Learning computer science (CS) can be both exciting and challenging. Perhaps, a major problem CS students face is *information overload*. With so much to learn and so much of information (and misinformation) easily available, learning CS concepts, staying in the *flow channel* (neither getting bored nor panicking), can be extremely difficult.

As a teacher and mentor, my key objective is to show students that learning CS is about grasping a relatively few fundamental ideas and principles that underlie a plethora of application-level technologies. For example, in a *programming* course, my main objective will be to help students learn the art of breaking down a solution to a problem into a set of *modules* and *steps* before teaching the syntactic nuances of a specific language.

Interests. With a strong CS background, I am qualified to teach a variety of foundational and advanced CS courses. I prefer teaching courses aligned with my research. However, many courses are within that scope because of my broad research interests, including courses related to Artificial Intelligence, Data Science, and Social Computing.

- The Bachelor and Master programs in CS at TU Delft offer several courses I am interested in teaching, including *AI Techniques*, *Logic and Reasoning*, *Data Mining*, *Patten Recognition*, and *Research Methodology for Data Science*. In addition, courses such as *Social Signal Processing*, *Intelligent Decision Making*, and *Crowd Computing* are close my research interests and I will be happy to contribute to these courses.
- Further, depending on the need and opportunities, I am interested in developing new courses such as *Engineering Intelligent Agents and Multiagent Systems*, *Values and Ethics for Intelligent Agents*, and *Explainable AI*.

Experience. I have been involved a variety of teaching-related activities since my early graduate school days. This experience has been both enjoyable and a great learning experience.

- *Instruction.* During my tenure as an Assistant Professor at RIT, I taught (1) a graduate course (thrice) on *Data Science*, which includes foundations of data mining techniques and their applications for Software Analytics. (2) an undergraduate course (twice) on *Engineering Secure Software*, which includes foundations of security and privacy for Software Engineering students. Further, at NC State, I served as a *teaching assistant* for three courses: *service-oriented computing* (four times), *social computing* (twice), and *graph theory* (once).
- *Mentoring.* I am advising one PhD, two MS thesis, and two MS capstone students at RIT. In addition, I advise three–four BS students each summer as part of the Research Experience for Undergrads program I organize.
- *Curriculum development.* I am a member of the curriculum committee for an upcoming MS in Data Science program at RIT. This was a very challenging task considering the diversity of students expected to join the program (those from CS, Statistics, as well as Humanities backgrounds). As part of the committee, I researched several competing programs. We proposed a curriculum that includes a set of core courses (e.g., Foundations of Data Science, which I developed), bridge courses (e.g., Introduction to Programming), and advanced electives (e.g., Deep Learning), suggesting different paths to students depending on their backgrounds. The curriculum was later approved by the University board. I am also involved in reviewing student applications to the program.

Methods. First, I incorporate *conceptual modeling*, an idea I explore in research, in teaching, too. Conceptual modeling advocates that a software (to be implemented) be understood via cognitive concepts, focusing not only on *what* and *how*, but more importantly on *why*. Following this intuition, in a data mining class, for example, I will not only describe the algorithmic steps of an approach, but also describe why the approach is designed a certain way.

Second, balancing theory and practice as an important aspect of teaching CS. I design my courses to educate students on the foundational concepts as well as to train them on using those concepts in concrete applications. This will both develop students' intellect and equip them with the skills their careers demand. I will make sure that the training exercises reinforce, but also be complementary to the material I teach in lectures.

Third, higher-education institutes are culturally diverse. Whereas diversity enhances students' experience, it makes teaching a challenging task. For example, I believe in engaging students via dialogue during a class. However, this may not be easy: whereas students from western countries tend to be outspoken, Asian students tend to be shy (some might even consider disagreeing with the instructor as disrespectful). A solution in this case can be to encourage students to first talk to their peers and then to the instructor. I have experienced many such cultural differences, personally. I respect such differences and will do my best to accommodate for them in my classes.

Fourth, students often do not have a bigger picture of CS careers, e.g., industry vs. academic jobs or development vs. testing in the industry. I am willing to help students understand various career options and prepare accordingly. For example, a student wishing to pursue a development job in the industry must master programming, whereas publishing a paper (or even attempting) can add a great deal of value to a student wishing to pursue a research career.

Finally, motivation is important for learning. I imagine that a student does well in a subject not just because of dedication, but also because he or she is passionate about the subject. To inspire students in my courses, I demonstrate how the concepts they learn could lead to applications that benefit millions of users. Similarly, in advanced courses, I will invite researchers to present cutting-edge works relevant to the course to inspire students about research careers.