Teaching Statement Yohan Jo

Learning is a continued process throughout life, and thus individual learning experiences should build on and benefit each other. When I teach a student, I envision my student becoming a self-regulated learner who (i) has intrinsic motivation for pursuing computer science—formulating real-world problems as computational and implementing efficient computational solutions, (ii) has mastered relevant knowledge and skills related to the class as well as to continued learning, and (iii) further advances the field of computer science by identifying new problems and developing their own computational solutions, concepts, and theories.

In order to achieve these goals, my teaching methods aim to make students' learning drawn by *personal goals*, centered around *problems*, and backed by *social support*. First, helping students set personal goals fosters intrinsic motivation to learn computer science and continue to pursue it even beyond the class. Second, problem-centered learning enables effective learning of knowledge and skills for computational thinking—solving problems using computational methods—as well as skills transferable to continued learning and advancing the field of computer science. Lastly, social support helps students sustain their learning in face of difficulties. I will elaborate on each aspect in the following sections.

As a note of background, this statement draws largely on my two teaching experiences connected to the School of Computer Science at CMU: (1) teaching the *Introduction to Natural Language Processing* course (*Intro to NLP* henceforth) for the spring and fall semesters in 2019 as the lead TA; (2) teaching computer science for local high school students at Pittsburgh Westinghouse Academy 6–12 in 2017–2018, through the college's partnership with Pittsburgh Public Schools and TEALS.

Personal Goals

When teaching, I try to get students intrinsically motivated to pursue their learning by helping them set personally relevant goals. For instance, when I delivered a lecture in the Intro to NLP course, I started each topic with how the topic may have affected the students' life (e.g., college entrance) and how learning the topic can help them find a job that benefits society or is financially stable, which is a major interest of the students in this course. However, connecting learning content with practical, personal goals is not always easy for students. Especially in introductory courses that cover a broad range of topics, teachers need to show how different topics are connected and relate to the students' learning goals. Students often came to my office hours with a puzzled face and asked why they ever needed to know certain concepts, such as a parsing algorithm. Then, I explained to them how those concepts can be used for their project or interesting applications. While I was teaching Korean in the Carnegie Library of Pittsburgh, most students wanted to understand Korean pop songs and dramas better. Hence, I tried to embed the learning content (e.g., grammar and vocabulary) in dramas and songs. Students showed strong passion evidenced by consistent attendance and active participation in the class.

Personal goals are key to unmotivated students especially. When I taught computer science at Pittsburgh Westinghouse Academy 6–12, the students had no prior exposure to this field nor to any computer scientists in their community. They had little to no motivation to learn computer science, and it was extremely challenging to win their attention. I wanted to convince them that computer science can be a fun field for their future jobs. Hence, I reached out to Facebook Reality Labs and Uber Advanced Technologies Group in Pittsburgh and organized two field trips. The students were astonished at intriguing technologies and expressed their desire to work at these companies. Such personal goals encourage students to commit themselves to learning with intrinsic motivation.

The importance of personal goals has been evidenced in my research as well. I conducted a computational analysis of how goal-setting is associated with students' learning behavior in an online course[1]. Students who set learning goals and their peers who observed them showed a higher degree of self-regulated learning behavior than other students, e.g., by engaging in hands-on exercises and revisiting prior learning content. This study and my teaching experiences convince me of many positive effects of personal learning goals.

[1] Yohan Jo, Gaurav Tomar, Oliver Ferschke, Carolyn P. Rose, Dragan Gasevic, Expediting Support for Social Learning with Behavior Modeling, in *Proceedings of the 9th International Conference on Educational Data Mining*, 2016

Problem-Centered Learning

I believe that learning is motivated and fostered by acknowledging problems that students are willing to solve, e.g., from making their programming code concise to forecasting the spread of a flu. Identifying such problems and reasoning about how to develop computational solutions are useful and transferable skills for computer science education. Hence, learning environments should be designed upon this principle so that when students learn certain concepts, they also learn why and how those concepts were invented in a broader context. In the Intro to NLP course, I structured my lecture such that I explained several NLP techniques by first pointing out a problem that people wanted to solve in the past, e.g., limitations of general sentiment lexicons and a need for domain-specific versions; and then I formulated the problem as computational and explained the techniques of our focus as solutions. Students can even discuss how they would solve a problem using what they already know, prior to learning new techniques. A learning environment should be full of intriguing and challenging questions, where students feel safe to share their ideas. The problem-centered learning is also effective for training students to be researchers and engineers to advance computer science, as research and engineering consist of identifying a problem and designing/testing methods using knowledge and tools at hand.

How can we help students find meaningful problems? Computer science traditionally has borrowed many problems from various fields, such as humanities, social sciences, and natural sciences, and the importance of interdisciplinary studies between computer science and other fields is growing rapidly. Therefore, I do my best to include relevant studies from other fields in my teaching. When I taught sentiment analysis and computational argumentation in the Intro to NLP course, I first introduced relevant theories from philosophy. It allows students to be exposed to meaningful and intriguing problems in the real world studied by domain experts and to be inspired by those studies.

I apply the problem-centric principle to assessment as well. I understand students' concerns about assessment, especially in a competitive environment such as CMU, since I also often struggled with timed exams that required intensive memorization or fast problem-solving. I think the two main roles of assessment should be to assess whether a student understands important concepts correctly and whether a student can apply the concepts for practical problems properly. In the Intro to NLP course, I created questions for every exam, and many of my questions first motivated a problem we want to tackle and then asked how to apply the concepts learned in the class. I also created open-ended questions that encouraged students to use their creativity based on what they had learned. Such questions help students ponder how key concepts could be connected to real problems and clarify whether they really understand a concept without confusion. Further, the same benefits apply to me as well while I create those questions, improving my own understanding and teaching. I will also use other assessment methods that complement the problem-centered learning, such as post-lecture surveys and game-based quizzes, to ensure students understand key concepts.

Social Support

Social support helps students sustain their learning in face of hardships. Soon after I started to teach the Intro to NLP course, it was evident to me that students who struggled and came to office hours the most were isolated from other students; they were from underrepresented ethnic groups or departments outside of computer science, and had no friends in the class. To help them, I matched students in need with other students. According to the feedback I gathered, the matched students felt supported and appreciative. But I also learned that peer groups formed in this way often need direct support from teachers for scheduling and moderating study sessions until they become stable.

Teachers should be mentors whom students feel comfortable approaching and sharing their concerns with. I made such an atmosphere through office hours in the Intro to NLP course, and some students shared their concerns about their failure in the course and uneven workload for the team project. If such concerns are not addressed, it significantly hurts their motivation and prevents them from reaching their full capacity. Social support may help students keep their interest in computer science and continue to pursue and even advance the field.

Building close relationships between students is even more important for younger and unmotivated students. When I taught junior high students from low-income families, many of them came to class against their will. I remember that four

students often ran away from the teachers and I chased them down. One day, I sat with them and asked about their dreams and family situations. I was surprised to see them open their hearts and actively share their stories. Building the close relationship opened their hearts to learning as well. I observed the same thing when I tutored high school students in my church. Some students listen to teachers only when they feel the teachers do not judge them and instead become friends with them. In such cases, effective teaching is difficult to even start without building personal relationships first.

In building proper relationships with students, teachers may experience the fear of losing authority. At the beginning of the Intro to NLP course, I was scared to be looked down upon by smart CMU students discovering my imperfect knowledge and skills. However, I gradually learned that showing my imperfection is fine; it is more important to properly follow up with help that the students need. During my office hours, I honestly told students up front if I was not familiar with certain topics, instead of avoiding them or making something up. However, I always sent them a follow-up email with proper answers after the office hours. In the post-semester TA evaluation, a student showed a deep appreciation for those emails.

Future Vision

My first learning experience of computer science, which captured me to pursue this field, was to develop an electronic billiards game. Drawn by the clear, personal goal of making my own game, I really enjoyed facing little problems and solving them one by one by studying mechanics and programming techniques, with the teacher's emotional support. This experience is formative to my teaching methods built upon personal goals, problem-centered learning, and social support, which I will keep in mind in my future teaching.

I am qualified to teach introductory computer science courses, such as programming and data structure, as well as advanced courses, such as natural language processing and applied machine learning. I can also teach topics in my research areas, such as (computational) argumentation, sentiment analysis, and educational technologies.

Teaching Experience

I have taught students with a wide range of ages (junior high students–middle-aged adults), degrees of motivation, ethnic groups, and class sizes (8–120).

Natural Language Processing (CMU): As the lead TA, I taught general topics in NLP for 120 undergraduate and masters students for the spring and fall semesters in 2019. I helped students through office hours, one-on-one meetings, Piazza, creating homework and exam questions, and delivering lectures. The office hours were an invaluable time where I got to know individual students better and could provide personalized support intellectually and emotionally.

Computer Science (Pittsburgh Westinghouse Academy 6–12): I taught an elective computer science course for 10 African-American high school students in the academic year of 2017–2018, through the TEALS program. I taught Snap! Programming language and motivated the young people to engage in computer science by giving lectures, helping with hands-on labs, and organizing field trips.

Problem Solving (KAIST): I taught problem-solving skills and algorithms for 20 undergraduate students for the spring and fall semesters in 2009. I graded homework and helped students with exercises via one-on-one meetings.

Korean (Carnegie Library of Pittsburgh): I taught Korean for 10 Pittsburgh residents in June 2015–June 2016. I organized lectures and exercises for grammar, writing, and speaking. I learned to lead and enrich lectures by incorporating various learning materials, such as textbooks, songs, and dramas.

Mathematics (Edushare): I taught mathematics for 8 middle school students from low-income families in Korea in February–May, 2010. I explained key concepts and helped students solve exercise questions one-on-one. I learned to empathize with students in difficult family environments and encourage unmotivated students with low self-efficacy.

Bible Study (The Church in Daejeon): I tutored 8 high school students and taught the Bible in Korea during February 2011–July 2014. In addition to leading Bible studies, I organized cooking and singing activities every week. I learned to approach adolescents, touch their hearts, and motivate them.

Period	Subject	Students	Type	Size	Roles	Organization
Spring & Fall 2019	Natural language processing	Undergraduate & masters	College major course	120	Lead TA & Lectures	Carnegie Mellon University
Aug 2017–May 2018	Programming (Snap! & Python)	9th grade	High school elective	10	Lectures & labs	Pittsburgh Westinghouse 6–12 & TEALS
June 2015–June 2016	Korean	Ages 14–50	Language class	10	Lectures	Carnegie Library of Pittsburgh
Feb 2011–July 2014	Bible	10th-12th grade	Bible study	8	Lectures & mentoring	The church in Daejeon
Feb 2010–May 2010	Mathematics	8th grade	Extracurricular class	8	one-on-one	Edushare
Spring & Fall 2009	CS problem solving	Undergraduate	College major course	20	Grading	KAIST