Teaching Statement

Pratik Chaudhari

Email: pratikac@seas.upenn.edu

I will first describe two courses that I have developed over the past six years, and then discuss how my teaching and mentoring methodology have evolved over time.

1 New curricula

ESE 546: Principles of Deep Learning I started developing this course after I joined in Fall 2019. A total 620 students (13.5% undergraduates, ∼70% Master's and ∼10% doctoral students) from across the schools of Engineering, Arts and Sciences, Wharton and Medicine have taken this course across its 6 offerings. This course begins from neural architectures, progresses through non-convex optimization, some mathematical topics like learning theory and the energy landscape of deep networks, and ends with variational inference and generative models.

This course has received good evaluations: 3.22 (out of 4) instructor quality, 3.08 course quality, and 3.5 course difficulty, on average. The median instructor quality is 4 (excellent) in the past four years. Many called it the best course they ever took. Many return to the course as TAs the year after. See Fig. 1 for details.

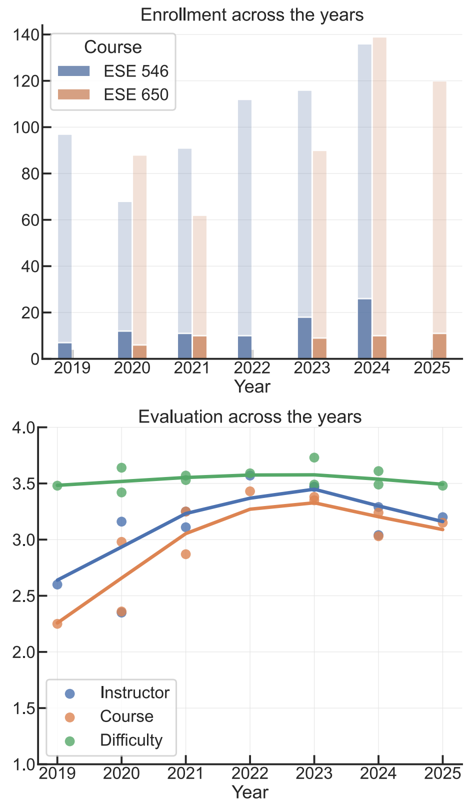

Figure 1: Top: Enrollment has gone up over the years, both classes are capped at 144. The number of undergraduate students (bold bars, 11.8% on average) is going up slowly as I have adapted ESE 546 over the years to target 4th-year undergraduates. See Section 2. Bottom: Both courses (data combined) are consistently rated highly on instructor quality and course quality. Both courses are rated above 3.5 in terms of their difficulty. My evaluations have gone up considerably after the pandemic subsided (post Spring 2021).

Selected student comments:

- 'goated",
- "An awesome course with a high quality...requires a lot of work",
- "The instructor was one of the best I have studied under",
- "Dr. Chaudhari is one of the best instructors that I have come across in all of my many years of education. . . This was by far, the best course that I have taken in recent memory, and I only wish the I could learn more from this excellent professor.",
- "...teaching methodology that went into this class are state of the art. Late [office] hours, accessible lecture notes, fantastic test design, homework structure...fairly high workload [but] the structure of this class managed to remove all of the worst "high stress" moments... I would highly recommend this course."

Deep learning is a relatively new field and lacks a good textbook. To fill this gap, I prepared detailed notes (pratikac.github.io/pub/24_ese546.pdf). These notes have been widely appreciated by both students and faculty, even by students from other universities who do not take the course. I have also given guest lectures on this material at Brigham Young University and Johns Hopkins University. I have talked about material from these notes at a number of venues for high-school and early undergraduate students (e.g., ESE 111, Penn Rising Scholar Success Academy, College of New Jersey, PennApps, and GRASP tours for high-school students and other visitors), summer/winter schools on deep learning, etc.

Online version of ESE 546 I also created an online version of this course that runs as one of the core courses in the Master of Computer and Information Technology (MCIT) program. The content of the course is near identical to that of the in-person offering, except that a few mathematical topics are treated as optional. As I tailored the content to the online audience, I had to think deeply about course design, and edited/re-edited the content extensively to create high-quality lecture videos. I conducted technical interviews with experts in computer vision, neuroscience, cloud computing. The most difficult aspect was developing robust automatic grading infrastructure that enables the class to scale, but keeps intact the complex, slightly open-ended homework problems that are a hallmark of the in-person offering at Penn. This course ran in Fall 2024 with 57 students from across the world, about 439 students have taken the Coursera version.

ESE 650: Learning in Robotics I took over this course in Spring 2020 and have revamped almost all the material since. This is an advanced robotics course and covers state-estimation, simultaneous localization and mapping (SLAM), dynamic programming, LQR/HJB and reinforcement learning. A total of 499 students (9.2% undergraduates, $\sim$70% Master's students and $\sim$20% doctoral students) have taken this course across 5 offerings. These students tend to be from the school of Engineering (typically ESE, CIS and MEAM).

This course has also received good evaluations: 3.03 (out of 4) instructor quality, 2.95 course quality and 3.60 course difficulty, on average. Many again called it the best course they took at Penn. See Fig. 1 for details.

Selected student comments:

- "Excellent. He's a genius."
- "Excellent course. Excellent lecture notes. Clear and comprehensive."
- "I really enjoyed the lecture and I believe it is one of the best lectures that I have taken out of the courses I have taken."
- "Difficult course but learned a lot!"
- "God level of teaching! Should be the benchmark of teaching at Penn."

This is covers a lot of ground from control theory, computer vision, and machine learning. Although there are numerous excellent textbooks on each of these topics, I have created my own notes that make a coherent argument as to how and why these fields come together in robotics (pratikac.github.io/pub/25_ese650.pdf). The students really appreciate this specific aspect of the course, and these notes have been widely used both inside and outside Penn. I have given invited lectures on this material in MEAM 520, ESE 615, and ESE 619 and tutorials at Amazon and Vanguard.

I developed material on reinforcement learning for a popular book named "Dive into Deep Learning" (d2l.ai) that is now published by Cambridge University Press. This is a unique textbook. It contains a textual narrative of a very wide variety of topics in deep learning together with extensive Python code for students to follow along. It is completely open-source and content is developed in public on Github by researchers in Amazon. This was done as a part of the "Deep Learning University" project within Amazon whose goal was to educate software engineers across the company in these new tools. This textbook has had a wide impact. It has been adopted by more than 500 universities across 70 countries in their coursework.

2 Evolution of my teaching methodology

In Fall 2019, I was guided by prof. Santosh Venkatesh who helped me craft my pedagogical style. I also worked with Dr. Bruce Lenthal, who directs the Center for Teaching and Learning (CTL) and took part in a semester-long Structured, Active, In-class Learning (SAIL) Seminar in Spring 2022. These interactions have helped me sharpen my goals as an Educator. I will next describe these below with a few examples.

Building foundational knowledge v.s. mere know-how Both deep learning and robotics are exceptionally broad and popular fields but they do not have a well-established educational curriculum yet. As a result, the student cohort in my courses tends to be very broad in its skill-set, interests and aptitude. To bring everyone on the same page, I have introductory lectures at the beginning of each module (e.g., on linear regression, convex functions, entropy, 2D rotation matrices, Dijkstra's algorithm etc.). Almost everyone has at least passing familiarity with these concepts. This helps bring everyone on the same page. And it gives the students a launchpad for these mathematically sophisticated courses, e.g., I introduce variational calculus, steady-state distribution of continuous-time Markov chains, state-of-the-art research on understanding generalization of deep networks in ESE 546 and concepts like the HJB equation, control-estimation duality, function approximation in Q-Learning, etc. in ESE 650. Students appreciate this technical depth and the ability to see how these concepts connect with each other in a single course.

My exams are designed to test breadth and creativity while homeworks are designed to test depth and exhaustive application of concepts. I set difficult homeworks that challenge the understanding of material gleaned from the lectures. Homeworks have about 50% programming assignments and about 50% mathematical questions. Programming questions are implemented by students from scratch—without much boilerplate code. In both deep learning and robotics, our theoretical understanding is pretty far from practice. E.g., seeing the acceleration of Nesterov's method is difficult even on a MNIST digit classification problem. Implementing SLAM using LiDAR data or a UKF on a drone is difficult because of concepts that one usually does not see in a textbook (sensors have different latencies, averages of quaternion errors have to be calculated correctly). With a few careful tweaks of MPC, one can simulate a humanoid walking in about 15 seconds—something that will take multiple days using the best RL algorithms. My homeworks are designed to showcase these lacunae using experiments on real data. They force students to think deeply about the course content and internalize both the virtues and failings of these ideas. This aspect of the homeworks has been widely appreciated by the students.

Adapting the course to undergraduate students Both ESE 546 and ESE 650 were initially designed as graduate-level courses. But over the years there has been a lot of interest in these courses among undergraduate students also. To support this, I have adapted certain parts of the course, e.g., incorporated more programming assignments over the years, expanded upon lecture notes and homework questions to ensure there is essentially no barrier to entry. All homeworks have some extra credit points to leave some wiggle room for minor mistakes. Course credit is spread quite evenly across different types of deliverables to ensure that a poor score on one of them does not excessively affect the grade.

My overall strategy is to keep track of the undergraduate students in the course and pay attention to their performance. I also have the TAs do the same thing. I require explicit permission of the instructor for undergraduate enrollment, so this is easy even in a large class. I personally encourage undergraduates to attend office hours (and many do). I am also empathetic to the heavy course-load that undergraduate students tend to take, e.g., while allowing extensions of deadlines on deliverables. Each semester, I have a few undergraduate TAs. This helps obtain a different perspective on the content and logistics.

Many undergraduate students prefer to learn via coding. I encourage this by doing live code demos in some lectures, many of the recitations, and pointing students to interesting software libraries. Course projects and any sort of extra-curricular tinkering typically gets the undergraduate students very interested in these ideas. I encourage this a lot in the office hours, and engage with them more extensively during the project. I also keep a careful tab on the scope and feasibility of undergraduate projects. In fact, in many cases, undergrads have done ambitious and exceptional projects (e.g., implementing supervised-learning based LLM alignment, automatically summarizing the highlights reel of a football match, etc.).

The number of senior and sub-matriculating undergraduate students who take my courses is increasing steadily. In some cases, I have personally mentored 2nd-year undergraduates who took this advanced coursework—they finished it with flying colors. I have interacted extensively with undergraduate students in a research capacity via on-campus research programs, see Section 3 on the next page. This has further improved my understanding of undergraduate students in my courses.

Creating an engaging learning environment Before every lecture, I ask students to write a 1–2 paragraph summary of the previous lecture. This keeps all the students engaged with the material throughout the semester. I often invite students to attempt to answer the question asked by someone else during the lecture. Through this, my goal is to make students comfortable enough to communicate their thought process and make mistakes. While lecturing, I am very open about our poor understanding of some of the topics, give examples of my hunches that proved correct/incorrect, and postulate hypotheses that should be explored. Because of how rich these two fields are, I have been able to inspire curiosity and suggest small experiments/course projects to the students to explore their questions. These courses are a canvas of ideas to actively play with.

Exposing students to research by inspiring their curiosity In both courses, students do a course project where they are free to explore a topic of their choice. I treat this as a mini-Independent Study and students receive a lot of help from the TAs and me on these projects. I make the project proposals completely public, this way everyone can see what everyone else is working on.

For ESE 650, I have procured hardware (Intel RealSense cameras and table-top robot manipulators) for students to do their course projects. Many course projects are quite sophisticated, e.g., a generative model for gene mutations; understanding the manifold of input images that is induced by a trained model; visual-inertial odometry and planning stack on an RC car, sophisticated VLM–LLM-based systems in recent times. Almost all the Master's theses that I have supervised came from students I interacted with through their course projects. Many course projects have become full publications, some with me and many others with other faculty advisors.

Continuous feedback I collect statistics about the time spent on, and the perceived difficulty of each homework, explicitly inquiring the students after the lecture and the office hour. This helps me adapt the course rapidly during the semester, and guide the students more individually even in these large classes (e.g., suggesting background books, advanced papers, pointing to new software libraries etc.).

In the first few years of teaching these courses, at the end of the semester I performed a statistical analysis of student performance (e.g., which homework problems were difficult for students in the lower percentile, correlation of exam performance with specific homework problems etc.). This analysis was useful to adapt the material (e.g., very few students grasped MCMC in the deep learning course so it was removed, many students struggled with NeRFs and RL so the problems were simplified, etc.)

Preparing students for the modern workplace Over the years, I have provided more than $125,000 worth credits to the students from the AWS Educate program, initially through my own contacts but later on with help from Penn ISC. Students can build much more ambitious projects using these compute credits. I have designed recitation sessions to go beyond the syllabus and provide a hands-on introduction to topics such as object detection, training large neural networks, standard industry software such as ROS, etc. This is critical to students getting jobs/internships and them being successful beyond the classroom. Exams in my courses have questions that ask students to think through a design problem (e.g., a machine learning system that is updated continually as new data is collected, imputation of missing data, selecting the sensor configuration of an autonomous forklift, etc.). Many students have told me that my exams/homeworks are the reason they did well at a job interview and got an offer.

3 Mentoring Philosophy

I am currently the primary doctoral advisor of 13 students (5 ESE, 6 CIS, 1 AMCS, and 1 Physics). Some of these students are co-advised with Deep Jariwala (ESE), Christos Davatzikos (Radiology/ESE), Jim Gee (Radiology), Vijay Balasubramanian (Physics) and Vijay Kumar (MEAM). Four students from my group have obtained their Ph.D. degrees so far. I was the primary advisor for all four, and the only advisor for three of them. These students are: Yansong Gao (2023; AMCS; now at ByteDance), Rongguang Wang (2024; ESE; now at Oracle), Rahul Ramesh (2025; CIS; now at Waymo) and Jialin Mao (2025; AMCS; now at GE Healthcare).

I have closely supervised 11 Master's students and 6 undergraduate students on long-term (at least one year) research projects. All these students worked on their individual research projects and were directly mentored by me. I have served on 33 doctoral committees (across ESE, CIS, MEAM, AMCS, Neuroscience, and some also from outside Penn such as the University of Padova, Johns Hopkins and Plaksha University). I have mentored some of these students closely and have also co-authored publications with some. I have served on 19 qualifying exam committees in CIS.

Students I have mentored have gone on to prestigious jobs and internships in companies like Google, Waymo, Meta, Amazon, Nvidia, SpaceX, Tesla, Skydio, Nuro, Zoox, etc., government agencies (ARL), and to graduate programs/postdoc positions at MIT, Berkeley, EPFL, Harvard, Princeton, Yale, Oxford, Brown, Cambridge, UT Austin, USC, UIUC, Michigan, MILA Montreal, Purdue, Penn, etc. Some have received prestigious fellowships (NSF GRFP).

I will next discuss some aspects of my research mentorship.

Student-centered mentoring My emphasis is on developing good researchers with the research project acting more as the scaffolding. I work closely with students (about 1 hr/week with each doctoral student, and about 0.5 hr/week for non-doctoral students) and also derive and code things myself to share with the students. This close partnership gives me a lot of opportunity to craft how they think—and share the sheer joy of finding things out. It also keeps me in touch with the details of the problems, all the way from patching the flight controller on a drone, to working out abstract concepts in information geometry. This has made me a better researcher, and a better advisor. Over the years, I have learned how to ask the right questions in these interactions. And how to push the students to think more and more deeply for themselves before I share my insights.

I strongly emphasize reading books ("cover to cover") over papers at the beginning of a Ph.D. I believe it is important to know the classical way to think about a problem and ascertain for oneself via calculations and experiments, the merits and failings of classical ideas. Students in my group inevitably take a lot of coursework in mathematics. I believe that regardless of whether one is theoretically or empirically inclined, a good scientist can rigorously think through an idea and deduce things—rather than relying on intuition or on what what they've read in publications. This "thinking before doing" strategy pays off immensely over time. Even if the students are a bit slow to get started, they become very good at picking the right problems and the right approaches to employ. In my opinion, the first publication in the doctoral program is critical to claiming intellectual ownership and developing a perspective upon a thesis topic. This is why I encourage all students to work on their first manuscript without collaborating with any other lab member. One cannot really teach "good" research taste but this mentoring approach at least enables everyone to discover theirs.

My goal is to develop researchers with exceptional technical depth, the wisdom to know how and why a field has evolved in a certain way, and the vision to imagine how it will evolve in the future. This is especially important in a field like deep learning where short-fuse arXiv submissions and non-peer reviewed dissemination of ideas can create a false impression of progress. A number of my doctoral and Master's students had never done research before they joined my group. And yet they have successfully published in premier venues now.

Promoting independence My second goal is to prepare students to chart their own agenda and become leading researchers/academics after they graduate. After they have their first result and thereby some confidence in their own ability, I encourage them to craft the second project on their own, with feedback from me. My goal is to have them (a) develop a hypothesis based on their past theoretical and empirical experience; (b) craft the right experiment or calculation to investigate it; (c) summarize often inconclusive findings; before (d) reformulating the hypothesis. Different students take different amounts of time to appreciate this cycle and successfully go through it. I emphasize a "write-first mentality" and encourage diligent documentation as they learn to do research systematically like this.

Nurturing a vibrant and honest intellectual environment I believe that a researcher should be able to address questions of varying levels of difficulty and naivete, communicate their work, and—at the end of the day—inspire others to think about it. My group meetings are usually driven by students asking questions (even early undergraduates). Addressing questions requires my students to, e.g., explain the basic idea of statistical learning theory or information geometry to a first year undergraduate, or for a computer scientist to explain to a mathematician why data augmentation is usually done on the CPU. All ongoing research progress within the group is public. Every student can look at the ongoing experiments/calculations of other students. This openness has been extremely fruitful. Many powerful ideas have come from students getting interested in each other's projects. There is very little intra-group competition because students work on sufficiently different problems, and are encouraged to share credit if they still end up overlapping.

3.1 Undergraduate Research

Over the past six years, I have closely worked with 33 undergraduates through on-campus research programs (1st or 2nd year students from PURM, VIPER, CURF, Work Study, Wharton Summer Research, Google Explore Undergraduate Research, First-year exposure to STEM, NSF REU, University Scholar Program). I personally work with these undergraduate students.

My primary goal while mentoring early undergraduates is to give them a taste of research, put them into the right mindset for future coursework/research and give them the confidence to explore their curiosity. To achieve this goal, I design about 3–4 weeks of a crash course on deep learning for the summer students based on the "Dive Into Deep Learning" (d2l.ai) textbook, and the "missing semester of your CS education" (missing.csail.mit.edu) course. Sometimes, the more advanced students will study my graduate course material on deep learning or robotics. I meet undergraduate students on a 1:1 basis, have them take part and present in group meetings, write reports and document their experiments. Over the course of the summer, I have students pick specific projects that catch their interests in a more-or-less choose your own adventure style, e.g., tracking birds using computer vision in an aviary, learning how to code FPGAs for image processing, assembling a drone from scratch and coding up navigation and perception algorithms, exploring different optimization algorithms in deep learning and domain adaptation, and even advanced projects like PAC-Bayes bounds and RLHF in LLMs.

I have found these interactions extremely enjoyable. Many of these students continue to attend my group meetings beyond their specific summer project, and end up picking a problem to work on during the semester. I have worked with some students for almost their entire 4 years at Penn.

In addition to this, I have also worked with 5 undergraduates on Independent Study projects and 36 students on year-long Senior Design projects. Projects ranged from hardware-based ones (e.g., active perception using a drone, oil spill containment using multiple boats, cleaning up litter using a drone) to software-based ones (e.g., estimating breast density using MRI data, generating novel Jazz solos), and theoretical ones (geometry of the predictions of deep networks, relationships between learning tasks). Almost all of the Senior Design projects have won awards, some were also published at student conference venues.