future training tasks. Nevertheless, muscular, and cardiovascular responses to a psychomotor training task vary from one muscle to another, and one person to another in terms of intensity and duration. As such, the Selfit tutoring module evaluates the practical immediate, short- and long-term effects of training tasks to adjust the following ones.

The sixth set of characteristics considers the motivation to train. Again, numerous factors might affect the willingness to train, for example, fatigue, emotional perception of a training task, or factors independent of the ITS, such as daily-life events. Consequently, the Selfit tutoring module evaluates the willingness to train and, if necessary, adjust the training sequence. The training planification process is generally accepted by the sports literature to be split into four phases [16], as can be seen in Fig. 3: planning (prescription of proper exercise units), realization (execution phase), control (comparison between exercises performed by the athlete versus the planned exercises), and evaluation (measurement of athlete's performance).

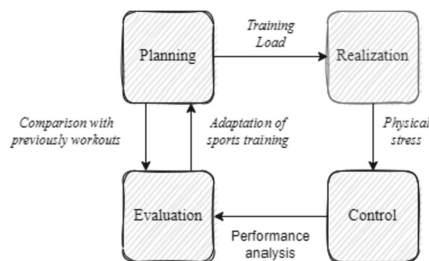


Fig. 3. The four phases of sports planning

As a learning sequence optimal for a trainee may be inefficient for another [17], an effective self-improving tutoring system for psychomotor development requires the collection and the update of a large amount of data before, during, and after the realization of training tasks. This is a significant challenge affected by the risk of lack of accurate information about trainees. As minimum data should be acquired while training, tutoring relies on trainee input, partially acquired data from the environment, and caution on learner's health—new sequences generated by the adaptive tutor should protect the learner

from injuries, medical issues, or physical exhaustion. However, the learner should be engaged in learning and maintain a long-term motivation for training. The learning should happen in the zone of proximal development for the trainee. In addition, when designing ITSs in open environments, there is a high level of uncertainty when assessing progress and while tutoring. Specific rules for building the learner profile were required while accounting for the prevention of injuries or health issues during workouts. The trainee is usually in-the-wild, with limited time, and Selfit is responsible for maintaining the trainee's motivation to achieve his/her training goals.

3.3 Challenges in Interactions with the Trainee

A tutor module may have the best student and teaching knowledge, but its value is limited without an efficient interface component to interact with the learner [14]. The guidelines for the Selfit interfaces are considered a clear, simple, attractive, and responsive design. Multiple iterations were performed, and we adopted an agile development methodology with incremental improvements based on feedback received from trainees.

Initially, Selfit had a learner calibration component that relied on the Functional Movement Screen (FMS) Test [18]. This approach was limited by the lack of variability of movement patterns required for proper assessment of students' levels across the diversity of exercises. Also, the initial system required the learner to record himself/herself while performing the FMS Test, within a strict protocol. Videos were afterward manually labeled by Selfit partners, and sports coaches, with a score from 0 (failure) to 3 (perfect execution). This initial learner calibration was difficult and long for both the learner and tutor. Also, learners had to wait until the coach manually revised the videos to start the first session; the same issue was valid for future calibrations.

Selfit v2 defined a new protocol for calibration, a custom FMS test, focused on six body movement patterns, defined on a difficulty scale from 1 to 4. Trainees are now able to manually input the number of repetitions to be performed, and the calibration is performed instantly. OntoStrength supports the mapping of training sessions exercises based on the custom FMS test.

In addition, Selfit v2 integrates a customizable training session. Trainees can now choose the location to train – home or gym, with specific training rules, time to train, and specific set of materials available. Each training session is generated from fundamental and complementary exercises. As the fundamental part is fixed, based on the trainee microcycle goal, the complementary exercises can be configured by the trainee before starting the session. Trainees may select preferred muscles to train or just leave the default ones, which are based on their previous ratings, while training.

3.4 Preliminary User Testing

Selfit v2 is currently being tested with real users. Google Analytics² was enabled to measure application performance, and engagement time, and to record platform traffic. Selfit v2 has an average of 170 users per month, from several countries: Romania (48 users), France (37 users), China (23 users), the United States (21 users), and other

² <https://analytics.google.com/analytics/web/>.

countries, as can be seen in Fig. 4.a and b. The average engagement time per user is 5 min and 45 s. As an average sports session has 45 min, we conclude there were many application sessions started by users without performing actual sports training. The analytics show 3.317 users' screen views and 89 training sessions completed in February 2022. Selfit is enrolling new users every day; from the 170 users who used the system in February 2022, 137 users were new.

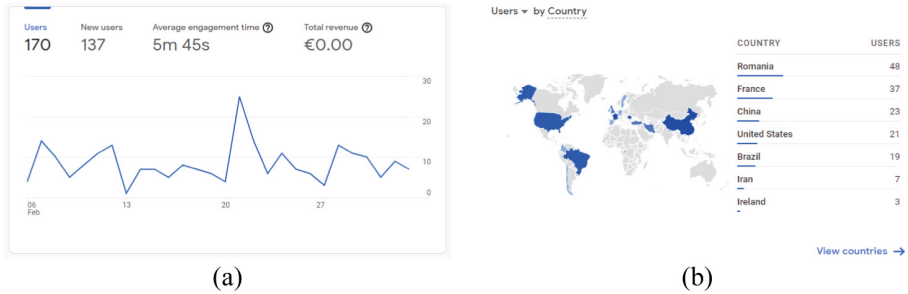


Fig. 4. a. Google analytics - Selfit total users in February 2022. b. Google analytics – Selfit total users per country in February 2022.

A user experience survey was conducted during this testing phase with 7 users who also tried Selfit v1 [8]. Similar to the evaluation of the first version, the AttarkDiff questionnaire [19] was used to assess user experience, and differences between the two versions were computed (see results in Table 2; values in bold mark a difference greater than 1 on the 7-point Likert scale between the evaluations).

Table 2. *Selfit* user experience feedback based on the AttrakDiff questionnaire (mean and standard deviation values corresponding to qualities scored on a 7-point Likert scale).

UX quality	V1 M (SD)	V2 M (SD)	UX quality	V1 M (SD)	V2 M (SD)	UX quality	V1 M (SD)	V2 M (SD)
Pleasant	5.44 (1.11)	6.42 (0.49)	Connective	4.61 (1.53)	4.85 (1.45)	Human	4.55 (1.25)	5.00 (1.30)
Inventive	4.94 (1.80)	6.57 (0.49)	Simple	4.50 (1.42)	5.85 (0.63)	Professional	4.83 (1.50)	6.57 (0.72)
Attractive	5.05 (1.22)	5.57 (1.29)	Practical	5.50 (0.89)	6.57 (0.49)	Likeable	5.83 (0.95)	6.57 (0.49)
Straight-forward	5.05 (1.17)	5.71 (1.27)	Stylish	5.00 (1.20)	6.14 (0.83)	Predictable	4.27 (1.19)	5.42 (1.29)
Premium	4.66 (1.29)	5.57 (1.39)	Integrating	5.72 (0.80)	5.00 (1.77)	Brings people closer	4.72 (1.32)	4.71 (1.57)
Novel	5.22 (1.35)	5.57 (1.98)	Motivating	5.44 (0.95)	6.42 (0.49)	Captivating	5.44 (0.89)	6.14 (0.83)

The overall user experience has improved in Selfit v2, as most of the hedonist and pragmatic quality values are better. In the survey, users had to also answer 3 open questions about the overall experience and new features to be implemented. Users recommended features such as ‘voice support while training’, ‘tips to improve movement execution’, ‘more guidance while training’, or ‘reminders to train’. A user recommended internationalization for French users and support for other languages. The overall feedback was positive, users wrote they were happy with the generated sessions and the diversity of exercises. However, we must emphasize the limitation that the survey was carried out with only a limited number of respondents; a larger scale evaluation will be conducted, but we wanted to emphasize the differences using trainees who experimented with both versions.

4 Conclusions and Future Work

The current work introduces a comprehensive description of the second version of Selfit, an ITS for psychomotor development, alongside encountered challenges and provided solutions. The first version of Selfit was tested with real users in anatomical adaption training and it raised several challenges in knowledge representation, self-improving tutoring, and trainee-tutor interaction. The actual version of Selfit integrates a comprehensive ontology in strength development – OntoStrength, an updated version of adaptive tutoring, and major updates in the communication component.

In terms of future extension, we envision a thorough evaluation of the suggested workout programs; however, this is a long-term process that requires at least 12 consecutive weeks (preferably 6 to 12 months for long-term effects), with continuously collected data. We also consider the integration of live recordings for exercise execution with computer vision techniques to perform automatic counts of repetitions, avoid user manual input errors, and enhance the overall user experience. Also, the next Selfit version should include Natural Language Techniques [14] to improve learner–tutor communication. Currently, trainees need to manually input feedback for each set of exercises in the graphical interface; instead, a voice-based interaction module may also improve user experience.

References

1. Owen, N., et al.: Sedentary behavior and public health: integrating the evidence and identifying potential solutions. *Annu. Rev. Publ. Health* **41**, 265–287 (2020)
2. Neagu, L.-M., Rigaud, E., Travadel, S., Dascalu, M., Rughinis, R.-V.: Intelligent tutoring systems for psychomotor training – a systematic literature review. In: Kumar, V., Troussas, C. (eds.) *ITS 2020. LNCS*, vol. 12149, pp. 335–341. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-49663-0_40
3. Neagu, L.-M., Rigaud, E., Guarnieri, V., Travadel, S., Dascalu, M.: Selfit – an intelligent tutoring system for psychomotor development. In: Cristea, A.I., Troussas, C. (eds.) *ITS 2021. LNCS*, vol. 12677, pp. 291–295. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-80421-3_32
4. Neagu, L.-M., et al.: OntoStrength: an ontology for psychomotor strength development. *IxD&A (Interact. Des. Arch. J. (under review))*

5. Neagu, L.-M., Rigaud, E., Guarnieri, V., Matei, G.-D., Travadel, S., Dascalu, M.: Selfit—accounting for sexual dimorphism in personalized motor skills learning. In: 6th International Conference on Smart Learning Ecosystems and Regional Development (SLERD 2021). Springer, Bucharest (Online) (2021). https://doi.org/10.1007/978-981-16-3930-2_7
6. Bompa, T., Buzzichelli, C.: *Periodization: Theory and Methodology of Training*, 6th edn. Human Kinetics Publishers (2017)
7. Noy, N.F., McGuinness, D.L.: *Ontology Development 101: A Guide to Creating Your First Ontology*. Stanford Knowledge Systems Laboratory, Stanford (2001)
8. Neagu, L.-M., Rigaud, E., Guarnieri, V., Travadel, S., Dascalu, M.: Selfit – an intelligent tutoring system for psychomotor development. In: Cristea, A.I., Troussas, C. (eds.) ITS 2021. LNCS, vol. 12677, pp. 291–295. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-80421-3_32
9. Bjørn-Hansen, A., Majchrzak, T.A., Grønli, T.-M.: Progressive web apps: the possible web-native unifier for mobile development. In: Proceedings of the 13th International Conference on Web Information Systems and Technologies, vol. 1, pp. 344–351 (2017)
10. Dermeval, D., Paiva, R., Bittencourt, I.I., Vassileva, J., Borges, D.: Authoring tools for designing intelligent tutoring systems: a systematic review of the literature. *Int. J. Artif. Intell. Educ.* **28**(3), 336–384 (2017). <https://doi.org/10.1007/s40593-017-0157-9>
11. Zouaq, A., Nkambou, R.: A survey of domain ontology engineering: methods and tools. In: Nkambou, R., Bourdeau, J., Mizoguchi, R. (eds.) *Advances in Intelligent Tutoring Systems*. SCI, vol. 308, pp. 103–119. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-14363-2_6
12. Neagu, L.-M., Guarnieri, V., Rigaud, E., Travadel, S., Dascalu, M., Rughinis, R.-V.: An ontology for motor skill acquisition designed for GIFT. In: Proceedings of the 8th Annual Generalized Intelligent Framework for Tutoring (GIFT) Users Symposium (GIFTSym8) (2020)
13. Brawner, K., Hoffman, M., Nye, B.: Architecture and ontology in the generalized intelligent framework for tutoring: 2018 update. In: 7th Generalized Intelligent Framework for Tutoring (GIFT) Users Symposium, p. 11. US Army Combat Capabilities Development Command–Soldier Center (2019)
14. Woolf, B.P.: *Building Intelligent Interactive Tutors: Student-Centered Strategies for Revolutionizing e-Learning*. Morgan Kaufmann Publishers, Burlington (2010)
15. Ropelato, S., Zund, F., Magnenat, S., Menozzi, M., Sumner, R.: Adaptive tutoring on a virtual reality driving simulator. In: 1st Workshop on Artificial Intelligence Meets Virtual and Augmented Worlds (AIVRAR) in Conjunction with SIGGRAPH Asia (2018)
16. Rajšp, A., Fister, I.: A systematic literature review of intelligent data analysis methods for smart sport training. *Appl. Sci.* **10**, 3013 (2020)
17. Clement, B., Roy, D., Oudeyer, P.-Y., Lopes, M.: Multi-armed bandits for intelligent tutoring systems. *J. Educ. Data Min. (JEDM)* **7** (2015)
18. Marques, V.B., Medeiros, T.M., de Souza Stigger, F., Nakamura, F.Y., Baroni, B.M.: The functional movement screen (FMS™) in elite young soccer players between 14 and 20 years: composite score, individual-test scores and asymmetries. *Int. J. Sports Phys. Ther.* **12**(6), 977–985 (2017)
19. Hassenzahl, M., Burmester, M., Koller, F.: AttrakDiff : Ein Fragebogen zur Messung wahrgenommener hedonischer und pragmatischer Qualität. In: Ziegler, J., Szwillus, G. (eds.) *Mensch & Computer 2003. Interaktion in Bewegung*, pp. 187–196 (2003)



Integrating Speech Technology into the iSTART-Early Intelligent Tutoring System

Renu Balyan¹ , Tracy Arner² , Tong Li² , Ellen Orcutt³ ,
Reese Butterfuss² , Panayiota Kendeou³ , and Danielle McNamara² 

¹ State University of New York at Old Westbury, Old Westbury, NY, USA
balyanr@oldwestbury.edu

² Arizona State University, Tempe AZ, USA

³ University of Minnesota, Minneapolis, MN, USA

Abstract. Speech technology (automated speech recognition – ASR and text-to-speech) offers great promise in the field of automated literacy and reading tutors for children. Students in third and fourth grades struggle with generating longer strings of text on a QWERTY keyboard because they still “hunt and peck” for the letters and symbols rather than typing fluently. Thus, in addition to reading comprehension, students’ performance is limited by their ability to translate their ideas into language and then transcribe those words into written text. Fourth grade students produce fewer words and recall less when typing or writing a response relative to speaking. Hence, speech technology is a crucial component in the development of iSTART-Early, an intelligent tutoring system that aims to provide online, automated reading strategy instruction and practice to improve deep comprehension for third and fourth graders. This paper discusses the key components and features of the speech technology incorporated into iSTART-Early. We also discuss some initial findings from pilot studies conducted with adults and youth for this ASR integrated tutoring system. Finally, we discuss considerations for future development of speech technology and integration with intelligent tutoring systems.

Keywords: Automated speech recognition · Text-to-speech · Intelligent tutoring systems

1 Introduction

The iSTART-Early project aims to develop an online, automated reading strategy tutor that provides instruction and practice designed to improve deep comprehension for students in 3rd and 4th grades. iSTART-Early builds upon a previously designed tutoring system (iSTART). iSTART was developed for relatively proficient readers, namely high school students and readers who were already able to read and type in their responses (i.e., self-explanations, summaries, and questions) using a standard QWERTY keyboard. iSTART is not appropriate for elementary school students in its current form because

students in 3rd and 4th grades may not be proficient readers nor are they proficient typists on a standard keyboard. Therefore, we are developing a new system that would offer students the *option to be able to read aloud the text or only the words they are struggling to identify* [1] by a pedagogical agent, as well as the *ability to speak aloud their explanations and edit their answers* generated by the automatic speech recognition system. Providing such scaffolding is imperative because 3rd and 4th grade students need options to read the text aloud and speak their responses while learning reading comprehension strategies. Students in 3rd and 4th grades struggle with generating longer strings of text on a keyboard because they still “hunt and peck” for the letters and symbols that they need rather than typing fluently [2]. Thus, in addition to reading comprehension, students’ performance is limited by their ability to *translate* their ideas into language and then *transcribe* those words into written text [3]. Fourth grade students produce fewer words and recall less when typing or writing a response relative to speaking [4]. Hence, the incorporation of speech technology into iSTART-Early constitutes a crucial feature to support students’ learning.

1.1 iSTART-Early

iSTART-Early is an intelligent tutoring system (ITS) that provides automated instruction and practice on higher-order reading comprehension strategies to 3rd and 4th grade students. iSTART-Early provides personalized, interactive, game-based strategy instruction on and practice with reading comprehension strategies with grade-appropriate informational texts. iSTART-Early provides students with real-time feedback and instruction on improving the use of effective comprehension strategies. Thus, each student receives personalized instruction from the system. Natural language processing (NLP) combined with speech technologies (automated speech recognition and text-to-speech) enable student interactions with immediate feedback.

The key components of iSTART-Early are encapsulated in a meta-game set in space in which students are space travelers traversing five planets based on target reading strategies modified from SERT [5]. The reading strategies are organized on planets named for each one: Ask it (question asking), Rerword it (paraphrasing), Find it (main idea identification), Explain it (self-explanation), and Summarize it (summarization). Each planet includes: a) video lessons providing guided instruction and demonstration of the five targeted reading strategies; b) practice games to increase engagement when interacting with the system and provide opportunities for deliberate practice; c) automated speech recognition (ASR) capabilities that allow students to practice reading strategies without the additional burden of typing; d) text-to-speech functionality that assists students’ reading by having the texts or difficult words read aloud to them; and e) a teacher interface that allows teachers to assign texts and monitor students’ performance to provide additional support and feedback when necessary, creating blended-learning opportunities. This paper focuses on describing the speech technologies (ASR and text-to-speech) and the complete workflow of these components.

2 Integration of Speech Technologies into iSTART-Early

The speech technologies were integrated into iSTART-Early to provide several functionalities that allow students to: a) have difficult (predefined) words or instructions read aloud by a pedagogical agent in English, Spanish, or Chinese, using text-to-speech; b) speak aloud their responses (e.g., paraphrases, questions, self-explanations) using ASR instead of typing responses; and c) edit and update the transcription generated by the ASR engine. The system also includes the capability to filter out and clean frozen expressions, curse words, and implements a spell checker.

2.1 The Process Flow

The overall process flow for the ASR and text-to-speech modules integrated within iSTART-Early is as follows (also see Fig. 1):

- 1) The student is shown the text to be self-explained, summarized etc. after logging into iSTART-Early. The student has an option to let the pedagogical agent read aloud the entire text or some predefined difficult words and their definitions. This feature is implemented via the text-to-speech component.
- 2) The student can also choose to type or speak aloud their response after logging in. The web API asks for permission to use the microphone, if the student chooses to speak their response.
- 3) The student is prompted to speak aloud the response, once the microphone permissions are accepted and asked to click the stop button once done speaking to stop the recording.
- 4) The student is presented with the ASR transcribed text and given the option to edit the ASR generated output. The student then submits the response.
- 5) The response generated by the ASR system and edited by the student is automatically filtered of spelling errors, frozen expressions, and curse words prior to algorithmic evaluation and the generation of pedagogical feedback.

2.2 Automated Speech Recognition Interface

Google Web Speech (GWS) API was used to create the ASR functionality. GWS provides cloud-based voice transcription services that transcribes speech into text in real-time. GWS relies on google chrome web browser to communicate and transfer data between the user interface and the GWS server. The acoustic models (statistical representation of sounds that make up words) for GWS were trained and developed using 5000 h of data that showed very high recognition rate for people with speech disorder [6]. The GWS in the iSTART-Early was implemented by adding a JavaScript event listener to the microphone button within the interface to process ASR requests. Once the connection request is detected, the interface creates a GWS controller to monitor user speech and transfer the audio data. The controller is responsive to “pause” or “stop” requests by the user.

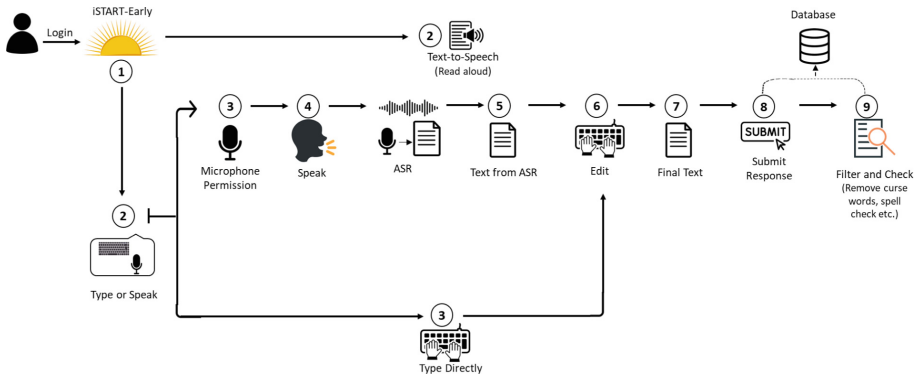


Fig. 1. The speech modules (ASR and text-to-speech) process flow

The student interface includes ASR to allow students to practice reading strategies without the additional burden of typing [7]. The ASR functionality added to the paraphrasing practice module is illustrated in Fig. 2. The interface consists of several components: a) a *text display* area (top section of the interface, marked as (1) - displays the text and target sentences that the students are tasked for paraphrasing; b) a *progress bar* (above the text display area) - indicates students' task completion progress; c) a *response box* (input box below the text display area, marked as (2) - shows the response typed by the student or the ASR transcription of the student's spoken response; d) a *pedagogical agent* (robot at the bottom-right corner) - provides students with instructions (spoken and text-based) for the tasks and step-by-step guidance on how to use the ASR; e) a *microphone button* (bottom-left of the response box) - is clicked to activate the ASR, which changes to a pause button once activated; and f) a *submit button* (bottom-middle of the response box) - to submit the final response.

2.3 Text-To-Speech: Audio Collection Integration

The text-to-speech component has been added to the student interface to request definitions for pre-identified vocabulary words or request that the entire text is read aloud by the pedagogical agent. We have utilized the pre-existing iSTART system module and created a set of new databases for all the texts and predefined words definitions. The E-learning authoring tool (Articulate Storyline) was used to generate MP3 audio files of texts and definitions for the pre-identified and difficult words. Figure 3 illustrates the text-to-speech functionality in a game-based *question-asking* strategy practice activity. The system retrieves the audio for the text from the database and plays it when the user clicks the speaker button in the middle of the interface (highlighted in blue). The users have the options to pause and replay the audio. The vocabulary text-to-speech functionality embedded in the interface (underlined word highlighted in red) is shown in Fig. 3. The users can hover over the underlined word to learn its pronunciation and definition. These definitions can also be read aloud by using the text-to-speech functionality.

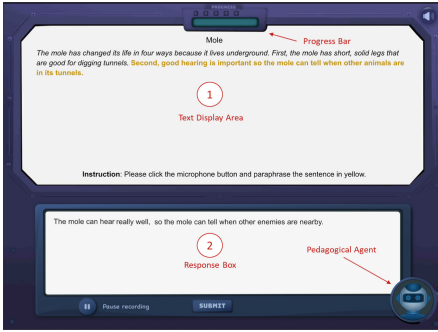


Fig. 2. Paraphrase module interface with ASR embedded technology

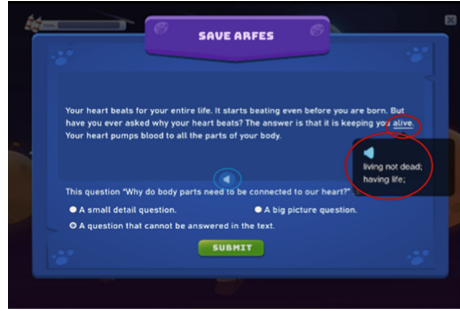


Fig. 3. Text-to-speech and difficult word vocabulary functionality

2.4 Key Features of the ASR and Text-To-Speech Interface

User Instruction. To help users become acquainted with the speech components of the system and provide smooth experience, several guiding components for the system were designed, including an *On-boarding Walk-Through*, *UI Status Indicator*, and *Action Reminder*.

On-Boarding Walk-Through. In the “walk-through” tutorial at the beginning of the system for first-time users, the pedagogical agent (a friendly robot named “Robo”) introduces features to the users after they log into the system. Following the feature introduction, Robo demonstrates the steps to activate the ASR and how it can be used to generate responses. Robo also instructs users to explicitly verbalize punctuation for each sentence, because the current ASR technology is not capable of automatically adding punctuation while transcribing. Robo then informs the user to identify any ASR-introduced transcription errors that may need correction. Finally, Robo demonstrates how to pause the ASR and edit responses.

UI Status Indicator. Once users permit the ASR system to use their microphone, a red circular marker (dot) appears on the web page tab that notifies users that they are being recorded. To further clarify the recording status, audio reminders such as “recording started” and “recording paused” were added to specify when the user changes recording status. In addition, when users hover over any button, a dialog box appears and displays information explaining the associated function.

Action Reminder. The pedagogical agent offers feedback to remind the user of the next steps, when using the ASR to produce responses. For instance, after the user switches

from the recording mode to the editing mode, the robot instructs the user on the possible actions that can be taken (see Fig. 4 bottom-right corner).

Editing Features. The editing feature enables users to pause ASR and edit the transcriptions generated from their spoken responses. To switch from recording mode to editing mode, users can click the microphone button or in the response box. While in editing mode, users can edit the transcribed text or add text using their keyboard or by reactivating their microphone and re-engaging the ASR technology. Figure 4. Action reminders by “Robo”.

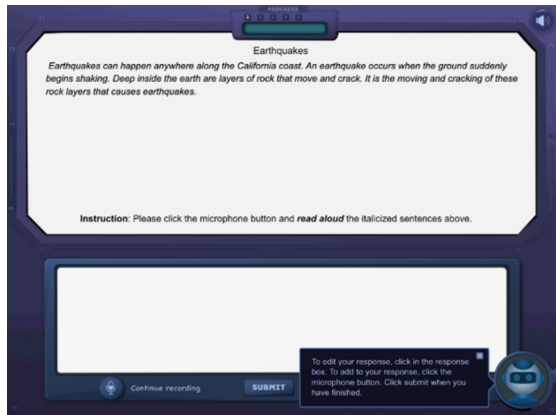


Fig. 4. Action reminders by “Robo”

Database Structure. The database was designed to collect information related to the user interactions with the system, particularly the speech technology needed to evaluate user responses generated by the ASR. Five different types of information are collected and stored in the database (Fig. 5): a) *text information* - includes the prompt text displayed to users (i.e., text title, text script, and target sentences); b) *original ASR transcripts* – all transcripts generated by the ASR, including ASR transcribed errors; c) *user-edited transcripts* - the edited transcripts for each practice session are stored; d) *user voice recordings* - user speech is recorded and stored as audio files in the database while they speak aloud the response; and e) *time-related features* – time taken by the user to read the text and edit the responses is also recorded and stored.

ID	Item	TextTitle	Status	TargetNumber	ReadingText	TargetSentence	ASR	EditResponse	ReadingTime	TimeTaken	Created at
100	Starfish	Starfish	Paraphrasing	1	Why are starfish called starfish?	Your blood carries the oxygen from the air to the ...	Starfish are a part of a group of species called inver...	Starfish belong to a group of species called inver...	1:2	1:47	2022-01-23 12:37:03
101	Starfish	Starfish	Reading	2	Invertebrates are animals that do not have a backb...	Eyepots aren't quite eyes, but they do help the s...	Invertebrates are animals that do not have a back...	Invertebrates are animals that do not have a backb...	0:59	5:5	2022-01-23 12:43:09
102	Starfish	Starfish	Paraphrasing	2	Invertebrates are animals that do not have a backb...	Eyepots aren't quite eyes, but they do help the s...	Starfish have icebox not are not quite like eyes ...	Starfish have eyepots that dont quite act as norm...	0:18	2:1	2022-01-23 12:45:31
103	Starfish	Starfish	Paraphrasing	2	Invertebrates are animals that do not have a backb...	Eyepots aren't quite eyes, but they do help the s...	Starfish have icebox not are not quite like eyes ...	Starfish have eyepots that dont quite act as norm...	0:18	2:1	2022-01-23 12:45:31
104	Stars	Stars	Reading	1	There are different types of stars. All stars are ...	Eyepots aren't quite eyes, but they do help the s...	There are different types of stars. All stars are ...	There are different types of stars. All stars are ...	0:33	0:29	2022-01-23 12:46:41
105	Stars	Stars	Paraphrasing	1	There are different types of stars. All stars are ...	Eyepots aren't quite eyes, but they do help the s...	Stars are how scientist tell them apart	The color of stars are how scientist tell them apa...	0:20	0:27	2022-01-23 12:47:31
106	Stars	Stars	Reading	2	Their color is related to their size. Smaller star...	You might think that bigger stars live longer, but...	Their color is related to their size. Smaller star...	Their color is related to their size. Smaller star...	0:4	1:43	2022-01-23 12:49:20
107	Stars	Stars	Paraphrasing	2	Their color is related to their size. Smaller star...	You might think that bigger stars live longer, but...	Bigars stars do not live longer when compared to ...	A star being big does not mean it lives longer.	1:9	2:25	2022-01-23 12:54:35

Fig. 5. Screenshot of the user interaction data stored in the database

3 Initial Evaluation of Speech Technologies into ISTART-Early

3.1 Adult Pilot Study

Participants and Procedure. Participants accessed the study via Amazon’s MTurk service. The original sample consisted of 38 adults; however, 5 participants were excluded from the dataset due to incomplete or irrelevant responses or poor response quality. Therefore, the final sample consisted of 33 participants (17 male, 16 female). The mean age was 39 years (Standard Deviation, $SD = 10$); 23 participants were White, 5 Black, 1 Asian, 1 Hispanic, and 2 were multiracial; 19 participants reported earning a four-year college degree, 6 a graduate degree, 5 had a high school diploma, and 4 reported attending some college. All participants reported that they were fluent in English. Participants first completed the ASR task that instructed participants to read the texts aloud at a comfortable pace while the ASR system transcribed their utterances. Participants were not permitted to edit the transcriptions. Upon completion, they were directed to Qualtrics [8] to complete the demographics questionnaire.

Texts. The current study included seven expository texts selected from a larger body of texts commissioned for the development of the iSTART-Early system. The texts used in this study were chosen based on word selection (i.e., number and type of content words), syntax, and punctuation. Texts were presented, one at a time, as individual blocks of text without paragraph breaks. See Table 1 for text length and Flesch-Kincaid Grade Level [9] scores.

Measures. Accuracy of the ASR system was assessed using the Word2Vec source overlap using the Tool for the Automatic Analysis of Text Cohesion (TAACO) [10].

Table 1. ASR accuracy and text characteristics

Text Topic	Accuracy Mean (SD)	Range (Min - Max)	Text Difficulty (Flesch-Kincaid Grade Level)	Text Length (Wordcount)
Avalanches	0.977 (.046)	0.817 - 0.999	5.5	248
Bioluminescence	0.971 (.025)	0.869 - 0.987	6.4	377
Combustion	0.980 (.025)	0.914 - 0.997	4.5	252
Domestication	0.970 (.078)	0.637 - 0.999	6.8	251
Fossil Fuels	0.983 (.036)	0.873 - 1.000	6.5	277
Hurricanes	0.990 (.017)	0.933 - 0.998	7.3	173
Inheritance	0.972 (.052)	0.794 - 0.995	11.1	217

Analysis. We examined the overlap between the participants’ ASR responses in order to evaluate the accuracy of the ASR system. The Word2Vec overlap provides an indicator of the accuracy of the ASR system, such that higher overlap values indicate more accurate transcriptions. Word2Vec [11] relies on a neural-network model to represent words

and phrases in a vector-space model. Words co-occurring in similar contexts are represented as being closer, whereas words with dissimilar contexts are represented as being farther apart, in different regions of the vector space. Overlap scores reflected the cosine similarity between the vectors corresponding to the compared segments (e.g., sentences, paragraphs, or documents), which were obtained by summing the vector weights for each word [12].

Results. With respect to overall accuracy, descriptive analyses revealed an overall mean Word2Vec overlap score, across all texts of .979 ($SD = .044$). Accuracy scores were generally high across all seven texts (see Table 1). Thus, results suggest that the ASR system accurately transcribed adult readers' speech as they read expository texts. However, it is critical to examine the ASR system's accuracy for young students because they are the target users for this new ITS i.e., iSTART-Early.

In addition to the adult pilot study, two late elementary students were recruited from a summer tutoring group in the Midwest with parental consent to test various features of the speech modules. Students generally reported positive experiences using the ASR system. The students were found to be fairly fluent readers. The results of these analyses also suggested that the ASR system transcribed students' reading fairly accurately. When errors were qualitatively analyzed some notable sources of error were found to stem from *compound words, homophones, and punctuations such as quotes and parentheses.*

4 Future Work/Next Steps

Our initial test of the ASR system was promising. The ASR transcription accuracy for fluent, adult readers suggests that errors occurring with students could be a result of their less fluent reading skills. However, it is important to note that the youth ASR pilot was conducted with only two students. Therefore, it is necessary to conduct a larger study with children to gain a better understanding of the biggest contributors of low accuracy. In addition to gathering a larger sample of read aloud data, we also need to evaluate the ASR system accuracy when students are generating responses such as verbalizing a self-explanation, a question, or a summary. The generation of constructed responses differs from read aloud activities because the students will be turning the microphone on and off as they generate thoughts and will have the opportunity to edit incorrect words that they speak or words that were transcribed incorrectly. Future studies will investigate the accuracy of students' read aloud protocols as well as generated responses in an effort to ensure that the ASR system is both accurate and sufficiently easy for students to use successfully. Further, we will collect keystroke and log data to better understand what types of errors students are making, what kinds of errors they choose to correct, and when they do so (i.e., as soon as the error is made or when they are done). The more we understand the error and correction patterns that frequently occur, the better we will be able to adapt the system to meet the needs of young learners.

Acknowledgements. The authors would like to recognize the support of the Institute of Education Sciences (IES), U.S. Department of Education through Grant R305A190050, the National Science Foundation - NSF through Award# 2131052, and the Office of Naval Research - ONR through









Grant N000142012623. The opinions expressed are those of the authors and do not represent views of the IES, the NSF, or the ONR.

References

1. Mayer, R.E.: *Multimedia Learning*, 2nd edn. Cambridge University Press, New York (2009)
2. Wijekumar, K.K., Meyer, B.J., Lei, P.: Large-scale randomized controlled trial with 4th graders using intelligent tutoring of the structure strategy to improve nonfiction reading comprehension. *Educ. Technol. Re. Dev.* **60**(6), 987–1013 (2012)
3. Berninger, V.W., Abbott, R.D., Augsburger, A., Garcia, N.: Comparison of pen and keyboard transcription modes in children with and without learning disabilities. *Learn. Disabil. Q.* **32**, 123–142 (2009)
4. Bourdin, B., Fayol, M.: Is written language production more difficult than oral language production: a working-memory approach. *Int. J. Psychol.* **29**, 591–620 (1994)
5. McNamara, D.S.: SERT: self-explanation reading training. *Discourse Process.* **38**(1), 1–30 (2004)
6. Anggraini, N., Kurniawan, A., Wardhani, L.K., Hakiem, N.: Speech recognition application for the speech impaired using the android-based google cloud speech API. *Telkomnika* **16**(6), 2733–2739 (2018)
7. Hagen, A., Pellom, B., Cole, R.: Highly accurate children’s speech recognition for interactive reading tutors using subword units. *Speech Commun.* **49**(12), 861–873 (2007)
8. Qualtrics: Version (2020). Qualtrics (2005). <https://www.qualtrics.com>
9. Flesch, R.: Flesch-Kincaid readability test, **26**(3) (2007). Accessed October 2007
10. Crossley, S.A., Kyle, K., McNamara, D.S.: The tool for the automatic analysis of text cohesion (TAACO): automatic assessment of local, global, and text cohesion. *Behav. Res. Methods* **48**(4), 1227–1237 (2016)
11. Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: Burges, C.J.C., Bottou, L., Welling, M., Ghahramani, Z., Weinberger, K.Q. (eds.) *Advances in Neural Information Processing Systems*, pp. 3111–3119. Curran Associates, Red Hook (2013)
12. Crossley, S.A., Kyle, K., Dascalu, M.: The tool for the automatic analysis of cohesion 2.0: integrating semantic similarity and text overlap. *Behav. Res. Methods* **51**(1), 14–27 (2018). <https://doi.org/10.3758/s13428-018-1142-4>



iSTART-Early: Interactive Strategy Training for Early Readers

Panayiota Kendeou¹ , Ellen Orcutt¹ , Tracy Arner² , Tong Li² ,
Renu Balyan³ , Reese Butterfuss² , Micah Watanabe² ,
and Danielle McNamara² 

¹ University of Minnesota, Minneapolis, MN, USA
kend0040@umn.edu

² Arizona State University, Arizona, USA

³ State University of New York College, Old Westbury, USA

Abstract. In this paper, we present iSTART-Early, an intelligent tutoring system that provides automated instruction and practice on higher-order reading comprehension strategies to 3rd and 4th grade students. iSTART-Early provides personalized, interactive, game-based strategy instruction and practice on comprehension strategies (i.e., *Ask It*, *Reword It*, *Find It*, *Explain It*, and *Summarize It*) with grade-appropriate informational texts. Natural language processing (NLP) combined with automated speech recognition (ASR) and text-to-speech technologies enable immediate formative and summative feedback. A teacher interface allows teachers to assign texts and monitor students' performance so that they can provide additional support and feedback when necessary, creating blended-learning opportunities. We describe the interface and the development of iSTART-Early, as well as our plans to examine the intelligent tutoring system for *usability*, *feasibility* and *promise* in improving reading comprehension strategies and outcomes for young readers.

Keywords: ITS · Reading strategies · ASR

1 Introduction

Recognizing the promise of educational technologies for literacy instruction, developers have created computer-based learning environments to support the development of reading skills [1]. The purpose of such tools is to integrate targeted, individualized instruction into existing literacy instruction. In this context, iSTART-Early was developed to improve students' use of higher-order comprehension strategies in the service of reading comprehension, including comprehension monitoring and question asking, paraphrasing, inferencing (prediction, bridging, elaboration), explanation, and summarization.

In the average-sized classroom, it is unrealistically demanding for teachers to provide consistent and timely individualized instruction and feedback on how well students are implementing reading comprehension strategies. iSTART-Early will provide students with real-time feedback and instruction aiming to improve the use of effective

The original version of this chapter was revised: The original version did not include the correct acknowledgement. The correction to this chapter is available at https://doi.org/10.1007/978-3-031-09680-8_36

comprehension strategies. Thus, each student receives personalized instruction from the system just-in-time. Furthermore, iSTART-Early will allow teachers to adjust their classroom instruction, using student performance data in the teacher interface to inform decision making. Teachers will also be provided with additional resources and instructional materials to facilitate blended learning.

2 Technological Foundations of iSTART-Early

iSTART-Early builds and expands on the existing iSTART. The original version of iSTART showed significant improvements in comprehension and self-explanation quality in comparison to students in a control condition who self-explained the target text without prior instruction on and practice with the strategies [2]. Practice games were added to iSTART in order to increase engagement when interacting with the system over longer periods. These subsequent versions of iSTART have yielded similar gains and have increased engagement and motivation through the addition of more interactive, game-based modules [3]. iSTART has been shown to improve comprehension for middle school [4], high school [3, 5] and college students [6, 7].

In its current form for adolescent and adult students, iSTART-3, students learn the five comprehension strategies inspired from the Self-Explanation Reading Strategy Training model (SERT) [8], in addition to question-asking and summarization through video lessons, guided demonstration, and game-based practice [9, 10]. Importantly, iSTART-3 has a responsive-design, allowing for practice on desktops as well as mobile devices (e.g., tablets, cell phones).

Three major technological advances are incorporated into iSTART-Early using iSTART-3 as a foundation. The first advance is the development and increased accuracy of automated speech recognition (ASR) technology, which means that students will not need to type their responses. The second is the expansion of Natural Language Processing (NLP) algorithms and linguistic features that can accurately assess shorter phrases that are less structured or syntactically incorrect. These advances will allow iSTART-Early to provide automated reading strategy training to a younger age group (grades 3–4). Third, iSTART-Early includes options to have the text read aloud by a pedagogical agent using text-to-speech technologies. Incorporating these advances is imperative because 3rd and 4th graders need options to read the text aloud and speak their responses while learning reading comprehension strategies. Students in 3rd and 4th grades struggle with generating longer strings of text on a keyboard because they still “hunt and peck” for the letters and symbols that they need rather than typing fluently [11].

3 Core Components of iSTART-Early

3.1 Background Story and Goals

iSTART-Early is encapsulated within a meta-game, in which students participate as space travelers traveling through planets making their way back to Earth (Fig. 1). A pedagogical agent, a dog named Arfes, guides students through the five instructional planet modules.

Each planet targets one of the five major reading strategies, modified from iSTART [8]: Ask It, Reword It, Find It, Explain It, and Summarize It. Each planet includes explicit instruction in the strategies (using interacting video lessons) and engaging games that provide identification and/or generative practice opportunities. Each instructional planet is designed to build upon one another, and students must attain a certain level of performance on one planet before traveling to the next. Along their journey, students earn system currency for their performance on games and can redeem these system currencies for rewards, such as different outfits for their avatar.



Fig. 1. iSTART-Early journey map.

3.2 Lessons

A series of video lessons provide guided instruction on and demonstration of the five targeted reading strategies. Each lesson includes a video lesson that provides explicit instruction in the major strategy and its sub-strategies, in addition to embedded practice questions to ensure comprehension of the strategy and its utilization in reading.

Lesson 1: Ask It. This lesson focuses on Comprehension Monitoring and Question Asking. Comprehension monitoring is the process of being aware of one's own understanding [12–16]. This lesson teaches students how to identify detail vs. big picture questions as well as identify whether an answer to a question can be found in the text vs. needs to be answered by an outside source of information.

Lesson 2: Reword It. This lesson focuses on Paraphrasing. Paraphrasing is restating the text in different words and, preferably, in a reader's own words. It is an important part of the comprehension process because readers often paraphrase the sentence in order to begin an explanation [4]. This lesson teaches students how to paraphrase by teaching four paraphrasing sub-strategies: using synonyms, restructuring the sentence's syntax, splitting a long sentence into two, and combining two short sentences.

Lesson 3: Find It. This lesson focuses on Identifying Important Sentences in the text. Identification of main ideas is an effective strategy to improve comprehension performance [17, 18]. Indeed, explicit instruction in main idea identification and summarization was identified as one of the highest impact instructional practices that teachers can use to improve reading comprehension [19–24].

Lesson 4: Explain It. This lesson focuses on Explanation. Self-explanation is the process of explaining orally, or in writing, the meaning of written text to oneself, which can improve deep-level comprehension of content texts [25]. Explanation serves as a vehicle for students to move beyond the text and generate inferences that connect ideas in the text and background knowledge. In this context, two sub-strategies are taught, bridging and elaboration. **Bridging inferences** link ideas and the relations between separate sentences in the text. Making bridging inferences is critical to text comprehension because texts normally do not (or cannot) state all of the relevant information [26]. **Elaborative inferences** link what is in the text to related knowledge, such as individual prior knowledge, different text passages, or other outside sources of information.

Lesson 5: Summarize It. Summarization helps reduce the text to its core ideas. During summarization, readers identify irrelevant information, integrate content with pre-existing knowledge [27], and better retain text material [28]. This lesson builds upon the skills taught in previous lessons and emphasizes a number of sub-strategies, such as identifying main ideas and writing a topic sentence.

3.3 Games

Practice games were developed to increase engagement when interacting with the system over longer periods and provide opportunities for deliberate practice. In particular, we designed a series of practice games based on the space-travel theme and the main characters of the meta-game. Each game presents a unique scenario with challenges that require students to practice targeted reading strategies.



Fig. 2. Question asking game: choosing good questions for Alien students

For example, in the Question Asking game, we created a scenario where the player and Arfes (the pedagogical agent) land on a planet and visit an alien school. The principal asks Arfes to become a substitute teacher and teach alien students using an informational text (see Fig. 2). The player needs to help Arfes decide which questions are the best to deepen the alien students' understanding of the assigned text. The game provides feedback, including alien students' reactions to indicate whether or not the questions chosen by the player benefit the alien students.

3.4 Texts

The iSTART-Early team decided to focus the reading strategy instruction specifically in the context of informational, scientific texts. This decision is supported by the scarcity of access to informational texts in elementary settings [29], despite the increasing demands to include informational texts in reading curriculum [30]. Furthermore, informational texts have differing comprehension demands than narrative texts [31]. Exposure to informational texts strengthens a students' prior knowledge [32], as well as their understanding of organizational structures and text features that are not common in narrative texts [33]. The texts of iSTART-Early have been developed by professional text writers, and they received extensive reviews and fact-checking. The final corpus consists of 84 original texts, 34 of which are paired versions of different lengths.

3.5 Automated Speech Recognition (ASR)

The student interface includes ASR technology to allow students to practice reading strategies without the additional burden of typing [34]. ASR technologies have advanced remarkably in recent years, and even conventional speech recognition systems now perform satisfactorily on children's speech input. Once the students have spoken aloud their response, the ASR system transcribes their verbal responses into text. The transcribed text is filtered for frozen expressions, cuss words, or any misspelled words. The transcribed text is then presented to the student with the option to edit their response for any transcription inaccuracies (Fig. 3).

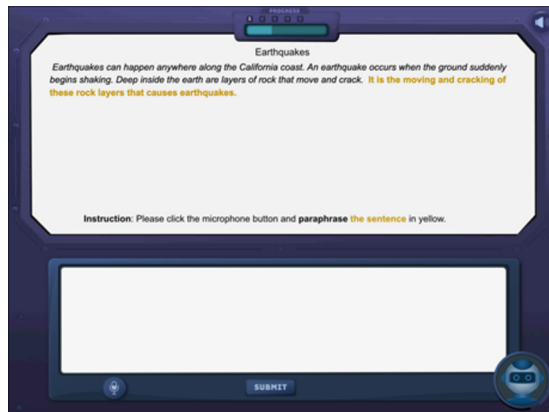


Fig. 3. ASR integration into iSTART-Early

3.6 Text-to-Speech

A text-to-speech (TTS) function has been implemented in the system to address potential difficulties with basic reading processes. TTS will read aloud full texts and difficult

words to the students. Previous research has shown that TTS improves reading fluency [35] and reading comprehension [36, 37], particularly benefiting students with reading difficulties. As illustrated in Fig. 4, when a student clicks the play button (the button in blue), the system plays the text audio from the database for the student. Students also have the capability to pause or replay the text audio. The difficult words are marked in blue to differentiate from the remaining text. A student can hover their cursor over each difficult word to learn the words' pronunciation and definition.

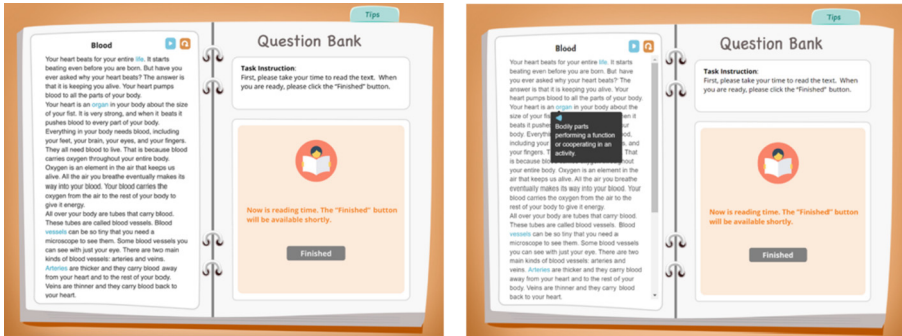


Fig. 4. Text-to-speech: full text read-aloud function (left) and difficult words read-aloud function (right panel)

3.7 Teacher Interface

The teacher interface was integrated with the iSTART-2 teacher interface [5] and allows teachers to assign students to the version of iSTART that is appropriate for their grade level. Within iSTART-Early, teachers can assign specific texts to students and add their own texts to the system library. Teachers are provided both class- and student-level data on lesson progress (including number of texts read, games played, modules completed) and reading strategy performance. The teacher interface allows for blended learning across environments, personalized instruction, and real-time feedback.

4 Conclusion

The core components and implementation features of iSTART-Early presented in this paper are designed to provide explicit instruction for comprehension strategies, with grade-level informational texts so that students can build relevant background knowledge while learning reading strategies. Build-in supports, such as ASR and TTS, are designed to address the needs of diverse and multilingual learners. Immediate feedback, gamification, and deliberate practice are designed to enhance motivation and self-regulation. Our future work will assess the usability, feasibility, and effectiveness of iSTART-Early with our target audience of 3rd and 4th grade students and teachers. Based on our initial pilot studies and prior work, we hypothesize that usable and feasible school implementation

of iSTART-Early with fidelity will improve students' reading comprehension strategy use during the reading of informational texts, and therefore improve overall reading comprehension performance.

Acknowledgment. The research reported here was supported by the Institute of Education Sciences, U.S. Department of Education, through Grant R305A190050 to Arizona State University. The opinions expressed are those of the authors and do not represent views of the Institute or the U.S. Department of Education.

References



1. Crossley, S.A., McNamara, D.S.: Say more and be coherent: how text elaboration and cohesion can increase writing quality. *J. Writ. Res.* **7**, 351–370 (2016)
2. McNamara, D.S., O'Reilly, T.P., Best, R.M., Ozuru, Y.: Improving adolescent students' reading comprehension with iSTART. *J. Educ. Comput. Res.* **34**, 147–171 (2006)
3. Jackson, J.G., McNamara, D.S.: Motivation and performance in a game-based intelligent tutoring system. *J. Educ. Psychol.* **105**, 1036–1049 (2013)
4. McNamara, D.S., O'Reilly, T.P., Rowe, M., Boonthu, C., Levinstein, I.B.: iSTART: a web-based tutor that teaches self-explanation and metacognitive reading strategies. In: McNamara, D.S. (ed.) *Reading Comprehension Strategies: Theories, Interventions, and Technologies*, pp. 397–421. Psychology Press, London (2007)
5. Snow, E.L., Jacovina, M.E., Jackson, G.T., McNamara, D.S.: iSTART-2: a reading comprehension and strategy instruction tutor. In: Crossley, S.A., McNamara, D.S. (eds.) *Adaptive Educational Technologies for Literacy Instruction*, pp. 104–121. Taylor & Francis, New York (2014)
6. Magliano, J.P., Todaro, S., Millis, K.K., Wiemer-Hastings, K., Kim, H.J., McNamara, D.S.: Changes in reading strategies as a function of reading training: a comparison of live and computerized training. *J. Educ. Comput. Res.* **32**, 185–208 (2005)
7. O'Reilly, T.P., Sinclair, G.P., McNamara, D.S.: iSTART: a web-based reading strategy intervention that improves students' science comprehension. In: *Proceedings of the IADIS International Conference Cognition and Exploratory Learning in the Digital Age*, Lisbon, Portugal, pp. 173–180 (2004)
8. McNamara, D.S.: SERT: self-explanation reading training. *Discourse Process.* **38**, 1–30 (2004)
9. McCarthy, K.S., Likens, A.D., Johnson, A.M., Guerrero, T.A., McNamara, D.S.: Metacognitive overload!: positive and negative effects of metacognitive prompts in an intelligent tutoring system. *Int. J. Artif. Intell. Educ.* **28**, 420–438 (2018)
10. McCarthy, K.S., Watanabe, M., McNamara, D.S.: The design implementation framework: guiding principles for the redesign of a reading comprehension intelligent tutoring system. In: Schmidt, M., Tawfik, A., Earnshaw, Y., Jahnke, I. (eds.) *Learner and User Experience Research*. Springer, Cham (2020) . https://edtechbooks.org/ux/9_the_design_impleme
11. Wijekumar, K.K., Meyer, B.J.F., Lei, P.-W.: Large-scale randomized controlled trial with 4th graders using intelligent tutoring of the structure strategy to improve nonfiction reading comprehension. *Educ. Tech. Res. Dev.* **60**, 987–1013 (2012)
12. McNamara, D.S.: *Reading Comprehension Strategies: Theories, Interventions, and Technologies*. Psychology Press, London (2007)
13. National Reading Panel: *Teaching Children to Read: An Evidence-Based Assessment of the Scientific Research Literature on Reading and Its Implications for Reading Instruction*. National Reading Panel, Bethesda, MD (2000)

14. Palincsar, A.S., Brown, A.L.: Reciprocal teaching of comprehension-fostering and comprehension-monitoring activities. *Cogn. Instr.* **1**, 117–175 (1984)
15. Palincsar, A.S., Brown, A.L.: Interactive teaching to promote independent learning from text. *Read. Teach.* **39**, 771–777 (1986)
16. Paris, S.G., Oka, E.R.: Children’s reading strategies, metacognition, and motivation. *Dev. Rev.* **6**, 25–56 (1986)
17. Biancarosa, C., Snow, C.E.: *Reading Next - A Vision for Action and Research in Middle and High School Literacy: A Report to Carnegie Corporation of New York*, 2nd edn. Alliance for Excellent Education, Washington, DC (2006)
18. Goldman, S.: Adolescent literacy: listening and understanding content. *Future Child.* **22**(2), 89–116 (2012)
19. Edmonds, M.S., et al.: A synthesis of reading interventions and effects on reading comprehension outcomes for older struggling readers. *Rev. Educ. Res.* **79**, 262–300 (2009)
20. Gajria, M., Salvia, J.: The effects of summarization instruction on text comprehension of students with learning disabilities. *Except. Child.* **58**, 508–516 (1992)
21. Garwood, J.D., Brunsting, N.C., Fox, L.C.: Improving reading comprehension and fluency outcomes for adolescents with emotional-behavioral disorders: recent research synthesized. *Remed. Spec. Educ.* **35**, 181–194 (2014)
22. Gersten, R., Fuchs, L.S., Williams, J.P., Baker, S.K.: Teaching reading comprehension strategies to students with learning disabilities: a review of research. *Rev. Educ. Res.* **71**, 279–320 (2001)
23. Scammacca, N.K., Roberts, G., Vaughn, S., Stuebing, K.K.: A meta-analysis of interventions for struggling readers in grades 4–12: 1980–2011. *J. Learn. Disabil.* **48**, 369–390 (2015)
24. Solis, M., Ciullo, S., Vaughn, S., Pyle, N., Hassaram, B., Leroux, A.: Reading comprehension interventions for middle school students with learning disabilities: a synthesis of 30 years of research. *J. Learn. Disabil.* **45**, 327–340 (2012)
25. Chi, M.T.H., Leeuw, N.D., Chi, M.-H., Lavancher, C.: Eliciting self-explanations improves understanding. *Cogn. Sci.* **18**, 439–477 (1994)
26. McNamara, D.S., Kintsch, E., Songer, N., Kintsch, W.: Are good texts always better? Interactions of text coherence, background knowledge, and levels of understanding in learning from text. *Cogn. Instr.* **14**, 1–43 (1996)
27. Wade-Stein, D., Kintsch, E.: Summary street: interactive computer support for writing. *Cogn. Instr.* **22**, 333–362 (2004)
28. Rinehart, S., Stahl, S., Erickson, L.: Some effects of summarization training on reading and studying. *Read. Res. Q.* **21**, 203–214 (1986)
29. Duke, N.K.: 3.6 Minutes per day: the scarcity of informational texts in first grade. *Read. Res. Q.* **35**, 202–224 (2000)
30. National Governors Association Center for Best Practices & Council of Chief State School Officers: *Common Core State Standards for English Language Arts and Literacy in History/Social Studies, Science, and Technical Subjects*. National Governors Association Center for Best Practices, Council of Chief State School Officers, Washington DC (2010)
31. McNamara, D.S., Ozuru, Y., Floyd, R.G.: Comprehension challenges in fourth grade: the roles of text cohesion, text genre, and readers’ prior knowledge. *Int. Electron. J. Elementary Educ.* **4**, 229–257 (2011)
32. Caswell, L.J., Duke, N.K.: Non-narrative as a catalyst for literacy development. *Lang. Arts* **75**, 108–117 (1998)
33. Pappas, C.C.: Is narrative “primary”? Some insights from Kindergarteners’ pretend readings of stories and information books. *J. Lit. Res.* **25**, 97–129 (1993)
34. Hagen, A., Pellom, B., Cole, R.A.: Highly accurate children’s speech recognition for interactive reading tutors using subword units. *Speech Commun.* **49**, 861–873 (2007)

35. Hartas, C., Mosley, D.: 'Say that again please': a scheme to boost reading skills using a computer with digitized speech. *Support Learn.* **8**(1), 16–21 (1993)
36. Pisha, B., Coyne, P.: Smart from the start: the promise of universal design for learning. *Remed. Spec. Educ.* **22**(4), 197–203 (2001)
37. Stodden, R.A., Roberts, K.D., Takahishi, K., Park, H.J., Stodden, N.J.: Use of text-to-speech software to improve reading skills of high school struggling. *Procedia Comput. Sci.* **14**, 359–362 (2012)



Correction to: Intelligent Tutoring Systems

Scott Crossley  and Elvira Popescu 

Correction to:

S. Crossley and E. Popescu (Eds.): *Intelligent Tutoring Systems*, LNCS 13284, <https://doi.org/10.1007/978-3-031-09680-8>

In chapters 9 and 10

Due to an oversight, the originally published version of this paper did not include the acknowledgement “We acknowledge the support of CRIAQ, the Natural Sciences and Engineering Research Council of Canada (NSERC), CAE, Bombardier, and BMU.” This has now been included.

In chapter 35

Due to an oversight, the originally published version did not include the acknowledgement for the grants that supported the research in this paper. This has now been included.

The updated version of these chapters can be found at
https://doi.org/10.1007/978-3-031-09680-8_9
https://doi.org/10.1007/978-3-031-09680-8_10
https://doi.org/10.1007/978-3-031-09680-8_35

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2022
S. Crossley and E. Popescu (Eds.): ITS 2022, LNCS 13284, p. C1, 2022.
https://doi.org/10.1007/978-3-031-09680-8_36

Author Index

- Aghajani, Mahsa 183
Akhuseyinoglu, Kamil 36
Alamri, Ahmed 227
Albacete, Patricia 88
Alrajhi, Laila 227
An, Sungeun 303
Anikin, Anton 293
Aprin, Farbod 264
Arner, Tracy 362, 371
- Badea, Gabriel 24
Balyan, Renu 362, 371
Ben Abdessalem, Hamdi 183, 190
Broisin, Julien 148
Brusilovsky, Peter 36
Butterfuss, Reese 362, 371
- Chandrasekaran, Deepak 139
Chang, Maiga 115, 139
Cheng, Yixin 238
Chounta, Irene-Angelica 264
Contractor, Maheen Riaz 323
Courtemanche, Marc-Antoine 95
Cristea, Alexandra I. 227, 313
Crossley, Scott 171
- Dagorret, Pantxika 161
Dare, Kodjine 190
Dascalu, Mihai 350
- Edwards, Lucas 336
Etcheverry, Patrick 161
- Fleming, Scott 36
Frasson, Claude 183, 190
Freedman, Reva 336
Fuchs, Johannes 3
- Gallagher, John 88
Gavrilova, Tatiana 51
Goel, Ashok K. 303
Graf, Sabine 139
Guarnieri, Vincent 350
Guin, Nathalie 148
- Hayashi, Yugo 343
Hodgson, Ryan 313
Hoppe, H. Ulrich 264
- Jakob, Benedikt 276
Jordan, Pamela 88
- Kamennov, Yaroslav 65
Kapuscinski, Tomasz 75
Katz, Sandra 88
Kendeou, Panayiota 362, 371
Krahn, Ted 115
Kraus, Matthias 3
Krouska, Akrivi 204
Kulyukin, Kirill 293
Kuo, Rita 115
- LaBarbera, Dean 336
Lee, Woojin 17
Lefevre, Marie 148
Li, Tong 362, 371
Litovkin, Dmitrii 293
Lomas, Derek 255
Lopisteguy, Philippe 161
Lugrin, Birgit 276
- Malvi, Shrey 255
Manrique, Rubén 238
Marquesuzaa, Christophe 161
McNamara, Danielle 362, 371
Morita, Junya 343
Murthy, Sahana 213
- Nana Tato, Gabrielle Joyce 105
Naples, Virginia 336
Nastase, Mariana Madalina 197
Neagu, Laurentiu-Marian 350
Nkambou, Roger 95, 105, 123
Nodenot, Thierry 161
- Ohmoto, Yoshimasa 343
Orcutt, Ellen 362, 371
Orlova, Yulia 65
- Patel, Darshan 255
Patel, Nirmal 255

Pathan, Rumana 213
Pereira Nunes, Bernardo 238
Platt, Thomas 88
Popescu, Doru Anastasiu 197
Popescu, Elvira 24

Rabadiya, Bansri 255
Rajendran, Ramkumar 213
Riedmann, Anna 276
Rigaud, Eric 350
Rivero, Carlos R. 323
Roux, Lisa 161
Rus, Vasile 36

Sablayrolles, Louis 148
Schaper, Philipp 276
Seo, Kangbok 17
Sgouropoulou, Cleo 204
Sharma, Aditya 255
Shimojo, Shigen 343
Silliman, Scott 88

Singh, Daevesh 213
Sullivan, Ian 336
Sychev, Oleg 51, 65, 293

Tamang, Lasang Jimba 36
Tato, Ange 95, 105, 123
Thakkar, Mithilesh 255
Tian, Yu 171
Travadel, Sébastien 350
Troussas, Christos 204

Ughev, Viktor 51

Wan, Qian 171
Wang, Jingyun 313
Wasowski, Radomir 115
Watanabe, Micah 371
Weigel, Emily 303

Yang, Tiffany 88