Designing Children's New Learning Partner: Collaborative Artificial Intelligence for Learning to Solve the Rubik's Cube

Forest Agostinelli
foresta@cse.sc.edu
University of South Carolina
Columbia, South Carolina, USA

Mihir Mavalankar
mihir.d.mavalankar@gmail.com
Independent Researcher
USA

Vedant Khandelwal
VEDANT@mailbox.sc.edu
University of South Carolina
Columbia, South Carolina, USA

Hengtao Tang
htang@mailbox.sc.edu
University of South Carolina
Columbia, South Carolina, USA

Dezhi Wu
dezhiwu@cec.sc.edu
University of South Carolina
Columbia, South Carolina, USA

Barnett Barry
barnettberry@sc.edu
University of South Carolina
Columbia, South Carolina, USA

Biplav Srivastava
biplav.s@sc.edu
University of South Carolina
Columbia, South Carolina, USA

Amit Sheth
amit@sc.edu
University of South Carolina
Columbia, South Carolina, USA

Matthew Irvin
irvinmj@mailbox.sc.edu
University of South Carolina
Columbia, South Carolina, USA

ABSTRACT

Developing the problem solving skills of children is a challenging problem that is crucial for the future of our society. Given that artificial intelligence (AI) has been used to solve problems across a wide variety of domains, AI offers unique opportunities to develop problem solving skills using a multitude of tasks that pique the curiosity of children. To make this a reality, it is necessary to address the uninterpretable "black-box" that AI often appears to be. Towards this goal, we design a *collaborative* artificial intelligence algorithm that uses a human-in-the-loop approach to allow students to discover their own personalized solutions to problems. This collaborative algorithm builds on state-of-the-art AI algorithms and leverages additional interpretable structures, namely knowledge graphs and decision trees, to create a fully interpretable process that is able to explain solutions in their entirety. We describe this algorithm when applied to solving the Rubik's cube as well as our planned user-interface and assessment methods.

CCS CONCEPTS

• **Computing methodologies** → **Discrete space search**; **Sequential decision making**; • **Human-centered computing** → **Empirical studies in interaction design**; **Usability testing**.

KEYWORDS

explainability, Rubik's cube, deep learning, reinforcement learning

ACM Reference Format:
Forest Agostinelli, Mihir Mavalankar, Vedant Khandelwal, Hengtao Tang, Dezhi Wu, Barnett Barry, Biplav Srivastava, Amit Sheth, and Matthew Irvin. 2021. Designing Children's New Learning Partner: Collaborative Artificial Intelligence for Learning to Solve the Rubik's Cube. In *Interaction Design and Children (IDC '21), June 24–30, 2021, Athens, Greece.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3459990.3465175

Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for third-party components of this work must be honored. For all other uses, contact the owner/author(s).
IDC '21, June 24–30, 2021, Athens, Greece
© 2021 Copyright held by the owner/author(s).
ACM ISBN 978-1-4503-8452-0/21/06.
https://doi.org/10.1145/3459990.3465175

1 INTRODUCTION

Developing problem-solving skills is a major challenge in education considered the single most important 21st Century skill, but it remains a major challenge in K-12 computer science education [12, 24]. Algorithmic thinking, described as the ability to develop effective algorithms to solve problems [21], is a core skill for students to foster computational thinking and solve computing problems [19]. We believe the artificial intelligence (AI) technologies can be a revolutionary change to the current education landscape to better equip our youth to be more prepared for future work by enhancing their ability to develop algorithms to solve problems. Therefore, it is critical for us to explore how we can design cutting-edge AI technologies to engage the youth in an AI-driven learning environment.

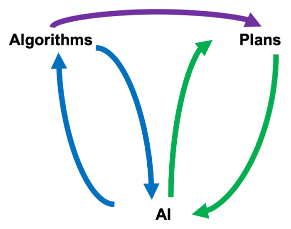

Figure 1: The student and AI agent exchange ideas to come up with a plan to solve the Rubik's cube using a set of algorithms.

AI has been used to solve many problems that involve algorithmic thinking, such as the Rubik's cube [6], Go [32], and chemical synthesis [11]. However, this possibility is impeded by the fact that AI is currently unable to explain its problem solving strategies to humans [1, 3]. To overcome this challenge, we seek to build a collaborative AI algorithm and then design an engaging interface to allow children to discover personalized solutions to challenging problems. In this paper, we focus on the Rubik's cube as a learning task for children due to its history of varied and creative solutions (demonstrated by its dozens of unique solutions [2]), its relationship to mathematics [20], and its decades long history of public interest ranging from grade school students to the frontiers of research [6, 23, 27]. By collaborating with AI, students will discover a *plan* to solve the Rubik's cube as well as a set of *algorithms* that they will use to fulfill that plan. An overview of this process is shown in Figure 1.

2 RUBIK'S CUBE BACKGROUND

The Rubik's cube is a three dimensional puzzle invented in 1974 by Erno Rubik. The Rubik's cube consists of 26 smaller cubes called cubelets. Each cubelet has between 1 and 3 stickers, where each sticker can be one of six colors. Cubelets are classified by their sticker count: center, edge and corner cubelets have 1, 2 and 3 stickers, respectively. Since there are 6 center cubelets, 12 edge cubelets, and 8 corner cubelets, the Rubik's cube has 54 stickers in total. The cube has 6 faces, where each face can be rotated 90° clockwise or counter-clockwise. By rotating these faces, the Rubik's cube can take on 4.3×10^{19} possible configurations. The goal is to rotate the faces to achieve the goal configuration where the stickers on each face are the same color. A visualization of a scrambled cube and cube that has been "solved" (returned to the goal configuration) can be seen in Figure 2. The 9 stickers that make up a face and their 12 adjacent stickers are referred to as a "layer".

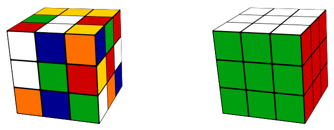

Figure 2: A visualization of a scrambled Rubik's cube (left) and a solved Rubik's cube (right)

Actions are represented using face notation: an action is a letter stating which face to rotate. *F*, *B*, *L*, *R*, *U*, and *D* correspond to turning the *front*, *back*, *left*, *right*, *up*, and *down* faces, respectively. Each face name is in reference to a fixed front face. A clockwise rotation is represented with a single letter, a letter followed by an apostrophe represents a counter-clockwise rotation, and a letter followed by a "2" represents two rotations. For example: *R* rotates the right face by 90° clockwise, *R'* rotates it by 90° counter-clockwise, and *R2* rotates the right face by two 90° rotations. Since these are the most basic actions that can be taken, we will refer to these as "atomic actions".

3 AI ALGORITHM DESIGNS: DEEPCUBEA AND CDEEPCUBEA

3.1 DeepCubeA

We have created an artificial intelligence algorithm, called DeepCubeA, that learns to solve the Rubik's cube without human guidance. DeepCubeA uses deep reinforcement learning [4, 33] to train a deep neural network (DNN) [29] that maps a Rubik's cube configuration to an estimate of the number of atomic actions left to solve it. It then uses this DNN with a search method, called A* search [17], to find a path (a sequence of atomic actions) to solve any Rubik's cube. The public also took interest in this work with coverage by news outlets such as the BBC [8], *Newsweek* [25], and *Forbes* [14]. To allow the public to interact with DeepCubeA, we created a web server that allows anyone to see the algorithm work in real time [5], which currently has over 31,000 unique visitors.

Since DNNs are generally considered to be uninterpretable, or "black boxes" [9], it is not possible to explain DeepCubeA's decision process. Therefore, DeepCubeA is useful for problem solving, but not helping someone else learn to solve a problem. Our core research task will be to extend DeepCubeA to produce explainable solutions to allow someone to learn to solve the Rubik's cube through collaboration.

3.2 CDeepCubeA

Each method for solving the Rubik's cube has its own high-level step-by-step *plan* and its own *algorithms* for achieving that plan [2]. We define a plan as a series of subgoals to be achieved in order to solve the cube where a subgoal can be any partial configuration of the Rubik's cube. A partial configuration specifies a subset of the cube that must be in a particular configuration while allowing the rest of the cube to be free to be in any configuration. An example of a plan that solves the Rubik's cube in a layer-wise fashion is shown in Figure 3. We define an algorithm as a series of atomic actions. The purpose of algorithms is to manipulate certain parts of the cube while keeping other parts fixed. An algorithm that swaps three edge pieces in the final layer of the cube while keeping the other layers and edge pieces in place is shown in Figure 4. We seek to create a collaborative version of DeepCubeA, collaborative DeepCubeA (CDeepCubeA), that collaborates with each student to find a high-level step-by-step plan that can be achieved using a set of algorithms. Communication necessary for collaboration will be made possible through the use of *knowledge graphs* [31] which can be used to describe Rubik's cube configurations and changes between configurations in human-interpretable terms.

In our framework, the AI agent with whom the student is collaborating begins with knowledge of only rudimentary algorithms and is given a high-level plan based on the student's own ideas. The agent then attempts to execute this plan with the rudimentary algorithms and reports back to the student about which parts of the plan were feasible and which appear to be infeasible. Some plans may be infeasible by nature while others require more sophisticated algorithms. Based on this feedback, the student has the opportunity to suggest a new plan or new algorithms or ask CDeepCubeA to suggest a new plan or new algorithms. If the student chooses to ask CDeepCubeA for suggestions, after CDeepCubeA gives its suggestions, the student can then choose which suggestions they would like to consider for the next iteration. This can be based on understanding, as some people may find some plans or algorithms easier to understand than others, or their personal preference. Learning new algorithms will also influence the kinds of plans the students comes up with. This process repeats until the

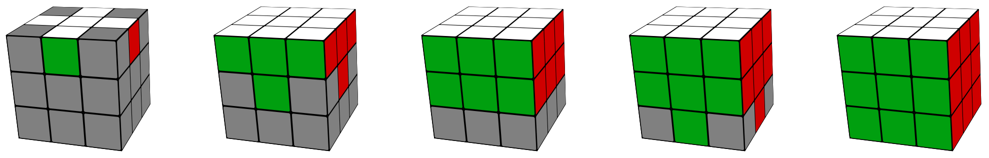

Figure 3: An example of a possible plan that solves the Rubik's cube in a layer-wise fashion. The grey parts of the cube are free to be in any configuration.

student finds a solution that they can understand. This exchange of ideas between student and AI is illustrated in Figure 1.

3.3 Knowledge Graph Integration

Children cannot be guaranteed to achieve the next subgoal by blindly applying algorithms. Instead, the algorithm selected for application must depend on the entire configuration of the cube. Instructions for solving the Rubik's cube describe a decision process based on certain queries. For example, Figure 4 shows a configuration that has three cubelets in the wrong place. However, one can notice that these three cubelets are in each other's correct position. Therefore, if one were to swap the cubelets, they would both be in the correct position. To figure out what to do, certain queries must be answered: "Is the first layer complete?", "Is the second layer complete?", "Is there a straight line at the bottom layer?", "Are there three edges that are out of place?". If all of the aforementioned queries return true, then we can execute the algorithm in Figure 4. Knowledge graphs [31] are able to encode such relationships among objects. We will construct a knowledge graph for describing a single configuration and changes between configurations by scraping Rubik's cube websites to find the most frequent descriptions of these two categories. [26, 30, 31].

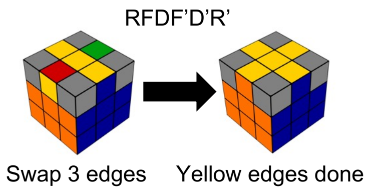

Figure 4: An algorithm that swaps three edge cubelets.

3.4 Learning Explainable Solutions

The original DeepCubeA algorithm can be seen as a special case of our framework where it executes a plan that has only one subgoal, the solved configuration, and where the only algorithms it had available to it are the atomic actions. DeepCubeA achieves the solved configuration by mapping each configuration to an atomic action. On the other hand, CDeepCubeA has a series of subgoals to achieve and only has to achieve each subgoal from the previous subgoal, significantly reducing the the number of configurations it must consider when learning to achieve each subgoal. Furthermore, CDeepCubeA will not use the atomic actions, but rather, algorithms, which are combinations of these atomic actions. To achieve any given subgoal, multiple algorithms may be needed and the order in which the algorithms are used will depend on the particular configuration. Moreover, a human must be able to understand the process behind determining which algorithms to use. To accomplish this, CDeepCubeA maps a configuration to a set of queries that are answered by a knowledge graph. CDeepCubeA then maps the results of those queries to an algorithm using an interpretable model, namely, a decision tree [28]. As a result, we would expect that CDeepCubeA can explain its entire strategy to a human by telling the human the queries it uses and the decision process behind mapping queries to algorithms. The difference between how DeepCubeA and CDeepCubeA find solutions to the Rubik's cube is shown in Figure 5.

3.5 Collaborative Problem-Solving

3.5.1 Suggesting New Plans. The user can suggest a new plan to CDeepCubeA through a visual interface where a user can click on cubelets and edit their colors to create a sequence of subgoals. CDeepCubeA can also suggest a new plan by showing the user a visualization of the sequence of subgoals. To discover those new plans CDeepCubeA will search for either 1) additional subgoals to simplify transitions between subgoals; 2) alternative subgoals to replace impractical or impossible subgoals.

To find additional subgoals, CDeepCubeA will monitor successful transitions between subgoals during training and cluster the configurations found in these transitions to see if certain types of configurations appear frequently. If so, then CDeepCubeA will create a subgoal from these frequently appearing configurations using the knowledge graph to describe their similarities. To find alternative subgoals, CDeepCubeA will remove a subgoal that is hard to achieve. For example, if CDeepCubeA cannot get from subgoal 4 to 5, it can remove subgoal 5 and try to go directly from subgoal 4 to subgoal 6. If successful, it will use this same clustering method to suggest an alternative subgoal to subgoal 5.

3.5.2 Suggesting New Algorithms. The user can suggest new algorithms to CDeepCubeA by specifying a sequence of atomic actions along with the queries that must be satisfied in order to use the algorithm. Similarly, CDeepCubeA can specify a new algorithm in the same manner. However, CDeepCubeA should also give the high-level intuition behind what the algorithm is doing. For example, in Figure 4, CDeepCubeA should be able to explain that this algorithm swaps three edge pieces. To accomplish this, we will use knowledge graphs to describe the relationship between two configurations.

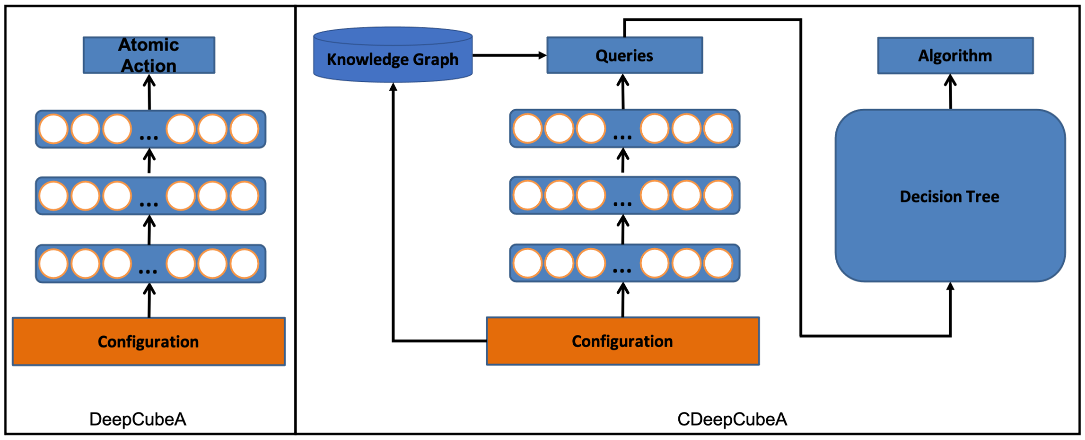

Figure 5: DeepCubeA (left) maps configurations to atomic actions using an uninterpretable DNN. CDeepCubeA maps configurations to human-interpretable queries that can be answered by a knowledge graph. It then maps these queries to human-interpretable algorithms. This creates a process that can be explained in solely human-interpretable terms.

To suggest new algorithms, CDeepCubeA will use knowledge graphs to infer what changes in configurations will be desirable and then search for algorithms that can fulfill this description. For example, CDeepCubeA can infer that swapping three edges that are out of place will get it closer to reaching a given subgoal. If it does not have an algorithm that can already do this, then it can search for one that does this while keeping all other relevant parts of the cube fixed.

4 USER INTERFACE DESIGN FOR AI-CHILDREN INTERACTIONS

To connect students to the CDeepCubeA algorithm, we plan to build a web-based interface embedded with a chatbot to translate queries and the decision process behind them to natural language. We will follow an iterative user-centered participatory design approach to design usable user interfaces (UIs) for end-users (i.e., children). First, user requirements for such a UI design will be acquired through a think-aloud method and observations with a small sample of 5 students who are familiar with Rubik's cube, 5 students who are not familiar with Rubik's cube, and the interface designer. The initial exploration lies in understanding how the CDeepCubeA algorithm behaves, what type of concepts that users will need to help solve the Rubik's cube, what user preferences are, and what types of social interactions/cues students would like to have when they are engaged in this game problem-solving process. Based on the user input, we will propose two alternative UI designs for initial A/B testing. With a few rounds of design iterations and usability testing, we will revise our initial UI design plans to select the more optimal one for prototype development. The objectives of this UI design are multi-folded: (1) to provide relevant and real-time feedback for students to cognitively engage and comfortably suggest and take suggestions regarding plans and algorithms in an understandable manner; (2) to provide visual feedback on the worked examples and guide students for next-step problem-solving presented on the main web-based interface; (3) to provide understandable natural language descriptions of suggestions provided by CDeepCubeA; and (4) to provide informative cues for students to understand how CDeepCubeA is collaborating with them with explicit signals on the UI, such as for when CDeepCubeA suggests new plans or algorithms. The current system will support American English. However, we recognize that children users can prefer other dialects and languages. One can create multiple variants of the system for other dialects and languages using localization and internationalization practices of programming for rendering strings [34] and using machine translators for content. We will defer this avenue to future work after the first stable release in American English.

5 ASSESSMENT

A design-based research [18] will be conducted to assure the usability of CDeepCubeA in fostering students' algorithm thinking. To improve the validity of the usability testing, twenty students will be recruited [7]. The selection of the participants will follow the maximum variation [13] methods to include participants from different academic backgrounds and grades and with different levels of mathematics and algorithmic knowledge. Formative and summative evaluations will be conducted to collect evidence about the effectiveness and usefulness of the platform. Specifically, formative evaluation activities will include a focus group interview with participants [16] and video recorded sessions of user-platform interaction [22]. Semi-structured interview protocols will be provided to structure the focus groups and also attend to novel insights that emerge during the focus groups [16]. The focus group will be audio/video recorded for analysis. Video recorded sessions of user-platform interaction will be collected through high-resolution camera (face-to-face) or Zoom (online) contingent upon the COVID-19 safety protocol. The video recorded sessions will be imported to Nvivo 12 for analysis. Thematic analysis [10] will be conducted on those two sources of formative evaluation data to identify problems in the prototype and uncover opportunities to improve the design. Summative evaluation activities will follow one-group pretest-posttest design [15] including a pretest and a posttest on students' algorithm knowledge and thinking [19]. Descriptive statistics and inferential statistics will be performed to gauge the effectiveness of the collaborative AI platform. The evaluation results will be converged to inform the revision to the prototype of CDeepCubeA. Another iteration of the design-based research [18] will be done to confirm whether the revision impacts student algorithm thinking.

REFERENCES

[1] Amina Adadi and Mohammed Berrada. 2018. Peeking inside the black-box: a survey on explainable artificial intelligence (XAI). *IEEE access* 6 (2018), 52138–52160.

[2] Anupama Aggarwal. [n.d.]. List of methods. https://www.speedsolving.com/wiki/index.php/List_of_methods/.

[3] Forest Agostinelli. 2021. How explainable artificial intelligence can help humans innovate. https://theconversation.com/how-explainable-artificial-intelligence-can-help-humans-innovate-151737.

[4] Forest Agostinelli, Guillaume Hocquet, Sameer Singh, and Pierre Baldi. 2018. From reinforcement learning to deep reinforcement learning: An overview. *Braverman readings in machine learning. key ideas from inception to current state* (2018), 298–328.

[5] Forest Agostinelli, Stephen McAleer, Alexander Shmakov, and Pierre Baldi. 2019. DeepCube Webserver. http://deepcube.igb.uci.edu/.

[6] Forest Agostinelli, Stephen McAleer, Alexander Shmakov, and Pierre Baldi. 2019. Solving the Rubik's cube with deep reinforcement learning and search. *Nature Machine Intelligence* 1, 8 (2019), 356–363.

[7] Roobaea Alroobaea and Pam J Mayhew. 2014. How many participants are really enough for usability studies?. In *2014 Science and Information Conference*. IEEE, 48–56.

[8] BBC. 2019. AI solves Rubik's Cube in one second. https://www.bbc.com/news/technology-49003996. Accessed: 2019-10-27.

[9] José Manuel Benítez, Juan Luis Castro, and Ignacio Requena. 1997. Are artificial neural networks black boxes? *IEEE Transactions on neural networks* 8, 5 (1997), 1156–1164.

[10] Virginia Braun and Victoria Clarke. 2006. Using thematic analysis in psychology. *Qualitative research in psychology* 3, 2 (2006), 77–101.

[11] Binghong Chen, Chengtao Li, Hanjun Dai, and Le Song. 2020. Retro*: learning retrosynthetic planning with neural guided A* search. In *International Conference on Machine Learning*. PMLR, 1608–1616.

[12] Charles Conn and Robert McLean. 2019. *Bulletproof Problem Solving: The One Skill that Changes Everything*. John Wiley & Sons.

[13] John W Creswell and Cheryl N Poth. 2016. *Qualitative inquiry and research design: Choosing among five approaches*. Sage publications.

[14] Forbes. 2019. This AI Can Solve A Rubik's Cube Super Fast. https://www.forbes.com/sites/jenniferhicks/2019/07/15/this-ai-can-solve-a-rubiks-cube-super-fast/. Accessed: 2019-10-27.

[15] Meredith Damien Gall, Walter R Borg, and Joyce P Gall. 1996. *Educational research: An introduction*. Longman Publishing.

[16] Joanne Gikas and Michael M Grant. 2013. Mobile computing devices in higher education: Student perspectives on learning with cellphones, smartphones & social media. *The Internet and Higher Education* 19 (2013), 18–26.

[17] P. E. Hart, N. J. Nilsson, and B. Raphael. 1968. A Formal Basis for the Heuristic Determination of Minimum Cost Paths. *IEEE Transactions on Systems Science and Cybernetics* 4, 2 (July 1968), 100–107. https://doi.org/10.1109/TSSC.1968.300136

[18] Christopher M Hoadley. 2004. Methodological alignment in design-based research. *Educational psychologist* 39, 4 (2004), 203–212.

[19] Chih-Chao Hsu and Tzone-I Wang. 2018. Applying game mechanics and student-generated questions to an online puzzle-based game learning system to promote algorithmic thinking skills. *Computers & Education* 121 (2018), 73–88.

[20] David Joyner. 2008. *Adventures in group theory: Rubik's Cube, Merlin's machine, and other mathematical toys*. JHU Press.

[21] Zoltán Kátai. 2015. The challenge of promoting algorithmic thinking of both sciences-and humanities-oriented learners. *Journal of Computer Assisted Learning* 31, 4 (2015), 287–299.

[22] TJ Kopcha, J McGregor, S Shin, Y Qian, J Choi, R Hill, J Mativo, and I Choi. 2017. Developing an integrative STEM curriculum for robotics education through educational design research. *Journal of Formative Design in Learning* 1, 1 (2017), 31–44.

[23] Richard E. Korf. 1997. Finding Optimal Solutions to Rubik's Cube Using Pattern Databases. In *Proceedings of the Fourteenth National Conference on Artificial Intelligence and Ninth Conference on Innovative Applications of Artificial Intelligence* (Providence, Rhode Island) *(AAAI'97/IAAI'97)*. AAAI Press, 700–705. http://dl.acm.org/citation.cfm?id=1867406.1867515

[24] Richard E Mayer and Merlin C Wittrock. 2006. Problem solving. *Handbook of educational psychology* 2 (2006), 287–303.

[25] Newsweek. 2019. Rubik's Cube Solved in 'Fraction of a Second' by Artificial Intelligence Machine Learning Algorithm. https://www.newsweek.com/rubiks-cube-artificial-intelligence-machine-learning-algorithm-1449476. Accessed: 2019-10-27.

[26] Hemant Purohit, Valerie L Shalin, and Amit P Sheth. 2020. Knowledge Graphs to Empower Humanity-Inspired AI Systems. *IEEE Internet Computing* 24, 4 (2020), 48–54.

[27] Tomas Rokicki, Herbert Kociemba, Morley Davidson, and John Dethridge. 2014. The diameter of the Rubik's Cube group is twenty. *SIAM Rev.* 56, 4 (2014), 645–670.

[28] S Rasoul Safavian and David Landgrebe. 1991. A survey of decision tree classifier methodology. *IEEE transactions on systems, man, and cybernetics* 21, 3 (1991), 660–674.

[29] Jürgen Schmidhuber. 2015. Deep learning in neural networks: An overview. *Neural networks* 61 (2015), 85–117.

[30] Amit Sheth, David Avant, and Clemens Bertram. 2001. System and method for creating a semantic web and its applications in browsing, searching, profiling, personalization and advertising. US Patent 6,311,194.

[31] Amit Sheth, Swati Padhee, and Amelie Gyrard. 2019. Knowledge graphs and knowledge networks: The story in brief. *IEEE Internet Computing* 23, 4 (2019), 67–75.

[32] David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, et al. 2017. Mastering the game of go without human knowledge. *Nature* 550, 7676 (2017), 354.

[33] Richard S Sutton and Andrew G Barto. 1998. *Reinforcement learning: An introduction*. Vol. 1. MIT press Cambridge.

[34] Wikibooks. [n.d.]. FOSS Open Standards/Standards and Internationalization/Localization of Software. https://en.wikibooks.org/wiki/FOSS_Open_Standards/Standards_and_Internationalization/Localization_of_Software, year=2020.