Teaching Philosophy

Rajee Ganesan, PhD Candidate, Carnegie Mellon

“The classroom with all its limitations remains a location of possibility. In that field of possibility we have the opportunity to labour for freedom, to demand of ourselves and our comrades, an openness of mind and heart that allows us to face reality even as we collectively imagine ways to move beyond boundaries, to transgress. This is education as the practice of freedom. (hooks 1994: 207)”

My commitment to this approach has been developed and tested in practice. Over five semesters teaching BIOF555 and BIOF556 — two courses offered through the NIH Foundation for the Advancement of Sciences focused on the intricacies of single-cell sequencing — I advanced from teaching assistant to sole instructor, taking on full responsibility for course conceptualization, design, and ongoing improvement. This course was conducted remotely with over 60 students, ranging from undergraduates to junior faculty members. I have also served as a teaching assistant for three semesters for BIO3133: Neurobiology of Disease at Carnegie Mellon University, where I supported the learning experience of over 280 students. In Fall 2026, I will serve as Recitation Instructor for Genetics, Guest Instructor for Genomics and Epigenetics of the Brain, and return as sole Instructor for BIOF555 and BIOF556.

Instructional Framework: “See one, do one, teach one”

My teaching philosophy is organized around the "see one, do one, teach one" framework — a principle originating in medical education, and one that I have adopted from instructors that I have learned from myself. Rooted in well-established principles from learning science, including retrieval practice, spaced repetition, and the concept of the zone of proximal development, this framework moves students through three distinct but interconnected phases: structured introductions to new material, individual application, and the extension that comes from having to explain and defend ideas to others. In a field like computational biology, where students must not only understand biological systems but develop the technical fluency to interrogate them, this progression from exposure to application to articulation directly reflects how effective science is conducted.

In the "see one" phase, students engage with lectures as a site of structured, foundational learning. My goal as an instructor is not simply to deliver content, but to make the architecture of the field visible: to show students not just what techniques do, but why they were developed, what problems they solve, and where they are limited. I integrate active learning strategies throughout lectures, including structured discussion, case studies drawn from recent literature — for example, breaking down and evaluating the Seurat toolkit in BIOF555 — and real-time surveys that allow me to adjust pacing and gauge misconceptions.

In the "do one" phase, students move from reception to application, working through problems that require them to extend and use content from lecture in practice. In BIOF555 and 556, this takes the form of weekly practicals assigned at the outset of each week, where students integrate code, tools, and methods introduced in lecture using their own dataset, creating an opportunity for scientific investigation within the classroom. Students are expected to work through interpretations, and common errors collaboratively with peers first. Instructor office hours are then offered at the end of the week to consolidate understanding and address any outstanding questions. This approach not only builds content knowledge, but also the resilience that students need to function as independent learners and researchers.

In the "teach one" phase, students consolidate understanding through cooperative learning. In BIOF555 and 556, this is implemented with structured weekly discussion posts, in which students select a topic from the course material and publish a written explanation for their peers. The prompts are designed to require synthesis rather than summary: in one discussion, for instance, students are asked to evaluate computational methods for bias detection in single-cell data: choosing a tool or case study from the literature, explaining the problem it addresses, and developing a reasoned argument for how they would apply it in their own research. Students then engage critically with each other's posts, offering alternative framings, limitations, and examples from their own datasets and research.

Throughout each module, I encourage students to ask how the material connects to their own research questions and scientific interests. My aim is never to produce students who can execute a workflow, but students who understand why a given approach exists, when it is appropriate, and where it breaks down. By the end of BIOF556, for example, students are able to not only apply best practices in single-cell RNA-seq analysis with their own datasets, but to evaluate competing methods critically, design their own analysis pipelines, and draw biologically meaningful conclusions from complex data.

To teach well is to believe our work is not merely to share information, but to share in the intellectual and spiritual growth of our students. The frameworks of structured content delivery, applicational practice, and articulation in discussions are centered in this conviction, and an attempt to support a classroom where every student, regardless of racial, socioeconomic, gender, or sexual identity, can fully engage in the intellectual work of science. My goal is not only to build technical proficiency in interdisciplinary biology, but to develop individuals who are rigorous, reflective, and genuinely curious — and who carry forward an understanding of what it means to learn and to teach with care.

Teaching ambitions

My educational training is in quantitative biology, genomics, data science and statistics, and neuroscience — a breadth that reflects the interdisciplinary nature of computational biology and directly shapes how I think about teaching. I am prepared and eager to educate across undergraduate, graduate, and postgraduate levels on topics ranging from genetics, evolution, neuroscience, and science communication. I am particularly interested in developing courses that ask students to synthesize across disciplines — to bring what they know from biology, computer science, statistics, and physics to tackle problems that none of those fields can solve alone.

One of the courses I am enthusiastic to develop is designed for entering students, before they have the technical prerequisites for hands-on computational work. Rather than beginning with code or methods, the course centers on the scientific literature and concepts themselves. Each week, a lecture models how to read and evaluate a research article — breaking down experiments, why the methodological choices matter, and where the work sits in the broader field. Students then work collaboratively to apply this to a recent paper of their choosing and present their analysis to peers in a weekly recitation. A second course, designed for students further along in their undergraduate training, shifts focus to the execution of computational analyses of functional genomics experiments, culminating in an independent pipeline project presented to the class.

The rapid integration of artificial intelligence is fundamentally changing how computational biology is practiced: from how analyses are designed and executed to how results are interpreted and communicated. Rather than treating AI as a threat to academic integrity, I approach it as a tool to be evaluated critically, understood mechanistically, and used deliberately — the same framework I apply to any method in the course. In practice, this means incorporating AI tools directly into coursework: asking students to use a large language model to generate an analysis pipeline, then identify where it fails, why it fails, and what biological knowledge is required to catch the error. In a field where AI-assisted analysis is becoming standard, the most important skill a computational biologist can develop is not the ability to prompt a model, but the critical fluency to know when to trust it — and when not to.