August Shi — Teaching Statement

I am excited to become a teacher and advisor for the next generation of students. I find teaching to be an enlightening and rewarding experience. I enjoy helping students grow and improve their skills to achieve their own future goals, and I appreciate how much I in turn learn from my students.

Teaching Philosophy

My teaching and mentoring experiences have helped me develop a teaching philosophy focused on the following aspects: (1) teaching requires establishing proper communication between the teacher and the students; (2) project-based assignments help students bridge the gap between theory and practice; and (3) lectures should be interactive, and they should contain proper motivation and examples to keep students engaged. I next describe my teaching experiences in detail and the lessons I learned that helped shape my philosophy.

Teaching Experience

I had several teaching opportunities during my time as a graduate student. (1) I was a teaching assistant (TA) for a large undergraduate Software Engineering course with 176 students. From my responsibilities in this course, I experienced teaching in both a large-scale setting (helping with lectures and homework assignments) and a small-scale setting (coaching the students' team projects). (2) I gave guest lectures on software testing in smaller courses for multiple professors. (3) I prepared and taught a one-hour class on software testing to high school students as part of my department's outreach program.

Large-Scale Teaching. As a TA in the large Software Engineering course, I answered student questions about lectures and homework assignments, with the goal to help them understand the material better and get them unstuck on homework. However, it is unfortunately impossible to personally help every student with their difficulties in such a large course setting, which is especially challenging given the diverse set of students in the course, all with varying expertise and backgrounds. Helping this large and diverse class necessitates the use of an online communications platform such as Piazza, which we used in our course. What I quickly discovered was that while Piazza is a great platform for students to get help, they started to take advantage of it once they found out that the teaching staff was very responsive. Students began to rely on Piazza as a platform to get quick answers instead of searching for answers themselves. The experience here helped me understand that while it is important to have a platform to get questions and feedback from students, we cannot just constantly give out the right answers to all the questions; we also have to help students "learn to learn." Throughout the course, I balanced giving out the proper hints to help the students get unstuck while avoiding giving out all the answers, prodding them to explore and learn on their own.

Small-Scale Coaching. As another part of my TA duties, I served as a coach for four teams of 7-8 students, each working on developing new features for an existing open-source project from GitHub. I helped the students with designing the features they wanted to develop, guided them to the resources they needed during the development phase, and challenged them to test their code properly. I encouraged the students to talk to me and ask questions about points they did not understand, with the goal to help them improve their own technical skills. At the end of the semester, I helped the students contribute their features back to the open-source community. Throughout this process, the students got the chance to apply the knowledge they learned from the lectures directly into their work. Projects such as these really give students a hands-on approach to learning the material, and I believe students can truly appreciate the content learned from lectures after experiencing those concepts as part of their own software development. In the future, I plan to incorporate such a project-based component into my courses so students can experience the "practice" in addition to the "theory" taught in lectures.

Lecturing. I gave four guest lectures on software testing topics related to my research for multiple professors in their smaller, graduate-level courses. As part of these lectures, I included plenty of examples to better illustrate both the problems and the solutions. With the examples, I interacted with the students by asking them how they would solve certain problems. I challenged them to think of edge cases or reasons why certain solutions would not work. I found this interaction important in keeping the students engaged, reinforcing my belief that proper communication between teacher and students is necessary to better facilitate learning.

Outreach. As part of my department's outreach program, which attracted high school students from all over Illinois, I, along with two of my fellow graduate students, prepared and co-taught a class on software testing to 19 high school students. We designed our class, "Software Testing for Fun, Fame, and Maybe Even Profit," to teach high school students the basics of software testing. During the lecture part of the one-hour class, I made sure to keep the lecture interactive by including examples of tests and bugs while encouraging the students to share their own thoughts, expectations, and experiences with software bugs with which testing would help. We also included a small team-based component, where we developed a buggy command-line calculator and had the students work in small groups to create test cases that could expose the bugs in this calculator program. I noticed the students seemed the most excited about this part, and they were delighted every time they came up with a test case to break our program. My experience here further reinforced my belief that lectures need to be interactive, involving two-way communication between students and teacher, and that a hands-on component is essential for students to truly appreciate the material taught.

Mentoring

Over the years, I have had the great pleasure to work with and mentor several undergraduate and junior graduate students in research projects. I mentored four undergraduate students and five Master's students, all from diverse backgrounds and with different levels of technical experience. My mentees and I submitted nine papers to top-tier conferences, with eight of them already published. I further informally co-advised the theses for two of the Master's students I worked with. One of the Master's students I worked with began a PhD in the CS program at TU Munich.

From my experiences, I found it necessary to adjust how I mentor based on the student and their experience with research. For students who are just starting out with research, I found it important to give them direct and concrete tasks to work on to allow them to "get their hands dirty" with research. While I would like students to explore and find their own interests, my experiences showed that students would often get lost and confused (which I also experienced early on when I was a junior student). Having the student quickly get to work on a project can help them build confidence as well as develop the foundation for conducting research. As the students grew more comfortable and became more experienced, I encouraged them to think more abstractly and try to find how their work fits into the overall research process. Throughout the mentorship, I always encouraged my mentees to discuss any difficulties they have. I also made it a point to discuss *why* they were doing any task; whether it was the concrete task I assigned them at the beginning or any task they started exploring themselves once they got comfortable with research. My overall goal is to help the students develop their own research taste and become better researchers overall.

Teaching Interests

Based on my background and experiences, I am eager to teach undergraduate courses on software engineering, software testing, programming languages, systems programming, compilers, data structures, and algorithms. In such courses, I want to prepare lectures that can help students develop their technical skills that can be useful for their future career goals, whether in an industrial or an academic environment.

At the graduate level, I am excited to teach a course on software engineering research topics. In this course, I want to give students the chance to work on cutting edge research in software engineering, while pushing them to consider how their research can connect with current industry challenges. I would also like to organize seminars on software engineering and related topics. Throughout my own graduate studies, I always appreciated such seminars, because students often connected with each other on their research topics. Discussions in seminars also pushed us to think more critically about research in general. I look forward to leading such seminars to help future students similarly.