Addressing challenges of constructionist modeling of adaptive systems

Uwe Lorenz
uwe.lorenz@fu-berlin.de
Freie Universität Berlin
Berlin, Germany

Ralf Romeike
ralf.romeike@fu-berlin.de
Freie Universität Berlin
Berlin, Germany

ABSTRACT

How should computer-based educational tools represent Machine Learning (ML) systems for didactic purposes? We address this question using constructionist learning theory and the intelligent agent paradigm of AI. ML in this context is understood as generating and improving "goal-directed" system behaviors by iteratively maximizing a "goal function".

We give a theoretical outline of the problem domain along the questions: How independent can ML concepts be from concepts of classical computer science (CS)? What are central concepts and processes that ML possesses? What are important properties of structural models of this kind of systems conducive to comprehension? Finally, we propose some design features of educational informatics tools for teaching ML and outline further research needs.

CCS CONCEPTS

• **Social and professional topics** → **Computing education**; • **Computing methodologies** → **Machine learning**; **Modeling and simulation**; *Artificial intelligence*.

KEYWORDS

machine learning, artificial intelligence, neural networks, teaching practise, CS education, constructionism

ACM Reference Format:
Uwe Lorenz and Ralf Romeike. 2022. Addressing challenges of constructionist modeling of adaptive systems. In *Proceedings of the 17th Workshop in Primary and Secondary Computing Education (WiPSCE '22), October 31-November 2, 2022, Morschach, Switzerland*. ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3556787.3556870

1 INTRODUCTION

In constructionist learning theory [6] new knowledge is build in an active construction process. It emphasizes the importance of creative and productive understanding and comprehension as well as mutual presentation and explanation. This means that learners and teachers in any domains should be empowered to prepare, adapt and create interactive learning materials themselves [10]. For this, appropriate educational tools are needed. These "Educational tools for learners should reduce entrance barriers as far as possible ('low floors'), should not restrict their interests and creativity ('wide walls') and at the same time allow the (step-by-step) creation of complex, sophisticated projects ('high ceilings')" [11]. They should also provide instructive insights behind the scenes [8].

This task raises some questions: What design features should educational informatics tools for teaching ML have? What are important properties of structural models conducive to comprehension? What are central processes and concepts that ML possesses? How independent are these from "classical CS" concepts?

In the following we provide a theoretical outline of the problem domain and related research gaps. For this we will address the mentioned questions starting with the last one. Finally, we make some proposals on the first question.

2 THEORETICAL OUTLINE

Even though most ML systems are currently running on (increasingly parallelized) classical digital computing hardware, that doesn't mean they are rooted in the CS tradition [12]. For example "Artificial Neural Networks" follow an alternative information processing paradigm that is inspired from biological nervous systems [4]. The universality of von Neumann or Turing hardware, allows to build and study virtual worlds that follow own principles. Also it is possible to implement models that come from other sciences such as philosophy and psychology, even though concepts that are rather foreign to computer scientists such as "interaction" or "behavior" may play an important role here.

A very fundamental concept of ML is the idea of iterative optimizing behavior [9] by reducing errors or maximizing rewards [13]. A criticism by cyberneticists in past AI debates was that formalist principles of representation and symbol processing are not central for that and have been improperly transferred to models of cognitive systems [1]. We want to follow this idea that ML is about automatically generating and improving useful system behaviors, like behaviours for detection or prediction tasks (supervised), agent controls (reinforced) or those that minimize the "generalization cost" of a model intended to represent a large data set (unsupervised). Bringing the different types of ML systems into one conceptual structure by understanding them all as behaviors of systems interacting more or less usefully with their environment provides didactic benefits such as supporting sustained understanding.

Among the central processes of classical CS, "problem solving and problem posing" is found in first place [15]. ML is used for solving problems, too. But the problem solving process looks fundamental different. In [14], some conceptual shifts are pointed out, e.g. "Implement the solution in a stepwise program." ("CT 1.0" problem solving stage 3) to "Train a model from the available data" ("CT

WiPSCE '22, October 31-November 2, 2022, Morschach, Switzerland

ACM ISBN 978-1-4503-9853-4/22/10.
https://doi.org/10.1145/3556787.3556870

2.0"). In [12], Langley is cited as having observed that ML is largely an empirical science, and similar epistemological approaches are taken here as in physics or chemistry with central processes such as "observing".

For prototyping complex behaviors in robotics visual data-flow programming is common used [2]. Similar properties show the representations of useful tools created in the last years like the "MemBrain" simulator, "Orange3" or the "google Tensorflow Playground" which allow handling and inspect ML systems without classical programming knowledge.

3 UNDERSTANDING ML-SYSTEMS BY CREATING: 5 DESIGN FEATURES

Based on the aforementioned works and considerations, some hypotheses on design features of educational tools for ML teaching are formulated in the following, which also result from a critical observation of the use of block-based implemented prototypes (language "Snap!") on the topic "How do neural networks learn?" [1] by student teachers in a didactic seminar and two high school students of grade 12 (1 male and 1 female). In this Snap! based ML systems, we tried to make functions, such as learning rules and activation functions, etc., transparent and accessible to the learner. In addition, we wanted to provide insightful opportunities for modification. But the block-based representation was not very well suited for that. For example things that happen independently in parallel, such as passing signals or modifying weights, are represented sequentially and contiguously. While functional dependencies conveyed via global variables remain invisible. Therefore, we propose to investigate the following design features for educational tools in terms of whether they are conducive to learning about ML systems:

DF1: "Dataflow perspective": We think of an ML system as a combination (where appropriate nesting,too) of partially parallel and sequential behaviors of components that ingest and forward data. "Control flow focuses on allowing operations to 'fire' only when an 'execution token' (program counter) arrives at their locations in the program. Data flow focuses on allowing operations to 'fire' when all their input data arrive."[14].

DF2: "Direct interaction": for the modification of activation functions or learning rules, the students using our program first tried to interact directly on the Snap! stage with the components (neurons) of the ML system. This is indicative of a more intuitive design. Such direct interaction possibilities with the computational nodes and their connections, combined with an immediate view of and intuitive ways to inspect system states and processes, could support comprehension according to experiential learning [7].

DF3: "Storytelling/contextualization": In our scenario for learning the Delta Rule, e.g. we used a story that included the topic of the "ignition point of flammable materials". This seems to have an activating effect and helped to better understand the application domains of the related ML approach ('supervised learning' in this case).

DF4: "Scaffolding": We could observe, that e.g. the provision of required blocks simplified the problem for learners significantly.

DF5: "Interdisciplinary combination of models/simulations, representations and methods": If methods of different disciplines must be used, e.g. for the construction of appropriate models and for checking correctness an instructive, critical, reflective comparison is possible not only of the kind of the descriptions and predictions, but also in terms of properties such as effort, reliability or ethical concerns.

[1] bit.ly/3bRrwhJ; https://bit.ly/3RjjoXP (in german)

4 DISCUSSION AND FURTHER WORK

We will apply the research framework of didactic reconstruction [3] [5] to structure our further research. It considers the "underlying perspectives", the "preparation of learning content" and the "design and implementation" of the desired learning environment. In terms of clarifying and empirical determination of central concepts and processes of ML there seems to be a gap in research at present. To contribute here we will conduct an empirical survey among CSE and ML experts. Existing educational tools like the mentioned above have different advantages and disadvantages with regard to their use for our educational purposes. This field needs to be further analysed, a prototype that implements our proposals needs to developed and evaluated. Furthermore, also with regard to teaching ML to learners without a CS background, we will elicit meaningful interdisciplinary application examples that can inspire instructive subject-specific creations of ML systems.

REFERENCES

[1] Rodney A. Brooks. 1991. Intelligence without Reason. In *Proceedings of the 12th International Joint Conference on Artificial Intelligence - Volume 1* (Sydney, New South Wales, Australia) *(IJCAI'91)*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 569–595.

[2] Sebastian Buck, Richard Hanten, C. Robert Pech, and Andreas Zell. 2017. Synchronous Dataflow and Visual Programming for Prototyping Robotic Algorithms. In *Intelligent Autonomous Systems 14*, Weidong Chen, Koh Hosoda, Emanuele Menegatti, Masahiro Shimizu, and Hesheng Wang (Eds.). Vol. 531. Springer International Publishing, 911–923.

[3] Ira Diethelm, Peter Hubwieser, and Robert Klaus. 2012. Students, teachers and phenomena: educational reconstruction for computer science education. In *Proceedings of the 12th Koli Calling International Conference on Computing Education Research - Koli Calling '12* (Koli, Finland, 2012). ACM Press, 164–173.

[4] J. Feldman and R. Rojas. 2013. *Neural Networks: A Systematic Introduction.* Springer Berlin Heidelberg.

[5] Andreas Grillenberger, Mareen Przybylla, and Ralf Romeike. 2016. Bringing CS Innovations to the Classroom Using the Model of Educational Reconstruction. In *International Conference on Informatics in Schools. ISSEP 2016. M"unster, Germany. Proceedings* (2016), Andrej Brodnik and Franccoise Tort (Eds.). 31–39.

[6] Idit Harel and Seymour Papert. 1991. Constructionism: Research Reports and Essays, 1985-1990.

[7] D.A. Kolb. 1984. *Experiential learning: experience as the source of learning and development.* Prentice Hall, Englewood Cliffs, NJ.

[8] Tilman Michaeli, Stefan Seegerer, Sven Jatzlau, and Ralf Romeike. 2020. Looking Beyond Supervised Classification and Image Recognition – Unsupervised Learning with Snap!. In *Constructionism 2020: Exploring, Testing and Extending our Understanding of Constructionism conference proceedings.*

[9] Tom M. Mitchell. 1997. *Machine Learning.* McGraw-Hill.

[10] Mareen Przybylla and Ralf Romeike. 2018. Empowering learners with tools in CS education: Physical computing in secondary schools. 60, 2 (2018), 91–101.

[11] Mitchel Resnick and Brian Silverman. 2005. Some Reflections on Designing Construction Kits for Kids. In *Proceedings of the 2005 Conference on Interaction Design and Children* (Boulder, Colorado) *(IDC '05)*. Association for Computing Machinery, New York, NY, USA, 117–122.

[12] R. Benjamin Shapiro, Rebecca Fiebrink, and Peter Norvig. 2018. How machine learning impacts the undergraduate computing curriculum. 61, 11 (2018), 27–29.

[13] David Silver, Satinder Singh, Doina Precup, and Richard S. Sutton. 2021. Reward is enough. 299 (2021), 103535.

[14] Matti Tedre, Peter Denning, and Tapani Toivonen. 2021. CT 2.0. In *21st Koli Calling International Conference on Computing Education Research* (Joensuu Finland, 2021-11-18). ACM, 1–8.

[15] A. Zendler, C. Spannagel, and D. Klaudt. 2008. Process as content in computer science education: empirical determination of central processes. 18, 4 (2008), 231–245.