Teaching Statement

Deepak Narayanan

I have enjoyed teaching since I was in high school. Walking my classmates through problems always helped me understand concepts better by forcing me to break down complex ideas into a series of simpler, logically consistent steps. I believe academics should be able to effectively communicate technical ideas to a broad audience, and I see teaching as an important way to hone this skill. Over the course of my time as a graduate student at MIT (M.Eng.) and Stanford (Ph.D.), I was a Teaching Assistant for various courses, where I taught recitation sections and designed assignments. I have also been a research mentor for younger undergraduate and graduate students.

I TAed three undergraduate algorithms classes and two graduate systems classes ("Principles of Data-Intensive Systems" and "Parallel Computing") at MIT and Stanford. My responsibilities included teaching and writing lecture notes for recitation sections of 20-30 students, where we reviewed material covered in lecture and applied some of those ideas in example problems. I also helped write and design problem sets and examinations from scratch: I fondly remember working on a programming assignment with Prof. Nancy Lynch that examined the benefits of using various data structures for a real-world data analysis problem, a multi-threaded task execution system with Prof. Kayvon Fatahalian that required students to carefully reason through using fine-grained locks and condition variables to ensure thread safety in the face of concurrent updates, and an in-memory relational data store with Prof. Matei Zaharia that walked students through the various tradeoffs associated with using row, column, and indexed data stores for different types of queries. In designing these assignments, I developed test suites to check for both functionality and performance, and set up grading infrastructure to ensure that assignments were graded automatically. In some cases, we needed to tweak the assignments to ensure the assignments actually achieved our learning goals; the assignments were generally well received, and used in subsequent iterations of the course.

In my opinion, the main objective of a class should be to communicate a small set of key concepts clearly and succinctly, even if the class covers a wide breadth of material over the course of a semester. From my experience both as a student and a teacher, I have found that reinforcing ideas repeatedly in various forms can help with this. I also believe that students learn best by doing: for example, teaching lectures using slides, while convenient, can be an ineffective medium of instruction for certain types of material; I have always found that I understand a mathematical proof better if I work through the individual steps myself. I hope to make classes more interactive, e.g., by using a whiteboard extensively when teaching more theoretical classes, or integrating programming exercises and interactive demos (e.g., showing a visualization of a machine learning model's parameters changing with time) through the course of a lecture. Outside of the classroom, I think homework assignments provide a way for students to interact with the material at a level much deeper than is possible in just a lecture. I found classes that made me think hard doing homework assignments the most effective, and I hope to design classes and assignments that teach students how to apply concepts to practical problems they might see in the real world, leveraging my teaching experience thus far.

As a faculty member, I would be qualified to teach introductory programming, algorithms, and machine learning classes, as well as undergraduate- and graduate-level computer systems classes (e.g., distributed systems, databases). Additionally, I would love to introduce a class focused on the intersection of systems and machine learning that discusses the practical issues (e.g., efficient model training and inference, model monitoring, training data collection) associated with learning and deploying models in production. This is one of my main research areas, and an area that I think will have a large impact in the coming years as more consumer-facing software uses AI.

I have also been a research mentor for younger graduate and undergraduate students as a Ph.D. student at Stanford: Keshav Santhanam worked closely with me on building a GPU cluster scheduler for ML training jobs (Gavel) as a M.S. student, and eventually joined our lab as a Ph.D. Student. I also worked with Parimarjan Negi (now a Ph.D. student at MIT) and Rahul Palamuttam to support new applications as part of the Weld project. I have mentored younger Ph.D. students at a higher level (Peter Kraft and Gina Yuan), providing feedback on paper pitches, communication, and technical ideas. In my experience, a single mentorship style does not work for everyone: some students prefer a more hands-on approach, while others prefer being given time to solve problems on their own. Much of being a good research mentor is recognizing the style students prefer early on and adapting accordingly. My ultimate goal with research mentorship is to equip every student with a toolbox that allows them to do research independently.