Dr. Alex J. Mercer

Teaching Statement

Department of Computer Science · Stanford University
amercer@stanford.edu · alexmercer.github.io · +1 (650) 555-0199

Teaching Philosophy: My teaching philosophy is grounded in active learning, cognitive scaffolding, and inclusive mentorship. I aim to demystify complex computer science concepts by building intuitive, hands-on bridges between abstract mathematical theory and real-world system implementations. As an educator, my goal is to equip students from diverse backgrounds with the analytical tools and confidence to become independent thinkers and future leaders in technology.

1 Teaching Philosophy

Teaching is not merely the transmission of facts; it is the construction of an environment where students learn how to formulate and test hypotheses. In my classroom, I implement three core pedagogical principles that guide students from novice consumers of technology to expert creators:

1.1 Active Learning and Peer Instruction

Traditional lectures can encourage passive listening. To counter this, I integrate peer instruction into every class. I break lectures into 15-minute segments, each followed by an anonymous conceptual question (a "concept test"). Students first vote individually, then discuss their reasoning in small groups before voting again. This peer instruction forces students to articulate their mental models, exposes misconceptions, and consistently improves the rate of correct answers by over 25%.

1.2 Cognitive Scaffolding in Project-Based Learning

In computer science, students often struggle with the gap between small homework exercises and large-scale systems. To bridge this, I design structured, modular projects that build incrementally. For example, when building a compiler, students first implement a lexical analyzer, then a parser, and finally a code generator. Each milestone is supported by automated testing suites that provide immediate feedback. By focusing on clean interface design, students learn to manage complexity without becoming overwhelmed.

1.3 Intuition-First Instruction

Before introducing formal mathematical abstractions, I focus on building intuition. When teaching natural language processing, instead of starting with the loss function of a Transformer, I begin with a visual simulation showing how attention weights shift over a sentence. Once students visualize how words interact, the mathematical formulation of self-attention becomes intuitive. Prioritizing conceptual models first lowers the barrier to entry and keeps students engaged.

2 Teaching Experience

Throughout my academic career, I have taught students across the undergraduate and graduate spectrum, ranging from co-instructing large introductory classes to designing specialized research seminars.

2.1 Teaching Portfolio Summary

Below is a summary of my teaching assignments, highlighting my roles, class sizes, and student evaluation scores:

Course	Title	Role	Term	Instructor Rating
CS 106B	Programming Abstractions	Co-Instructor	Fall 2024	4.85 / 5.00
CS 224N	Natural Language Processing	Head TA	Winter 2024	4.90 / 5.00
CS 294S	AI Ethics & Large Language Models	Co-Developer	Fall 2023	4.92 / 5.00
CS 109	Probability for Computer Scientists	Guest Lecturer	Spring 2023	4.78 / 5.00

At Stanford, I co-instructed *CS 106B: Programming Abstractions* (280+ students), lecturing on recursion, dynamic memory, and graph algorithms, and managing 15 section leaders. In evaluations, students highlighted my clarity: "*Dr. Mercer made pointer manipulation visual and accessible. He taught us how to think like a debugger, not just memorize code.*"

As Head TA for *CS 224N: Natural Language Processing* (450+ students), I coordinated grading, led recitations, and designed the sequence-to-sequence programming assignment. I restructured office hours to include specialized "conceptual queues" alongside "debugging queues," reducing wait times by 30%.

3 Course Development & Innovation

Educational tools should evolve alongside the field. To keep pace with modern developments in systems and machine learning, I have introduced several curriculum innovations:

3.1 New Seminar: AI Ethics and Large Language Models

In Fall 2023, I co-developed and taught a new seminar, *CS 294S: AI Ethics & Large Language Models*, exploring the societal, environmental, and ethical implications of generative AI. I designed a syllabus combining technical readings (e.g., bias mitigation, watermarking) with philosophical texts on fairness. For their final projects, students built auditing tools to detect biases in commercial LLM outputs.

3.2 Interactive Visualization Platforms

To enhance understanding of deep learning dynamics, I developed `NeuralPlayground`, an open-source library of interactive Jupyter notebook visualizations. Students use the tool to dynamically adjust learning rates, momentum, and batch sizes, observing the optimization path on 2D loss surfaces in real-time. This module was adopted in introductory machine learning courses, where 89% of surveyed students reported it significantly helped them grasp gradient descent.

4 Mentoring & Inclusion

An outstanding educator must also be a supportive mentor who actively fosters an inclusive environment. In my academic career, I strive to dismantle systemic barriers in STEM through intentional classroom design and research mentorship:

4.1 Creating an Inclusive Classroom Environment

To ensure all students feel valued, I employ inclusive teaching practices, such as using gender-neutral and culturally diverse examples in assignments. I implement anonymous Q&A boards (e.g., Ed Discussion) so students can ask questions without fear of judgment. I also use a transparent "slip day" policy for assignments, which accommodates personal, health, or financial challenges without requiring students to justify extensions.

4.2 Mentoring Undergraduate Researchers

Over the past four years, I have mentored 5 undergraduate students through Stanford's CURIS research program, focusing on underrepresented backgrounds. I meet with mentees weekly, providing feedback on research and professional development. Two have co-authored papers with me at CVPR and ICML, and three have pursued computer science PhDs. One former mentee noted: *"Alex did not just hand me tasks; he taught me how to read papers critically, write code, and present with confidence."*

5 Future Teaching Goals

As a faculty member, I am eager to contribute to your department's educational mission by teaching core curriculum courses and developing new, interdisciplinary classes:

5.1 Teaching Core Curriculum

I am prepared and enthusiastic to teach core undergraduate and graduate courses in computer science, including introductory programming and data structures, probability, machine learning, and natural language processing.

5.2 New Course Proposal: Trustworthy and Safe AI Systems

I plan to design a new project-based advanced course, *CS 3xx: Designing Trustworthy and Safe AI Systems*, bridging ML theory and systems. Students will learn conformal prediction, runtime verification, adversarial robustness, and model compression. For the final project, students will deploy an ML model onto local physical hardware (e.g., a robot arm) while guaranteeing safety bounds.

5.3 TA Training and Mentorship

Graduate teaching assistants are the backbone of large courses, yet they are rarely trained in pedagogy. I will establish a structured TA mentoring program in the department, organizing pre-quarter workshops on active learning, rubric design, and inclusive communication to improve instruction quality department-wide.