Limed: Teaching with a Twist
Season 3, Episode 7 – AI as Friend or Foe in Computer Science – Part 1

Matt Wittstein (00:00:11):

You are listening to Limed: Teaching with a Twist, a podcast that plays with pedagogy. Artificial intelligence, large language models, chat GPT. These have become the topic of serious discussion and challenges in higher education. Over the next two months, we take a look at what that means and what could be possible for the future of computer science education. Ryan Mattfeld, associate professor of computer science at Elon University asks for some practices to safeguard his students' learning while wondering what might be possible if the floodgates to AI were left open. This month, we get some of the student perspectives on the topic with Anna David, a first year student at Carnegie Mellon University and Arav Patel, a graduating student at the University of Maryland. Next month we'll let some faculty respond and share their insights as well. If you're enjoying the show, please rate, review and share. It is the best way for us to get out there, stay relevant and talk about the topics that are most interesting to you. Thanks for listening and enjoy this episode of Lyme Teaching with a Twist. I'm Matt. Hey Ryan. Welcome to the show. I am so excited to talk to you today. Before we get into conversation, would you just introduce yourself to our audience?

Ryan Mattfeld (00:01:44):

Sure, yes. I'm Ryan Mattfeld. I'm an associate professor of computer science at Elon University. It's my seventh year here. I teach a variety of different courses from introductory to upper level and I also teach a core capstone.

Matt Wittstein (00:01:56):

And so this episode is going to start sort of like a joke. There's three computer scientists sitting at a bar and I walked into that bar and y'all were talking about your teaching a little bit. And here we are having a episode about AI in the computer science classroom. So do you want to share a little bit about that conversation that you were having to sort of frame where we're going to start this off today?

Ryan Mattfeld (00:02:21):

Yeah, sure. We chat regularly in our department about the impact of a lot of these large language models, which I may call LLMs as I go through this. Things like chat, GPT copilot and their impact on computer science in particular. And of course as we're teaching it we also talk about the impacts on computer science education. And so we've talked about a couple of different things. I mean we could talk about how we currently are approaching it, but the question we were talking about then was how could we radically shift this to adjust to these tools? And really my idea was how could we fully embrace these, right? We know that these tools are used by professionals in computer science. We know that these tools are useful and helpful to people who program. Is there a way to write or to create a computer science one course that fully embraces this and says, alright, from day one, let's use these as well as we can and let's see what we can actually do with these from a complete novice perspective.

Matt Wittstein (00:03:22):

So when I came into that conversation, it was a little bit more of a, felt more like a venting session where you all were talking about how your students were using LLMs maybe inappropriately for some of

their coursework. So do you want to share a little bit more about how that course currently runs, how you are open to the use of AI in the classroom as a department and for you individually?

Ryan Mattfeld (00:03:49):

Ever since Chat GBT really became popular and a couple of years ago, we've kind of been adapting. So we could view it as a problem that we now have this new tool that makes it maybe more easy for students to find alternative ways to solve problems than through their own work. But I consider it not necessarily a problem, but something that we can adapt to because it also has a lot of benefits. I view it as a tool that if we use it the right way, we can do things with it. That said, in developing that process, we run into situations that we have to handle where students clearly did not submit their own work and believe that they should be getting full credit for something that they haven't actually done themselves. And so we've talked about and we've adapted our courses some certainly recently we've shifted focus away from projects and more toward in-class graded assignments.

(00:04:39):

But that said, you can't learn computer science without actually doing it. And the best way to do it is to write a program. But if we let set students loose and let 'em write a program, they're going to try to use whatever resources they can at times and that can cause certainly frustrations. One thing that we've done, for example in our project work is we have this criteria that says if you submit work that uses skills that we have not covered in class, then you must explain what that skill is and how it works. And you have to be able to apply it to another problem. So if they submit something and we look at it and say, this seems like something that in a really advanced programmer or maybe in a large language model like chat GBT might produce, you can still earn credit for it, but you've got to come in and show us that you actually learned this thing, that you didn't just copy and paste it and submit something you don't understand, but you at least have to understand what you learn.

Matt Wittstein (00:05:32):

That's a really clever approach to sort of double checking if they used AI or if they're actually learning the material because it sounds like to an extent you don't care if they use AI as long as they learn while using it. That if they use it intentionally and learn the steps that are appropriate and are able to apply it to other things, well then the learning actually did happen in some way. And so you're okay with that than say they just submit something that's not theirs. So that's a really good example and it makes me wonder what are some of the other ways that you sort of know or have a hunch that students are using LLMs to enhance their work?

Ryan Mattfeld (00:06:15):

We can see it a lot of times in a discrepancy between what a student is capable of in class and what they're capable of outside of class. We have in-person tests that we take and we have shifted more and more, especially at computer science one, that really introductory level to having tests written on paper that's very conceptual and focuses on the fundamentals and has students writing solutions out. And then we give similar problems. That's a piece of what they have to do on a project. And so if they can't do it in person on a test in a simple version and then they suddenly are doing more advanced versions of that on a project without discussing with us, it's possible that they learned on their own and put in time and effort, but it's more common that they didn't. Of course, the hard part is there's no guaranteed way to say that this was generated by an LLM versus the student did this on their own.

(00:07:11):

Regardless, we can also see signs of students using these tools through using more advanced skills than what we cover in class. So we teach the basic fundamental easy to read and understand version of a loop, for example. And in Python, there is also a way to write a more advanced version of the loop. That's one line of code, much more condensed, harder to read and pretty popular, but you have to actually learn that and understand it to be able to do it. And so if you jump to the more advanced version of displaying the skill without displaying the basic version, then either something is helping you or you've learned a lot and we got to figure out which of those it is, which is why we take the time to actually bring students in and say, alright, which of these things happened and try to figure it out and go from there.

Matt Wittstein (00:08:00):

So I do want to ask a quick question about how you think students could use it intentionally and how you guide them to use it intentionally for the benefit of their learning.

Ryan Mattfeld (00:08:16):

Yeah. Students have a number of different ways that they could be helpful. If they're coming across a piece of code that they don't understand, feeding that code into a large language model and saying What's happening here, it actually does a pretty good job of explaining the things that are going on and that can help you understand what's happening on a lot of the individual assignments we give them, many of which are not graded anymore since they're at homework. But students can we offer to students, Hey, create your solution and if you can't get it working, send it to us and we'll help you try to figure out what's going on or bring it up in class and we can walk through it. But also large language models can do it too, right? Those problems that we give them to solve on their own, these large language models are capable of programming at the level of at least a computer science two student.

(00:09:07):

So certainly cover everything we learn in computer science one, if you give it the questions we give you, it will produce the code. The key to the learning process though, at least in my opinion, is that you attempt the problem yourself first and then you ask chat GPT for a solution if you can't solve it. So you can figure out, hey, I went through the thought process thinking I need to do X, then Y, then Z. But when I put it through chat, GPT, the solution actually did A and then B and then C. So the approach I was using was wrong, but now I can see what the right approach would be and maybe apply it to a new problem.

Matt Wittstein (00:09:40):

Do you have any strategies to ensure that that process happens other than just encouraging that they try it first? Do you have any way to sort of make sure that they try it first before they go to chat GPT?

Ryan Mattfeld (00:09:52):

I'm not sure that there is a way to do that because chat GPT is accessible immediately. We can tell students, you should do it this way, this way. This is the way that we know helps people learn better. In our years of experience, we've seen people who just look straight at the solution, feel very confident that they understand it and then have to do it again on a test for a different problem and cannot translate that skill. I don't know if there's a way to truly ban it. I mean, you can't ban these things and you shouldn't cut off student access to these because when they become professionals, and we know this from speaking with professionals in the field, these are useful, they're helpful tools that help speed up your coding process and taking care of, honestly, a lot of the fundamental skills that we teach in CS

one and that have been believed as the fundamental skills in computer science for decades, which is kind of leading me to the question of the course I'm getting to, which is what if the fundamentals we think are fundamentals are no longer the fundamentals? What if those skills have changed with these tools? We don't think they have, but we can't know that without trying, without experimenting.

Matt Wittstein (00:10:57):

Well, that's a great segue and it sort of makes me think of, there's a lot of commercials on TV now of how copilot can increase your productivity and coding and stuff like that. And I imagine students see this and they see this as not a tool for cheating, but as a tool of productivity as a tool that they would actually get to use in the real world. So let's talk about this sort of big idea conceptual class that you're thinking about and to frame it, I mean, you were talking about doing this with a computer science 1 0 1 style course, but just taking off the guardrails and saying, Hey, here are the LLMs. Here's the things that you can do. Let's see what we can accomplish. So when you frame that in the context of an actual course, how do you set those learning goals differently than in your traditional computer science one-on-one course now?

Ryan Mattfeld (00:11:52):

Yeah, absolutely. It's an interesting question to think now that we have these tools, what are the fundamentals? And if we provide, and this is the real question, if we provide a novice with these tools who knows nothing about programming, what can they do and how quickly can they get there? We focus on, again, what I talked about, the fundamental skills. Can you write a loop? Can you write a conditional statement? Can you solve a word problem? If we have an LLM, can we jump to the more advanced skills that students have, which is a little bit closer to can you figure out what you need to ask to solve a problem? Can you break a large problem into smaller chunks that LLMs can handle? And can you interpret correct or combine code generated by these tools? And to be clear, these skills are ones that we cover later in computer science.

(00:12:46):

They're still important variations of these have been taught for decades, but can we jump to that or can we jump to that? Can we skip to that? Can we approach that more quickly in an introductory course and a true introductory course? And so how do we have to design it differently? Well, when I'm thinking about this, it's still an experimental course and an idea. So I'm formulating, and honestly, this is where I'm hoping maybe you can get some ideas to this one area at least. My thought is to approach this as giving a bigger project than we would normally give for CS one, since you have access to this tool, then let's make the project bigger and more interesting because we should be able to get to that more quickly. And then also focus on can you make the program work? Can we now focus on the program works and runs and does everything we want it to, and that's the grade as opposed to do you understand what the pieces of this program are and how each of them interact and how we get to the results.

(00:13:46):

So it's a little bit more focused on does it work than do you understand, I've done this in the past with a little bit more performance-based, does this thing work or not? Partial credit is a lot harder because in computer science, if one misplaced comma or one misplaced tab can make the whole program not run the whole thing just fail. And so if you did 99% of the work and made one silly error and it doesn't run, and we're evaluating on a model of does this thing do everything it's supposed to, there's no credit to give, the program doesn't run. And so that's certainly a challenge with this idea for this course, and I'd

have to figure out different ways to potentially scaffold it or handle that kind of situation. And of course there's all sorts of other concerns.

Matt Wittstein (00:14:29):

I'd love to know what some of your bigger concerns are to share with the panel.

Ryan Mattfeld (00:14:33):

First of all, if you are a CS major, there could be a concern that we run this course, you're successful in this course, and then if you were to move on to another cult class, you would've skipped some of the fundamental understandings that you might've gotten in a traditional CS one course. Does that significantly impede and make it much more difficult to complete the other coursework that's normal or that's included? And so my way of addressing this would be to run this as an experimental course that does not give computer science one credit. In other words, you'd have to take that course again if you wanted to be a CS major. There are many departments around Elon that are wanting to send their students to computer science one to learn enough programming to help with research projects, to help with basic programming along the way.

(00:15:18):

And so I wonder whether this course would be better for that than our traditional CS one course. Another concern of mine is that this fails completely, not just in future semesters, but in the current class. The experience that we have and that allows us to use large language models effectively does not translate at all to a novice. And that if you give a novice this tool, they will generate a bunch of things that they don't know how to put together or run or do anything with. And of course I'd be there to help try to guide it, but that we may end up having to just say, man, nope, this doesn't work. We can't do this without teaching the things that we consider foundational right now. And if we learn that, that's great information for us. But I also would feel for the students in the course not getting as much out of it as I would hope, and of course we'd have to adapt if that ends up happening in the course. Maybe this idea of going full GPT is actually better placed after computer science too, and then we have an upper level course that then says, okay, now that we know the foundations, now that we have the foundational information, let's make something really cool and just go full GPT with it and see what's possible. And that's kind of why I went run the experiment. I want to figure out what level do we need to be at to get something useful out of these tools.

Matt Wittstein (00:16:35):

Awesome. Well, I really appreciate you sharing this with our show, and I can't wait to talk to our panel and get back to you after we have that conversation.

Ryan Mattfeld (00:16:43):

Sounds great. Thank you for having me.

Matt Wittstein (00:16:55):

Hi Anna. Hi Ara. I'm so excited to have you on the show. To introduce yourselves to our audience, please let them know your name, where you're at, what year you are, what you're studying, and generally do you think that current college students are using AI ethically and appropriately in the classroom and for their learning?

Arav Patel (00:17:12):

Hi, my name's Arav Patel. I'm currently a senior studying computer science at the University of Maryland. Recently I've been getting super involved in AI engineering and just the AI world on the side. I love to ski, I like to hike, listen to music, go to the gym, pretty normal college stuff. And to answer your question, I think for the most part, or at least around the people I surround myself with, I think we're generally using AI ethically when it comes to our schoolwork and the way that we go about school to learn things. But there are definitely some cases where we do rely on it sometimes and we'll use it not the most ethical way, but it just allows us to get stuff done faster.

Anna David (00:17:53):

Hi, I'm Anna David. I am a freshman in robotics and computer science at Carnegie Mellon University, and I'm really interested in ai, especially in robotics applications and also generative ai. So I've done a lot of work in generator of ai. And to answer your question, I'd like to think that all students are using chat GBT and other generative AI sources ethically, but obviously certain people are not, and I think that in some cases maybe it's not impeding their learning, but in other cases it definitely is. So I think it's really a balance between making sure that we're using the resources we have but not impeding the learning.

Matt Wittstein (00:18:47):

So I want to ask real quick for both of you, what's sort of that line? You both said generally people are good, their students are good, they're not trying to use AI to cheat generally, but there are cases where it's not being used in the most appropriate manner. So my question for you is what's sort of that line that you think distinguishes when a student decides it's okay to use it and another student decides, whoa, it's off limits?

Arav Patel (00:19:18):

One area that I think is a little bit off limits is when it comes to things that should be creative and spark creativity. So when it comes to art such as music for example, and I guess drawing and just showcasing some form of artistry, I think that's where there's a little bit of a line that if AI generates it, it's AI art. But that kind of defeats the purpose of it being called art in the first place because in my opinion, art is made by human and shows creativity of a human. But when it comes to schoolwork, not all schoolwork, but sometimes to just generate some boilerplate code or to help you understand something that's not really too understandable and sometimes a Google search isn't really helping out some form of a copilot. That's where I think it is ethical and useful to use as a tool and a resource versus something that sparks creativity.

Anna David (00:20:13):

Yeah, I definitely agree with that. I think in any sort of creative things using AI kind defeats the whole purpose because the whole point of something creative is to represent human emotions and something like that. So you're kind of taking all of the human out it if you're just using ai. But I definitely agree that especially in bigger projects where a single function, you may have learned that in an intro class, but maybe you're now in a more advanced class, that one function knowing how to write it may not be the most important thing. It may not be the best use of your time when maybe it's a pretty common function you can find on Google or chat GBT generally if you're working on maybe a bigger project where in individual function isn't the whole point of the project, the point of the project is to get those bigger ideas and really connect everything. Then maybe using AI to write an individual function isn't a big deal because you're still doing all of that learning on the bigger project. Whereas in some smaller

scenarios where you're in an intro class and the whole point is for you to actually understand how maybe binary search function works, then you don't want to use chat GPT for that because then you're not actually learning what the goal was for you to learn to

Arav Patel (00:21:40):

Kind of add on to her. I just thought of something for some of my classes when we're introducing ourselves or starting the class, we go on to a discussion board and kind of post things about ourselves, but when we get those personal moments where we should explain a personal experience or say something to introduce ourselves, I think those are also moments where you don't need to ask generative AI or LLMs to generate your response to write that down. It doesn't feel authentic, whereas you normally just write that yourself and explain to whoever you want to explain to who your true self is, whereas other times it's not necessarily needing to be authentic, so then you can use AI to help you get something out quicker.

Matt Wittstein (00:22:24):

So y'all have probably figured out at this point that we're talking about artificial intelligence use. We're talking about in a computer science classroom and specifically one of my colleagues here at Elon, Ryan Mattfeld is a professor, an associate professor of computer science, and I spoke to him after he was sort of frustrated with students that clearly were using artificial intelligence chat, GPT, other LLMs as a means to sort of get the answer, and he was really concerned about their ability to learn some of the foundations. As we had that conversation, we also started talking about what it might mean to just allow them to use chat GPT. I'm curious how explicit you want your faculty, your professors to be with you when they're giving you an assignment of how you're allowed to use ai, how you're not supposed to use ai. So I guess this question is twofold. One is in your experience, how explicit are your faculty members and two, how would you want them to teach you about using ai?

Anna David (00:23:32):

Yeah, so I think my professors in general have been very clear, especially as a freshman. I'm in mostly intro classes, so most of them are absolutely no AI because we really want to teach you those basics so that then you actually understand what's going on when you get into those more advanced classes where maybe using AI is fine, but I'm currently in a class right now that's intro to Feedback control systems, which is like a robotics software class. And that class, the first day my professor came in and said, I want you to use chat g pt, I want you to use Google. Everything in this class is open computer, open Google Open Chat, GPT, because the goal of this class is to work on those big projects. So because we have chat GPT to use in that class, we're going to be able to do some really cool things as our final project. Whereas if we had to write every single function by hand and build off of, maybe we need some sort of matic math equation or something that maybe we didn't learn, maybe we can go to chapter 15 and it can explain it to us and then either help us program it or then we have that knowledge to actually program those complex functions that we wouldn't be able to otherwise.

Arav Patel (00:24:51):

None of my professors really have said so far to not explicitly use it except for one professor in my writing course. In one of my writing courses, she specifically said, don't use ai, but for most of my CS courses, they kind of allow us to use it because for projects it does help us get things out a lot quicker. However, for exams, all of our exams or most of our exams are handwritten. So you can use AI to get you through a lot of the schoolwork and classwork and homework assignments. However, once you get

to the exams, you kind of can't use AI because you're in an exam room and taking an exam with multiple people. So it is in your best interest to learn that information yourself and just get the best knowledge of it. So when you're taking an exam, you know how to solve the problems on the exam.

(00:25:39):

But when it comes to projects, using AI is so useful. You can finish projects so much quicker now and you still need to understand some of the core concepts. And when using ai, you'll be able to see the core concepts that you're coding or building out, but you don't necessarily need to be there intricately changing the numbers and figuring things out in that sense, whereas before you kind of had to do that and it would take a lot longer. But for exams, you do need to understand it deeply. So I understand that professors would be like, yeah, you can use it, but when you come to the exams, you're not going to know anything.

Matt Wittstein (00:26:13):

Generally, how do you want your professors to approach you with AI rules and expectations? And I want to be explicit that while we are talking a little bit more about a CS 101 type of course here, that I think some of this conversation also belongs in that English class and some of those general education classes that may be more writing intensive.

Anna David (00:26:37):

So I think I definitely when professors are pretty explicit with that because in some cases it is really useful and they want you to use it. And if they don't say that, then I think most students are hesitant, even if they may think, oh, it seems like we could use AI for this, but the professor didn't say it, they won't, and then it's a lot harder than some other people that are using ai. Whereas there are some classes where they're really strict and if they don't tell you that ahead of time, then you can get into some bad situations if you assume the opposite. So I think it's really important for professors to be very explicit and even if it's with individual assignments, if it's not something that's of the whole course, maybe it's the first assignment, you can't use it, but after that you can. I think professors really need to be explicit about

Arav Patel (00:27:31):

When it comes to English classes and those types of areas. I think it does require for the context of what is being asked of you. So for example, if it's asking to get either your opinion or just a different outlook on a certain situation, I think that you should totally write that yourself and maybe use AI to help you with the wording and help you with how to write these and format things. However, when it comes to the actual content, using AI doesn't necessarily do anything because it's not creative in that sense and thinking of something new and giving a different outlook on a certain situation, whereas with, for example, a business document they need to write or some form of legislation or something when it comes to just concrete things, then using AI to help you just automate writing, that is definitely something that I can see being helpful because you don't want to just sit there and look at those random facts all day and having an AI to just do that for you is pretty helpful in my opinion. But yeah, when it comes to Simons, that's my perspective. So

Matt Wittstein (00:28:33):

I'm curious if you all can think back to your first introductory type of course. Do you think that chat GPT enhances learning in those foundational courses or does it actually stunt the learning in some ways?

Anna David (00:28:51):

Yes, a lot sense of a freshman on taking those classes right now, and I think in most cases in intro class it would impede the learning, except if you're asking Chad teacher to explain that concept to you, I think that can be really, really useful, which I don't think most professors are really utilizing. So that might be a place where it could definitely be used. But I think in actually doing your programming assignments and such in an intro class, it's really important for it to be the students work over chat GPEs because we leave the whole point of those foundational classes is to understand the foundations because once you've learned all those foundations, everything to do those more advanced topics that maybe you can automate some of the lower level easier things because you already understand how they work and now you can debug them and you actually know what's going on because you have actually taken most foundational classes and understood it all versus letting chat gt write everything for your foundational class and then realizing once you're in a junior or senior level class that you don't actually understand what binary research or something like that was even doing and the thinking behind it, because I feel like those foundational classes really get you to think like a computer scientist, and that's something really important that I think students come out of those classes with which they were just using in those classes, they might not get that as much.

(00:30:35):

Anna,

Matt Wittstein (00:30:36):

Sorry Anna, would you mind explaining binary search to me and our listeners that may not know exactly what that is? You brought it up a couple times.

Anna David (00:30:45):

So binary search is just a pretty basic algorithm. It's usually one of the first algorithms that you learn in computer science where it's basically you just have an array or a list of numbers that is sorted and binary searches an algorithm that efficiently finds whether say you had a list of 10 numbers and you're trying to see if 60 is in the list, binary searches, a really efficient algorithm to figure out is 60 in that list. So it's an algorithm that you learn really early on and it's one of those things that really gets you thinking like that and thinking, how can I make this efficient, which is a really important topic in any computer science, how can I make the most efficient algorithm?

Arav Patel (00:31:38):

I have a question for Anna. As a freshman, taking your intro classes, are you getting when you have to get your exams and take your exams, especially last semester, they're all handwritten exams, I'm assuming? Yeah. And they're not open note at all?

Anna David (00:31:57):

No. So my computer science classes, at least last semester, it was no notes on paper. We had weekly quizzes for my first class last semester, so we were actually tested on paper every single week to make sure that we actually understood the content from that week. So I think that really forced people to actually understand it. It also wasn't like you couldn't just put it all off until the final and try to figure everything out. Then you really had to make sure that you understood it on your own throughout the whole course.

Arav Patel (00:32:33):

I agree with that heavily. I think one way to really enforce learning and make people want to learn a lot more is having those handwritten quizzes and handwritten exams where you can't look at your notes and you're not able to do those notes. Then you're kind of forced to learn these things and learn the concepts such as binary search for example, and then sorting algorithms as you get further learning those as well and understanding them deeply. You can only really do that if you're handwriting it and handwriting how these algorithms are working, how the nodes are moving around and that sort of stuff.

Matt Wittstein (00:33:11):

So you both clearly have some pretty strong feelings and values about your own learning, but I'm curious a little bit if you feel like your peers share those values or if some of them are taking the shortcuts, we know for a fact that students are doing this, why do you think that's okay for some students, but not okay for others? Or to phrase it a different way, why do you think some of your friends decide, I'm going to take the shortcut on this one and other times decide, oh no, I'm going to do this the long way.

Anna David (00:33:46):

I think sometimes it really comes down to how much they not really care about that class, but how much they value their learning in that class or on that assignment. So I think in a lot of cases, someone who's a lot more interested in STEM may be more likely to use CHA T on a writing assignment. They don't enjoy writing or they don't really care about their learning in that class, whereas they might be less likely to use it on a computer science project because they really enjoy that and they actually coding it themselves. Whereas on the other end, someone who's more a humanities major, if they're in a computer science class, they might just not, I care as much about learning in that class because that's not the field they're going into. So they may be more likely to use because they don't feel it's impeding their learning as much. A,

Matt Wittstein (00:34:43):

Maybe I can point this one to you. What are some tangible things a professor could do with an assignment to make it more relevant and interesting to the learner that is maybe somewhat misplaced in their course? Maybe for Anna's example, for the STEM student in the English course, what could an English professor do to make the assignment relevant or for the humanities student in the CS course, what could the CS professor do to make the coding assignment relevant? Are there any tips, tricks or suggestions that you can think of?

Arav Patel (00:35:16):

Maybe allow free reign for the student to do what they want? So essentially say a humanities major comes into a CS course and the project is build a project based on what you want to do. Maybe format the project in a way where it's relating something humanities related, so they're building a way to make some sort of humanities idea more efficient or something along those lines. I dunno, I'm just saying something. And maybe they'll get more interested on it and care a lot more. So they're going to maybe use AI less or they'll use AI to help them do that. I don't necessarily think that using AI impedes learning completely. I think they can go hand in hand with each other. So even in humanities, for example, if I'm in a humanities class and I want to write something or understand how something's working such as historically what's going on, I will sometimes use chat myself to learn that information a little better and understand it more because I'll ask it more and more questions to understand how did things happen or

how does this sort of concept work? And yeah, I don't think that it necessarily impedes the way a person understands information, but when it comes to writing that whole assignment out, that's when maybe you want to write that yourself and work on it yourself and maybe presentations slash slides and having people present what they're learning, that's something that could maybe enforce people to want to learn more because then they have to explain something or a certain concept to other people.

Anna David (00:36:53):

Yeah, I definitely agree with making the assignments more relatable to anyone. I know when I was in a humanities class last semester, I was a lot more motivated to do the assignments that were more open-ended where it may have been find any recent news article and write a response to it where I could then look at more STEM or computer science articles, whereas other people could look at things that they were interested in. I was a lot more motivated to do those assignments and do them earlier on, whereas the assignments where it was I had to write about a specific thing that I wasn't interested in, I found myself procrastinating on it a lot because I just wasn't excited about doing the assignment. So then you end up doing it later on. And I think that's also the point where a lot of people then switch to chat GT because they just didn't have the motivation and now it's an hour before the deadline and they don't have time to write the whole thing, so now they're going to go to chat gt. And so I think really keeping the students motivated with something they can relate to would be really helpful in getting everyone to do it on their own.

Matt Wittstein (00:38:06):

And I think bringing up that point of time pressure and other pressures because we know our students have complex lives just like we have complex lives and you might be stressed by different things at different points. So those types of things can sort of push you to take some shortcuts. So I really appreciate that point. I want to switch gears a little bit, and part of Ryan's idea was what would it look like if we got rid of all of the rules around AI and we said, just build the coolest thing. You both have talked a little bit about how useful it could be for projects, but what do you see as the challenges that a professor might have in making sure it's still used appropriately, but allowing that full creativity, that full opportunity when they're creating a new assignment or a new course

Arav Patel (00:38:56):

With just free reign? I think that there's positives and negatives that could come out of it. I want to speak more on the technical side of things. When it comes to a CS project, for example, if you just allow free reign one, their project might get done faster and a lot quicker. However, some issues that could come about is that if the code doesn't work, which we all know sometimes in CS the code and with the way LMS work, sometimes the code just doesn't work or it's hard to debug not understanding those fundamental things such as how end points might get routed or how webhooks were when building a full stack application. Those are fundamental ideas that you kind of need to understand in order to debug, whereas I am using an LLM to try to help you debug and it continues to not work and continues to not work, then you're kind of screwed and you won't get your project done. So there's the balance of just letting free reign, but then it's also part of a computer scientist to just figure it out and figure out how to debug. So I don't really know. It's a hard double-edged sword.

Anna David (00:40:01):

Yeah, I think definitely if the students have that foundation and they really know how to debug, it can be really good to have sort of free reign because you can make those really big cool projects. I had an

internship last summer where we were really encouraged use Chachi BT to write as much as we could, but that meant that we could make a really big project and then you are actually doing it all on your own because chat GPT can't write a huge project. It can write maybe part of a single file at a time. So you really still have to understand what's going on and how everything connects. So it's not, in that case, in a bigger project, you're still learning everything that they wanted you to learn, but you're speeding up those smaller, maybe one function here and there, or half a file or some formatting of something, you're speeding that up so you can make something much bigger in a shorter period of time, but you're still getting all of that knowledge of you have to know how everything fits together for it to work. And you have to know how to debug and what's actually going on, and you have to actually understand the code that check UT is giving you because otherwise you can't debug it. So once you're past those fundamentals, I think it's fine to use chat in those bigger projects because you do still have to understand what's happening to actually make it work.

Arav Patel (00:41:41):

And also with math-based classes, and this is more in engineering fields, when you have large math equations, cha chi petite is awful at really high level math. It's just not good at it. So when you get to those areas as well, you do need to understand those fundamental concepts in mathematics, otherwise you can't proceed further in whatever you're trying to solve. So yeah, then the free reign, you can be like, yeah, you could have free reign on your math problems, but will you really get there and solve it? Probably not because it's also not that great at it. I was taking a cybersecurity course the last semester and it was pretty much a full math class, which was very interesting. But the entire time I was trying to use chat GBT to help me with some of the problems, and when I was using chat GBT for one of them was actually funny.

(00:42:39):

It said that I'm not allowed to use chat g BT, because the problem I was asking was violating their security breaches because the problem that was given was can you decrypt a message being sent from person A to person B on WhatsApp, for example? And then it was like try to decrypt it. And when I asked chay, we need to help me with that problem, it was like, we can't allow you to do that because that's a security breach with their LLM guardrails. But otherwise, even when I tried to start with other math problems, sometimes it's just wrong and then you have to understand why it's wrong and how to fix that and get the equation to work out properly. And so yeah, free reading can help, but it also sometimes won't even matter.

Anna David (00:43:20):

Yeah, and I think building off of that, I think also figuring out what's going on or checking chat GPTs work or trying to debug it also gives more learning because you're actually understanding how this works and maybe chat gpt found a more efficient or more concise method, and now you can actually learn a lot from that. So I think in that sense it can also be really useful.

Matt Wittstein (00:43:47):

You both have talked about inaccuracies and mistakes that our AI companions, our copilots can make, and the ability or the need to be able to understand the fundamentals so you can debug and assess what's wrong on a base level. How much do you all trust the output from A GPT when you're using it in sort of a googly sort of way where you're just trying to get a search and an answer on something? How do you know when to trust versus when to really dig in deep and check the accuracy of the output?

Arav Patel (00:44:22):

I'd say when I need something that's factual and I need to make sure that it is a fact, fact, I will go to Google and usually ask Google what I'm asking for and check through four different sources before I insert that maybe statistic or whatever factual statement into whatever I'm doing. But when it comes to basic stuff such as my workout tomorrow or if I'm making something code-based, I'd usually trust chat, give or take 80% of the time. And then there's the 20% of times where it's hallucinating and that what it's saying is just not correct.

Matt Wittstein (00:45:04):

Do you have an example of when you clearly knew it was hallucinating?

Arav Patel (00:45:09):

When I'm coding sometimes and building out a project, I'll look at the code itself and be like, that loop is an infinite loop. There's no stop point in that loop, so it's not going to work. I had to edit it myself. But when it comes to general facts and statistics, I don't usually ask chat GBT for statistics of anything. I don't trust it for statistics of anything, I usually just go to Google to find statistics. So I can't really say I've seen it hallucinate because I also just have never prompted it for factual information or statistics.

Anna David (00:45:48):

If it's anything factual, you can't really trust it more than maybe finding out where to find that fact. But if it's something like a computer songs project where you can ask it for some code, look through, make sure it's seems to make sense, and then just copy and paste it and share code and test it.

Arav Patel (00:46:08):

I also used ChatGBT to help me with my taxes. One time I was filling out my tax return stuff, but regardless, usually you would ask your dad, and that's what I did my first freshman year and sophomore year I would ask my dad to help me. And the way I kind of treated chat this time around was it's kind of like asking your dad those questions. How does this form work? How does the W2 form work and all that kind of stuff, just figuring that out and then chat chi. But he kind of explained each thing step by step better than my dad did. So it's like, okay, I understand how this works a little bit better now, but sometimes if your dad's telling you a statistic, you might believe him, you might not. You kind of want to do your research yourself every now and then. And that's kind of how I view it with ai. If AI gives me a stat, I'm like, I'll take that with a grain of salt and then also do my own research and look into the statistic itself.

Matt Wittstein (00:47:02):

So if Ryan were to develop a course that was completely, you can use as much AI as you want to, how would you advise him to set the bar on the quality and how he might assess the quality of work that students do in that course? And for some context here, I think we're probably thinking about an introductory computer science course that's not actually geared towards computer science majors. So thinking about the business major that wants to learn some computer science skills or the entrepreneur that wants to learn some computer science skills.

Anna David (00:47:37):

So I think a class like that would definitely benefit from having, especially at the beginning of the course, maybe a whole lesson or two odds, prompt engineering and actually figuring out how to get chat GBT to give you what you want. Because I think that becomes really important if you're going to be using it, especially in programming. You want to make sure that it's making the function that you want and not something completely different. So I think that would be really swollen a course like that.

Arav Patel (00:48:15):

One thing that I've kind of noticed while I've been coding with chat is file structure sometimes doesn't necessarily work properly. And you have to understand when building a project using VS code for example, and in an introductory course, I guess for entrepreneurs slash people that aren't in cs, one thing that they might struggle with is having the right libraries installed and having the dependencies there and making sure that everything's set up properly, which is something that I struggled with a lot even now to make sure if I'm running a project, I have everything ready to go and set up in my environment. So maybe setting up your files properly and your environments properly and learning that kind of stuff when building out a project is something to teach more than just teaching is an integer and then a string is this, and a float is this because those are very AI able topics, whereas understanding environments and dependencies are not necessarily AI able because a person that's not in CS doesn't even know to prompt an AI about those types of topics. And those are some things that are important when building out a project.

Anna David (00:49:22):

I think if the goal of the course is for business majors to understand enough to work with their developers, then I think using chat petite writes and stuff can be really useful because in that case, they don't actually need to understand the fundamentals that an actual computer science major does need to understand. They need to understand the bigger things and how those things work, and a lot those libraries and stuff that chat GT can't actually do. So I think those things would really be the most important to teach in those classes. What chat GT can do and just enough that they need to know to work with developers and understand what limitations the developers have. Because I think in a lot of cases, like a project manager who doesn't know anything about computer science, might think that something is very possible when in reality it's completely impossible. As a developer,

Matt Wittstein (00:50:24):

I want to thank you for your time and your thoughtful responses to the questions today. I can't wait to share some of this with Ryan and also with our other faculty panelists a little bit later on.

Arav Patel (00:50:35):

Thank you. It was great to be on the show. And yeah, thank you for your time and this conversation.

Anna David (00:50:40):

Yeah, thank you for having me. This was great.

Matt Wittstein (00:50:56):

Hey, Ryan, welcome back to the show.

Ryan Mattfeld (00:50:58):

Hey, thanks. Glad to be back. Glad to hear what the students had to say.

Matt Wittstein (00:51:01):

So we took your questions about navigating AI use in your introductory computer science class and exploring the potential of unrestricted AI use in courses to a couple of budding computer science students. Anna David, a first year computer science student at Carnegie Mellon University with a passion for robotics and to Arav Patel, a senior at the University of Maryland. Heading into the AI sector offered some really fascinating insights that I think can help shape your approach. The conversation revealed an interesting tension between AI as a tool and the importance of foundational learning. While both students believe that their peers generally use AI ethically to help define some important boundaries that they see amongst their peers, Anna echoed what she's hearing from her faculty that building strong foundations without AI assistance is really crucial in early computer science courses. This was reinforced by Arav's observation that spotting and fixing AI generated code errors requires really solid fundamentals of their understanding in computer science.

(00:52:01):

When we explored potential positive use cases, some practical suggestions came out. Both students saw value in using AI as a learning aid, helping explain concepts during assignments or checking work during the debugging process, which Anna noted was particularly valuable for novices. Our of shared and interesting perspective about AI accelerating work once foundational skills are established, while also pointing out AI's current limitations with complex math and engineering problems. One of the most valuable insights came from Anna's internship experience. While AI could assist with smaller tasks throughout a project, accountability for the final product remained entirely human. This mirrors their broader point about context mattering AI use might be more appropriate for quickly assembling code blocks than for tasks requiring personal reflection or creativity. The students also emphasize the importance of assignment design. Anna pointed out that students are more likely to take AI shortcuts when they don't see value in the learning activity.

(00:53:01):

This suggests that making assignments relevant and meaningful could naturally guide appropriate ai. Use the recommendations for implementation. Were practical, teach prompt engineering introduce relevant coding libraries and scaffold the setup thoughtfully. As we talked about the prospect of an unrestricted AI use course with the context that this may not be for computer science majors, Anna made the exceptional point that she would really want her project managers in the future to understand the limitations of AI and really have a sense of what is and is not possible for developers. They both acknowledged and agreed with you that this type of setting could open the door to some very large and creative projects, which would be fun to implement. To me, it doesn't seem like they actually see AI bringing a huge shift in how to teach computer science or what we might want a non-major to get out of a computer science course. That said, is there anything from their conversation that strikes a chord with you?

Ryan Mattfeld (00:53:55):

I would say that several of the things that they said are topics and concepts that we've discussed before and kind of felt were accurate, but it's good to have some evidence. So for example, one thing that stood out is Anna saying that if their student doesn't see value in the course, then they're more likely to take a shortcut. That's something that we had kind of suspected and discussed, and certainly a point of emphasis really always, but especially since chat GPT came out, was to emphasize trying to show

students how what we're teaching is relevant, how it matters, and what they can do with it to help them see the value in the course. So it's good to see a lot of that confirmation, but also certainly the new piece is, I think the piece that you highlighted as well. I had not considered the what is possible aspect, living in the world of computer science and working here for years, sometimes you lose sight.

(00:54:56):

You just know what are the limitations of code, what are the possibilities with it, what are the things we can do and can't do. You just know these things. But that relevant experience she brought in where her supervisor didn't know what was possible, and having had a pushback and teach that in the course might be a very valuable lesson. I also really appreciated the idea of talking about prompt engineering and what AI is and how it works as a good start to the course, I think would make sense as well. I think this perspective of a first year student in particular really helped being in that new to programming, new to computer science field. That's a lot of really good advice.

Matt Wittstein (00:55:35):

One of the things that struck me in the conversation is when I first asked them what do they think about their peers use of ai, they generally thought that their students, that their classmates were using AI generally, ethically. There's obviously some exceptions to that, and when I have conversations in the faculty world, it feels very different than that. It feels sort of like we think a lot of students are using it inappropriately. So I'm curious what you think about maybe that disconnect between what these students' perspectives are about themselves and their peers, as well as how that's different from our faculty perspective.

Ryan Mattfeld (00:56:13):

That's a very, very surprising point to me as well. And part of the surprise is because I have asked other students the same question and they've answered differently. So maybe it depends on the group that students are working with. Maybe it depends on the year, maybe it depends on, now that we're a few years intuit, people are starting to realize what's ethical and not ethical students in particular. Maybe it's an experience thing, but I certainly, most students that I have at least known well enough to feel comfortable asking 'em the question of like, Hey, class is over. Do you think your classmates were using AI ethically or was anybody not using it ethically? And I'd get mostly answers of, yeah, there was X student was using it all the time. And it was really frustrating because put in all the work and then they'd get something else and submit it, and they didn't really know what they were doing, but they got an answer. And really the perspective from a lot of students that I've talked to has been frustration that their peers are misusing it and benefiting from it. So it's interesting to hear that Anna's peers were not misusing it, and I suspect that there are a good contingent of students that are using it properly. A good fair chunk, maybe the ones that are highlighted in my discussions are just the ones that are specifically not using it because of the frustration and the emotion that comes out of that.

Matt Wittstein (00:57:43):

So I have one last question for you. Since we're going to take this conversation, both your questions and ideas, but also what the students said. Is there anything you want the panel to comment on specifically from the students' feedback or questions that are still just nagging at you that you really want them to focus in on?

Ryan Mattfeld (00:58:03):

I would say one piece that I feel like I'm maybe still missing and certainly would appreciate some guidance on is obviously this is an experimental course would be the first time I run it, and so finding a way to set expectations when I don't know how elaborate of a project we will be able to create, giving students expectations for where that goal will be. I know I can set clear expectations for the usage of ai, which is something that students pointed out to highlight when and how you can use AI and make those policies transparent. I feel confident I can do that in that class. The piece I'm uncertain about is, again, what level of challenge for the assignments can I get, and then how to maybe standardize or grade it or design it when I'm not exactly sure exactly how much students will be able to do.

Matt Wittstein (00:58:56):

That's great. I will definitely take that to our panel. I think that's a really good pedagogical question that can go beyond just the AI conversation, so thank you for letting me share this feedback and I can't wait to see what the panel says too. Sounds great. Thank you. Limed: Teaching with A Twist is produced in collaboration for the Center for Engaged Learning at Elon University. For more information, including show notes and additional engaged learning resources, visit www.centerforengagedlearning.org. Limed: Teaching with a Twist is a creation of Matt Wittstein, associate professor of Exercise Science at Elon University. Episodes are developed and hosted by Matt Wittstein and Dhvani Toprani, assistant Director of Learning Design and Support at Elon University. Olivia Taylor, a class of 2026 music production and recording arts major is our Summer 2024 intern and serves as a producer and editor for the show. Original music for the show is composed and recorded by Kai Mitchell and alumni of Elon University. If you enjoyed our podcast, please take a few moments to subscribe, rate, review, and share our show. We aim to bring insightful and relevant content to educators each month, and we would love to hear from you. If you're interested in being a guest on the show, do not hesitate to reach out. Our most updated contact information can be found on the Center for Engaged Learning website. Thanks for listening and stay zesty.