My teaching philosophy is to promote learning by engaging students using active learning techniques, encouraging critical thinking skills, and respecting students' opinions and backgrounds. This is based on my experience of attending classes as an undergraduate and graduate student, working as a teaching assistant aiding professors to develop teaching material and conduct in-class activities, delivering guest lectures, and tutoring students online. I plan to apply the following practices to keep my students motivated and engaged.

Engaging students using active learning. In my experience *learning by doing* is more effective in understanding and remembering a topic than just reading or listening about it. Therefore, in addition to lectures, homework, and exams, I will conduct in-class activities and group discussions where students can apply their learning from the course and have a better understanding of topics. As a TA for the course *Theory and Practice of Software Engineering*, I helped develop and conduct the in-class exercises to teach students how to use version control systems. I also conducted in-class discussions to help students come up with interesting group projects ideas where students applied the software engineering principles taught in the course to develop real-world software applications such as Elevation-based Navigation. Sometimes students, especially those who do not have a computer science background, may struggle with technical issues in doing such projects and in-class exercises. I develop teaching materials being cognizant of such factors. For example, I created in-class exercises[1] to teach version control systems using Jupyter notebooks that students can run on browsers and do not require any installation. This ensures that students do not get off track of the main topic of interest.

Teaching experience and feedback. I am enthusiastic and energetic while teaching as it helps to keep students focused. I try to relate course topics to students' interests, knowledge, experiences, and possible future occupations. This enables students to apply what they have learned in class to solve real-world problems and helps students retain the learning. My experience of working in both industry and academic environments allows me to teach students about the real-world challenges practitioners struggle with and how academic researchers help in solving those challenges. This also helps them make informed decisions about their future career choices. To improve my pedagogical skills, I took a course called *Teaching Assistants as Tomorrow's Faculty* offered at UMass Amherst in which I learned to: (1) use evidence-based teaching practices for creating student-responsive, scalable classes, (2) create an inclusive environment to broaden participation and support diversity, and (3) use discipline-specific instructional strategies to scaffold learning. An important aspect of becoming a strong teacher is getting feedback and actively improving. When I TAed, I used the OIA Peer Review of Teaching Protocol[2] to get student feedback on my teaching. This protocol provides information on 29 parameters related to the presentation (e.g., *explained ideas or demonstrated skills with clarity, projected enthusiasm and confidence, maintained eye contact with students, related new ideas to familiar concepts*, etc.), student-instructor interaction (e.g., *responded appropriately to student disruptions, maintained students' attention*, etc.), and student responses (e.g., *students were eager to ask questions, students appeared to understand the lesson material*, etc.) skills. For each parameter, a score in the range 1–4 was assigned where '4' denotes *very evident throughout the class session*, and '1' denotes *not evident to any degree during the class session*. Considering the scores for all 29 parameters, I scored 3.24, on average. The main feedback I received was to show more enthusiam and vary my tone while responding to students' questions. I have improved my communication skills based on that feedback. I also conduct one-on-one online sessions with students to teach them about introductory computer science concepts and programming. I have tutored 43 students so far and the average rating provided by students on my teaching skills is 4.8/5.

Research mentoring. While working in the industry and pursuing my PhD, I have had multiple opportunities to mentor students and colleagues from diverse academic backgrounds on various research projects. Some of these projects even ended up being published in top software engineering publication venues. I try to keep my

[1] `https://bitbucket.org/manishmotwani/vcs-tutorial/`

[2] `https://teachingprotocol.oia.arizona.edu/`

mentees motivated by asking them to think about the impact and utility of the research problems they work on, and what aspects of the research work make them more excited. Every student has different interests, strengths, and weaknesses. I adapt my mentoring based on these aspects to make students progress and have a satisfying experience. I have mentored multiple undergraduate and graduate students from diverse CS backgrounds and even non-CS backgrounds. For example, one of my mentees started working with me on a research project as a 1st year MS student. He was more interested in developing software than identifying and solving research problems. To give him a flavor of doing research activities considering his interest in developing software, I asked him to think about *what kind of software should he create that is useful to practitioners?* When he came up with a bunch of ideas, I asked him to find out *if such software already exists?* And if it does, then *what new features can he add to his software that would make his software more preferred than existing ones?* And if not, *what are the challenges in building such software that he is proposing and how can we solve them?* While asking him such questions in the context of developing software, I also pointed him to the open-source tools and datasets released by researchers related to his ideas and asked him to read about them in their corresponding research papers. This made him develop two crucial research skills: (1) how to look for the state-of-the-art and (2) how to identify the good research problems and formulate research hypotheses.

Teaching plan. My research background, coursework, and professional experience have prepared me to teach all introductory undergraduate computer science courses such as Data Structures and Algorithms, Introduction to C/C++/Java/Python Programming, Object-Oriented Design, Databases, and Operating Systems. My expertise makes me best-suited to teach both undergraduate and graduate-level courses related to software engineering (e.g., Theory and Practice of Software Engineering, Advanced Software Engineering: Verification and Analysis, Requirements Engineering etc.) and research methodology (e.g., Empirical Research Methods). In addition to teaching existing courses, I am interested in designing new courses to teach about the challenges in developing software in open-source and closed-source settings, importance of ethics in software development, and software engineering for ML-based systems. These courses will make students learn the desired skills to go out and apply their knowledge in real-world settings. I am also interested in teaching non-technical courses or seminars to help students develop their technical writing and communication skills, and topics from humanities to develop skills on time management and managing work pressure.