Demystifying Black Boxes: Analysis of Neural Network Transparency Research

Improving Computer Science Pedagogy for Dynamic Values in Society

A Thesis Prospectus
In STS 4500
Presented to
The Faculty of the
School of Engineering and Applied Science
University of Virginia
In Partial Fulfillment of the Requirements for the Degree
Bachelor of Science in Computer Science

By
Parker Hutchinson

12/17/2023

On my honor as a University student, I have neither given nor received unauthorized aid on this assignment as defined by the Honor Guidelines for Thesis-Related Assignments.

ADVISORS

Kathryn A. Neeley, Department of Engineering and Society

Lu Feng, Computer Science

Improving Computer Science Pedagogy for Dynamic Values in Society

Overview:

I was introduced to the world of computer science through unconventional means, but it opened many doors for me and has been a passion of mine since I discovered it. I will review the most prominent methods currently being used to teach computer science, and then offer a value-sensitive design analysis of how the system could be improved to reach more students and unlock their interest, just as mine was. Readers can expect to walk away from this paper with a greater understanding of the way computer science is currently taught, and with knowledge of how this education can be improved for learners of all stages.

Positionality:

My research goals arise from my personal experience in trying to teach computer science to myself and others. Programming always seemed like a magical concept to me, some sort of wizardry experts were able to perform by flipping 1s and 0s. This perception prevented me from seriously considering looking into the topic, and I believe many of my friends had a similar experience. As a freshman in high school, however, I took a free online Codeacademy course for the Python programming language and was illuminated to how accessible this world was if one simply had the right introduction and tools. On the first day I wrote a "Hello, World!" program, and by the end of the roughly month-long course I was able to write functions and use advanced language features like list comprehension. I'm incredibly fortunate to have had the opportunity I had to learn this subject, given that I had both the resources, like a laptop and internet connection, to complete the course, as well as the free time. These variables are not present for all people and represent an advantage I had that creates a somewhat unlevel playing field.

Later, I would work at Code Ninjas, a company that teaches computer science principles to young children. This was a fulfilling experience, and I felt I was guiding people through a complex topic in a way I wished someone could have done for me at the same age. I would like to continue this effect in my research. Learning computer science opened a whole new world to me, and with this project I will help open that door for others who otherwise might not reach it.

Problematization:

Computer science education isn't reaching enough people – there is still a great demand for more software engineers. In addition, the people who are being educated are receiving an education that doesn't adequately prepare them for the work they will expected to do in their careers, as Garousi et. al discuss in their paper "Closing the Gap Between Software Engineering Education and Industrial Needs" (2020). The authors describe how computer science graduates "face difficulties when beginning their professional careers" due to a "misalignment" of their university education and what the industry expects of them. This deficit hurts these graduates, the companies they work for, and society at large when the general population uses their products which could be better.

Guiding Question:

What is the best way to teach computer science so that it can reach more people while also giving them a proper education of the field?

Projected Outcomes:

As a result of this research, I will aim to suggest a policy change for the University of Virginia computer science curriculum. This policy change would advocate for certain teaching methods and ways of measuring student performance, which although possibly bearing a

resemblance to past methods would likely diverge in many ways from conventional teaching. Students, employers, and teachers would all benefit as a result of the outcomes of this research.

Technical Project Description:

My experience in this area encompasses internships, school education, and my own extracurricular learning. I've completed internships in software engineering, where I used concepts that I had learned in class but also had to pick up skills on the spot. I've completed many computer science courses, both in high school and in college at the University of Virginia. Through online courses, I've also learned a significant amount outside of traditional schooling. I've taken a course in Machine Learning, as well as in cloud computing, culminating in the achievement of an Amazon Certified Cloud Practitioner Certification. I've also worked at Code Ninjas, where I was an instructor for computer science lessons taught to younger students, giving me some background in software education.

Preliminary Literature Review & Findings:

Much research has gone into this topic. Garousi et. al, as previously noted, published an article in IEEE in which they claim that many fresh computer science graduates "face difficulties when beginning their professional careers", and that these difficulties are a result of a "misalignment" between the topics they are taught in their university education "and what is needed in industry" (2020). In ""Software Engineering Education Knowledge Versus Industrial Needs", Liargkovas et. al sought to understand exactly which topics were taught adequately by educational institutions to current software engineering professionals. The authors found that subjects like "computer science fundamentals, software design, and mathematical concepts" were adequately addressed by the prior education of the engineers studied. However, the authors also noted that topics like "the World Wide Web, software engineering (SE) components, and computer graphics" were lacking in terms of educational preparation (2022).

Other, more niche areas of computer science exist that could be improved upon with modifications to educational curricula. In "Cognitive Biases In Software Engineering: A Systematic Mapping Study", Mohanani et. al analyze cognitive biases, defined as "systematic deviations from optimal reasoning", and how they can become a source of "software project challenges and failures". The authors argue that there is a "scarcity of research on mitigation techniques" and a lack of understanding for these biases. They also argue that "specific bias mitigation techniques" are needed for software engineers to avoid the negative effects of cerebral shortcomings on their work (2020). An education that notes these biases as well as mitigation strategies for them might help prevent them from arising in future software engineers. Eidenskog et. al have also attempted to bring a modernized update to an engineering program, but noted difficulties in teaching topics like sustainability to engineers because of their "fluffy" perception (2023).

I aim to build off of the existing literature and create an educational solution which addresses the industrial needs of software engineers, and prepares them for success by applying a modern analysis and decomposition of their most important skills. I hope to synthesize the best findings from studies that analyze teaching methods, problems in the current system, and others into a coherent set of policy recommendations that can be enacted by higher-level learning institutions.

STS Project Proposal:

STS is the study of how society and culture impact our engineering process and ethics, as well as an analysis of how we can and should affect the opposite by creating an impact on the world with our designs. STS is what separates UVA engineers from the thousands of others

across the country. By studying the implications of our creations, we are making ourselves unique in our understanding and appreciation of the effects that our designs can have on our fellow humans.

This approach will align with the ethics and values as well as the policy ecosystems of knowledge of STS. The implications of a certain method of instruction will be analyzed, which is connected to ethics in terms of finding a solution that is equitable for all stakeholders, and connected to policy in that the findings may inform decisions from school administrators. I plan on basing my research on the foundations of previous STS writers and their findings. One specific piece that I plan to use as a keystone for my research is "Philosophy, Rhetoric, and the End of Knowledge: The Coming of Science and Technology Studies", by Steve Fuller and Jim Collier (1993). I think this book will be fruitful for my research because the authors discuss an STS approach to creating knowledge, and analyze the social process of how it is created. Also, given that the book was published in 1993, the research is reasonably recent and likely applicable to modern teaching methods.

My approach will integrate value sensitive design throughout the research to inform areas of interest for the research and maintain ethical standards. I specifically plan on using the value sensitive design process as defined by David Hendry, Batya Friedman, and Stephanie Ballard from the University of Washington and described in the article "Value sensitive design as a formative framework". This approach is one in which the researcher is "systematically accounting for human values during design" and incorporating human values from all interested stakeholders to arrive at a decision that creates the most benefit without violating any defined principles (Hendry et. al 2021). I will use this approach to ensure that my research respects the values of those who would benefit from it and those who are being researched. To do this, I will conduct interviews or use other methods of eliciting the values held by teachers and professors, and will also explicitly state the research goals and outcomes to all participants. I think the contributions of this approach are important because they ensure that everyone is heard throughout the research process, which increases the likelihood of finding a solution that benefits everyone. Following this approach should greatly help me conduct research that harms nobody and possibly helps many.

I plan on using the value source analysis method to inform my value sensitive design approach. The project values that I explicitly support are increased awareness and knowledge of computer science skills and cognitive mitigation techniques which are currently lacking in conventional curricula. My personal values include creating a project which I can realistically conduct and which I think could create value for the computer science community at the University of Virginia. Stakeholders, including students, professors, and employers, have other values which must be distinguished as well. Some values I expect students to have a desire for an education which prepares them adequately for their careers, and an education which isn't too strenuous or too easy for them to complete. I expect professors to hold values of wanting to effectively instruct their students in their course material and wanting to teach lessons that fit within their teaching abilities and timeframe. I expect employers to value students who know many of the skills that they will need once they arrive in the workplace. I can further distinguish these values in my project by conducting interviews and questionnaires with direct and indirect stakeholders.

In my project, I plan to supplement a traditional computer science course at the University of Virginia with lessons through which I seek to address the areas of education which my research has indicated are lacking. With permission from a professor, I envision giving a

lecture every three weeks to a computer science class on these topics. I plan to conduct questionnaires, for quantitative analysis, and interviews, for more qualitative and formative analysis, at the end of the semester to gauge the effectiveness of the teachings and understand student and professor responses to the lectures. I hope to also have employers interview some of the students at the end of the course and indicate their perception of the students' readiness for industry work, signaling how well the research addressed the values of employers. Once the research has been completed, I will summarize the findings and indicate whether it appears as if this project sufficiently prepared students for a career in software engineering, while also serving the needs of those students and their professors.

Barriers & Boons

Most of the research I've found appears to be centered around the United States. This makes sense, with America having more software engineering companies and employees than any other nation. However, it could introduce a limitation on the generality of my findings when considering how they could be applied to other nations or types of institutions.

Conducting research on learning methods may inherently take a long time – about 3-4 months for a semester-long course, and 5-6 for a high school semester-long course. These constraints would have to be considered if I hoped to conduct testing at the scale of an entire course.

I have some experience teaching software though my work at Code Ninjas, but no experience in a traditional classroom setting. To address this, I could spend some time shadowing a lecturer in a computer science classroom.

References

Eidenskog, Maria, et al. "Changing the World One Engineer At a Time – Unmaking the Traditional Engineering Education When Introducing Sustainability Subjects." International Journal of Sustainability In Higher Education, vol. 24, no. 9, 13 Mar. 2023, pp. 70 - 84.

Fuller, S. (1993). Philosophy, Rhetoric, and the End of Knowledge: The Coming of Science and Technology Studies. Madison, Wisc: University of Wisconsin Press.

Garousi, V., et al. "Closing the Gap Between Software Engineering Education and Industrial Needs." IEEE Software, Software, IEEE, IEEE Softw, vol. 37, no. 2, 1 Mar. 2020, pp. 68 - 77.

Hendry, D. G., Friedman, B., & Ballard, S. (2021, March 1). Value sensitive design as a formative framework. Ethics & Information Technology, 23(1), 39 - 44.

Liargkovas, Georgios, et al. "Software Engineering Education Knowledge Versus Industrial Needs." IEEE Transactions On Education, vol. 65, no. 3, 1 Aug. 2022, pp. 419 - 427.

Mohanani, R., et al. "Cognitive Biases In Software Engineering: A Systematic Mapping Study." IEEE Transactions On Software Engineering, Software Engineering, IEEE Transactions On, IIEEE Trans. Software Eng, vol. 46, no. 12, 1 Dec. 2020, pp. 1318 - 1339.