Learning Technology

Publication of

IEEE Computer Society's

Technical Committee on Learning Technology (TCLT)

Volume 21 Issue 4 **ISSN 1438-0625** **December 2021**

Editorial Board

Call for Articles

The IEEE Technical Committee on Learning Technology (TCLT) has been founded on the premise that emerging technology has the potential to dramatically improve learning. The purpose of this technical committee is to contribute to the field of Learning Technology and to serve the needs of professionals working in this field.

The Bulletin of the Technical Committee on Learning Technology aims to report (1) the up-to-date outcome of the emerging learning technologies, (2) the review of learning technology related books, instruments or reports, (3) the collaboration opportunities of work-in-progress research ideas and projects, (4) the current development status of learning technology in the developing countries, and (5) the announcements of the upcoming activities that the learning technology community may interest. It would also serve as a channel to keep everyone aware of Technical Committee's activities.

The bulletin is calling for articles in the following sections:

- **Emerging Learning Technologies:** an article with up to 8 pages the research outcome of learning technologies, including systems, tools, apps, etc., no theoretical or concept only research would be accepted.
- **Equity, Diversity & Inclusion (EDI):** an article with up to 4 pages to discuss the issues for minorities in STEM education and how the community deal with the matter.
- **Book & Report Reviews:** an article with up to 4 pages.
- **Collaboration Opportunities:** an article with up to 4 pages to talk about the research progress and stage outcome as well as the aspects and needs of looking for collaborations.
- **Report from Developing Countries:** an article with up to 6 pages to describe the current research progress/difficulties/needs/limitations of the learning technology in the developing countries.
- **Event Info & Call for Event Host:** 1 page.

The bulletin articles have to give readers clear idea and vision of the advanced learning technologies with rich and proper figures, screenshots, and diagrams.

For preparing your manuscript, please follow the IEEE guidelines and use the template at https://ieeeauthor.wpengine.com/wp-content/uploads/Transactions-brief-short-or-communications-article-template.doc. Please submit your manuscript to tclt-bulletin@ieee.org in Word format with the subject title "Bulletin Submission for [*section*]" (*section* indicates which section you would like to submit). All figures should be in high resolution and embedded in the main text.

The bulletin is included in Emerging Sources Citation Index (ESCI). The first decision for the submission is in 24 days.

Editorial

Maiga Chang, Rita Kuo, Jun Chen Hsieh, Danial Hooshyar, Jerry Chih-Yuan Sun

As the Covid-19 pandemic spread around the world in the past two years, researchers are working hard on the work-family balance [1]. The IEEE Technical Committee on Learning Technology (TCLT) hopes that the bulletin can be the platform for researchers presenting their works, looking for collaborators, and delivering the academic activities information during the pandemic. There are eight articles published in this issue after the rigorous review process. The articles include one in Emerging Learning Technologies section, one in Equity, Diversity & Inclusion section, one in Book & Report Reviews section, and five in Event Info & Call for Event Host section.

The first article entitled "Neural-symbolic computing: A step toward interpretable AI in Education" written by Hooshyar and Yang discusses an innovative application of neural-symbolic computing (NSC) in education. The research team aims to propose a general framework that embeds the NSC paradigm into the educational context, offering a valuable starting point for interpretable AI in education. The proposed framework addressed the challenges of machine learning and AI about explainability and a lack of training data, by combining symbolic reasoning and neural learning models. The novel approach of NSC can provide better transparency in learning, leading to nurturing educators, practitioners, and learners' trust in data-based decisions.

In Equity, Diversity & Inclusion section, the article "The Women in Computer Science Program in New Mexico Tech" written by Knowles summarizes the four-year work in the Women in Computer Science (WiCS) program in a STEM focused university – New Mexico Tech. The program changes the introductory programming course with two parallel sections for students with and without programming experience before attending the college. To engage students' interests in learning computer science, the program also integrates game design topics in the courses. Last but not least, the program got the funding form the National Center for Women & Information Technology for developing content for outreach and bridging the knowledge gap for transfer students and incoming freshmen. The program will keep working with the local public schools to encourage more students enrolling computer science majors.

The article "Comparative Analysis of Object Visualization Tools with Respect to Their Use in Education" written by Dennis and Ramyaa is published in the Book & Report Reviews section. This work presents a comparative analysis of object visualization tools regarding their use in Education. Dimensions that authors used to compare the tools include detailed graphics, object class graphics, single instance view, data structure view, additional diagram generation, and usability in introducing object-oriented programming (OOP) concepts. Findings of this research shows that while the existing tools provide sufficient support for the named dimensions, they ignore underlying ideas of OOP like object design.

There are five articles published in the Event Info & Call for Event Host section; three of the events are in the technology enhanced language learning area. The Annual Symposium of the Computer-Assisted Language Instruction Consortium (CALICO) will take place from May 31st to June 4th, 2022 in Seattle, Washington in the Seattle Renaissance Hotel. The five-day conference gathers scholars around the globe features workshops, presentations, panels, technology showcases, poster presentations, special interest group meetings, and social events that engage scholars in an in-depth discussion on topics related to technology and language learning.

The 1st International Conference on Future Language Learning (ICFULL) will be held online from July 1st to 3rd, 2022, with the theme of "Future Language Learning". With diverse issues and topics included in keynotes, paper presentations, and panel discussions, ICFULL 2022 offers a channel through which scholars, practitioners, and researchers could share and exchange research findings regarding the potentials of cutting-edge technologies for language pedagogical practices.

The 5th Pedagogy and Practice in Technology Enhanced Language Learning (PPTELL) will take place from July 1st to 3rd, 2022 as an online conference by Shaanxi Normal University and the PPTELL Association. Through oral presentations and poster presentations, the attendees gain a better understanding of "Developing learner agency through smart technology", aiming to help language learners become autonomous learning agents with the use of smart technology.

The last two articles in the Event Info & Call for Event Host are from two conferences in Romania. The Intelligent Tutoring Systems (ITS) Conference directs researchers and academics, the industry, and end-users to fields of computer and cognitive sciences, artificial intelligence and deep learning in tutoring the education from June 27th to July 1st, 2022 at University Politehnica of Bucharest, Romania. To enhance learning through interdisciplinary intelligent systems, ITS 2022 leads the attendees to the evaluation of the use of Intelligent Systems in education, modeling innovative applications of technologies and the adaptation of systems to specific groups of learners.

The University Politehnica of Bucharest also hosts the 22nd IEEE International Conference on Advanced Learning Technologies (ICALT) which will be held from July 1st to 4th, 2022 as a hybrid event. As ranked the second (2nd) by saliency in Educational Technology area in Microsoft Academic from 2000 to 2020, ICALT 2022 provides an academic platform featuring 15 tracks, engaging researchers in sharing and discussion related to design, development, use, and evaluation of technologies in learning. Because there is one-day overlapping between ITS 2022 and ICALT 2022 in the same venue, both ITS and ICALT participants can attend extra keynote and activities, from ITS's last day and ICALT's first day, for free.

The current submission statistics in the Bulletin of TCLT show that authors receive the first decision notification in average 20.66 days, and for the accepted articles the authors get the acceptance notification in average 43.59 days. The accepted articles are published online in average 88.01 days after they were submitted. The Editorial Board of TCLT wishes the learning technology community stay healthy and has a productive year in 2022.

References

[1] Landolfi, A., Barattucci, M., De Rosa, A., & Lo Presti, A. (2021). The Association of Job and Family Resources and Demands with Life Satisfaction through Work–Family Balance: A Longitudinal Study among Italian Schoolteachers during the COVID-19 Pandemic. *Behavioral Sciences, 11*(10), 136.

Neural-symbolic computing: A step toward interpretable AI in Education

Danial Hooshyar and Yeongwook Yang

Abstract— **The application of machine learning in education has some challenges, including data bias, sparsity, dependency on large amounts of data, and a lack of interpretability. Further, many learning methods cannot obtain additional information out of the scope of the training data. While the application of neural-symbolic computing (NSC) has been investigated in different domains, by addressing the ambiguities in data and reasoning behind the learning outputs, these frameworks have never been studied in the educational context. We argue that integrating NSC frameworks into educational contexts can be greatly important. Thus, we propose a general framework for embedding NSC paradigm into the educational context, offering a valuable starting point for a framework of interpretable AI in education. The proposed framework can facilitate providing better transparency in learning, leading to nurturing educators, practitioners, and learners' trust in data-based decisions. Besides, it can open a door to many scholars dealing with the issue of the lack of data and data inconsistency for obtaining accurate predictive results by improving learning and reasoning.**

Index Terms— **Interpretability, Machine learning in education, Neural-symbolic computing, Transparency in AI**

I. Introduction

Machine learning (ML) has proven to be successful in developing models for pattern recognition in engineering. Aside from its classical application, (deep) neural-network-based models have shown great success in domains such as speech recognition (e.g., [1]), computer vision (e.g., [2]), and natural language processing (e.g., [3]). Education has also benefited greatly from the application of such techniques. While prediction of students' performance (e.g., [4]), procrastination behaviors (e.g., [5]), and grades in online courses (e.g., [6]) are interesting of application of classical machine learning techniques in education, educational researchers have begun applying (deep) neural network techniques to improve previous state-of-the-art results (e.g., for student modeling). Thus, Piech et al. [7] modeled student learning through knowledge tracing problems using Recursive Neural Networks (RNNs) and achieving higher predictive capability compared to Bayesian knowledge tracing models. The reason for such success is the data-driven nature of these approaches, with models learning from examples. Nonetheless, these data-based methods can also result in unsatisfactory outcomes or reach their limits. A lack of enough data to train generalized and well-performing models is the most obvious scenario. On the other hand, there might be many heterogeneous data in some areas, which requires further knowledge description sources. More importantly, purely data-driven models may fail to meet necessary constraints, such as being dictated by learning principles, or educational guidelines. During the past decade, many governmental institutes have funded the promotion of machine learning and AI, although the effectiveness of educational programs and research results can be difficult to assess due to poor explainability and limited reproducibility of ML-based computations [8], [9]. This results in some limitations when it comes to the systems' accountability, and can lead to challenges like poor reliability and an overall lack of trustworthiness. Thus, there is an urgent need for such models to be explainable and interpretable.

These challenges have resulted in growing research around improving the accuracy of ML-based models and making them interpretable. A prospective solution is neural-symbolic computing (NSC), which aims to integrate robust neural learning and sound symbolic reasoning. The NSC framework revolves around grounding symbolic knowledge into the learning process of neural networks and translating implicit knowledge from the networks to explicit, interpretable symbolic knowledge [10], [11]. This insertion of domain knowledge aims to provide the network with an alternative or additional pre-training, whereas extraction of knowledge after the training offers an explanation of the decision. For instance, Diligenti, Roychowdhury, and Gori [12] and Karpatne et al. [13] have added logic rules and algebraic equations as constraints to the loss function of neural networks.

While the application of NSC has been thoroughly investigated in different domains (such as biology, game AI, engineering) to deal with issues related to a lack of data, data inconsistency, and improving reasoning and interpretability, such frameworks have not been studied in the educational context. Given that educational studies have frequently employed (deep) neural networks in their predictive models (e.g., [14]) and that some challenges can have a serious negative impact on the prediction power of educational predictive models, we argue that integrating NSC frameworks into an educational context can be greatly important. Thus, we propose a general framework for integrating the NSC paradigm into an educational context (e.g., predictive models), which could potentially offer a valuable starting point for the framework of interpretable AI in education. Application of this framework in education would not only lead to nurturing educators, practitioners, and learners' trust in such data-driven decisions, but also it assist researchers dealing with the issue of the lack of data and data inconsistency for obtaining accurate predictive results.

The structure of this article is as follows: Section two presents related research on machine learning in education and neural-symbolic computing; Section three puts forward the proposed framework for applying neural-symbolic computing to education; and Section four provides discussion and future work.

Received September 12, 2021, Accepted October 13, 2021, Published online November 14, 2021.

This research was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (No. 2021R1C1C2004868), and the Institute for Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No. 2020-0-00368, A Neural-Symbolic Model for Knowledge Acquisition and Inference Techniques).

D. Hooshyar is with the Department of Computer Science and Engineering, Korea University, Seoul, Republic of Korea and School of Digital Technologies, Tallinn University, Tallinn, Estonia (e-mail: danial.hooshyar@gmail.com).

Y. Yang is with the Division of Computer Science and Engineering, Hanshin University, Osan, Republic of Korea (e-mail: yeongwook.yang@gmail.com).

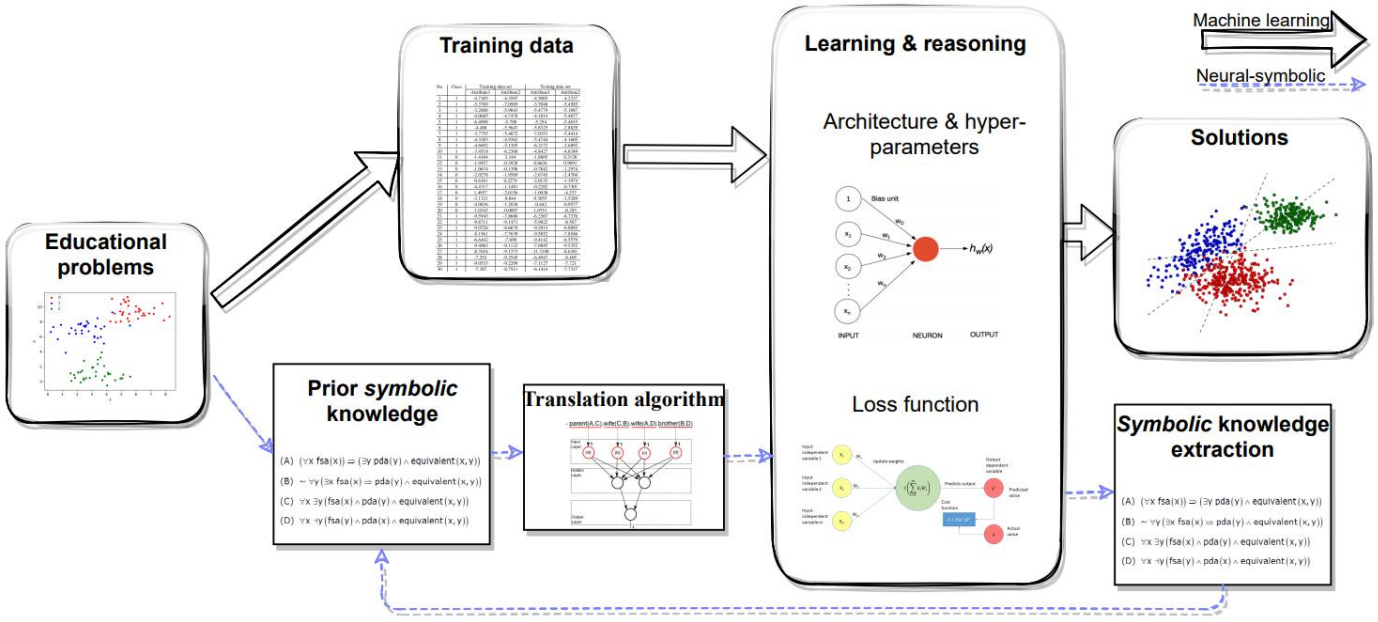

Fig. 1. Overall framework for integration of neural-symbolic computing in Education.

II. Literature Review

A. Machine Learning in Education

The use of data science in education has eased the collection and mining of educational data to unveil learning needs and patterns. This can further help to offer personalized feedback and learning paths to learners or provide them with tailored interventions and learning strategies [15], [16]. To achieve such goals, advanced computational technologies, such as machine learning and artificial intelligence, are key. Machine learning (ML), considered as among the most rapidly growing methods at the core of data science and AI, has shown to be among the most effective techniques for optimizing teaching and learning (e.g., [17]). ML aims to build computer systems that can automatically learn from past experiences without explicit programming [18]. Employing ML in education provides new insights, recommendations, and predictions regarding individualized learning, classifying patterns and profiles, etc. Some examples of ML application in education are modeling students' emotional reaction (e.g., [19]–[21]), knowledge, and abilities (e.g., [22]), automatic generation of hints and feedback (e.g., [23], [24]), prediction of students performance and grades (e.g., [25], [26]), early detection of students at risk of failure (e.g., [27]), prediction of procrastination behaviors (e.g., [5]), etc.

Nonetheless, there are two main challenges in applying ML in education: unavailability of enough training data and a lack of interpretability. Depending on the used approach and the parameter that is being predicted, some ML methods require different data amounts in order to make a highly accurate prediction. To develop highly performed predictions, many existing ML-based methods suggest using a lot of data for training their models. However, a large amount of data does not ensure a highly accurate prediction. The main issue is that, regardless of the amount of data provided for training ML models, purely data-driven models may fail to meet necessary constraints, such as being dictated by learning principles or educational guidelines. Another challenge is the inability of most ML-based models to explain their decisions. Similar to other studies (e.g., [8]), we argue that these techniques and other methods in educational context should be featured with interpretability, so that educators, learners, and decision-makers can understand the reasoning behind the actions or decisions. Such explanation would improve pedagogical effectiveness by helping learners understand why the system decided to recommend a specific topic to be taught, or why their answer is incorrect. Further, it can result in nurturing educators, practitioners, and learners' trust in such decisions, and bringing about their willingness to follow such decisions (e.g., [28]).

B. Neural-Symbolic Computing

Neural-symbolic computing (NSC), an active branch of AI research, combines the two most important cognitive abilities of learning from experiences and reasoning from what has been learned. To this goal, NSC reconciles the symbolic and connectionist paradigms of AI by presenting knowledge in a symbolic form and using neural networks for learning and reasoning [29]. In other words, NSC allows robust learning and inferences in neural networks, accompanied by reasoning with logical systems and interpretability offered by symbolic knowledge extraction.

Recently, researchers have developed numerous NSC methods to address the challenges of ML and AI, including explainability and a lack of training data. For instance, Tran and Garcez [30] introduced a new method, confidence rules, representing propositional formulas in deep neural networks and restricted Boltzmann machines aimed at an easier learning and reasoning process. In their novel approach of injection of the prior knowledge to deep networks, knowledge in the form of rules was translated to the network as weights. This allowed regulating parameters or the network's structure, aimed at guiding learning. The results of their experiment on image-processing tasks revealed that the approach can improve the accuracy of deep belief networks. Hu et al. [31] presented a general distillation framework for transferring knowledge (in the form of first-order logic rules) to neural networks, wherein first-order logic constraints are integrated via posterior regularization. Their approach was similar to Tran's and Garcez's [30], as it injects the prior knowledge into neural networks' weights. However, it was different to previous works in that it used a process of iterative rule distillation to systematically embed structured knowledge to neural networks' parameters. In a different attempt, Serafini and Garcez [32] proposed a framework for learning and

reasoning which translated logical statements into the loss function rather than the network architecture. Their experiments show that it can be efficiently used for knowledge completion and data prediction tasks. Li and Srikumar [33] introduced constraints to the architecture of neural networks through the translation of first-order logic into differentiable components of networks without additional learnable parameters. Experimenting with their framework of natural language processing tasks revealed that it successfully equips neural models with additional domain knowledge during learning and prediction, especially when training data is limited. A general overview of natural-symbolic computing methods is presented by Garcez et al. [11]. Although NSC methods have been extensively studied in different fields and their effectiveness in handling a lack of interpretability and data has been frequently reported, their application in education is limited.

III. Neural-Symbolic Computing for Education: A Framework

According to Garcez et al. [11] and Bader and Hitzler [34], neural-symbolic systems can be constructed through the integration of four components, namely knowledge representation, learning, reasoning, and knowledge extraction. Knowledge representation deals with the translation of symbolic (background) knowledge into the network; learning and reasoning revolve around gaining additional knowledge from examples (and generalization) by the network and executing the network: and knowledge extraction involves symbolic knowledge extraction from the network. While there might not be a solid consensus on the crucial components of NSC, several researchers agree that the knowledge representation is the main cornerstone of NSC, and successful reasoning (or problem-solving) is dependent on an appropriate knowledge representation (e.g., [35]).

Inspired by the ML framework introduced by von Rueden et al. [36], this research proposed a generalizable framework for the integration of NSC in the educational context. Fig. 1 shows the proposed framework's overall architecture. Aside from the available training data, prior knowledge of different forms—such as logic rules, knowledge graphs, etc.—has been integrated into the ML (process). Conventional ML approaches merely feed training data into a machine learning process. Accordingly, this process produces the final hypothesis, solving the task using a model, wherein the knowledge is used at pre-processing stages, particularly in feature engineering. Unlike this, in NSC, represented by dashed arrows in Fig. 1, the domain knowledge is pre-existent and separate from training data, constituting a second source of information that is usually translated to the machine learning process through explicit interfaces. This knowledge can be integrated through regulation of a neural network's architecture and hyper-parameters (e.g., selecting a model structure), or guiding loss function (e.g., by designing an appropriate regularizer). Note that a mere integration of knowledge without training data or using prior knowledge to augment training data cannot be considered as knowledge representation in NSC.

First, a translation algorithm is used to inject the symbolic knowledge into the loss function or initial architecture of neural networks. For instance, bottom-clause propositionalization can be used to convert symbolic knowledge into propositional clauses, which are then embedded into neural networks through setting weights and biases in the network. Second, using a neural learning algorithm, the neural network is trained with training data, revising the theory via the prior background knowledge. Inductive logic programming is among the methods in neural-symbolic learning that benefits from NSC's learning capability to automatically build a logic program from examples (e.g., bottom-up type of logic programming). Thereafter, the reasoning can take place within neural networks through different techniques of model-based or theorem proving, enabling the network to act as a powerful parallel system for computing the logical consequences of the knowledge or theory. The information gained through a computation that takes place during the learning and reasoning process could be employed to optimize the network and have a better representation of the problem domain. Thus, inconsistencies between the training examples and domain knowledge can be resolved. Third, the extraction of revised symbolic knowledge represents the result of the training. Similar to the insertion of rules/knowledge, the extraction algorithm must be demonstrably correct, so that the extracted rules or knowledge is guaranteed to be a rule of the network. Finally, an expert can analyze the extracted knowledge and decide if it can be fed to the system to close the NSC cycle.

An example of the application of the proposed framework in education is the prediction of students' risk of failure in online learning. In such systems, many data on students' patterns of online learning behavior, frequency, intensity, their change over time, etc. are employed to model students' learning behavior. Once the model is trained using these data, the system can predict students' risk of failure (hopefully on time), so that educators and parents can intervene when necessary. However, there are still concerns regarding the predictions by such models: "why should I follow these specific interventions", "why I am the one who's at risk of failure", and so forth. If the trained model can show the path to the decisions, e.g., decision tree, one may observe that a specific student is tagged as the risk of failure mainly because he or she was an introvert and not engaged in chatting with peers in the learning environment. Such inaccurate decisions or predictions could have deteriorating effects on some students and their parents. By including symbolic knowledge into the model (e.g., on students' personality traits), the application of neural-symbolic computing would eliminate the need for the big load of data that may fail to learn the necessary educational guidelines or some impractical rules. Therefore, it offers highly accurate predictions that take into account the desired educational rules or patterns with minimal domain-specific training, also offering explanations of its predictions.

Another example could be when a teacher tries to address the problem of identifying suitable students for entering the Physics Department. After the training, some models may give more weight to students' grades on different courses, ignoring the relationship between the courses and the guidelines for entering the Physics Department. In this situation, the relationship between the subjects can be given to the model as symbolic knowledge so that the model can have better reasoning over the knowledge to make more accurate decisions or predictions.

Finally, Fig. 2 illustrates the application of NSC to solve a visual question answering problem (taken from [37]). In this example, the system requires a huge amount of supervision and training data to answer the question. More specifically, the predictive model takes the question as an input and maps it to an answer using end-to-end reasoning, ignoring that concepts and reasoning are entangled. This may make it hard to completely solve the task, but also to create transfer to similar tasks. On the other hand, NSC can relax such challenges and learn to solve the task by bridging the gap between symbolic and sub-symbolic methods (in other words, by marrying learning and reasoning). It firstly employs neural learning (convolutional neural network or CNN) to parse the scene into a sequence of characteristics of the objects like shape and color (the structured representation of the scene). Thereafter, it uses a recurrent neural network (RNN) to semantically parse the question and generate a symbolic program (i.e., filter and query). Finally, it implements symbolic reasoning by running the program on the structured

representation, filtering the red and querying its shape to get an answer to the question.

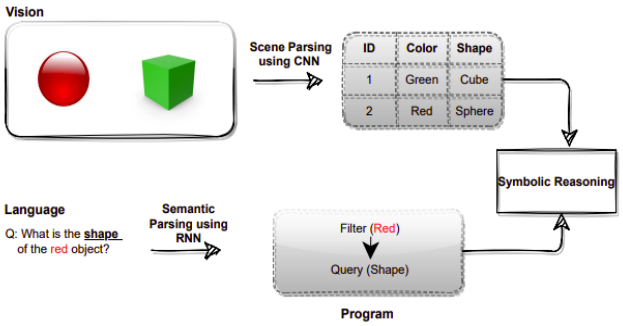

Fig. 2. Solving a visual question answering task using NSC.

IV. DISCUSSION AND FUTURE WORK

Machine learning plays an important role in educational technology through cognitive and non-cognitive student modeling. However, the application of machine learning-based models in education, possesses two main challenges, namely unavailability of enough training data and a lack of interpretability. Regarding the former, although many existing ML-based methods suggest using a big load of educational data for training their models, a mere data-driven model might not meet specific constraints, such as being dictated by learning principles or educational guidelines. The latter stresses the fact that a lack of interpretability and explainability could result in losing educators, practitioners, and learners' trust in decisions made by machine learning-based models. Borrowing ideas from other fields, we introduce the concept of neural-symbolic computing (NSC) to the educational context. NSC systems combine both the symbolic and neural models to relax the weakness of symbolic systems by neural learning and inference under uncertainty. They also bring about better reasoning and explanation to neural networks by solving their main issues, which are forgetting or having difficulties with extrapolation. Employing NSC has proven to relax both above-mentioned challenges, as prior knowledge can be translated and injected into the network so that it has better learning and reasoning with fewer data. On the other hand, extraction techniques can be employed to translate the existing knowledge in the network to interpretable and explainable rules. Such explainability would help educators, decision-makers, and learners understand the computation behind the decision and evaluate the system's outputs. Employing NSC in education would also facilitate providing knowledge description for some application areas that might have many heterogeneous data (e.g., when video and audio data are tagged with ontological metadata and multimodal data fusion for information retrieval).

NSC can be employed in the educational context to solve tasks, such as image and video processing, natural language processing, sequential tasks, etc. In terms of natural language processing tasks, such methods can be employed for improving procedural text comprehension or relationship extraction (e.g., [38]). Regarding video or image processing tasks, NSC approaches can be used for emotional understanding or explainable video action reasoning (e.g., [39]). Regarding sequential tasks, these methods can improve the existing (deep) knowledge tracing approach, as some educational principles can be fed to the model while monitoring and tracing learners' knowledge through sequential models such as RNNs.

Further, NSC can contribute to the educational data mining and learning analytics that often employ machine learning models for prediction of students' performance, achievement in online courses, procrastination, and failure risk in courses; for student modeling, that is employed in educational recommender systems and knowledge tracing; in open-learner models (OLMs) that provide information on learners' level of knowledge, motivation, and competencies (e.g., [40]), and many more. For instance, NSC has the potential to improve both accuracy of the educational prediction models and their interpretability by providing a separate source of domain knowledge to the network while being trained on the available training data and extracting hidden knowledge from the predictive models. In a similar vein, NSC can be employed in knowledge transfer learning—by extracting symbolic knowledge from a domain and transferring it to another domain—when there is a lack of domain knowledge [41].

NSC methods can also relax the long-standing issue of OLMs, namely their incapability to transparently explain their development process concerning the assessment mechanism by which learners' knowledge is evaluated. As reported by Hooshyar et al. [40], aside from opening learner models to feedback information on learners' knowledge and competency level, transparency or explainability could be educationally valuable. For instance, it would be pedagogically useful to transparently explain the means by which the model derives information and the assessment mechanism employed for inferring respective information in the domain model.

In conclusion, more research is required to provide a comprehensive framework for interpretable machine learning and AI. However, our proposed framework—which is based on the empirical findings gleaned from the work on NSC in other fields—has implications for addressing interpretability of AI models, and, thus, can be considered as a step toward interpretable machine learning and AI in education. As our future work, we plan to apply an NSC-based method to educational data and evaluate its effectiveness in the educational context.

REFERENCES

[1] L. Deng, G. Hinton, and B. Kingsbury, "New types of deep neural network learning for speech recognition and related applications: An overview," in *2013 IEEE international conference on acoustics, speech and signal processing*, 2013, pp. 8599–8603, DOI: 10.1109/ICASSP.2013.6639344.

[2] L. Hongtao and Z. Qinchuan, "Applications of deep convolutional neural network in computer vision," *Journal of Data Acquisition and Processing*, vol. 31, no. 1, pp. 1–17, 2016.

[3] S. Lee, D. Lee, D. Hooshyar, J. Jo, and H. Lim, "Integrating breakdown detection into dialogue systems to improve knowledge management: encoding temporal utterances with memory attention," *Information Technology and Management*, vol. 21, no. 1, pp. 51–59, 2020, DOI: 10.1007/s10799-019-00308-x.

[4] M. Kumar and Y. K. Salal, "Systematic review of predicting student's performance in academics," *Int. J. of Engineering and Advanced Technology*, vol. 8, no. 3, pp. 54–61, 2019.

[5] D. Hooshyar, M. Pedaste, and Y. Yang, "Mining Educational Data to Predict Students' Performance through Procrastination Behavior," *Entropy*, vol. 22, no. 1, pp. 1–24, Jan. 2020, DOI: 10.3390/e22010012.

[6] D. Hooshyar and Y. Yang, "Predicting Course Grade through Comprehensive Modelling of Students' Learning Behavioral Pattern," *Complexity*, vol. 2021, pp. 1–12, 2021, DOI: 10.1155/2021/7463631.

[7] C. Piech *et al.*, "Deep Knowledge Tracing," Jun. 2015, Available: http://arxiv.org/abs/1506.05908v1

[8] C. Conati, K. Porayska-Pomsta, and M. Mavrikis, "AI in Education needs interpretable machine learning: Lessons from Open Learner Modelling," Jun. 2018, *arXiv preprint arXiv:1807.00154*.

[9] H. Hagras, "Toward human-understandable, explainable AI," *Computer*, vol. 51, no. 9, pp. 28–36, 2018.

[10] A. S. d'Avila Garcez, K. B. Broda, and D. M. Gabbay, *Neural-symbolic learning systems: foundations and applications*. London: Springer London, 2002, DOI: 10.1007/978-1-4471-0211-3.

[11] A. d'Avila Garcez, M. Gori, L. C. Lamb, L. Serafini, M. Spranger, and S. N. Tran, "Neural-Symbolic Computing: An Effective Methodology for Principled Integration of Machine Learning and Reasoning," May 2019, Available: http://arxiv.org/abs/1905.06088v1

[12] M. Diligenti, S. Roychowdhury, and M. Gori, "Integrating prior knowledge into deep learning," in *2017 16th IEEE international conference on machine learning and applications (ICMLA)*, 2017, pp. 920–923, DOI: 10.1109/ICMLA.2017.00-37.

[13] A. Karpatne, W. Watkins, J. Read, and V. Kumar, "Physics-guided neural networks (PGNN): An application in lake temperature modeling," Oct. 2017. *arXiv preprint arXiv:1710.11431*.

[14] A. Hernández-Blanco, B. Herrera-Flores, D. Tomás, and B. Navarro-Colorado, "A systematic review of deep learning approaches to educational data mining," *Comp*lexity, vol. 2019, 2019, DOI: 10.1155/2019/1306039.

[15] C. R. Cook, S. P. Kilgus, and M. K. Burns, "Advancing the science and practice of precision education to enhance student outcomes," *Journal of School Psychology*, vol. 66, pp. 4–10, 2018. DOI: 10.1016/j.jsp.2017.11.004.

[16] O. H. Lu, A. Y. Huang, J. C. Huang, A. J. Lin, H. Ogata, and S. J. Yang, "Applying learning analytics for the early prediction of Students' academic performance in blended learning," *Journal of Educational Technology & Society*, vol. 21, no. 2, pp. 220–232, 2018.

[17] C. A. Bacos, "Machine learning and education in the human age: A review of emerging technologies," in *Science and Information Conference*, pp. 536–543, 2019, DOI: 10.1007/978-3-030-17798-0_43.

[18] M. I. Jordan and T. M. Mitchell, "Machine learning: Trends, perspectives, and prospects," *Science*, vol. 349, no. 6245, pp. 255–260, 2015, DOI: 10.1126/science.aaa8415.

[19] C. Conati and H. Maclaren, "Empirically building and evaluating a probabilistic model of user affect," *User Modeling and User-Adapted Interaction*, vol. 19, no. 3, pp. 267–303, 2009, DOI: 10.1007/s11257-009-9062-8.

[20] N. Bosch *et al.*, "Detecting student emotions in computer-enabled classrooms," in *the Twenty-Fifth International Joint Conference on Artificial Intelligence (IJCAI-16)*, 2016, pp. 4125–4129.

[21] H. Monkaresi, N. Bosch, R. A. Calvo, and S. K. D'Mello, "Automated detection of engagement using video-based estimation of facial expressions and heart rate," *IEEE Transactions on Affective Computing*, vol. 8, no. 1, pp. 15–28, 2016, DOI: 10.1109/TAFFC.2016.2515084.

[22] M. Mavrikis, "Modelling student interactions in intelligent learning environments: constructing bayesian networks from data," *International Journal on Artificial Intelligence Tools*, vol. 19, no. 06, pp. 733–753, 2010. DOI: 10.1142/S0218213010000406.

[23] J. Stamper, M. Eagle, T. Barnes, and M. Croy, "Experimental evaluation of automatic hint generation for a logic tutor," *International Journal of Artificial Intelligence in Education*, vol. 22, no. 1–2, pp. 3–17, 2013, DOI: 10.3233/JAI-130029.

[24] L. Fratamico, C. Conati, S. Kardan, and I. Roll, "Applying a framework for student modeling in exploratory learning environments: Comparing data representation granularity to handle environment complexity," *International Journal of Artificial Intelligence in Education*, vol. 27, no. 2, pp. 320–352, 2017, DOI: 10.1007/s40593-016-0131-y.

[25] Y. Yang, D. Hooshyar, M. Pedaste, M. Wang, Y.-M. Huang, and H. Lim, "Predicting course achievement of university students based on their procrastination behaviour on Moodle," *Soft Computing*, vol. 24, no. 24, pp. 18777–18793, 2020, DOI: 10.1007/s00500-020-05110-4.

[26] Y. Yang, D. Hooshyar, M. Pedaste, M. Wang, Y.-M. Huang, and H. Lim, "Prediction of students' procrastination behaviour through their submission behavioural pattern in online learning," *Journal of Ambient Intelligence and Humanized Computing*, pp. 1–18, 2020, DOI: 10.1007/s12652-020-02041-8.

[27] O. Sukhbaatar, T. Usagawa, and L. Choimaa, "An artificial neural network based early prediction of failure-prone students in blended learning course," *International Journal of Emerging Technologies in Learning (iJET)*, vol. 14, no. 19, pp. 77–92, 2019.

[28] V. Kostakos and M. Musolesi, "Avoiding pitfalls when using machine learning in HCI studies," *Interactions*, vol. 24, no. 4, pp. 34–37, 2017, DOI: 10.1145/3085556.

[29] T. R. Besold *et al.*, "Neural-Symbolic Learning and Reasoning: A Survey and Interpretation," Nov. 2017, Available: http://arxiv.org/abs/1711.03902v1

[30] S. N. Tran and A. S. d'Avila Garcez, "Deep logic networks: Inserting and extracting knowledge from deep belief networks," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 29, no. 2, pp. 246–258, 2016, DOI: 10.1109/TNNLS.2016.2603784.

[31] Z. Hu, X. Ma, Z. Liu, E. Hovy, and E. Xing, "Harnessing deep neural networks with logic rules," Aug. 2020, *arXiv preprint arXiv:1603.06318*.

[32] L. Serafini and A. S. d'Avila Garcez, "Learning and reasoning with logic tensor networks," in *Conference of the Italian Association for Artificial Intelligence*, 2016, pp. 334–348, DOI: 10.1007/978-3-319-49130-1 25.

[33] T. Li and V. Srikumar, "Augmenting neural networks with first-order logic," August 2020, *arXiv preprint arXiv:1906.06298*.

[34] S. Bader and P. Hitzler, "Dimensions of neural-symbolic integration-a structured survey," Nov. 2005, *arXiv preprint cs/0511042*.

[35] J. Townsend, T. Chaton, and J. M. Monteiro, "Extracting relational explanations from deep neural networks: A survey from a neural-symbolic perspective," *IEEE transactions on neural networks and learning systems*, vol. 31, no. 9, pp. 3456–3470, 2020, DOI: 10.1109/TNNLS.2019.2944672.

[36] L. von Rueden *et al.*, "Informed Machine Learning--A Taxonomy and Survey of Integrating Knowledge into Learning Systems," May 2021, *arXiv preprint arXiv:1903.12394*.

[37] M. Jiayuan *et al.*, "The neuro-symbolic concept learner: Interpreting scenes, words, and sentences from natural supervision," Apr. 2019, *arXiv preprint arXiv:1904.12584*.

[38] X. Du *et al.*, "Be consistent! improving procedural text comprehension using label consistency," Jun. 2019, *arXiv preprint arXiv:1906.08942*.

[39] T. Zhuo, Z. Cheng, P. Zhang, Y. Wong, and M. Kankanhalli, "Explainable Video Action Reasoning via Prior Knowledge and State Transitions," Aug. 2019, Available: http://arxiv.org/abs/1908.10700v1

[40] D. Hooshyar, M. Pedaste, K. Saks, Ä. Leijen, E. Bardone, and M. Wang, "Open learner models in supporting self-regulated learning in higher education: A systematic literature review," *Computers & Education*, vol. 154, Sep. 2020, DOI: 10.1016/j.compedu.2020.103878.

[41] D. L. Silver, "On Common Ground: Neural-Symbolic Integration and Lifelong Machine Learning," in *Proceedings of the 9th International Workshop on Neural-Symbolic Learning and Reasoning NeSy13*, 2013, pp. 41–46.

Danial Hooshyar received his Ph.D. degree from artificial intelligence department at University of Malaya, Malaysia, in 2016. He had worked as a research professor in department of computer science and engineering at Korea University for nearly four years. He is currently an associate professor of learning analytics. His research interest includes artificial intelligence in education, adaptive educational systems, and educational technology.

Yeongwook Yang received his master degree from computer science education, and Ph.D. from 2021-01011-Proof-reading-V2 the department of computer science and engineering at Korea University, Seoul, South Korea. He had worked as a research professor in the department of computer science and engineering at Korea University for one year. He was a Senior Researcher at University of Tartu, Tartu, Estonia. He is currently an assistant professor in the division of computer engineering at Hanshin University. His research interests include information filtering, recommendation system, educational data mining, and deep learning.

The Women in Computer Science Program in New Mexico Tech

Amy Knowles

I. Introduction

Although 56.86% of undergraduate enrollments are females [3], only 22.5% of bachelor's degrees are awarded to women [1]. The situation is worse in the STEM focused universities, such as New Mexico Tech. Historically, the female enrollment in the Department of Computer Science and Engineering has been less than 15%. Therefore, three alumni from New Mexico Tech decided to support the program called Women in Computer Science (WiCS), funded since 2017, to increase the number and percentage of females both entering and graduating from the Computer Science bachelor of Science program.

This article first describes the overview of the WiCS program in Section 2. Section 3 summarizes the work we did to change the introductory programming courses. Integrating game design in the computer science course in order to engage students' interests is illustrated in Section 4. Section 5 outlines the recruitment and retention plan we made for the program via a nonprofit organization. The future plan in the program is summarized in Section 6.

II. WiCS Program Overview

Numerous studies have described the need for integrating in courses a diverse set of applications to show possible careers in Computer Science and Information Technology. Careful analysis of best practices and evaluation of the exceptional programs visited will be used to implement changes to CSE 101 (Introduction to Computer Science and Information Technology, CSE 107 (Python Programming), CSE 113 (Introduction to Programming with C), and CSE 122 (Data Structures) courses with the intent to increase retention rates and use CSE 107 for recruiting. We have already added a python course for students who do not have the mathematical maturity to take programming in C, but will focus on promoting that course for students with limited programming experience.

The program has indicated that "boot camps" and summer "catch up" courses can be successful in bringing students up to speed for those who may, for various reasons, not be ready academically to initially be in the same classes as their peers or who fall behind. There are various solutions that will be assessed through visits to programs such as Harvey Mudd and through discussions with high school counselors. Based on these findings, a program will be designed using best practices. For example, we have scheduled the Program Lead to teach CSE 122 in the summer for students who are a semester behind; this would allow them to catch up in the summer after the first year.

While the department has had a mentoring program for a number of years, many students do not take advantage of the opportunity. In addition to assigning every new female student a mentor in our program, and a female advisor, we will build a database of mentors – including students, alumni, faculty, and others – that will facilitate matching each student's interests with a mentor. We will arrange mentoring events (to meet and get to know each other) and provide guides to mentoring and being mentored.

For each major component of this program, an assessment process will be developed along with the component. The assessment process will allow us to evaluate the effectiveness and perceived impact of the component. Ultimately, this will allow us to improve the operation and effectiveness of the component or if it is not having a positive impact, to eliminate components or replace them with more effective approaches.

III. Changes to Introductory Programming Course

Two female faculty in the program visited Harvey Mudd College (HMC) on October 29, 2019. There are two pathways to create the foundational knowledge in programming for both CS majors and non-majors at HMC: the first pathway consists of CS5 (Introduction to Computer Science), CS60 (Principles of Computer Science) and CS70 (Data Structures and Program Development); the second pathway consists of CS42 (Principles and Practice of Computer Science) and CS70 (Data Structures and Program Development). The CS5 course is separated into black and gold sections – one for students who come into computer science with experience and the other for students without any experience.

The two sections (Black and Gold) are offered at the same time to allow students to switch between the two until a certain point in the semester. This allows students progressing quickly in Gold to switch to Black and those who feel they were not as prepared as they thought to switch from Black to Gold. They chose Python as the introductory language because it is relatively simple to have beginner students creating something they think is really cool due to the large number of libraries and packages available in Python. Students in the Gold section of the course will progress at a slightly slower pace than students in the Black course, but students in both sections will complete the course knowing the basics required for CS60. Students in the Black section of the course may delve a little deeper into a more diverse range of applications across the computer science discipline, but will not be ahead of the Gold course or gain more experience/knowledge toward CS60 or CS70.

A compromise was reached with the New Mexico Tech faculty on how to implement the Harvey Mudd model at New Mexico Tech. The first programming introductory core course – CSE 113 – was split into two sections starting in Fall 2020, but it was still taught in C, as opposed to Python. One instructor taught the CSE 113 Silver section for inexperienced coders and another taught the CSE 113 Blue section for students who had coding experience before attending New Mexico Tech. Despite limitations due to Covid-19, both sections made active use of pair programming.

Over the summer, the Program Lead tested online tools to facilitate distance pair programming. Some of the tools used to do this were recommended during an online meeting with HMC faculty. The primary tools for the online pair programming activities were Visual Studio Code with MS LiveShare, which allowed students to share the same coding environment despite not being in close physical

Received December 30, 2021, Accepted January 2, 2022, Publish online January 12, 2022.

A. Knowles is with the Department of Computer Science and Engineering, New Mexico Institute of Mining and Technology, Socorro, NM 87801 USA (e-mail: amy.knowles@nmt.edu).

proximity. Discord was used for screen share and discussion during labs so that TAs could jump between groups and aid students when needed. Significant issues that were dealt with in the HMC model include: (a) Placement Test – a placement test was given to all enrolled students at the beginning of the semester. Recommendations for which section students were likely to be most successful in were based on the student performance on the test. Students who were not enrolled in the recommended course were told they should change to the recommended section; (b) Course Switch – unlike at Harvey Mudd, all but three students in the two semesters took the placement exam and followed recommendations as to which section (Blue-experienced or Silver-novice) would be the best fit; (c) Overall Student Performance Statistics for CSE 113 – data was gathered over 6 semesters to show pass/fail rates for our introductory coding course. Passing is defined as performance of a C or better and Failing is defined as performance below passing.

IV. Integrating Game Design in CS Courses

The Program Lead developed and taught a Video Game Design course that counts as a technical elective for current CS students and as a Fine Arts credit for non-CS students. This course is designed as a recruitment tool to generate more diverse interest in the CS major at New Mexico Tech. New Mexico recently made changes in the core curriculum for all NM college students that would require 3 credit hours in Fine Arts. Since all students starting with the 2019/20 freshman will be required to take 3 credit hours in fine arts, we believe that this Video Game Design course will act as a great recruitment tool to generate more diverse interest in CS as a possible major and/or as a minor. Note that students could take this course twice, once as a fine arts course meeting the arts requirement and once as a computer science technical elective. The requirements for the two courses are substantially different, but the students in the two sections work together on projects, which gives those in the fine arts section exposure to coding and computer game development.

Despite the setbacks of COVID-19 on the Spring-2020 semester, each of the 10 teams in the "Elements of Game Design: Programming" and "Elements of Game Design: Art" courses successfully completed much of their planned games. Teams were made up of four students - two programmers and two art students. Art students were responsible for designing assets, music, and sound for the games, and the programming students were responsible for bringing the designs to life and implementing game mechanics. This course is being offered a second time in the Spring semester of 2022 with an additional instructor who will be teaching the Art and Assets section of the course.

Besides the Video Game Design course, the Game Jam – an activity that brought together the student community to work over a weekend to make video games – has been used during the past few semesters to interest a diverse group of students in pursuing computer science. In 2019, the WiCS program host the Game Jam with the Cybersecurity Education Center in New Mexico Tech. We also have outreach underway that involves video game development with the Cybersecurity Education Center.

The outcome of the 2019 cybersecurity game outreach was presented in a webinar hosted by Asia-Pacific Society for Computers in Education (APSCE, https://apsce.net/) in December 2020. The 2020/2021 Cyber Game Jam kicked off in October of 2020 and was fully digital through Zoom due to COVID-19. Participants met bi-weekly via Zoom to discuss problems and updates, and demonstrate the current progress of their games. The winning game, which has been purchased by the Cybersecurity Centers and will be used for outreach.

V. NCWIT Learning Circle Funding

The National Center for Women & Information Technology (NCWIT) is a nonprofit organization founded in 2004 which works on increasing participation of females in computing. The Extension Service Learning Circles of NCWIT assists departments in developing and implementing strategies to increase female enrollments. The WiCS program joined the Learning Circle in 2020. After 12 meetings with two other departments of computing, an Extension Service consultant, and an Extension Service staff member, the WiCS program received the 10,000 Gift Fund.

We plan to use our 10,000 Gift Fund to develop content for outreach and to bridge the knowledge gap for transfer students and incoming freshmen. Through outreach, we plan to inspire local female and underrepresented minority students not currently majoring in computer science through online exposure experiences. Once a post-COVID-19 normal has been established, we also plan to have face-to-face outreach experiences as well.

For the online exposure experience, we have a large quantity of Galileo boards (Intel Arduino boards). We plan to build kits including these Galileo units that can be mailed to workshop participants. A team of 3 undergraduate computer science students will develop the curriculum utilizing the Galileo boards. By creating fun and engaging content centered around active learning, we hope to build interest in computer science as a major and as a career. As we do have a limited number of Galileo units, we plan to shift this online experience to utilize a Galileo simulator/emulator for future workshops and events.

For in-person events, we would like to utilize Orzobots, small robots that utilize a light sensor to follow a line or path. These little robots are very flexible in their range of appropriate content across a wide range of ages. We plan to have a team of 3 undergraduate students create content for these Orzobots that can be used across a wide variety of outreach events and workshops. We first came across these little robots during a trip to New Mexico State University to learn more about their Young Women in Computer Science (YWiCS) program.

In addition to the outreach content development, we plan to utilize our gift fund to build transition content for our introductory computer science courses. We teach our introductory courses in ANSI C, a language known for its hellish memory management. Although this allows us to produce high-quality computer scientists upon graduation, it is not necessarily conducive to recruitment or retention.

In New Mexico, only 44% of all public high schools teach a foundational computer science course [2]. When taught, the language for the course is generally an object-oriented language such as Java or Python. Only 371 exams were taken in AP Computer Science by high school students in New Mexico in 2020 (113 took AP CS A and 258 took AP CSP) [2]. There are fewer AP exams taken in computer science than in any other STEM subject area. This knowledge gap is one reason our introductory course retention hovers around fifty percent. We plan to hire 4 undergraduate computer science students to focus on creating high-quality videos and associated documents that would aid in bridging the content gap between incoming freshman and transfer students and the introductory courses at New Mexico Tech. We plan to make this content available on the website and our computer science community discord server. Once created, these videos and instructional materials would be maintained by current course TAs and instructors.

VI. Next Steps

While we do think we are beginning to make significant progress, we have much more work to do. Here are a few of the next steps we are pursuing. Only 44% of New Mexico public schools teach computer science courses. The Randolph Program in New Mexico Tech is focusing on how to inspire New Mexican middle and high school teachers using innovative teaching methods to teach computer science in classrooms. The WiCS program will work with the Randolph Program to ensure seamless integration between WiCS educational outreach and that of the Randolph Program. On the other hand, the WiCS program formed connections with the local junior high and high school through the Socorro Educational Mentoring Alliance – a collaboration between New Mexico Tech, the Socorro Consolidated School District, and Mentoring Kids Works. We are looking forward to providing computer science related activities to local school students ranging in topics from cybersecurity to video games.

References

[1] American Society for Engineering Education (2020). *Engineering and Engineering Technology by the Numbers 2019*. Washington, DC. [Online]. Available: https://ira.asee.org/wp-content/uploads/2021/02/Engineering-by-the-Numbers-FINAL-2021.pdf

[2] Code.org. (2021). *Support K-12 Computer Science Education in New Mexico*. [Online]. Available: https://code.org/advocacy/state-facts/NM.pdf

[3] E. Duffin (2021, May 4). *Undergraduate enrollment numbers in the United States from 1970 to 2029, by gender*. [Online]. Available: https://www.statista.com/statistics/236360/undergraduate-enrollment-in-us-by-gender/

Amy Knowles is a Computer Science instructor who teaches beginning to advanced topics for undergraduate students at New Mexico Tech(NMT). In addition to their teaching responsibilities, Amy is the Women in Computer Science (WiCS@nmt) Coordinator and is dedicated to encouraging young women and underrepresented minorities to pursue careers in computer science. Amy believes that computer science is more than just coding and analytics as it requires quite a bit of creativity and problem solving.

Comparative Analysis of Object Visualization Tools with Respect to Their Use in Education

Brandon Dennis and Ramyaa Ramyaa

***Abstract*— There are many visualization softwares for Object-Oriented Programming (OOP) in the literature. Each has different capabilities that allow them to visualize different aspects of OOP. However, to the author's knowledge, there is no survey of such visualization tools. Such a survey might help in comparing and contrasting the different tools across various dimensions. Further, a survey may uncover some crucial aspect of OOP that is not currently visualized, but could benefit from visualization.**

In this work, we survey visualization tools for OOP with respect to their use as an education tool. Specifically, we compare and contrast "JaguarCode", "jGRASP", "Javalina Code", "JIVE", "Python Tutor", "BlueJ", "Jeilot3", and "PandionJ". We consider the following dimensions of comparison: Detailed Graphics, Object Class Graphics, Single Instance View, Data Structure View, Additional Diagram Generation, and usability in introducing OOP concepts.

We found that while the existing technologies for object visualization are sufficient for most applications in education, there is a lack of tools that target introducing OOP. In particular, the tools lack features designed specifically to educate students on fundamental ideas of OOP such as object design.

***Index Terms*— computer science education, object-oriented programming, software, visualization**

I. Introduction

Object Oriented Programming (OOP) can be a difficult concept to master. Learning what an object is and how to design object classes can be particularly difficult. Visualization tools can help with this. A graphical representation of objects and what they encapsulate can help them understand how objects can be built and used.

There are many visualization softwares for Object-Oriented Programming (OOP) in the literature. Students may be confused at many different levels and so, different visualization tools may choose to focus on different levels of detail. Graphics such as Class Diagrams can be good for aiding in object design, whereas more complex visuals like sequence diagrams can assist in a student's understanding of the order of events occurring in a program. Visualizing individual objects can be helpful for understanding how data in an object can change over time while a tool allows for the visualization of a complex data structure may be more desirable for a user who needs to see how objects interact with each other.

However, to the author's knowledge, there is no survey of such visualization tools in an educational context. Such a survey comparing and contrasting existing tools could highlight a gap in current visualization software. Here, we provide such a survey focusing on using visualization tools in OOP education. Specifically, we compare and contrast "Jaguar Code", "jGRASP", "Javalina Code", "JIVE", "Python Tutor", "BlueJ", "Jeilot3", and "PandionJ". We found that object visualization is most commonly done in a format similar to the UML Class Diagrams, but there are several approaches used by different modeling tools. Some give a surface level view of objects, while others can be used to model large-scale data structures. Though not all of these tools were specifically designed for education, many of the authors of these tools had intended education to be one of the tool's uses, and had reported experimental results of the tools' use in an educational setting. We consider the following dimensions of comparison: Detailed Graphics, Object Class Graphics, Single Instance View, Data, Structure, View, Additional, Diagram, Generation and usability in introducing OOP concepts.

We conclude that, while the tools had varying degrees of performance across the dimensions of our analysis, they all miss concepts that could make them more effective as educational tools. For instance, they all lacked a focus on foundational principles of OOP such as object design and the difficulty students have transitioning from procedural-style languages to an Object-Oriented approach. Instead, these tools were intended for users who already have a fundamental knowledge of objects and how to design them. Adding more features designed to educate students on fundamental ideas of OOP like object design would be a step in this direction

Received November 18, 2021, Accepted December 22, 2021, Publish online January 12, 2022.

Both authors are affiliated with the New Mexico Institute of Mining and Technology, Socorro, NM 87801 USA (e-mail: brandon.dennis@student.nmt.edu and ramyaa.ramyaa@nmt.edu).

II. Overview of Existing Technology

While all of the tools covered in this analysis are used to visualize object-oriented code in some form, the level of detail and capabilities of each tool are not the same across the board. Some tools are simplistic in their graphics and only aim to give the user a base-level view of data, while others allow the user to view full scale data structures in detail. This section gives a brief introduction of each tool.

A. JaguarCode

JaguarCode ([1]) is a web-based programming tool designed for dynamically visualizing Java code. JaguarCode generates static diagrams for classes, as well as dynamic views of each instance of an object used in a program being ran in the tool. Additionally, the tool visualizes relationships between associated objects such as inheritance or implementation as shown in Fig. 1.

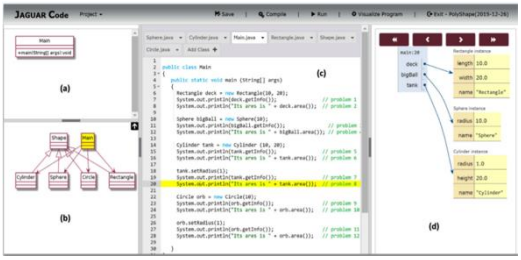

Fig. 1. An annotated screenshot of JaguarCode taken from [1]. The left-hand side depicts the generated Class Diagrams, the middle section is for user-given code, and the right-hand side is used to visualize running code.

B. *jGRASP IDE*

The jGRASP IDE ([2]) is a Java programming tool with a wide range of visualization capabilities. It is mainly focused on visualizing multi-object data structures (such as linked lists shown in Fig. 2.) rather than individual classes, but there are some UML visualization capabilities.

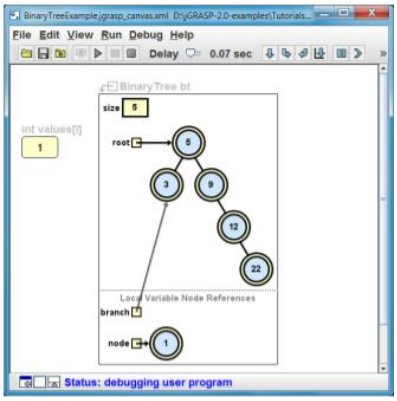

Fig. 2. A view of a binary tree generated by the jGRASP IDE taken from [2].

C. *JavalinaCode*

JavalinaCode ([3]) is an online Java visualization tool. It uses a cloud service to allow users to create Java projects, compile code, and visualize programs. The tool generates Class Diagrams for each class in the project and table-like graphics for each object used in a running program much like the JaguarCode example in Fig. 1.

D. *JIVE*

The Java Interactive Visualization Environment (or JIVE) [4] is a plugin for the Eclipse IDE designed to dynamically visualize Java code. Unlike some of the other tools presented so far, JIVE does not offer Class Diagrams as a form of static visualization. Instead, the tool generates Contour Diagrams and dynamic, table-like graphics for objects used in the running program (shown in Fig. 3.). JIVE also offers Sequence Diagram generation.

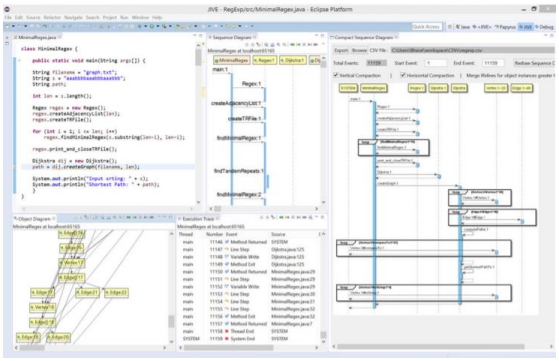

Fig. 3. Screenshot of the JIVE user interface taken from [4]

E. *Online Python Tutor*

Online Python Tutor ([5]) is a web-based programming tool intended for both students and instructors that allow the user to step through Python code with the aid of visualizations. The tool allows users to step both backwards and forwards step-by-step through a program while providing a graphic representation of the execution stack, global variables, and local variables. Complex objects are given their own graphic representations depicting the current value of their attributes such as the graphics in Fig. 4.

Fig. 4. Graphics generated by the Online Python Tutor taken from [5].

F. *BlueJ*

BlueJ ([6]) is a simplistic Java visualization tool that allows users to interact with objects graphically. The visual representation of objects generated by BlueJ are lacking in detail and offer no description of a class's methods or attributes, which can be seen in Fig. 5. They are, however, capable of visualizing inheritance relationships and individual instances of an object.

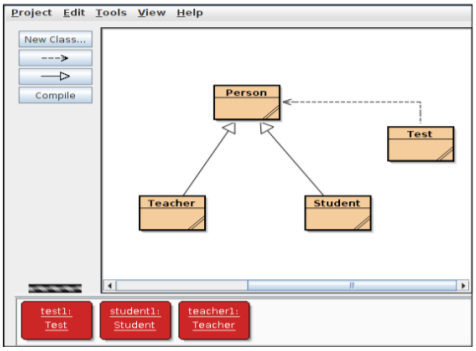

Fig. 5. Visualization generated by the BlueJ tool taken from [6].

G. *Jeliot 3*

Jeliot 3 ([7]) is an evolution of the Jeliot visualization tool for Java. This tool allows the user to view and interact with the graphics it generates alongside the written code. Similar to JIVE, Jeliot 3 focuses more on visualizing programs as a whole rather than individual classes. The graphics generated by the tool include graphics similar to a contour diagram with separate graphics for object variables (see Fig. 6. for details).

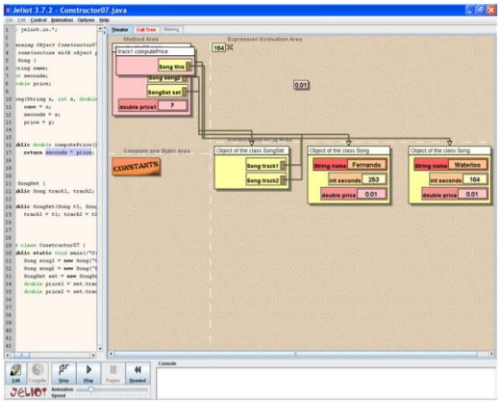

Fig. 6. Screenshot of the Jeliot 3 visualization tool taken from [7].

H. *PandionJ*

PandionJ ([8]) is an Eclipse plugin designed to visualize Java code. The plugin allows users to see objects and function stacks visualized alongside the running code. PandionJ also allows for step-by-step code execution and debugging.

JaguarCode, jGRASP, JavalinaCode, Online Python Tutor, BlueJ, Jeliot 3, and PandionJ were tested in educational settings and the results were detailed in [1] [2] [3] [5] [6] [7] [8] respectively.

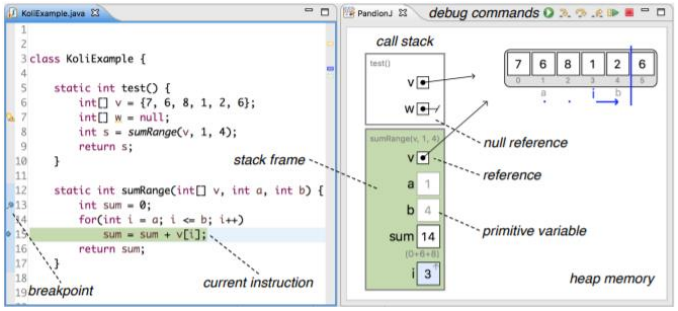

Fig. 7. Screenshot of the PandionJ plugin taken from [8].

III. Tool Comparison

We first give brief overview of the comparison criteria. The functionalities of these tools being compared in this paper can be grouped into 6 categories: visualization detail, object class graphics, individual instance view, data structure view, additional diagram generation, and usability in introducing OOP. A reference of what criteria a given tool offers can be found in Table 1.

A. Visualization Detail

The amount of detail provided by a visualization tool can greatly impact usability. Some of the tools covered in the previous section give graphics that allow the user to view the names of individual instances generated by the program, or even the data types of an object's attributes. This information can greatly benefit the user, especially when using multiple objects of the same class in a single program.

B. Object Class Graphics

A key feature of these visualization tools is the ability to produce graphics depicting the objects being used in a program and the data being stored inside them. Some tools, however, offer a broader view of the data as well in the form of Class Diagrams (or their own equivalent form). These diagrams allow the user to view the characteristics of an entire class of objects rather than just the data in an object instance. This information can be helpful when designing objects since diagrams like these can often serve to condense large amounts of code into a more digestible format.

C. Individual Instance View

The ability of a tool to generate a visualization of individual instances being used in a program is arguably what classifies a program as a visualization tool. Each tool being reviewed is capable of this functionality, although with varying levels of detail provided by each.

D. Data Structure View

On top of being able to visualize individual objects, some visualization software allows users to generate graphical representations of complex data structures like binary trees or linked lists. This can be an especially useful tool when creating programs that utilize data structures like these since visuals like this can help the user determine if each structure is functioning correctly in the large scale.

E. Additional Diagrams Generated

While being able to generate graphics for a single snapshot of a program's runtime is certainly useful, sometimes additional data is needed or desired. This is where visual aids such as Sequence Diagrams can be useful. A Sequence Diagram is a single graphic that can describe the interactions between objects over time. Typically, these are generated manually, but some visualization tools give the user the ability to generate these views along with the traditional graphics.

F. Usability in introducing OOP

In order for a tool to be useful in introducing OOP to students, it must offer details about the fundamental elements of OOP. For example, teaching students about proper object design is a vital part of OOP education. Another helpful feature would be a method of aiding a student's transition from procedural programming to an Object-Oriented approach. Features like these would provide a helpful teaching aid to students just beginning their programming education.

TABLE 1
Tool Comparison Overview

	Jaguar Code	jGRASP	Javalina Code	JIVE	Python Tutor	BlueJ	Jeliot 3	PandionJ
Detailed Graphics	X	X	X	X	X	X	X	X
Object Class Graphics	X		X				X	
Single Instance View	X	X	X	X	X	X	X	X
Data Structure View		X						X
Additional Diagram Generation	X			X				
Usability in introducing OOP								

This table serves as a brief reference of what comparison criteria are and are not offered by each tool covered in this paper.

IV. Tool Analysis

A. JaguarCode

JaguarCode's dynamic visualizations are displayed in the right-hand pane of the window and include a visual representation of the function stack, as well as names of the variables contained in each frame. If necessary, JaguarCode also generates a graphic for objects in each frame with an arrow back to the frame they are used in as displayed in Fig. 1. While JaguarCode does not have the ability to visualize large scale data structures, it can generate sequence diagrams for user given code.

JaguarCode is a good visualization tool for students who already have some programming knowledge but struggle with understanding objects as a concept. The tool offers some debugging tools along with dynamic visualization for each stage of the code, which can be very useful in understanding how objects can change over the course of a program's execution. The static Class Diagrams also help the user understand the content of a class file, as well as gains some knowledge of how an object can be used. The addition of a sequence diagram generation feature can also be useful for more advanced users who plan on using JaguarCode for more sophisticated projects.

B. jGRASP IDE

The jGRASP IDE is a great tool for visualizing multi-object data structures. It allows users to visualize commonly used data structures such as linked lists, hash tables, and binary trees. jGRASP also common debugging features like user given breakpoints and the ability to step through code. What jGRASP offers in data structure visualization, however, it lacks in general object visualization. There are no static visualizations for object types, though the names and values of instance attributes are given in the generated graphics.

While it may not be a particularly useful tool for learning how to design classes, the jGRASP IDE is a great tool for any application that requires tracking data in a large-scale data structure. It may also be useful for students who are learning about these types of structures and could use a visual aid for comprehension.

C. JavalinaCode

JavalinaCode is fundamentally similar to both JaguarCode and the Online Python Tutor in the sense that all of them visualize the

function stack and point to individual instances of objects. In further similarity to JaguarCode, JavalinaCode also gives static visualizations for object classes along with visual representations of inheritance and other object relationships. Unlike JaguarCode, however, JavalinaCode lacks the ability to generate sequence diagrams over the course of program execution.

D. JIVE

The JIVE visualization tool may be the most detailed visualization tool covered in this paper in terms of the level of detail given for an individual object. JIVE offers users a table view of each object where attribute names, values, and data types are given. The fact that this information is given is especially important given that JIVE does not give users a static Class Diagram view of objects. Instead, the tool focuses on giving users as much information about an object as possible directly in the dynamic view. JIVE also gives users the option of generating Sequence Diagrams, which can also be useful for understanding interactions between objects.

E. Online Python Tutor

The Online Python Tutor acts as a basis for tools like JaguarCode and JavalinaCode. While Online Python Tutor may not offer as many features these tools, it was published well before either of them and likely served as a basis for both given how similar their visualization styles are to Online Python Tutor. And, while this tool may not be as feature heavy as others, it does still offer a fairly streamlined and convenient view of data. That, along with the convenience of being an web-based tool and the ability to execute a program step-by-step make the Online Python Tutor a good education tool.

F. BlueJ

BlueJ is a bare bones approach to object visualization. Its figures offer little detail (seen in Fig. 5.), and no general object class graphics are given. BlueJ does, however, name object instances and represent inheritance graphically, which is useful information to have visualized. As the oldest tool covered by this paper though, BlueJ is a solid foundation for the field of object visualization to build on.

G. Jeliot 3

Jeliot 3 is another tool that allows a user to step through execution while giving dynamic visualizations of variables and objects in their code. While no static diagrams are generated by the tool, Jeliot 3 does give a description of data types in its graphics. Another interesting feature of Jeliot 3 is the ability for users to interact with the visualizations it generates. For example, Fig. 6. shows a user moving objects around in the data view.

H. PandionJ

PandionJ offers a similar experience to that of the Online Python Tutor with a few added features. Like the Online Python Tutor, PandionJ offers detailed visualizations of single object instances, but does not offer general class descriptions. PandionJ does also give the user visual indication of null references (a node cut off by a slash) and differentiates between objects and primitive data types (primitive data type attributes have desaturated variable value boxes).

V. Discussion and Conclusion

Visualization software fills a gap in OOP education left by traditional instruction alone. These tools allow students to experiment in a safe environment at their own pace, and visualization allows students who are less familiar with code to gain a better understanding of their programs. However, there are a few areas where current visualization tools could improve.

The tools do not focus on introducing OOP concepts. To be effective at this level in OOP education, a tool must have functionalities that address more fundamental principles of OOP. Many of the tools covered in this article, for example, seem to be designed for users who already know how to design objects. A foundation in OOP is almost a pre-requisite for using these kinds of tools, which doesn't help students just learning how to program in an Object-Oriented space. A potential solution to this issue is to offer a series of guided tutorials that teaches the user some of the more fundamental concepts of OOP like object design, the difference between private and public attributes or methods, etc. Giving a student these sorts of instructional guides before allowing them to interact in the tool freely may help students build a strong foundation in OOP.

Another area that current visualization software is lacking is the level of detail offered by their graphics. While a majority of the tools covered in this paper give some way of identifying object instances, many of them do not give this information in a convenient location. For example, both JaguarCode and the Online Python Tutor name object instances in their visualization of the stack but not in the graphics generated for the objects themselves. Another detail many of these tools do not offer in their graphics is the data type of an object's attributes. The tools that offer static Class Diagram generation have data types listed, but this level of detail is dropped in the dynamic visualization. Offering instance names and attribute data types all in one place provides the user with a greater level of detail in a more convenient location. This improvement can help with usability, especially for a novice in OOP.

The last area we believe visualization software can improve is by giving users who have programming experience but are new to OOP a way to transition between the two styles of coding. The transition between procedural programming and OOP can be jarring, and many students struggle with adapting to the idea of objects that can contain their own data and methods. While some procedural languages have mechanisms that can be considered similar to objects (such as structs in C), shifting your thinking to fit with a near-fully Object-Oriented environment like Java can be difficult. Visualization software currently doesn't do anything to aid in this transition. A possible way to accomplish this is to generate a C-like pseudocode representation of user given code along with more traditional visualizations.

VI. References

[1] J. Yang, Y. Lee and K. H. Chang, "Evaluations of JaguarCode: A web-based object-oriented programming," *Journal of Systems and Software*, no. 145, pp. 147-163, 2018.

[2] J. H. Cross II, T. D. Hendrix, L. A. Barowski and D. A. Umphress, "Dynamic program visualizations: an experience report," in *Proceedings of the 45th ACM technical symposium on Computer science education*, Atlanta, 2014.

[3] J.-s. Yang, *JavelinaCode: A Web-Based Object-Oriented Programming Environment with Static and Dynamic Visualization*, Auburn, Alabama, 2016.

[4] S. Jayaraman, B. Jayaraman and L. Demian, "Compact visualization of Java program execution," *Software: Practice and Experience*, vol. 47, no. 2, pp. 163-191, 2017.

[5] P. Guo, "Ten Million Users and Ten Years Later: Python Tutor's Design Guidelines for Building Scalable and Sustainable Research Software in Academia," in *The 34th Annual ACM Symposium on User Interface Software and Technology*, 2021.

[6] B. Y. Alkazemi and G. M. Grami, "Utilizing BlueJ to teach polymorphism in an advanced object-oriented programming course," *Journal of Information Technology Education*, vol. 11, pp. 271-281, 2012.

[7] M. Ben-Ari, R. Bednarik, R. B.-B. Levy, G. Ebel, A. Moreno, N. Myller and E. Sutinen, "A decade of research and development on program animation: The Jeliot experience," *Journal of Visual Languages & Computing*, vol. 22, no. 5, pp. 375-384, 2011.

[8] A. L. Santos, "Enhancing Visualizations in Pedagogical Debuggers by Leveraging on Code Analysis," in *Proceedings of the 18th Koli Calling International Conference on Computing Education Research*, Koli, 2018.

Brandon S. Dennis was born in 2000 in Socorro, New Mexico, USA. He is an undergraduate student at the New Mexico Institute of Mining and Technology in Socorro, New Mexico and is currently pursuing a Bachelors of Computer Science with an accelerated Masters program.
His current field of interest is computer visualization for use in education.

Ramyaa Ramyaa is an assistant professor of Computer Science at New Mexico Tech. She did her PhD at Indiana University, USA, Post-doctoral research at Ludwig Maximilian University, Munich and has masters degrees from Carnegie Mellon University, USA and University of Georgia, USA. She was a fellow at Simons institute of Theory, UC Berkeley, USA.
Her primary fields of research are Theory of Computation (and complexity) and Logic, focusing on implicit complexity (relating logical complexity of concepts to computational, resource-based complexity). Her secondary research area is Artificial Intelligence, focusing on machine learning.

Computer-Assisted Language Instruction Consortium (CALICO 2022) (May 31st to June 4th, 2022)

Randall William Sadler

I. Introduction

The Annual Symposium of the Computer-Assisted Language Instruction Consortium (CALICO) will take place from May 31st to June 4th, 2022 in Seattle, Washington in the Seattle Renaissance Hotel, which is only a 10-minute walk from the main waterfront of the city. The annual conference has gathered scholars from around the world for five days of discussion on all topics related to technology and language learning since 1984 (Baltimore, Maryland) at locations across the United States and Canada. Recent conferences include Virtual (2021), Montreal (2019), University of Illinois (2018), Northern Arizona University (2017), Michigan State University (2016), and University of Colorado, Boulder (2015). CALICO 2022 will take place face-to-face, with provisions for a select number of online presentations on the last day of the conference. We invite you to attend and meet your colleagues!

II. CALCO Presentation Types, Meetings, and Social Events

CALICO includes a number of session types, including *Workshops* (half day, full day, or two day) that take place before and after the main presentation days. These workshops are hands-on and must be registered for ahead of time. During the main days of the conference the presentation types include *30-Minute Presentations* that can include those theory-driven focusing on a range from quantitative/qualitative studies of language technology, application of technology for classroom use, or any of a range of language technology uses/applications for language acquisition; *75-minute Panels* offering multiple perspectives on a topic; *Technology Showcases* where the presenters demonstrate new and interesting technologies; and *Poster Presentations*.

In addition to the presentations, each CALICO symposium also includes special interest group meetings focused on a variety of topics: Computer-Mediated Communication, Gaming, Grad Student SIGN, Language Teaching and Learning Technologies, Second Language Acquisition and Technology, Teacher Education, and Immersive Realities. These SIG meetings are an excellent opportunity to engage with fellow teacher scholars who share your interests.

CALICO always host multiple social events during the time of the conference, with the event types varying depending on the site of the event. In recent years, these events have ranged from live music, to bowling, to dancing, shared meals, tourist trips, and more. We are a very collegial group and welcome attendees creating their own social gatherings as well!

III. Registration

CALICO 2022 registration fees include an option for early registration:

Early Registration (before May 1st)

- Presenters: $225 US
- Members not presenting: $275 US (this rate applies also to IALLT and EUROCALL members)
- Nonmembers: $345

Registration after May 1st

- All prices increase by $25 US

IV. Important Dates

- May 1st, 2022: Deadline for early registration
- May 9th, 2022: Deadline for group rate at conference hotel
- May 31st-June 4th: CALICO, Seattle, WA

V. Important Websites

- CALICO 2022 Website: https://calico.org/calico-conference-2022/
- Hotel Information: https://calico.org/calico-conference/lodging-calico-conference/
- Airport Information: https://www.portseattle.org/sea-tac
- Visit Seattle (tourist info): https://www.portseattle.org/sea-tac

Seattle is a wonderful site for tourism, and the Puget Sound, Olympic Peninsula, and Washington coast are all very easily accessible from Seattle.

Received November 11, 2021, Accepted November 14, 2021, Publish online November 16, 2021.

R. W. Sadler is with University of Illinois at Urbana-Champaign, Illinois, USA (e-mail: rsadler@illinois.edu).

The 1st International Conference on Future Language Learning July 1st – 3rd, 2022

Yanjie Song

I. Overview

Language learning mediated by technologies is evolving rapidly over the past three decades with the fast development of technologies: from Computer-Assisted Language Learning (CALL), to Mobile Computer-Assisted Language Learning (MALL), and now to Intelligent Computer-Assisted Language Learning (ICALL). In recent years, this research area from a transdisciplinary perspective has attracted attentions from more and more academics and researchers from different fields, including cognitive neural scientists, linguistic researchers, pedagogical designers, theorists of language learning as well as practitioners in various language learning arenas in the real world. This changing landscape of research by cross-disciplinary researchers implies a collective endeavor to achieve a goal: a momentous advancement of language learning in the future!

II. ICFULL 2022

The 1st ICFULL 2022 International Conference on Future Language Learning (ICFULL) 2022, organised by the Asia-Pacific Society for Computers in Education (APSCE), aims to involve scholars, practitioners and researchers to share and exchange their research findings in exploring the potential that cutting-edge technologies can provide for language pedagogical practices in terms of AI-supported language learning, personalized / adaptive learning, digital game-based learning, technology-supported self-directed learning, learning analytics in language learning, assessment with emerging technologies, interdisciplinary studies, etc.; looking into emerging language learning theories, pedagogies, research methods and professional development models; and addressing ethical issues caused by new technologies in language education in an attempt to shed light on future language learning. We invite you to attend the conference and get together with friends in the promising research field!

III. ICFULL Presentation Types

The conference program includes keynotes, paper presentations and panelists. All the accepted papers in the conference will be published in ISBN-coded proceedings. The 1st ICFULL 2022 will be held online.

Received January 4, 2022, Accepted January 7, 2022, Publish online January 12, 2022.

Y. Song is with the Education University of Hong Kong, Hong Kong, PRC. (e-mail: ysong@eduhk.hk).

IV. Registration

ICFULL 2022 registration fees:

Early Bird Registration (on or before 11 April)
- APSCE member: USD 92 / HKD 720
- Non-member: USD 120 / HKD 940
- Student APSCE member: USD 44 / HKD 350

Normal Registration (after 11 April)
- APSCE member: USD 115 / HKD 900
- Non-member: USD 150 / HKD 1170
- Student APSCE member: USD 55 / HKD 430

V. Important Dates

- Conference period: 1st July – 3rd July 2022 (Friday – Sunday)
- Paper submission due: **28 February 2022**
- Paper review due: **28 March 2022**
- Camera-ready paper submission: **11 April 2022**
- Early-bird registration deadline: **18 April 2022**
- Author registration deadline: **6 June 2022**

VI. Important Websites

- ICFULL 2022 Website: https://www.eduhk.hk/ICFULL2022/
- APSCE Website: https://www.apsce.net/

VII. Contacts

- *For questions regarding paper submission*:
  Please contact the 1st ICFULL 2022 at icfull2022@eduhk.hk
- *For academic related matters*:
  Please contact the Programme Chair
  Yanjie Song
  Department of Mathematics and Information Technology, The Education University of Hong Kong
  Email: ysong@eduhk.hk
  Weblink:
  https://pappl.eduhk.hk/rich/web/person.xhtml?pid=142850&name=SONG-Yanjie

The 5th Pedagogy and Practice in Technology-Enhanced Language Learning (July 1st to 3rd, 2022)

Hongliang Ma, Liwei Hsu*

I. Overview

The expeditious development and progress of technologies have made the use of various kinds of technologies in educational praxis the norm and language education is no exception. Particularly when the COVID-19 has changed our lifestyles drastically and such an unprecedented change also facilitates the applications of technologies in teachers' pedagogy and practice. More empirical evidence as well as practical examples on technology-enhanced language learning are in great need and hence call for further explorations. Taiwan Pedagogy and Practice in Technology-Enhanced Language Learning Association (PPTELL Association) was founded in July 2020 with Prof. Yu-Ju Lan as the first Chairman and Prof. Meei-Ling Liaw as the first Vice-Chairman. With the aim to promote the collaboration between researchers and practitioners in the fields of Technology-Enhanced Language Learning (TELL) and Computer-Assisted Language Learning (CALL), PPTELL is keen to make contributions to advance our current knowledge on the role technologies play in language education.

II. Previous PPTELL Conferences

In past years, four PPTELL conferences have already been held even before the Association was founded. The National Taiwan Normal University (NTNU) hosted the first and the second PPTELL Conference in 2018 and 2019 on its Heping campus and Linkou campus respectively. In the year of 2020, PPTELL conference was jointly hosted by the NTNU and the University of North Texas (UNT), U.S.A. Particular gratitude must go to the financial support of the Ministry of Science and Technology as well as the Office of Research and Development of the NTNU as these conferences were held successfully and fruitfully. The global situation of COVID-19 pandemic was still devastating in 2021, which made PPTELL 2021 be held online again. Even so, more attendees and participants joined the conference to share their recent research in the concurrent sessions and also socialized on the virtual platform of Second Life during break time.

The themes of each PPTELL conference focus mainly on understanding the issues of striving for the better implementation of TELL, including what to do and how to do it; equally important, the underlying reasons why it should be done. Since 2018, the PPTELL conferences have convened researchers and the front-line practitioners to have thorough discussions from the vantage points of theory as well as practice on the potentials of integrating language learning and advanced technologies for providing language learners with better learning outcomes. Various appealing yet important topics have been covered in the past PPTELL conferences, including smart learning environments, AI, robotic technology, augmented/virtual reality, educational analytics/big data, mobile computing, and educational game-based learning. It is worth noting that PPTELL 2020 and 2021 conferences were held amid the COVID-19 pandemic and hence they both were conducted solely online. With the joint efforts of many researchers and scholars, the aforementioned two PPTELLs were a breakthrough when international travel was still unrealistic. These two online conferences received remarkable success and so will be the following PPTELL 2022, which will take place online again. The details are listed in the following section.

III. PPTELL 2022

The 5th Pedagogy and Practice in Technology Enhanced Language Learning (PPTELL 2022) will be held as an online conference by Shaanxi Normal University and the PPTELL Association, still with a possibility of face-to-face conference if the pandemic comes to an end. The conference sincerely invites extended abstract submission within 500 words that shows originality based on theoretical or practical works focusing on global connectivity in technology-enhanced language learning with the topics related to the theme- "Developing learner agency through smart technology." This year will focus on using smart technology to help language learners become autonomous learning agents.

The PPTELL 2022 conference features two types of presentations: oral presentation and poster presentation. Prior to submission, please read and follow the conference templates (PDF version / MS Word version) carefully. Submissions must not be published previously. All the submissions will be double-blind peer-reviewed. Below are important dates of PPTELL 2022:

- Deadline for Abstract Submission: January 31, 2022
- Notification of Abstract Acceptance: February 28, 2022
- Deadline for Early Registration: March 20, 2022
- Conference Dates: July 1 ~ 3, 2022

Please visit https://conference.pptell.org/index.php for detailed information. For questions regarding this submission, please contact PPTELL 2022 at pptell2022@gmail.com.

We look forward to seeing you at PPTELL 2022.

Received November 21, 2021, Accepted November 27, 2021, Publish online December 13, 2021.

H. Ma is with the Shaanxi Normal University, Xi'an, Shaanxi, China (e-mail: mahl@snnu.edu.cn).

L. Hsu is with National Kaohsiung University of Hospitality and Tourism, Hsiao-Kang District, Kaohsiung City, Taiwan (e-mail: liweihsu@mail.nkuht.edu.tw).

ITS2022 -18th International Conference on Intelligent Tutoring Systems (June 27-July 1, 2022)

Stefan Trausan-Matu, Mihai Dascalu

I. Introduction

ITS2022 (https://iis-international.org/its2022-bucharest-romania/) is the upcoming Conference of the series of Intelligent Tutoring Systems (ITS) Conferences on Computer and Cognitive Sciences, Artificial Intelligence and Deep Learning in Tutoring the Education. It will be held in Bucharest, Romania, in June/July 2022 and will be hosted by University Politehnica of Bucharest. The theme of the conference is "NEW CHALLENGES FOR ITS DURING AND AFTER COVID".

The Intelligent Tutoring Systems conference attracts researchers and academics, the industry and the end-users since 1988. It started out as an enthusiastic project by Professor Claude Frasson and in 2018 it celebrated its 30th anniversary at its birthplace in Montreal.

The ITS conference series enables, supports and enhances human learning, through an interdisciplinary research approach on Intelligent Systems. By linking together the study of the use of advanced computer technologies and various other themes, such as education, tutoring and etc, the ITS conference series builds the foundation for the evaluation of the use of Intelligent Systems in education, modeling innovative applications of technologies and the adaptation of systems to specific groups of learners.

We want to emphasize the strong link we have established with another prestigious conference, the 22th IEEE International Conference on Advanced Learning Technologies (ICALT2022; https://tc.computer.org/tclt/icalt-2022/ organized in the same venue, with one overlapping day (July 1, 2022).

II. Topics of Interest

The topics of interest of the ITS2022 Conference include, but are not limited to:

- 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), 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 Visualisation 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 Modelling 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 Modelling and Constraints-based Modelling in Intelligent Tutoring

The proceedings of the conferences are published with Springer's Lecture Notes in Computer Science.

III. Important Dates

- January 15, 2022: Submission deadline for all papers (Full paper, Short paper, Posters, Doctoral Consortium, Workshop Proposals, Tutorial Proposals, Industry Track Proposals)
- February 28, 2022: Acceptance notification
- March 31, 2022: Final version submission
- June 27-28, 2022: Workshops and Tutorials
- June 29 - July 1, 2022: ITS 2022 Conference

Received December 23, 2021, Accepted December 27, 2021, Publish online January 12, 2022.

S. Trausan-Matu is Professor in the Computer Science Department of the University Politehnica of Bucharest, Romania, Senior researcher at the Research Institute for Artificial Intelligence, and Full Member of the Romanian Scientists Academy (stefan.trausan@upb.ro)

M. Dascalu is Professor in the Computer Science Department of the University Politehnica of Bucharest, Romania and Corresponding Member of the Romanian Scientists Academy (mihai.dascalu@upb.ro)

The 22th IEEE International Conference on Advanced Learning Technologies (July 1-4, 2022)

Mihai Dascalu, Stefan Trausan-Matu

I. Introduction

The International Conference on Advanced Learning Technologies (ICALT, https://tc.computer.org/tclt/icalt-2022/) is an annual conference organized by IEEE Computer Society and IEEE Technical Committee on Learning Technology. It aims to bring together people who are working on the design, development, use and evaluation of technologies that will be the foundation of the next generation of e-learning systems and technology enhanced learning environments. After its kick-off as IWALT in Palmerston North, New Zealand (2000), ICALT has been held in Madison, USA (2001), Kazan, Russia (2002), Athens, Greece (2003), Joensuu, Finland (2004), Kaohsiung, Taiwan (2005), Kerkade, The Netherlands (2006), Niigata, Japan (2007), Santander, Spain (2008), Riga, Latvia (2009), Sousse, Tunisia (2010), Athens, Georgia, USA (2011), Rome, Italy (2012), Beijing, China (2013), Athens, Greece (2014), Hualien, Taiwan (2015), Austin, USA (2016), Timisoara, Romania (2017), Mumbai, India (2018), Maceió, Brazil (2019), and online in 2020 and 2021 due to COVID-19 pandemic. This year (ICALT 2022) will be organized and conducted as a hybrid event at University Politehnica of Bucharest, Romania.

ICALT conference is ranked as the second (2nd) by saliency in Educational Technology area in Microsoft Academic for the time period of 2000 to 2020, see https://tc.computer.org/tclt/ieee-icalt-conference-ranked-second-2nd-saliency/ for more details.

In addition, we want to emphasize the strong link we have established with another prestigious conference, the 18th International Conference on Intelligent Tutoring Systems (ITS2022; https://iis-international.org/its2022-bucharest-romania/) organized in the same venue, with one overlapping day. Both ITS and ICALT participants can attend two keynotes, one from ITS's last day and one from ICALT's first day, for free.

Moreover, please consider joining any of the social networking groups (e.g., Discord, LINE, Telegram, see https://tc.computer.org/tclt/social-media/) to ensure that you could always get all updates and information that IEEE Technical Committee on Learning Technology has. On Discord (https://discord.gg/wdhQrMa), TCLT has created three channels to announce various webinars, panels, events, talks, keynotes, podcasts, etc. opportunities for both of students and professionals.

Received December 30, 2021, Accepted January 2, 2022, Publish online January 12, 2022.

M. Dascalu is Professor in the Computer Science Department of the University Politehnica of Bucharest, Romania and Corresponding Member of the Romanian Scientists Academy (mihai.dascalu@upb.ro)

S. Trausan-Matu is Professor in the Computer Science Department of the University Politehnica of Bucharest, Romania, Senior researcher at the Research Institute for Artificial Intelligence, and Full Member of the Romanian Scientists Academy (stefan.trausan@upb.ro)

II. Theme-based Tracks

The ICALT conference this year features the following tracks on various thematic topics, including: Track 1. Technologies for Open Learning and Education (i-OPENLearn); Track 2. Adaptive and Personalised Technology-Enhanced Learning (APTeL); Track 3. Mobile Applications of Learning Technologies for Education and Development (MALT); Track 4. Digital Game and Intelligent Toy Enhanced Learning (DIGITEL); Track 5. Computer Supported Collaborative Learning (CSCL); Track 6. Big Data in Education and Learning Analytics (BDELA); Track 7. Technology-Enhanced Science, Technology, Engineering and Math Education (TeSTEM); Track 8. Technology-Enhanced Language Learning (TELL); Track 10. Technology-supported education for people with disabilities (TeDISABLE); Track 11. Artificial Intelligence and Smart Learning Environments (AISLE); Track 12. Augmented Reality and Virtual Worlds in Education and Training (ARVWET); Track 13. Motivational and Affective Aspects in Technology-enhanced Learning (MA-TEL); Track 14. Technology-Enhanced Technology-enhanced Assessment in Formal and Informal Education (TeASSESS); Track 15. Internet of Everything (IoE) for Smart Education (IoESE). This year IEEE Special Technical Community (STC) on Internet of Everything joinly organizing Track 15 on Internet of Everything (IoE) for Smart Education (IoESE). IEEE Special Technical Community (STC) on Internet of Everything can also be joined freely. See https://tc.computer.org/tclt/icalt-2022-track-15-ioese/ for more details.

III. Incentives

The high-quality accepted papers will be invited to submit the extended version of the paper to International Journal of Distance Education Technologies (https://igi-global.com/ijdet included in Web of Science's ESCI, SCOPUS, EI and other indexes), Bulletin of Technical Committee on Learning Technology (https://tc.computer.org/tclt/bulletin/, open access and included in ESCI), and Special Issue of Sustainability (https://www.mdpi.com/journal/sustainability/special_issues/sustainability_edu, open access with article processing fee and included in SSCI).

IV. Important Dates

- January 14, 2022 (Friday): Submission deadline for all papers (Full paper, Short paper, Discussion paper)
- April 1, 2022 (Friday): Authors' Notification on the review process results
- May 6, 2022 (Friday): Author's registration deadline
- May 6, 2022 (Friday): Final Camera-Ready Manuscript and IEEE Copyright Form submission
- May 20, 2022 (Friday): Non-authors' early bird registration deadline
- July 1-4, 2022 (Friday to Monday): ICALT 2022 Conference