Generative AI in assessing Comp0233. An educators’ tale.

The COMP0233 is Research Software Engineering with Python. This is a level seven module and it's delivered in the autumn term. So we're talking about the delivery in autumn 2023. It has approximately 80 students, mostly from STEM backgrounds. And they're a combination of one year master's students, students from PhD programs in the first year and some final year undergraduates. We have a flipped classroom approach in the. There is a lot of upfront learning, um, in terms of the notes. And then when they come to the sessions, they are practical so that the three hour lecture session is effectively a lab, and they're assessed in two ways. There's two pieces of coursework. One is an individual piece of coursework, and the other one is group coursework.

So what did we know at the start of term one? So over the summer we'd run a number of workshops, and we saw that students were googling large language models to solve problems. We also know that these tools have varying coding capabilities depending on the platform and the embedded LLM. We also know that our students had various capabilities. We saw students using AI based translation tools in timetabled sessions. So we knew that that this was something they were already using. And we also knew that the Bing chat, now Co-Pilot, was available to staff via Microsoft Edge.

So what did we do? Well, we thought we would try it for ourselves. We're interested in how well it could or could not address the assignments. We could then make an informed decision about what to say to our students.

So personally, I made use of Microsoft Edge chat which was then branded as Bing Chat. I created a repository on GitHub and added all the files that we shared with students. I started from the same place as them. I made sure I used the precise mode mode for the chat. So at the time it had the capability of GPT4. I kept a log of all the chat prompts and the generated outputs. And it was very much an iterative process. So on the right hand side, here is a screenshot of the log of my initial input, what it generated, and then how I responded to that to tweak the output to what was required in the assignment. The reason why there is a train is because the first assignment was all around route planning, and the second assignment was also around trains and the London Underground.

So decisions. There were several gotchas in using this. There were a number of errors in some of the code generated, including logical errors. So when importing data from a file, you can't import it straight into a bool, otherwise everything is something, which is what the chat suggested initially. So you'd have to go back and fix it and various other things. It didn't always create things in the way that met the assignment brief. But because of this and because we knew that actually that there was a strong emphasis in the assignment on how they managed their project as well as the output. We decided that some use would be permitted, but we said that they could only use free to access tools or those available through an educational license. And if they had used it, we asked them to submit a file outlining what they'd done and why, and if they could give us an example and the exact text from the assignment. One brief is kind of given on this screenshot.

So what did the students do? Re: assignment one.

Well.

We received usage files from 8 out of 82 students. There was a mix of use, really. Some of them just said ChatGPT and didn't specify which version. Others specifically mentioned that they'd used GPT 3.5 and others were using GitHub Co-Pilot. One person was using the PyCharm plugin, which meant that it was kind of always there as he was writing code. And the other one was using the student developer pack, which is all part of the educational license for GitHub.

So how do they use it?

Them did various things. One of them actually just wanted to see what you could do, much the same as us. They were curious as to whether it actually improves the quality of their coding or not. Others where maybe their coding wasn't quite strong, we'd seen that there were varying levels in the class, were using it to understand some of the concepts. Specifically those that weren't used to object orientated program. They were asked to create a data class and they didn't necessarily know what that was. Some use was around kind of specific syntax and generating examples of things that they could then modify themselves for their assignments. And for others it was it was for simple tasks or more straightforward tasks like generating the docstring. A doctoring is the help for methods that you would put in your code.

On the whole though, they found that it can contain errors.It's not always correct. Like the the issue of importing data straight into a bool. So bool is a boolean 0 (False) or 1(True). If you bring data straight in, because there is data there, it always assumes it's true. The generated outputs didn't always match the assignment requirements. There were very specific things in the assignment. It didn't always kind of test things the way they wanted to, or the variable names weren't right or the logic wasn't quite right. They almost always had to modify the output. They couldn't just use what they'd been given, there was always something they had to tweak.

So in kind of summary.

Well, actually the student outcomes were comparable to previous years. Students generally do quite well on the module, they continue to do so this year. Regardless of whether they used the large language models or not. For those that did use them, they used them as a learning aid,, very much to kind of improve their understanding of the language and some of the key concepts. But I think the most important thing for this is that they were double checking the outputs, that they were critical of what it was giving them, and they were checking across other sources.

We know these capabilities are increasingly in built into development environments. It's there. It's in the tools that we use to teach them, it's now in the Microsoft estate that they that they can access across the institution, in their accounts and so on. So actually using it in their assignments is authentic, and it's going to be what's available to them, when they leave the institution and go to work. So experiencing use, seeing what it can and can't do is going to be really important. And there's also some debate, actually ongoing debate about the impact of these tools and code quality. So there's some reports that it produces poor code. Others aren't so clear. So them having that experience of seeing what it can and what it can't do, is incredibly important. And and for us, it was, it was really useful to see what it could do and how and what that meant for our assignment itself.

Here is just the image credits. Some of the images came from Pixabay and are CC0. The rest were from the Bing image search within PowerPoint and are Creative Commons.

Thank you for listening.