Teaching Philosophy of Tanay Kumar Saha
LEARN, (Re) BUILD, BREAK

My dad was a great teacher. He used to teach math and science in high school. His style of teaching always inspired me. He used to articulate first the motivation of learning the topic and then cover the subject. I had the opportunity to lecture at two universities in Bangladesh before coming as a PhD student. During three year time as a lecturer, I have experimented with various techniques including techniques learned from my dad in teaching undergraduate level courses which I thought would have significantly helped me during my sophomore, junior, and senior years. After coming as a PhD student, I worked as a Teaching Assistant, supervised summer students and course projects in which I always blended my thoughts and experience to make the students succeed.

As I had enthusiasm about teaching, I always strove to take more than required responsibilities when I had the opportunity. During my PhD studies, I gave a guest lecture in Prolog in a Programming Language course and several others in Artificial Intelligence course. Occasionally, I used to teach graduate Data Mining class. I also answered most of the questions in Piazza[1] to help students either understanding the topic or help with the assignments. I won the *best TA award* from Computer Science Department in 2017. My philosophy of teaching is: LEARN, (Re) BUILD, AND BREAK which I explain in the next paragraphs.

Learn: I believe that the very first step in teaching is to make the students engaged in the lecture time and help them learn the fundamentals of a topic through small hands-on examples and motivational questions. Small hands-on examples stimulate students' thinking during the class time and make them more engaged in the subject matter. From my experience in teaching, I have also observed that sometimes asking students to come to the white board and solve the problem in an impromptu way makes students who were already engaging more engaged and confident, and draws the attention of students who were not following. Moreover, I love to use white-board to explain ideas or showing simulation in addition to slideshow presentation so that students have less struggle at home going through the lecture, and be able to invest the spare time to go through additional resources/questions which I plan to provide after every class. Apart from teaching the basics of a topic from the introductory level, I also plan to explain the underlying theories to understand the concept better once they are comfortable and motivated with the high-level ideas. I am also aware of the fact that there may be students of different caliber and disabilities and I am always happy to give an extra effort for those who need the help the most. I would like to follow the open door policy as much as possible. Regarding assessment of active learning, I plan to take pop-quizzes, provide regular feedback on projects and give points for in-class quizzes.

(Re) Build: I think, learning without building makes the learning short-lived. In an undergraduate algorithm course, I divided my whole class into four groups and helped them implement all the relevant algorithms in four different topics: (1) Dynamic Programming, (2) Greedy technique, (3) Graph Algorithms, and (4) String Matching Algorithms. I did similar for the Data Structures course. After these two classes, I realized that students enjoyed the learning if they know beforehand that they have to build a product which can be a simple program or an app out of the course. It also stimulates the sharing. All the groups shared their implementation and struggle with each other which made the class more engaging. I realized that encouraging students to help each other is often the most efficient way of teaching. Students were also more spontaneous for all of the topics in the lecture as each responsible group were prepared before the class and asked fascinating and intriguing questions which helped everybody to understand the subject matter better. In addition to assigned TA hour, in one graduate Data Mining course, I supervised five project groups. We had a weekly meeting, and I always maintain an agenda for each week and incorporated the weekly feedback in the grading. In every week, they had to show some program or a particular portion of an app that works. I also guided them for completing the project reports. This strategy worked well

[1] https://piazza.com/

and taught me another lesson that encouraging students to start early and continuously giving them feedback is essential for quality teaching. Sometimes what you build breaks down which I explain later, but, finding the cause of the breaking point and solve them makes the (re) building phase more interesting, motivating and practical. I believe, most of the innovative ideas come in the (re) building phase.

Break: I firmly believe that without breaking something you build you do not know enough about your creation or own it. When students create an app or a program that perform some tasks, I often instructed them to find cases where it will break. The breaking points can come from different perspectives: invalid input, scalability issues, algorithmic issues, and many other ways. This process encourages them to think through their learning and often stimulate new ideas which give robust builds. In one class, I was supervising an honor's project where the students were building an AI agent playing Michigan Rummy. After the initial design was complete, I instructed them to think some of the cases where their plan may fail and gave them pointers for more robust design pattern and AI strategy to use which would make their build more robust. The one other general mistake I have found most of us do is that we think very big at first and then struggle substantially to meet the expectation within the time frame. For the project mentioned above, at the very beginning project meeting, the students came up with ideas of implementing three/four types of AI agents for the game. I listened to them, and after a thorough conversation they understood that the very first step would be to design a simple Michigan Rummy Game where two players can play several moves, and the program can compute the final score correctly. To be able to break problems into smaller problems and solve in pieces is an art which I believe personally and I have applied it in practice through examples so that students can also understand it. This process also works very well with the research students who are trying to solve a particularly hard problem.

Planned courses: As a Professor, I plan to teach (1) Data Mining, (2) Deep Learning, (3) Artificial Intelligence, (4) Algorithms for Data Science/Machine Learning: Analysis of Data Mining Algorithms, and (5) Data Structures for Data Science/Machine Learning in various capacities for graduate and undergraduate level. In each of these courses, I plan to incorporate both the solid theoretical foundation with hands-on experience in leading students into thinking about the problem and formulating a hypothesis, before engaging in their solution. In Data Mining course, I aim to cover basic statistics, dimensionality reduction, pattern mining, and classification and clustering techniques along with the qualitative as well as quantitative evaluation of each of these methods. In the Deep Learning course, I intend to cover state-of-art deep learning techniques, such as Convolution Neural Network (CNN), Sequence Modeling (Recurrent Neural Network (RNN), Recursive Tensor Neural Network (RNTN)) and other models for learning representation. In Algorithm for Data Science/Machine Learning, my idea is to cover traditional classification, clustering, summarization, representation learning techniques as well as their underlying learning algorithms such as Expectation Maximization, Variational Inference, MCMC sampling, and Stochastic Gradient Descent (SGD) with the particular emphasis on their computational complexity. For the Data Structures for Data Science/Machine Learning, I plan to cover Graph as a Data Structure and its usage in Network Analysis and Mining in addition to traditional data structures and algorithms.

Other Plans: For undergraduate students, I plan to create a competitive programming contest platform for brushing up their knowledge of data structure and algorithms, and possibly prepare them for the **ACM-ICPC International Collegiate Programming Contest**. I created a similar environment when I was a lecturer in Jagannath University, Dhaka, Bangladesh (`http://jnu.ac.bd/`) which helped my students to put one step forward in the job market.

Moreover, I also plan to gradually develop a GitHub repository of implementation of data structures and algorithms as well as crucial derivations covered in Algorithms for Data Science/Machine Learning and Data Structures for Data Science/Machine Learning course so that students can incrementally build up knowledge. For providing real-life usage of these topics, I will borrow applications from Natural Language Processing, Text Mining, and Computer Vision domain. I also intend to use both the textbook and the relevant research papers for the course. My research experience in the field will also help me to relate the topics to the mindset of real-life problem-solving.