Teaching Statement | Sumon Biswas

A key reason for my pursuit of an academic career is teaching, advising, and working with students, as I see the journey as a collective enhancement through a synergy between teaching and research. In a rapidly expanding field like computer science, we need to continuously create and disseminate knowledge, which my academic ancestor Edsger W. Dijkstra elicited – *"It's not the task of the University to offer what society asks for, but to give what society needs"*. Not only teaching drives my academic pursuit, I find it rewarding to guide students grasp a concept, explore new ideas, and grow as proficient computer scientists and engineers.

Teaching Experiences

I worked as a lecturer and instructed CS courses at the Bangladesh University of Business and Technology. I also gave several guest lectures such as in the graduate course – *Responsible AI: Risk Management in Data-Driven Discovery* at ISU, and CREATE SE4AI group at NSERC, Canada. During my Ph.D., I taught the following two required CS courses as a TA and mentored several undergraduate and early graduate students.

COMS 309-Software Development Practices: I was a TA for this large class comprising approximately 250 students. The course focused on building a software from ideation to release throughout the semester, in a team of four. I supervised eight project teams in each of the three semesters. During supervision, I closely guided the teams by conducting weakly meetings, explaining requirements, reviewing code in Git, creating user stories in Trello, etc. Most of my supervised teams built complex software with Android clients, Spring back-end, and MySQL database. My responsibilities also included weekly recitation lectures in a section (30 students), covering practical exercises of design patterns, client-server architecture, etc., and creating screencast videos offline on the topics.

COMS 327-Advanced Programming Techniques: This course taught advanced programming concepts using unmanaged programming languages (C and C++), including memory management, network programming, etc. I taught the course for three semesters and guided students in problem-solving, debugging, and testing. I conducted long programming sessions (4 hours), assisted the instructor in designing tests and rubrics, and graded programming assignments. Being heavy on the practical side, my teaching of programming methodologies in live sessions such as *think-aloud* coding and brainstorming, optimizing the runtime, etc, were very well-received.

Teaching Methods and Philosophy

I am a strong proponent of students' success on the subjects, within and beyond the class. I am committed to design a course and deliver materials with three key principles: **1)** For adapting to the rapidly changing field, I emphasize the *core fundamentals* and introduce state-of-the-art techniques. **2)** I explain the *applications* with relevant examples and hands-on exercises. **3)** With a goal of effective knowledge sharing and inspiring students to *learn independently*, I adopt the following educational techniques:

Active learning. Students learn more when they participate in the classroom [1]. I follow the Socratic method of teaching by asking thought-provoking questions and starting dialogues. The method might be impractical in a large class, where I plan to combine different modes such as peer instruction [2]. Furthermore, I decrease the gap between theory and practice by adopting a *learn-by-doing* strategy, e.g., creating both close- and open-ended assignments. Depending on the subject, I will design projects that require building from scratch or contributing to an existing codebase. To ensure steady progress throughout the semester, I will break large projects into incremental milestones and assign group tasks. To adapt with the new capabilities of LLMs, I will focus more on the formative assessments that encourage students to learn by co-creating with generative AI and prompt engineering. Leveraging my experience as a Program Committee (PC) member in ICSE, ASE, etc., I will simulate PC activities (e.g., reviewing, writing rebuttals) in graduate courses, rather than only reading and presenting scholarly works.

Accessible resources and classroom. As an instructor, it is my responsibility to provide appropriate resources to the students for their academic and professional development. I will share a carefully prepared syllabus containing learning objectives, logistics, schedules, and policies, e.g., integrity, accommodation, etc. Next, I will ensure the landing webpage, where I would share the digital contents (textbook, slides, software, VM). I do not strictly follow the slide or blackboard-based class but create a balance between the two. Reflecting on my experience of organizing the SPLASH'20 (virtual) and SPLASH'21 (hybrid) conferences as the *Accessibility Chair*, I will develop engaging contents with various modalities for such as closed-captioned videos, assistive notes, etc. Because of the increased enrollment in introductory CS courses, I plan contributing to the Massive Open Online Courses (MOOC) offered by the department. I will leverage my research experience of **static program analysis** and large-scale mining to create automatic grading and testing infrastructure for programming-related MOOC.

Feedback and interactions. I continuously adapt and improve content through feedbacks. I plan to take students' opinions through anonymous surveys in each quarter and take a data-driven approach for adapting based on the learning curve, difficulty of topics, and reflections. Likewise, I will provide feedback to the students on their performance, individually and as a whole. Additionally, in my classes, I combined interactive mediums such as online forums (Piazza) and IDEs, literate programming in Jupyter Notebooks, breakout rooms, etc., to encourage continuous knowledge-sharing, which have been further useful during the pandemic to scale up collaborations.

Diverse and inclusive environment. Drawing from my research experience on *Fairness*, I will ensure equitable treatment of students irrespective of their background and abilities. Throughout my team-projects supervision, I proactively reached out and talked to the introverted students during office hours. I adopted the pedagogy of instructing students to build software features that is culturally relevant they care about. I did not evaluate only the correctness of the solution but also students' skills of code comprehension, documentation, and communication, which are essential in real life. Furthermore, as academia can often be demanding, I ensure a student-centered approach through empathy and support so that students can grow by learning from their failures [3], such as by allowing to revisit missed exam questions to make up points, exclusion of one low-scoring homework, etc.

Advising Experience and Approach

I have mentored 13 students at CMU and ISU, including undergraduate, NSF REU, master's and early Ph.D. students in different research projects, which resulted in impactful contributions including five conference publications in ICSE and FSE. I also mentored for a Senior Design project, Student Mentoring Workshop at ICSE'23, and K-12 outreach program. I always ensured the freedom of expression of my mentees so that they are comfortable in asking questions. My advising philosophy is intertwined with three main approaches. **First**, I am flexible with students and mentees since *each person is different* with their own way of thinking and working style. **Second**, I always encourage students to motivate their work with sufficient real-world examples and keep the *big picture* in mind. **Third**, starting with a hands-on advising style (i.e., finding problems and guiding through building solutions), I will transition to a hands-off approach as the students develop the skills, sense of ownership, and *grow as independent researchers*. Looking ahead, as a faculty, I envision a research lab that fosters innovation, curiosity, and growth. I will establish an environment where the students can collaborate among themselves. I will hold a weekly group meeting, individual meeting, and other discussions as necessary. In the meetings, I will discuss the progress and next steps, and guide them to make an impact towards the overarching goal of innovation.

Teaching Interests

I am primarily interested in teaching *software engineering* and *programming languages* related courses that cover topics such as software design and architecture, program analysis, mining software repositories, software evolution, modularity, lambda calculus, compilers, formal methods and verification. My research and doctoral study qualified me to teach both undergraduate and graduate-level courses on the topics. I would also be interested to teach or co-teach large classes including *machine learning* and foundational courses such as *algorithms*, *object-oriented design*, *data science programming*, for both CS and non-CS majors. Additionally, I am also keen to develop the following graduate-level courses.

1) Software engineering (SE) for AI/ML systems: Because of the *under-specified*, uncertain, and data-driven nature, AI/ML-based systems require careful engineering around modeling and architecture (e.g., ML pipeline), deployment (e.g., MLOps), quality assurance (e.g., ML testing), and maintenance (e.g., data drift, technical debt, feedback loop). The contents will go beyond the algorithms taught in ML classes and target future *software engineers* and *data scientists* building AI/ML systems in production. My research and working experience will facilitate to develop the course with core concepts of SE and AI, recent developments in the field, and a real-world project.

2) Responsible AI Design: Several ethical and safety risks raised by AI applications are major challenges for the next generation. Therefore, I envision the course along four dimensions — fairness, robustness, safety, and explainability. I plan a hybrid format class (lecture and seminar), with a capstone research-oriented project that will teach responsible design (algorithmic and engineering), novel mitigations, and interdisciplinary thinking.

References

[1] S. Freeman, S. L. Eddy, M. McDonough, M. K. Smith, N. Okoroafor, H. Jordt, and M. P. Wenderoth, "Active learning increases student performance in science, engineering, and mathematics," *Proceedings of the national academy of sciences*, 2014.

[2] E. Mazur and R. C. Hilborn, "Peer instruction: A user's manual," 1997.

[3] C. S. Dweck, *Mindset: The new psychology of success*. Random house, 2006.