Jonathan Chang jkchang2
Technology Review

Online/Scalable Education

Online education, after emerging within recent years, rapidly gained acceptance and adaptation by universities and students alike. Major reasons include reducing costs, saving time, and increasing flexibility of learners. Costs of on-campus learning require living expenses and or commuting, as well as instructor time to read the same lectures over and over again. Instead of that, online education primarily relies on pre recorded lecture content for students. That means educational materials from a single University became sharable to the world. However, with great benefits come noticeable downsides. The overall quality of lectures, with indirect teacher student interactions, completely changes the learning experience. Students now must digest as much of the content as possible, and then interact with peers in online forums instead of getting direct live feedback from instructors. Furthermore, students end up utilizing web resources to enhance understanding of course content. There are a couple of studies addressing these ongoing issues of online education. First, the AXIS paper which acts as an added feature of online learning platforms to enhance explanations to exam answers. Second, the Tutorons paper, which adds an extension to the web to help learners understand code. Both the pros and cons will be addressed, with added feedback on future steps.

Firstly, the AXIS project. The team built a useful feature on top of online learning platforms. It collects continuous feedback of answer explanations via student ratings. Also, it collects student written explanations. Afterwards, that data is passed to a Machine Learning model, which outputs improved answer explanations (Williams, 1). The paper refers to the improvement iteration cycle as the Multi armed bandit problem, where the model seeks to gather higher student ratings to provide better explanations to students (Williams, 4). Major pros include saving instructor time to work through manual updates of answers. It may be that instructors and students have different perspectives, so allowing for optimized student collected and rated opinions makes sense. Students are also looking to better understand “why” something is correct, in perhaps simpler terms from someone new to the concept. Over time with many students providing feedback, there would arise optimal explanations to answers for those to learn from. Otherwise, answers may be unclear or even misinterpreted by students. There are certainly cons to call out as well. The biggest gap is confirming whether this model is accurate enough to be deployed and accepted. In practice this is difficult for a couple main reasons. One, is that the reviews (selecting 1 to 10 rating) is entirely subjective, and may greatly vary depending on the student’s personal background knowledge (Williams, 3). There would need to be enough test rounds to understand the proper threshold of ratings to mark as a “good” answer. Two, you would need a lot of student based data to train the model. The paper mentions as much as 540 students, but to fully deploy and utilize online platforms you would need more time to iterate and improve (Williams, 6). This limitation blocks actual progress and confidence to roll out this feature. Raw answers also are variable, as some instructors may provide much better answers than others, so ratings could be subjective relative to the instructor individually. I had a few suggestions to improve upon this idea. For the ratings, instead of a 1 to

10 scale, simply use an upvote or downvote. That avoids more subjectivity and ambiguity on everyone and the model to determine if an explanation is useful or not. It would be more realistic to collect the data and train the model by simplifying the set. Also, showing various examples of explanations to vote on to students. Then students can self-select answers that provide the most value. Typically we also see this today in online forums, where good explanations have many “upvotes” while unclear explanations may have no “upvotes”. Upvote scoring can be a direct way to rank and surface explanations. Addressing the issue of proving the system works via user testing, the best way would be to “beta test” on live students studying online. This can involve partnerships with online universities looking to improve upon their design, and provides the candid feedback required to make improvements. AXIS overall makes sense and fills a missing piece in making online education better.

Secondly, the Tutorons paper. Tutorons means providing context relevant and on demand information (Head, 1). In context, Tutorons are specifically applied to help students understand CSS code, regex, and wget. These are a common interest for people to learn and feasible enough to produce relevant Tutorons guidance for enhanced student understanding. It aims to avoid forcing students to constantly manually browse other websites for various contexts to understand the basics of the code. The tool itself is a web extension which relies on web-scraped data to generate a mapping between code text and relevant explanation. It gets specific on details as well, such as for CSS, using a tree to map the relations between tags. Major pros include surfacing student-friendly tips for understanding the basics of code to develop their skills. Cuts down on the variability of other websites students may rely on today to decipher unclear code without meaningful explanations. Also from the student perspective, flexible to turn on in the browser and on and off for specific types of tips. Major cons include poor recall scores during testing, as well as only having tested performance on a handful of students (Head, 6). The scalability would also be in question, such as how to account for vastly growing amounts of scrape data for various types of code, and how to monitor and adjust the explanations surfaced by collecting user feedback (actual text, size, accuracy). As the paper states, each style of code also takes on its own implementation of backend mapping logic (Head, 4). It would realistically have low ceilings of extensibility to very simple code examples. Some thoughts are expanding the test group, similar to the above idea of partnering with online universities to “beta” feature this extension and get more feedback. Also there needs to be a consistent “fact-checker” running, either automated or by approved instructors, to surface proper data. There needs to be more user experience based generated testing as well for UI interaction, and potentially extension based activity (adding/removals, which features are being turned on or off). Overall, Tutorons seem promising and would provide extra value to students learning how to code.

In conclusion, online learning clearly includes multiple areas of opportunity to improve upon, which researchers are working on through projects such as AXIS and Tutorons. Online education received even more attention due to the COVID pandemic, and adding improvements now will yield countless benefits to future students. The couple of ideas discussed today prove promising, though in early stages, and hopefully fellow online students will reap the educational rewards soon.

Works Cited

Head, Andrew. “Tutorons: Generating Context-Relevant, On-Demand Explanations and Demonstrations of Online Code.” IEEE, 2015.

Williams, Joeseph Jay. “AXIS: Generating Explanations at Scale with Learnersourcing and Machine Learning.” SIGCHI Conference Proceedings Format, 2016.