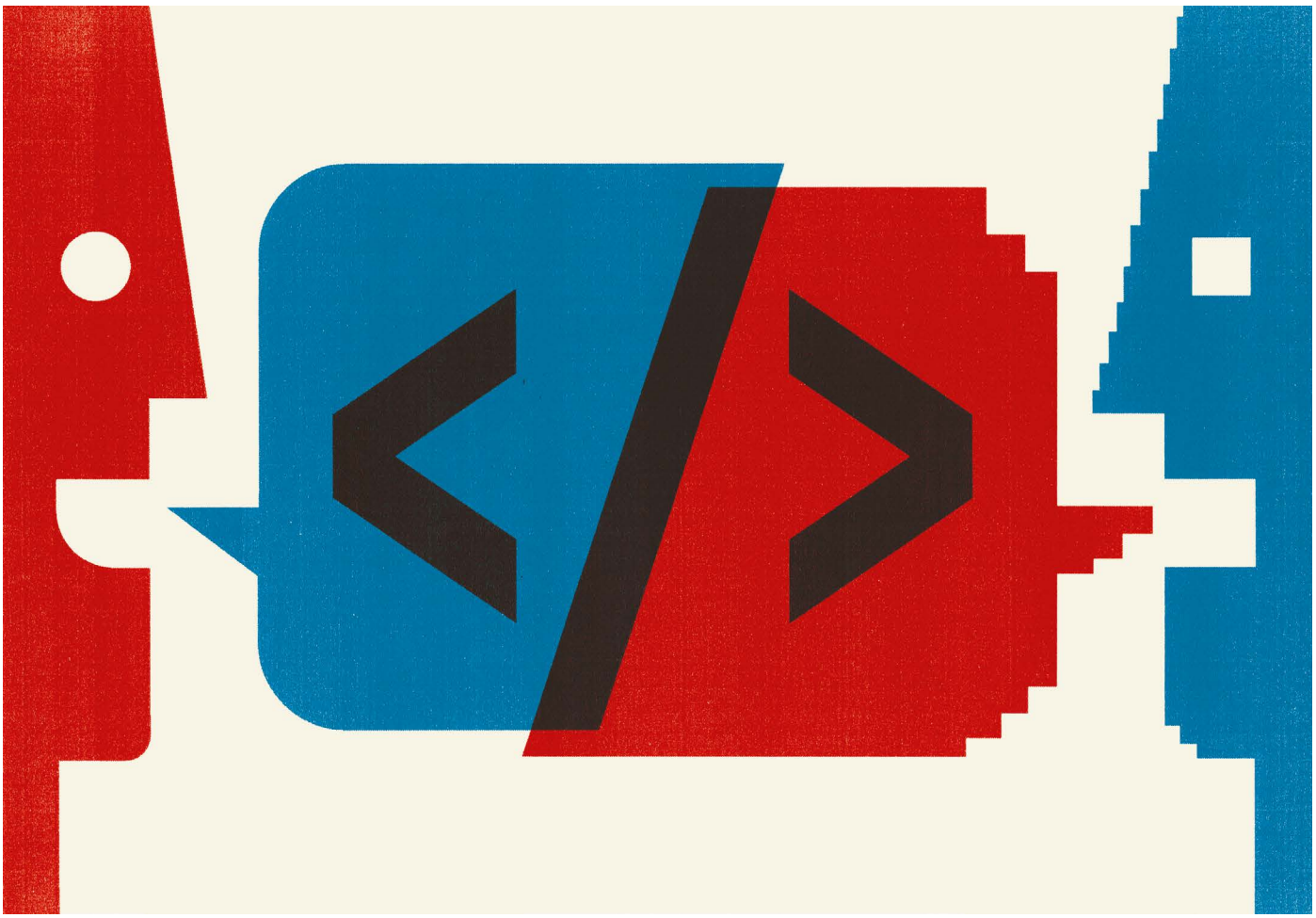

ILLUSTRATION BY THE PROJECT TWINS

SIX TIPS FOR BETTER CODING WITH CHATGPT

Although powerful, the tools are not as intelligent as they seem. Use them with caution, computer scientists warn. **By Jeffrey M. Perkel**

Unless you've been living under a rock, you know about ChatGPT. The chatbot, driven by artificial intelligence (AI) and created by OpenAI in San Francisco, California, provides eerily human-like responses to user questions (called prompts) on almost any subject. ChatGPT is trained on a vast corpus of text, and its ability to engage in text-based conversation means that users can refine its responses. Even if its initial answers are wonky, it often eventually produces accurate results, including software code.

Researchers can use ChatGPT to debug and annotate code, translate software from one programming language to another and perform rote, boilerplate operations, such as plotting data. A March preprint reported that the program could solve 76% of 184 tasks in an introductory bioinformatics course, such as working with spreadsheets, after a single try and 97% within seven attempts[1].

That's good news for researchers who feel uncomfortable coding, or who lack the budget to employ a full-time programmer – for them, chatbots can be a democratizing tool.

Yet for all their apparent sentience, chatbots are not intelligent. They have been called stochastic parrots, randomly echoing back what they've seen before. Amy Ko, a computer scientist at the University of Washington in Seattle, invokes a long-running US quiz show to describe the tool's limitations, writing on the Mastodon social-media site: "ChatGPT is like a desperate former Jeopardy contestant who stopped following pop culture in 2021 but really wants to get back into the game, and is also a robot with no consciousness, agency, morality, embodied cognition, or emotional inner life." (The data used to train ChatGPT only extend into 2021.)

In short, ChatGPT and related tools based on large language models (LLMs), which include Microsoft Bing and GitHub Copilot, are incredibly powerful programming aids, but must be used with caution. Here are six ways to do so.

Choose your applications

Chatbots work best for small, discrete programming tasks, such as loading data, performing basic data manipulations and creating visualizations and websites. But that's not the same as software engineering, says Neil Ernst, a computer scientist at the University of Victoria in Canada.

"Software engineering is a lot more than just solving a programming puzzle," Ernst explains. "There's thinking about test frameworks,

writing maintainable code and understanding the trade-offs involved in building a system" – such as that between speed and readability. "I don't think that the current tools are solving any of those issues."

That leaves many tasks they can do, such as remembering the syntax for creating visualizations with Matplotlib, a graphing library for the programming language Python. In that sense, chatbots are like a conversational interface to Stack Overflow, an online question and answer forum for programmers. "That's not stuff that anyone particularly enjoys writing," says Ernst, "and it saves time for us to ask the tough analytical questions that we might have about the data."

Chatbots are also good at explaining why code doesn't work. Emery Berger, a computer scientist at the University of Massachusetts Amherst, has exploited those abilities to build several helpful tools. One, called cwhy, uses ChatGPT to explain compiler errors in code written in the programming languages C, C++ and Rust. Another, ChatDBG, provides a conversational interface for debugging, and a third, Scalene, uses the AI to suggest code optimizations to improve performance.

Chatbots can even translate code from one programming language to another. Mathieu Coppey, a biophysicist at the Curie Institute in Paris, is using ChatGPT to help him move from MATLAB, his preferred language, to Python. Using Google and online forums, he typically needs days to get his Python code working. "Now, I can do that in an hour or so," he says.

Trust, but verify

Chatbots might not always know what they're talking about, but they certainly sound like they do. In some cases, the AI doesn't understand the question; at other times, it provides an incorrect answer. When the code fails to run, such mistakes are obvious. Sometimes, however, the code runs but yields the wrong result.

According to a study[2] co-authored by linguist Emily Morgan at the University of California, Davis, chatbots – like the human-written code on which they were trained – often create what she calls "simple, stupid bugs". These single-line errors, such as using > instead of >= in a conditional statement, are easy to fix, but hard to find. "If you don't know enough to tell the difference between something correct and something that's effectively nonsense, then you could get yourself in trouble," she says.

Iza Romanowska, a complexity scientist who studies ancient civilizations at the Aarhus Institute of Advanced Studies in Denmark, has used ChatGPT to produce code in a language called NetLogo. Because there's less online code written in NetLogo than in the languages Python and R, ChatGPT is less fluent in it. Sometimes, the AI peppers its suggested code with functions that don't actually exist, she says – a behaviour sometimes called hallucination.

The bottom line is not to blindly accept what ChatGPT gives you – read it carefully and test it. Make sure it performs as expected on 'edge cases' – for instance, does an algorithm to sort *n* numbers include the *n*th number? Patrick Lam, a computer scientist at the University of Waterloo in Canada, says: "I wouldn't trust this further than I can throw it."

Think safety

Chatbots output code that reflects their training data. That's not always a good thing, says Ko. "The aggregate quality of code on the web that's shared, that these [chatbots] are trained on, is actually quite low."

Just as random code online is unlikely to be particularly efficient or robust, so too is chatbot-generated code. It might not work well on large data sets, for instance, and can contain security vulnerabilities.

"I wouldn't trust this further than I can throw it."

Brendan Dolan-Gavitt, a computer scientist at New York University, says that when Github's Copilot programming tool launched in 2021, he and his team tested it in 89 security-relevant scenarios. One was the ability to check for malformed queries using the language SQL that could corrupt a database – known as an SQL-injection attack[3]. "About 40% of the time, Copilot was producing code that was vulnerable." That's a moving target – when Dolan-Gavitt put those scenarios to a newer version of the LLM underlying ChatGPT, called GPT-4, the error rate fell to 5%.

Still, it pays to check your code. But also consider the application – not everything is mission-critical. The web interface to a database or visualization tool, for instance, might require extra vigilance. But if you know what the answer to your programming problem should look like, "Just go for it", says computer scientist Sayash Kapoor at Princeton University in New Jersey, "because it's easy to check if you're wrong."

Iterate

Chatbot-based coding, says Ko, "is not a single-shot sort of experience". It's a conversation. "You write something, you get something back, you read it sceptically, you ask for more detail, you ask for it to fix something."

Gangqing (Michael) Hu, who runs the bioinformatics core facility at West Virginia University in Morgantown, capitalized on that iterative workflow to develop a method that beginners in bioinformatics can use to optimize chatbot prompts, called OPTIMAL[4]. Users provide detailed prompts, test the replies and feed back to the chatbot to tweak its responses. That can include questions about errors as well as tweaks to the prompt itself. "Communication is the key," Hu explains.

If you get stuck, try adjusting the settings, suggests Xijin Ge, a bioinformatician at South Dakota State University in Brookings. The ChatGPT 'temperature' setting, for instance, controls creativity – the higher the temperature, the more creative the output. "Sometimes it works," Ge says.

But not always – in some cases, "you'll have to intervene and take over", says Ko.

Anthropomorphize

Chatbots aren't people, but it can be helpful to treat them that way. "Treat this AI as a summer intern," Ge advises – a college student who is hard-working and eager to please, but also inexperienced and error-prone.

Avoid ambiguity, and break your problem up into smaller pieces, suggests Paul Denny, a computer scientist at the University of Auckland, New Zealand.

Another tip: direct the chatbot to assume a role, such as a biologist who is fluent in Python. Specify the tools or programming libraries you would like to use. Such directives can help the chatbot to get "into the right probabilistic space", says Ko – that is, home in on the text that is most likely to follow the prompt.

For instance, one prompt in Hu's study[4] asked ChatGPT: "Act as an experienced bioinformatician proficient in ChIP-Seq data analysis, you will assist me by writing code with number of lines as minimal as possible. Reset the thread if asked to. Reply "YES" if understand."

And, if possible, provide starting code, comments and expected results. "Examples can really help ChatGPT to target it in the right direction," says Dong Xu, a computer scientist at the University of Missouri, Columbia.

Embrace change

Finally, LLMs are constantly evolving, and becoming more powerful. That's good news for researchers, although it will keep them on their toes. Prompt lengths are increasing, allowing for more nuanced responses. And new tools are constantly emerging. One plug-in called Code Interpreter turns ChatGPT into a digital data analyst, allowing users to upload data sets, ask questions of their data and download results. As one blogger on AI put it, "It's like having a conversation with your data. How cool is that?"

Jeffrey M. Perkel is technology editor at *Nature*.

1. Piccolo, S. R. *et al*. Preprint at https://arxiv.org/abs/2303.13528 (2023).
2. Jesse, K. *et al*. Preprint at https://arxiv.org/abs/2303.11455 (2023).
3. Pearce, H. *et al*. Preprint at https://arxiv.org/abs/2108.09293 (2021).
4. Shue, E. *et al*. *Quant. Biol.* https://doi.org/10.15302/J-QB-023-0327 (2023).