CONSTRUCTIONISM 2020

The University of Dublin

Trinity College Dublin

IRELAND

"*26-29 May*"

Constructionism 2020

26th to 29th of May 2020 in Dublin, Ireland

Proceedings of the 2020 Constructionism Conference

Edited by Brendan Tangney, Jake Byrne and Carina Girvan

ISBN 978-1-911566-09-0

Constructionism and AI: A history and possible futures

Ken Kahn, *toontalk@gmail.com,*
Dept of Education, University of Oxford, Oxford, UK

Niall Winters, *niall.winters@education.ox.ac.uk,*
Dept of Education, University of Oxford, Oxford, UK

Abstract

Constructionism, long before it had a name, was intimately tied to the field of Artificial Intelligence. Very soon after the birth of Logo, Seymour Papert set up the Logo Group as part of the MIT AI Lab. Logo was based upon Lisp, the first prominent AI programming language. Many early Logo activities involved natural language processing, robotics, artificial game players, and generating poetry, art, and music.

In the 1970s researchers explored enhancements to Logo to support AI programming by children. In the 1980s the Prolog community, inspired by the Logo community, began exploring how to adapt logic programming for use by school children. While there has been over forty years of active AI research in creating intelligent tutoring systems, there was little AI-flavoured constructionism after the 1980s until about 2017 when suddenly a great deal of activity started.

Among those activities were attempts to enhance Snap! with new blocks for speech synthesis, speech recognition, image recognition, use of pre-trained deep learning models, and word embeddings, as well as blocks to enable learners to create and train deep neural networks.

We close with speculations about possible futures for AI and constructionism.

Keywords

Artificial intelligence, constructionism, deep neural networks, machine learning, Snap!, logic programming, Logo

AI was there from the beginning

The core constructionism idea is that learning "happens especially felicitously in a context where the learner is consciously engaged in constructing a public entity" [Papert & Harel 1991]. To "consciously engage" one needs to bring to bear concepts while introspecting. Computational concepts are an important well-known part of this within constructionism but here we focus on the learner's ideas about how they learn, represent things, solve problems, and create. AI-oriented constructionist projects may help learners acquire deeper and more effective ways of learning and creating.

These ideas influenced the design of the Logo language [Solomon et al 2020] and early project ideas. The paper "Twenty Things to Do with a Computer" [Papert and Solomon 1971] suggests on the first page that artificial intelligence should deeply affect our thinking about "computers in education". Several of the twenty project suggestions were AI projects (as it was conceived in the 1970s) such as programming robots with sensors and effectors, composing music, and generating poetry. These were soon followed by projects such as generating grammatical sentences and programming AIs to play simple games like *Tic Tac Toe* and *Nim*.

In "Three Interactions between AI and Education" [Kahn 1977] AI plays three roles: (1) students use AI programming tools in their projects, (2) students create artefacts by interacting with AI programs, and (3) and the use of computational theories of intelligence and learning in the design of learning activities. In the 1970s research prototypes were built upon Logo to support student projects involving natural languages and semantic networks. LLogo [Goldstein et al 1975], an implementation of Logo in Lisp, was particularly well-suited for this.

[Kahn 1977] lists a few reasons why children should create and interact with AI programs:

1. "Children are encouraged to think explicitly about how they solve problems. Hopefully the children will thereby improve their ability to describe and understand their own thoughts"
2. "The problem domain to which the AI programs are applied is learned, and in a new and perhaps better way"
3. "If children are to program, then AI can an interesting open ended problem domain for that programming"
4. "The children will learn about AI which is a subject … that is as important as spelling or history".

Regarding the first point consider the role self-reflection plays in learning according to Marvin Minsky [Minsky 2019; Minsky 2009] (original emphasis):

- An adequate theory of learning should also cover the "reflective skills" that people use to recognize exceptions to generalizations, to eliminate tactics that waste too much time, and more generally, to make longer-range plans and form broader perspectives.
- But I'm convinced that these "self-reflective" processes [the methods that people use for *thinking about what we've been thinking about*] are the principal ones that people use for *developing new ways to think*.

Efforts in the 1980s to bring AI programming into schools

The articles in the book *New Horizons in Educational Computing* [Yazdani 1984] in the Ellis Horwood Series on Artificial Intelligence describe many efforts to make AI programming tools accessible to children. The logic programming language Prolog inspired two very different approaches: (1) providing children with a simple syntax for using a powerful deductive engine and (2) giving children a powerful symbolic programming language that provides support for building semantic networks and doing backtracking searches. Prolog programs have declarative and procedural readings and different research groups focussed on each of these different aspects.

"Logic as a computer language for children" [Kowalski 1984] describes the declarative approach. Students can build deductive, typically symbolic, databases. Ten-year-old children constructed semantic networks using a front end to Prolog with a very simple syntax. [Sergot 1984] instead

built a meta-interpreter for Prolog for use by children. While much slower, it was able to provide explanations for any deductions it made. [Nichol and Dean 1984] describes how students could use Prolog to explore history.

An example of using Prolog as a powerful procedural tool is described in [Kahn 1984]. A grammar kit was built upon LM-Prolog that enabled students to build grammars to both parse and generate sentences. It relied upon a "backtracking turtle" that erased its trail when backtracking. This visualisation helped students understand how their programs and queries worked and to fix bugs.

POP-11 [Sloman 1984] is an AI language similar to Lisp that supports semantic databases. Development of POP-11 started in 1974 and was heavily influenced by the Logo efforts at MIT and the University of Edinburgh. The goals of Pop-11 were to enable students to, instead of "making toy cranes or toy aeroplanes, or dolls", to make "toy minds". The POP-11 research team wanted "people to experience themselves as computation". Accessible AI programming tools were developed with these goals in mind.

POP-11 later incorporated Prolog into their system (renaming it Poplog) after seeing the successes others were having introducing Prolog to children. From 1983 to 1998 Poplog was an expensive commercial product but after it became free in 1998 it was once again used by school students for AI programming. Examples of Poplog projects include puzzle solving, chatbots, analogical reasoning, and planning [Sloman 2012].

Everything changes in 2017

The AI research community, the general public, and educators view of AI began changing in 2012 as neural networks began attaining impressive performance in image recognition, machine translation, transcription of speech, game playing (Atari games, Go, and chess), and natural language processing. Deep neural networks are what most people think of today when they hear AI. Deep learning combined with big data and special hardware for doing tensor operations now dominates the field.

2017 saw the appearance of several excellent machine learning tools and resources for children. They include

1. April 2017 Dale Lane created *The Machine Learning for Kids* website. Learners can create neural networks for images, text, or numbers, train them, and use them in Scratch programs. The site includes a long list of well-designed project suggestions [Lane 2020].
2. May 2017 Stephen Wolfram added a machine learning chapter to the *Mathematica* online programming textbook. He published a blog describing the AI capabilities of the Wolfram Language. He suggests a range of projects suitable for middle school students [Wolfram 2017].
3. May 2017 Google announced its first AIY project where students can build standalone artefacts capable of speech synthesis and recognition. This was followed by a similar kit for image recognition [Google 2020].
4. September 2017 Kahn and Winters released the eCraft2Learn project's Snap! blocks that access web services for speech synthesis, speech recognition, and image recognition [Kahn & Winters 2017]. This was followed by Snap! blocks for using pre-trained TensorFlow models, and word embeddings, [Kahn & Winters 2018]. Subsequently blocks for creating, training, and using neural nets were added a year later [Kahn et al 2020].
5. October 2017 Google released its Teachable Machine which is a web page where one can train a model to classify three kinds of images [Google 2017]. Unlike the other 2017 machine learning tools this one does not have a programmer interface. While it is a good introduction to an aspect of machine learning it lacks the open-ended nature of most constructionist tools. It did, however, directly inspire similar projects in Snap! using the eCraft2Learn blocks.

This explosion of activity has continued. AI programming by children projects have since started at many universities. The AI for K-12 Initiative is growing and has organised two international workshops [Touretzky 2020].

A closer look at one AI programming for children project

eCraft2learn was a European research project from 2016 to 2018. It focussed on bringing ideas and technology from the Maker Movement to STEAM education. As part of the project the University of Oxford developed Snap! blocks for AI programming. They have continued to develop the blocks, guides, project suggestions, and sample projects after the eCraft2Learn project finished. They recently added blocks for defining neural networks, training them, and using them for predication and classification [Kahn et al 2020].

eCraft2Learn partners in Greece and Finland field tested the blocks and learning resources. Workshops using the blocks have been run in Sri Lanka, Singapore, and Indonesia. Collaborators at the Beijing Normal University have begun efforts to test these resources with both school children and non-CS major university students.

A unique aspect of this set of programming tools and resources is that it is completely web-based and largely server-free. There is no need to register or log in. By relying upon Google's TensorFlow.js [Smilkov 2019] the machine learning occurs in the user's device (laptop, tablet, or phone). It runs at competitive speeds to other technologies since it can accelerate computations by using the device's graphical processing unit (GPU). Nothing needs to be installed. No data needs to be transmitted to servers thereby enhancing privacy. It can run solely upon the local file system when Internet connections are not available or reliable. The Snap! blocks that use pre-trained neural networks and those for creating and training models all rely upon TensorFlow.js to perform all computations locally. Apps using the blocks can be very responsive since they avoid any latency due to communication overhead with servers.

A response to a criticism

[Ames 2018] is a critique of constructionism. She apparently rejects good-old fashioned symbolic AI and a computational view of the mind. She seems to view learning as primarily a social activity. Consequently, it is not surprising she is so critical of constructionism. Most relevant to this paper she wrote "Papert described cybernetics rapturously in Mindstorms as a potent framework for understanding learning by thinking about brains like we think about computers. This was not unusual: at the time, there was widespread belief that human brains were particularly sophisticated computers; Minsky, in fact, famously called them 'meat machines.' While the problems with this equivalence eventually contributed to the collapse of cybernetics and to the 'artificial intelligence winter' during which the field stagnated for decades, fragments of the belief persisted."

One can disagree with this view of AI history (and we do). The important question regarding the value of introducing AI programming to children is whether there really are "problems with this equivalence". Indeed, if one rejects the idea that thoughts are computations and brains are computational machines then one might question the value of helping children to construct AI programs. If thoughts are something other than computations (contrary to most research in cognitive science and neuroscience), then students may acquire a false mechanical concept of their thought processes. But what models of thinking might children acquire instead? Papert considered the alternative to be harmful "Pop-Ed theories" [Papert 1971]. These are the theories that children do acquire about thinking. One is the "blank-mind" theory that what one should do is make your mind a blank and wait for an idea to come. Another Papert named "getting-it" where one expects understanding to come in a flash, the opposite of the Piagetian process of accommodation. Another common theory that interferes with learning Papert called "faculty". For example, some people have the faculty of mathematics and some don't. Failure is because the problem is too hard to address given one's innate faculties. Our hope is that by engaging in AI programming students might acquire models of their thinking and learning that, while crude, can help them become better thinkers and learners.

Possible Futures

The early history of AI and constructionism was focussed on symbolic AI ("good-old-fashion AI"). More recently the focus has shifted greatly towards machine learning with neural networks. This is good since it provides students with very powerful capabilities enabling them to creatively build very capable artefacts. Students now have the tools to build apps that have the potential to make a positive difference in their community or the entire world. [Tissenbaum, Sheldon & Abelson 2019]. Students can acquire a first-hand experience with a technology that is rapidly changing the world. It is a technology they can use to become scientists, engineers, business executives, artists, musicians, doctors, or simply engaged citizens.

But something has been lost. Aaron Sloman wrote of students making "toy minds". Minsky and Papert spoke of acquiring concepts to make one better at reflection and learning. But with neural nets students are now making "toy brains". They are not programming at the level of concepts, symbols, and plans, but instead are relying upon crude approximations to how neurons work in brains.

An ideal future would involve a synthesis of these two schools of AI. Students could not only choose between computational building blocks that are symbolic or neural but use hybrids (a current topic of AI research). When symbolic-neural computation is well-enough understood we should provide accessible hybrid building blocks to learners. Studies are needed to explore more closely whether, indeed, learners' hands-on exposure to AI and machine learning concepts has any effect on their self-reflection and learning skills. And whether the effects are evident regardless of whether the AI is symbolic or neural. Another open question is what the minimum level of intellectual ability a child needs to benefit from AI programming.

AI is also being used in non-constructionist ways in schools. Automatic grading systems are proliferating. Learning analytics are applied to data generated from online courses. Research on intelligent tutoring systems continues to improve.

Currently these uses of AI in teaching work best when the software knows what the students were assigned to do. Maybe someday AI systems will be able to help guide students doing their own projects. Maybe AI systems will be able to provide suggestions and nudges that help students do projects without providing so much guidance that students are deprived of the joy and passion of self-directed creation of artefacts.

References

Ames, M. G. (2018). Hackers, Computers, and Cooperation: A Critical History of Logo and Constructionist Learning. Proceedings of the ACM on Human-Computer Interaction, 2(CSCW), 18.

Goldstein, I., Lieberman, H., Bochner, H., & Miller, M. (1975). LLOGO: An implementation of LOGO in LISP. MIT Artificial Intelligence Lab Memo Number 307a.

DGoogle (2020) Do-it-yourself artificial intelligence. https://aiyprojects.withgoogle.com/

Google (2017). Teachable Machine https://experiments.withgoogle.com/ai/teachable-machine

Kahn, K. (1977) Three interactions between AI and education. In Elcock, E. and Michie, D., editors, *Machine Intelligence 8: Machine Representations of Knowledge*. Ellis Horwood Ltd. and John Wylie & Sons.

Kahn, K. (1984). A Grammar Kit in Prolog. In Yazdani, M., editor, *New Horizons in Educational Computing*. Halsted Press.

Kahn, K. & Winters, N. (2017) Child-friendly Programming Interfaces to AI Cloud Services, *Proceedings of the EC-TEL 2017 Conference*, Tallinn, Estonia, September.

Kahn, K. & Winters, N. (2018). AI Programming by Children, *Constructionism Conference*, Vilnius, Lithuania, August.

Kahn, K., Lu, Y., Zhang, J., Winters, N. and Gao, M. (2020). Deep Learning Programming by All. *Proceedings of Constructionism Conference 2020*, Dublin, Ireland.

Kowalski, R. A. (1984). Logic as a computer language for children. In Yazdani, M., editor, *New Horizons in Educational Computing*. Halsted Press.

Lane, D. (2020). Machine Learning for Kids. https://machinelearningforkids.co.uk/#!/about

Minsky, M. (2009). OLPC Memo 5: Education and Psychology. https://web.media.mit.edu/~minsky/OLPC-5.html

Minsky, M. (2019). *Inventive Minds: Marvin Minsky on Education*. Edited by Solomon, C. and Xiao, X., MIT Press.

Nichol, J. & Dean, J. Pupils (1984). Computers, and History. In Yazdani, M., editor, *New Horizons in Educational Computing*. Halsted Press.

Papert, S. (1971). Teaching Children Thinking. MIT Artificial Intelligence Lab Memo Number 247.

Papert, S., & Solomon, C. (1971). Twenty Things to Do with a Computer. MIT Artificial Intelligence Lab Memo Number 248.

Papert, S. & Harel, I. (1991) *Constructionism*. Ablex Publishing Corporation.

Sergot, M. (1984). A Query-the-user Facility for Logic Programming. In Yazdani, M., editor, *New Horizons in Educational Computing*. Halsted Press.

Sloman, A. (1984). Experiencing Computation: A Tribute to Max Clowes. In Yazdani, M., editor, *New Horizons in Educational Computing*. Halsted Press.

Sloman, A. (2012). Examples of General and AI Programming Teaching Materials Illustrated using Poplog/Pop-11. http://www.cs.bham.ac.uk/research/projects/poplog/examples/

Smilkov, D., Thorat, N. Assogba, Y., Yuan, A., Kreeger, N., Yu, P., Zhang, K. (2019) "TensorFlow.js: Machine Learning for the Web and Beyond." *arXiv preprint arXiv:1901.05350*.

Solomon, C., Harvey, B., Kahn, K., Lieberman, H., Miller, M., Minsky, M., Papert, A. and Silverman, B. (2020). History of Logo. *Proc. ACM Program. Lang. 4, HOPL*, Article 79 (June 2020), 66 pages. https://doi.org/10.1145/3386329

Tissenbaum, M., Sheldon, J., & Abelson, H. (2019). From Computational Thinking to Computational Action. *Communications of the ACM*, 62(3), 34-36.

Touretzky, D. (2020). Ai4k12 wiki. https://github.com/touretzkyds/ai4k12/wiki

Wolfram, S. (2017) Machine Learning for Middle Schoolers. https://writings.stephenwolfram.com/2017/05/machine-learning-for-middle-schoolers/

Yazdani, M. (1984). Editor, *New Horizons in Educational Computing*. Halsted Press.