Teaching Statement

I am passionate about strengthening connections between computer science and related social science fields. I have taught in both CS and Linguistics departments, and wherever I am teaching, I look for ways to cultivate cross-disciplinary conversations and to show the value of collaborative, interdisciplinary work.

Computing skills are valuable for students in many disciplines, and mastering the tools and techniques of computer science can open up new possibilities for social science and humanities research. Teaching computer science classes that are welcoming to students from other disciplines can also help students from backgrounds that are underrepresented in computer science to find a place in the field.

I also believe that the social sciences and humanities have a lot to offer for CS students. The social sciences can provide the rigorous training in experimental design and data validity that future machine learning and data science experts need to address the replicability issues in their fields, while the humanities can provide insight into the questions of ethics and fairness that are so important for the future of CS.

I am also passionate about working closely with undergraduates. When class sizes are small, instructors can really get to know students: their goals, aspirations, and prior experiences. Being able to contextualize students helps me adapt the course content to their needs and interests.

My teaching philosophy encompasses three core principles:

- **Interactive, participatory learning**: students should be active participants in classes; ideally, instructors should interact with every student in every class session.
- **Multidisciplinary learning**: integrating insights from other fields enriches student learning. By opening up new pathways into computer science, it can also help diversify the field.
- **Fairer pedagogy through uncovering the hidden curriculum**: expectations and foundational assumptions should be made explicit. This helps students from diverse backgrounds to succeed.

1 Programming Languages

In Fall 2018, I taught Programming Languages as an upper-level systems elective in the Mt. Holyoke CS department. Although my graduate research is in computational social science, I previously worked on programming languages for networks as a research intern for Nate Foster at Cornell. I drew upon this experience and my experience studying PL at Swarthmore to re-design the Mt. Holyoke course.

I structured my course into four units. The first unit introduced students to functional programming and covered core PL concepts like higher-order functions, scope, and evaluation order. The second unit was a multi-week project in which students wrote an interpreter for a subset of Scheme. Next came a unit that I called ‘Programming Languages in the Wild’, which highlighted less common kinds of programming languages: a probabilistic programming language (Figaro); a verification language (Dafny); and a DSL for deep learning (TensorFlow). This unit allowed me to relate the class’s core concepts to my own research. In the final unit, students researched different programming languages and presented demos to their classmates.

I designed the course to be highly interactive: short portions of lecture time were interspersed with coding exercises. During the exercises, I was able to help students individually, gauge how well the class was understanding the material, and adapt my lecture. In course evaluations, students commented positively on the live coding style that I used in lectures. I connected one student, Jenna Hammond, with CS professors at UMass so that she could pursue summer research. Her project was recently submitted to AAAI.

2 Cognitive Modeling

In Fall 2017, I was the TA for Cognitive Modeling. I developed a weekly supplemental class for students who had little prior programming experience. One challenge was that students needed to gain skills very quickly to keep up with the assignments. Many students came into the class having never written a function, but were able to successfully complete assignments such as implementing a probabilistic parsing algorithm.

In this “Python for Linguists” supplement, I covered core CS concepts as well as basic Python skills;

to make it more engaging for the students, I tailored the in-class exercises to focus on natural language applications. For example, in my class on hierarchical data structures, I showed how tries can be used to implement a rudimentary auto-complete program. Courses like this are one way of broadening access to computational tools in a targeted, discipline-motivated way; students in this section were highly motivated because they understood that grasping these methods would help them in their own research.

I planned and developed the materials for this section. They have since been incorporated into a standalone online Python for Linguists course.

3 First Year Seminar

I am currently teaching a first year seminar on Technology for Language Revitalization at UMass, which seeks to bridge between computational and social sciences. This class, which I proposed and designed, addresses the practical and ethical considerations involved in designing technologies for language revitalization. It draws upon my experience as a Fulbright scholar in Northeastern Canada, where I collaborated with First Nations communities on web-based language learning resources for endangered languages.

The seminar explores the ways in which technology can both help and hurt revitalization efforts. We discuss ethical issues in research on minority languages, sustainability and preservation issues in the digital humanities, and the role of the internet in language change. In the final weeks, students will contribute to an ongoing digital language revitalization project for indigenous Mexican languages.

4 Teaching outside of the classroom

I believe that teaching does not take place during class time alone. I have sought out ways to build connections between departments and mentor students outside of the classroom as well. In Linguistics, there is a growing interest in computational methods, but many students and faculty members lack formal CS experience. I am regularly consulted for help with research-related programming, such as R and Python scripts for data analysis. I have helped several students design new experiment control programs for IbexFarm, the widely-used JavaScript-based framework for web-based linguistics experiments.

As a way of strengthening ties between Linguistics and Computer Science at UMass, I organize a reading group on neural networks in linguistics. The group has proven to be a fruitful space for interdisciplinary discussion and is regularly attended by students and faculty from both CS and Linguistics.

I have also had the privilege of supervising undergraduate research assistants: Alicia LeClair worked with me in Spring 2017 on semantic fieldwork materials, and Tessa Patapoutian has been working with me since Spring 2019. I have been mentoring Tessa in experimental and computational methods and am currently working with them on a neural network probe task on grounded language. I also recommended them for the internship at BBN Technologies that they received in Summer 2019.

5 Courses

I would be ready to teach introductory CS classes, as well as upper-level classes such as Natural Language Processing, Programming Languages, AI, and Data Science. With preparation, I could offer other upper-level classes like Deep Learning, Theory of Computation, Speech Recognition, and Compilers. Additionally, I could teach interdisciplinary classes like Computational Linguistics and Cognitive Modeling.

Figure 1: Summary of teaching experience

Course title	Course number	Enrollment	Role
Technology for Language Revitalization	HFA 191-A49	19	Instructor
Language Processing and the Brain	LING 412	32	TA
Programming Languages	CS 343 (Mt. Holyoke)	17	Instructor
Introduction to Linguistic Theory	LING 201	102 (34 in section)	TA
Cognitive Modeling	LING 692c	13	TA