Why your boss isn't worried about AI

Boyd Kane

2025-10-14

(a note for technical folk)[1]

When it comes to understanding the dangers of AI systems, the general public has the worst kind of knowledge: that what you know for sure that just ain't so.

After 40 years of persistent badgering, the software industry has convinced the public that bugs can have disastrous consequences. This is great! It is *good* that people understand that software can result in real-world harm. Not only does the general public mostly understand the dangers, but they mostly understand that bugs can be fixed. It might be expensive, it might be difficult, but it can be done.

The problem is that this understanding, *when applied to AIs like ChatGPT*, is completely wrong. The software that runs AI acts very differently to the software that runs most of your computer or your phone. Good, sensible assumptions about bugs in regular software actually end up being harmful and misleading when you try to apply them to AI.

Attempting to apply regular-software assumptions to AI systems leads to confusion, and remarks such as:

> "If something goes wrong with ChatGPT, can't some boffin just think hard for a bit, find the missing semi-colon or whatever, and then fix the bug?"

or

> "Even if it's hard for one person to understand everything the AI does, surely still smart people who individually understand small parts of what the AI does?".

or

> "Just because current systems don't work perfectly, that's not a problem right? Because eventually we'll iron out all the bugs so the AIs will get more reliable over time, like old software is more reliable than new software."

[1]This article is attempting to bridge a gap between the technical and the non-technical, so I'm going to be quite lax with the jargon here and there. By "AI" I'm referring to 2025 frontier LLMs. I'm also going to be making some sweeping statements about "how software works", these claims mostly hold, but they break down when applied to distributed systems, parallel code, or complex interactions between software systems and human processes. Feel free to debate me in the comments if you think this piece discussing how experts struggle to emphasise with novices should have had more jargon (:

If you understand how modern AI systems work, these statements are all painfully incorrect. But if you're used to regular software, they're completely reasonable. I believe there is a gap between the experts and the novices in the field:

- the experts don't see the gap because it's so obvious, so they don't bother explaining the gap
- the novices don't see the gap because they don't know to look, so they don't realise where their confusion comes from.

This leads to frustration on both sides, because the experts feel like their arguments aren't hitting home, and the novices feel like all arguments have obvious flaws. In reality, the experts and the novices have different, unspoken, assumptions about how AI systems work.

1 Some example false beliefs

To make this more concrete, here are some example ideas that are perfectly true when applied to regular software but become harmfully false when applied to modern AIs:

1.1 Software vulnerabilities are caused by mistakes in the code

In regular software, vulnerabilities are caused by mistakes in the lines of code that make up the software. There might be hundreds of thousands of lines of code, but code doesn't take up much space so this is only around 50MB of data, about the size of a small album of photos.

But in modern AI systems, vulnerabilities or bugs are usually caused by problems in the data used to train an AI[2]. It takes thousands of gigabytes of data to train modern AI systems, and bad behaviour isn't caused by any single bad piece of data, but by the combined effects of significant fractions of the dataset. Because these datasets are so large, *nobody knows everything that an AI is actually trained on.* One popular dataset, FineWeb, is about 11.25 trillion words long[3], which, if you were reading at about 250 words per minute, would take you over 85 thousand years to read. It's just not possible for any single human (or even a team of humans) to have read everything that an LLM has read during training.

1.2 Bugs in the code can be found by carefully analysing the code

With regular software, if there's a bug, it's possible for smart people to carefully read through the code and logically figure out what must be causing the bug.

With AI systems, almost all bad behaviour originates from the data that's used to train them[4], but it's basically impossible to look at misbehaving AI and figure out parts of the training data caused that bad behaviour. In practice, it's rare to even attempt this, researchers will

[2]It can also come from the reward model used during RLHF, but the reward model is still trained from data at the end of the day. It can also come from prompt injections, but those also only work because of the data.

[3]FineWeb is 15 trillion tokens, each token is about 0.75 words, 11.25 trillion words.

[4]It can also come from the reward model used during RLHF, but the reward model is still trained from data at the end of the day. It can also come from prompt injections, but those also only work because of the data.

retrain the AI with more data to try and counteract the bad behaviour, or they'll start over and try to curate the data to not include the bad data.

You cannot logically deduce what pieces of data caused the bad behaviour, you can only make good guesses. For example, modern AIs are trained on lots of mathematics proofs and programming tasks, because that seems to make them do better at reasoning and logical thinking tasks. If an AI system makes a logical reasoning mistake, it's impossible to attribute that mistake to any portion of the training data, the only answer we've got is to use more data next time.

I think I need to emphasise this: With regular software, we can pinpoint mistakes precisely, walk step-by-step through the events leading up to the mistake, and logically understand why that mistake happened. When AIs make mistakes, *we don't understand the steps that caused those mistakes*. Even the people who made the AIs don't understand why they make mistakes[5]. Nobody understands where these bugs come from. We sometimes kinda have a rough idea about why they maybe did something unusual. But we're far, far away from anything that guarantees the AI won't have any catastrophic failures.

1.3 Once a bug is fixed, it won't come back again

With regular software, once you've found the bug, you can fix the bug. And once you've fixed the bug, it won't re-appear[6]. There might be a bug that causes similar problems, but it's not the *same* bug as the one you fixed. This means you can, if you're patient, reduce the number of bugs over time and rest assured that removing new bugs won't cause old bugs to re-appear.

This is not the case with AI. It's not really possible to "fix" a bug in an AI, because even if the AI was behaving weirdly, and you retrained it, and now it's not behaving weirdly anymore, you can't know for sure that the weird behaviour is *gone*, just that it doesn't happen for the prompts you tested. It's entirely possible that someone can find a prompt you forgot to test, and then the buggy behaviour is back again!

1.4 Every time you run the code, the same thing happens

With regular software, you can run the same piece of code multiple times and it'll behave in the same way. If you give it the same input, it'll give you the same output.

Now *technically* this is still true for AIs, if you give them exactly the prompt they'll respond in exactly the same way. But practically, it's very far from the truth[7]. Even *tiny* changes to the input of an AI can have dramatic changes in the output. Even innocent changes like adding a question mark at the end of your sentence or forgetting to start your sentence with a capital letter can cause the AI to return something different.

Additionally, most AI companies will slightly change the way their AIs respond, so that they say slightly different things to the same prompt. This helps their AIs seem less robotic and

[5] Anthropic is doing very good work to try and figure out why AIs think the way they do, but even the state of the art does not have a full understanding of these AIs, and what understanding we do have, is often partial and with significant gaps.

[6] You *are* writing tests to prevent regressions, right? *right?!*

[7] see for example, this blog post from Mira Murati's Thinking Machines Lab

more natural.

1.5 If you give specifications beforehand, you can get software that meets those specifications

With regular software, this is true. You can sit with stakeholders to discuss the requirements for some piece of software, and then write code to meet those requirements. The requirements might change, but fundamentally you can write code to serve some specific purpose and have confidence that it will serve that specific purpose.

With AI systems, this is more or less false. Or at the very least, the creators of modern AI systems have far far less control about the behaviour the AIs will exhibit. We understand how to get an AI to meet narrow, testable specifications like speaking English and writing code, but we don't know how to get a brand new AI to achieve a certain score on some particular test or to guarantee global behaviour like "never tells the user to commit a crime". The best AI companies in the world have basically one lever which is "better", and they can pull that lever to make the AI better, but nobody knows precisely what to do to ensure an AI writes formal emails correctly or summarises text accurately.

This means that we don't know what an AI will be capable of before we've trained it. It's very common for AIs to be released to the public for months before a random person on Twitter discovers some ability that the AI has which even its creators didn't know about. So far, these abilities have been mostly just fun, like being good at Geoguessr (Figure 1).

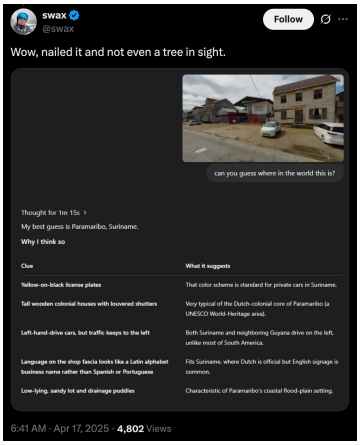

Figure 1: Geoguessr map

Or making photos look like they were from a Studio Ghibli film (Figure 2).

But there's no reason for these hidden abilities to always be positive. It's entirely possible that some dangerous capability is hidden in ChatGPT, but nobody's figured out the right prompt just yet.

While it's possible to demonstrate the safety of an AI for a specific test suite or a known threat, it's impossible for AI creators to definitively say their AI will never act maliciously or dangerously for any prompt it could be given.

Figure 2: Ghibli tweet

2 Where to go from here

It is *good* that most people know the dangers of poorly written or buggy software. But this hard-won knowledge about regular software is misleading the public when it gets applied to AI. Despite the cries of "inscrutable arrays of floating point numbers", I'd be surprised if a majority of people know that modern AI is architecturally different from regular software.

AI safety is a complicated and subtle argument. The best we can do is to make sure we're starting from the same baseline, and that means conveying to our contemporaries that if it all starts to go wrong, we cannot just "patch the bug"[8].

If this essay was the first time you realised AI was fundamentally different from regular software, let me know, and share this with a friend who might also not realise the difference.

If you always knew that regular software and AIs are fundamentally different, talk to your family and non-technical friends, or with a stranger at a coffee shop. I think you'll be surprised at how few people know that these two are different.

If you're interested the dynamics between experts and novices, and how gaps between them arise, I've written more about the systemic biases encountered by experts (and the difficulties endured by novices) in this essay: Experts have it easy.

Thanks to Sam Cross and Caleb for reviewing drafts of this essay.

[8] Personally, I think some more empathy is needed when having good faith discussions with non-technical folk. Communication is *empirically hard*, in that it often goes wrong in practice, even if it feels easy to do.