The key challenge of teaching in my view is being mindful of and reconciling different goals in a coherent way: A good class should communicate the clarity of thought of the professor. A good class should give students the practical experience necessary to implement the concepts in the class without subsequent hand-holding. From a social perspective, teaching should ensure that bright students benefit from the experience of the professor in furthering their career in research, engineering, or teaching. Furthermore, I believe one of the underlying duties of an academic institution is to provide opportunities and support for underrepresented sections of society.

During my Ph.D., I was head TA for Percy Liang’s graduate introduction to AI course, with an enrollment of around 200 students. My greatest takeaway from this experience was the importance of a simple framework for explaining complex concepts clearly. For instance, online learning with various loss functions allows an introduction to machine learning which bypasses complex optimization algorithms, as well as introduces a number of classifier formulations (perceptron / logistic regression / max-margin classifiers) as intuitive variations on a theme (perceptron loss / logistic loss / hinge loss). Similarly, an exposition of Markov Networks in the same context as Constraint Satisfaction Problems is an invaluable complement to and motivation for the probabilistic definitions. I have done my best to emulate this clarity in my office hours and section lectures.

I believe there is great value in even simple habits that promote clarity: machine learning is notorious for having multiple technical terms for the same concept – minimally, one from the statistics community and one from the machine learning community. Simply identifying and clustering these terms is invaluable to a new student. Relatedly, most concepts we teach in AI appear both as a mathematical formalism, and a block of concrete code. The interleaving of equations and live coding in Percy’s class has been a great modern technique in solidifying a student’s understanding of the material.

The other important aspect of a good class is the ability for students to take the knowledge they gained and apply it to future tasks without the hand-holding of the classroom setting. In my opinion, this is most naturally done through substantial hands-on projects. The natural language processing class with Chris Manning does an exceptional job in accompanying each of the technologies taught in class with a relatively free-form project. Some framework code is provided for dataset management and coding, but the student is left free to implement a high-level interface through any means they can. While TAing this class, I got to both develop a new assignment for coreference resolution, and see firsthand both the difficulty and value of these project-based assignments. Many important classes of mistakes either do not surface, or are explicitly pointed out when implementing isolated functions in a conventional programming assignment. For example, unknown words in language modeling, or unary closures in parsing. Furthermore, the type of error analysis which is required to get good accuracy on machine learning tasks is far more naturally taught in the context of open-ended projects.

In my teaching, I intend to pursue both goals: teaching with simplicity and clarity to convey the underlying concepts in the course, and to incorporate practical projects which allow students to become comfortable with the unassisted implementation of the technology. Given that students
have limited time, I believe the purpose of lectures is to convey the professor’s understanding of the topic, sections should be interactive and ensure students’ understanding of the topic, and homework should (in most cases) be practical assignments to ensure a student’s ability to implement the topic semi-independently.

To conclude, I’d like to highlight the social context of teaching. A university education is the gateway to the high-tech workforce, and this places a certain responsibility on the university to shape the demographics of that workforce. This past year, Stanford started a free 2-week outreach for rising high-school sophomores, for which I served both as the chair of the financial committee and a co-teacher of one of the hands-on projects (2–3 hours per day). The goal of the program is to expose underrepresented minorities to CS through a more exciting use-case than the conventional intro class, and to teach them some of the skills that will help them if they do choose a career in CS or AI.

The most salient “minority” in computer science are women, and this was the target of the program’s first year. However, this is certainly not the only demographic we could and should be targeting. Ethnically, African Americans are underrepresented in computer science. Socioeconomically, the working class is underrepresented as well. As a professor, I feel I have the ability and therefore the responsibility to devote some of my effort to outreach programs to encourage more women, more ethnic minorities, and more socioeconomic diversity in our community.