

KAYUR PATEL | TEACHING STATEMENT

While I enjoy research, I originally entered the PhD program to teach. To me, teaching is one of the most exciting and compelling tasks for faculty members. I've pursued teaching and mentoring throughout my career, first as an undergraduate at CMU, and then as a masters student at Stanford and more recently throughout my PhD program at UW. In this statement, I'll describe my teaching philosophy and experience in detail.

I started teaching as an undergraduate at Carnegie Mellon, serving as a teaching assistant for introductory computer science courses, and during this time I received some of the highest teaching reviews in the department. I believe that strong introductory courses are key to expanding the reach of computing, and for this reason, I focused on teaching introductory programming courses. These were the same courses that originally got me excited about and ultimately kept me in computer science.

Carnegie Mellon allowed me to experiment with teaching recitation sections. I increased engagement by making the sections interactive. For example, I taught linked lists by selecting a group of students, labeling each student, and physically linking them together with string. I then demonstrated basic linked list functionality using the metaphorical representation provided by the students. Now, years later, ex-students have communicated that this was both an effective teaching strategy and a memorable moment in their undergraduate experience.

I continued teaching as a graduate student, serving as a teaching assistant for Stanford's graduate artificial intelligence course and the University of Washington's graduate human-computer interaction course. Stanford's AI course resembled an advanced undergraduate course. It had a large, diverse class and focused less on current research in artificial intelligence and more on applicable skills. In contrast, the HCI course at Washington was smaller and designed to familiarize students with human-computer interaction research. Through these experiences, I learned how to successfully build and teach several drastically different advanced-topic courses.

During graduate school, I was deeply involved in mentoring undergraduate researchers, advising 12 students on various projects. Mentoring has been more of a challenge for me than teaching. My research requires expertise in advanced programming skills, such as designing graphical user interfaces and writing machine learning code. Working with a broad set of students has taught me how to gauge a individual student's skill set and how to help them independently expand their skills to provide meaningful contributions to a project. My reflections on mentoring led me to co-instruct a seminar entitled "Research in Computer Science", which was designed to smooth the transition from course work to research for young graduate students and older undergraduates.

Building an atmosphere conducive to learning requires a commitment to service. In the research community, I served as the student innovation competition chair for the *ACM Symposium on User Interface Software and Technology*. Events like the student innovation contest provide an important avenue for motivated undergraduates and graduates from under-represented institutions to participate in the research community. For example, three of the eight winners were undergraduates from teaching institutions when I ran the contest.

My experiences have reinforced my goal of teaching and expanded my conception of what teaching is. Although mentoring my undergraduates is my favorite weekly activity, I am excited to return to the classroom. I would love to teach introductory computer science courses as well as courses in human-computer interaction, artificial intelligence and software engineering. I am also excited about teaching new courses that build off of my research and courses targeted at increasing participation of under-represented students in computing.

The following are examples of courses I would be particularly enthusiastic to teach:

Undergraduate Courses

Computational Thinking: An introductory course designed to introduce non-majors to computation and expose them to basic concepts such as decomposition and abstraction. Modeled on similar courses at institutions such as UC Berkeley, Harvey Mudd and Georgia Tech.

Introduction to Human-Computer Interaction: Teaches basic tools and techniques for building, evaluating and improving user interfaces. Assignments would focus on techniques used in practice such as rapid prototyping, contextual inquiry and heuristic evaluation.

Human-Computer Interaction Design Studio: Teaches students how to design, build and iterate on an interactive application in a group setting. Provides experience with modern prototyping and design tools. Projects involve collaborating in mixed teams of computer science students and students other disciplines such as art, design and business.

Introduction to Artificial Intelligence: Basic introduction to artificial intelligence. Covers topics such as machine learning, planning, natural language processing, intelligent agents and search. Students are assigned problem sets designed to familiarize them with common algorithms.

Programming with Data: An advanced course that teaches students how to build end-to-end machine learning systems for problems such as sketch recognition, spam classification and activity recognition. Covers algorithms but focuses on the machine learning process. Teaches how to collect data, generate features, run experiments and visualize data and results.

Graduate Courses

Intelligence in Interfaces: Covers emerging research at the intersection between artificial intelligence and human-computer interaction. Provides an overview of the current research through readings, discussions and course projects.

Human-Computer Interaction: Provides an introduction to research in human-computer interaction. Readings focus on current breakthroughs and historical results in sub-areas such as user interface toolkits, interaction techniques, computer-supported collaborative work and technology for development. Readings are supplemented by assignments and a course project.

I pursued a PhD because I wanted to work with students. I have been active in teaching, mentoring and creating atmospheres conducive to research and learning. I am excited about innovating in the classroom. As a faculty member, I look forward to designing new curricula, interacting with students in the classroom and advising young researchers to help shape the next generation of computer scientists.