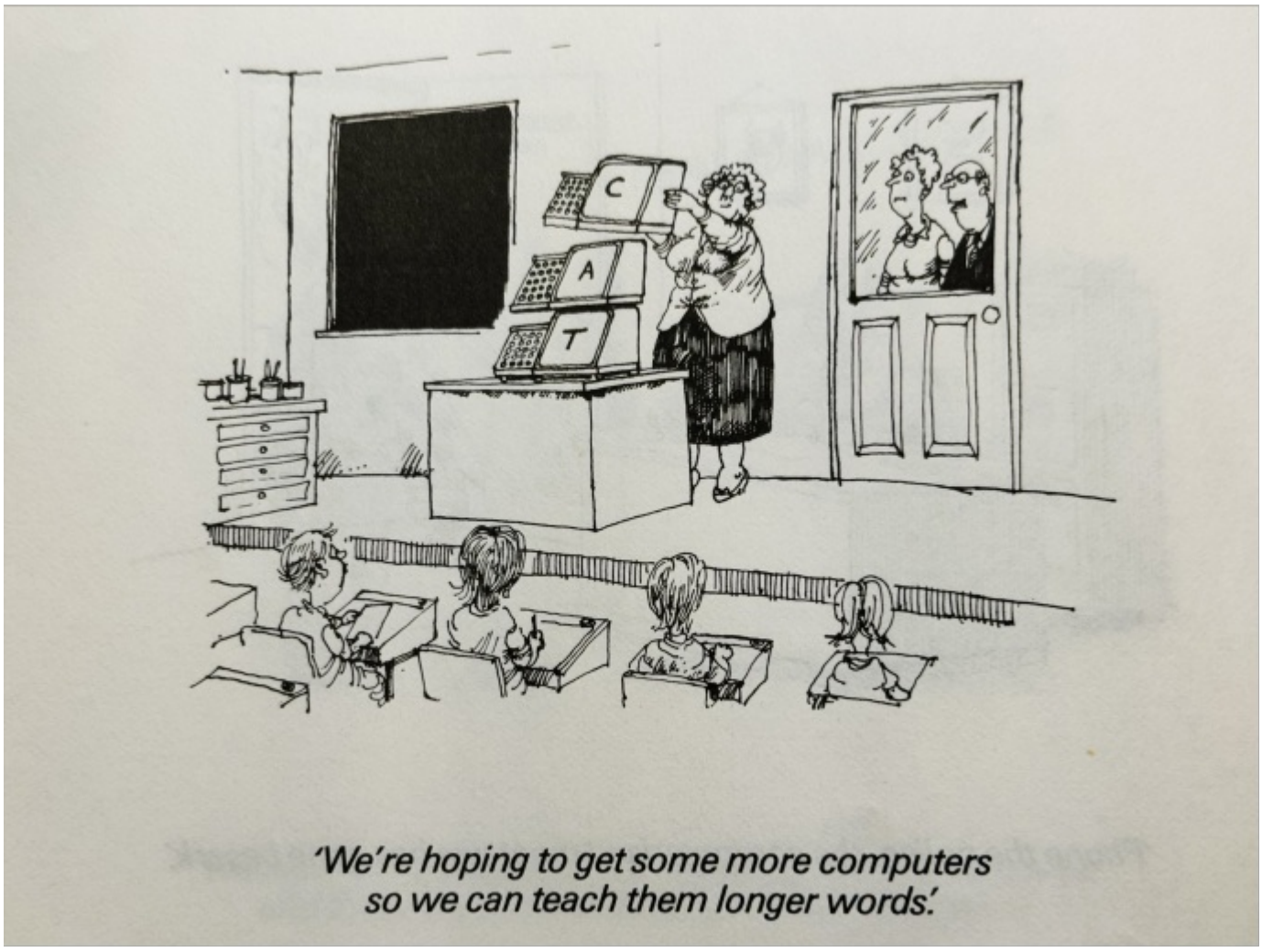
Looking for AI use-cases

We've had ChatGPT for 18 months, but what's it for? What are the use-cases? Why isn't it useful for everyone, right now? Do Large Language Models become universal tools that can do 'any' task, or do we wrap them in single-purpose apps, and build thousands of new companies around that?



This image comes from a book by Martin Honeysett called 'Microphobia', published in 1982. It's full of great jokes and I could have used almost any of them, because they're all making the same point - what are you supposed to do with this thing? I played Saboteur on the ZX Spectrum that my father bought that year, but what else?

A couple of years earlier, Dan Bricklin had found one answer: he saw a professor making a spreadsheet with chalk, on a blackboard, and realised that you could do this in 'software'. So he made VisiCalc, the first successful computer spreadsheet, and when he showed it to accountants it blew their minds: they could do a week's work in an afternoon. An Apple II to run VisiCalc cost at least $12,000* adjusted for inflation, but even so, people reached for their cheque-books the moment they saw it: computer spreadsheets changed the world, for accountants.

However, if you had showed VisiCalc to a lawyer or a graphic designer, their response might well have been ‘that’s amazing, and maybe my book-keeper should see this, but I don’t do that’. Lawyers needed a word processor, and graphic designers needed (say) Postscript, Pagemaker and Photoshop, and that took longer.

I’ve been thinking about this problem a lot in the last 18 months, as I’ve experimented with ChatGPT, Gemini, Claude and all the other chatbots that have sprouted up: ‘this is amazing, but I don’t have that use-case’.

The one really big use-case that took off in 2023 was writing code, but I don’t write code. People use it for brainstorming, and making lists and sorting ideas, but again, I don’t do that. I don’t have homework anymore. I see people using it to get a generic first draft, and designers making concept roughs with MidJourney, but, again, these are not my use-cases. I have not, yet, found anything that matches with a use-case that I have. I don’t think I’m the only one, either, as is suggested by some of the survey data - a lot of people have tried this, especially since you don’t need to spend $12,000 on a new Apple II, and it’s very cool, but how much do we use it, and what for?

This wouldn’t matter much (‘man says new tech isn’t for him!’), except that a lot of people in tech look at ChatGPT and LLMs and see a step change in generalisation, towards something that can be universal. A spreadsheet can’t do word processing or graphic design, and a PC can do all of those but someone needs to write those applications for you first, one use-case at a time. But as these models get better and become multi-modal, the really transformative thesis is that one model can do ‘any’ use-case without anyone having to write the software for that task in particular.

Suppose you want to analyse this month’s customer cancellations, or dispute a parking ticket, or file your taxes - you can ask an LLM, and it will work out what data you need, find the right websites, ask you the right questions, parse a photo of your mortgage statement, fill in the forms and give you the answers. We could move orders of magnitude more manual tasks into software, because you don’t need to write software to do each of those tasks one at a time. This, I think, is why Bill Gates said that this is the biggest thing since the GUI. That’s a lot more than a writing assistant.

It seems to me, though, that there are two kinds of problem with this thesis.

The narrow problem, and perhaps the ‘weak’ problem, is that these models aren’t quite good enough, yet. They will get stuck, quite a lot, in the scenarios I suggested above. Meanwhile, these are probabilistic rather than deterministic systems, so they’re much better for some kinds of task than others. They’re now very good at making things that look right, and for some use-cases this is what you want, but for others, ‘looks right’ is different to ‘right’. Error rates and ‘hallucinations’ are improving all the time, and becoming more manageable, but we don’t know where this will go - this is

one of the big scientific arguments around generative AI (and indeed AGI). Meanwhile, whatever you think these models will be in a couple of years, there's a lot that isn't there today. These screenshots are a nice example of a use-case that I do have, that should work, and doesn't - yet.

The deeper problem, I think, is that no matter how good the tech is, you have to think of the use-case. You have to see it. You have to notice something you spend a lot of time doing and realise that it could be automated with a tool like this.

Some of this is about imagination, and familiarity. It reminds me a little of the early days of Google, when we were so used to hand-crafting our solutions to problems that it took time to realise that you could 'just Google that'. Indeed, there were even books on how to use Google, just as today there are long essays and videos on how to learn 'prompt engineering.' It took time to realise that you could turn this into a general, open-ended search problem, and just type roughly what you want instead of constructing complex logical boolean queries on vertical databases. This is also, perhaps, matching a classic pattern for the adoption of new technology: you start by making it fit the things you already do, where it's easy and obvious to see that this is a use-case, *if you have one*, and then later, over time, you change the way you work to fit the new tool.

However, the other part of this pattern is that it's not the user's job to work out how a new tool is useful. Dan Bricklin, and in principle all software, had three steps: he had to realise that you could put a spreadsheet into software, then he had to design and code it (and get that right), and then he had to go out and tell accountants why this was great.

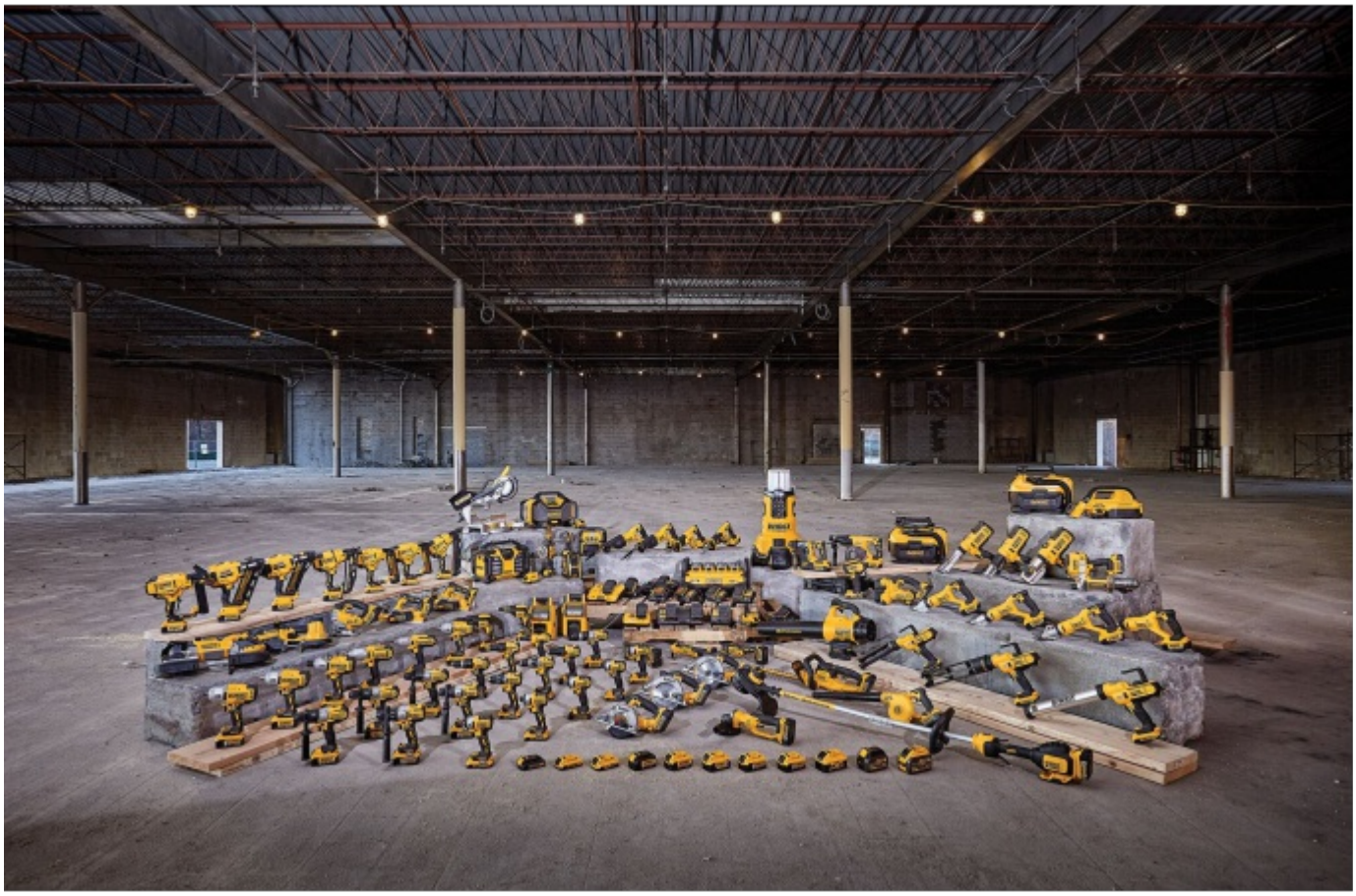
In that case he had perfect product-market fit almost immediately and the product sold itself, but this is very rare. The concept of product-market fit is that normally you have to iterate your idea of the product and your idea of the use-case and customer towards each other - and then you need sales. The great recurring fallacy in productivity software startups is that you can sell bottom-up without a sales force, because the users will see it and want it. The reality, with a tiny number of exceptions, has always been that only a very small percentage of your target users are interested and ready to explore a new tool, and for the rest, you will need to sell to them.

Hence, one hypothesis today might be that generative AI could remove or minimise Dan Bricklin's work actually to build the product, but you still need to realise that you could do this, make something tangible that expresses that, and then go out and tell people. People know they're doing taxes, but most of the things we automate are things we don't really see or realise we're doing as a separate, discrete task that could be automated until someone points them out and tries to sell us software.

Meanwhile, spreadsheets were both a use-case for a PC and a general-purpose substrate in their own right, just as email or SQL might be, and yet all of those have been unbundled. The typical big company today uses hundreds of different SaaS apps, all them, so to speak, unbundling something out of Excel, Oracle or Outlook. All of them, at their core, are an idea for a problem and an idea for a workflow to solve that problem, that is easier to grasp and deploy than saying 'you could do that in Excel!' Rather, you instantiate the problem and the solution in software - 'wrap it', indeed - and sell that to a CIO. You sell them a problem. And meanwhile, you probably don't want to give ChatGPT to Dwight or Big Keith from The Office and tell them to use it for invoicing, anymore than you tell them to use Excel instead of SAP.

Hence, the cognitive dissonance of generative AI is that OpenAI or Anthropic say that we are very close to general-purpose autonomous agents that could handle many different complex multi-stage tasks, while at the same time there's

a ‘Cambrian Explosion’ of startups using OpenAI or Anthropic APIs to build single-purpose dedicated apps that aim at one problem and wrap it in hand-built UI, tooling and enterprise sales, much as a previous generation did with SQL. Back in 1982, my father had one (1) electric drill, but since then tool companies have turned that into a whole constellation of battery-powered electric hole-makers. One upon a time every startup had SQL inside, but that wasn’t the product, and now every startup will have LLMs inside.



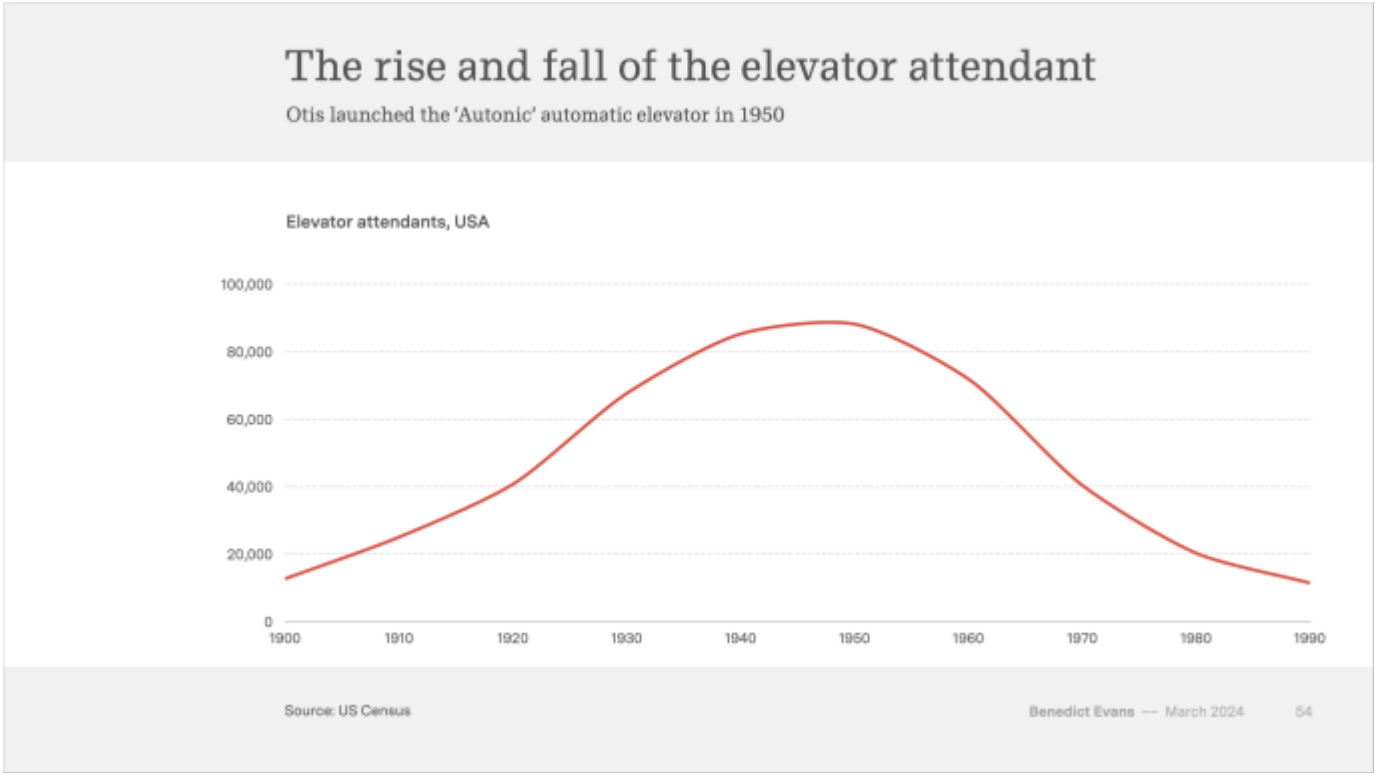
I often compared the last wave of machine learning to automated interns. You want to listen to every call coming into the call centre and recognise which customers sound angry or suspicious: doing that didn’t need an expert, just a human (or indeed maybe even a dog), and now you could automate that entire class of problem. Spotting those problems and building that software takes time: machine learning’s breakthrough was over a decade ago now, and yet we are still inventing new use-cases for it - people are still creating companies based on realising that X or Y is a problem, realising that it can be turned into pattern recognition, and then going out and selling that problem.

You could propose the current wave of generative AI as giving us another set of interns, that can make things as well as recognise them, and, again, we need to work out what. Meanwhile, the AGI argument comes down to whether this could be far, far more than interns, and if we had that, then it wouldn’t be a tool anymore.

But even if I had an actual human intern, it might be quite hard for them to solve the ‘one-shot’ request in my screenshots above. You’d have to know that I’m asking for a time-series dataset, with probably one number per year but perhaps one per decade, of people as employees by occupation (not, say, people employed *by* elevator operators), on a national and not state basis, and then you’d go to the US Census website and discover that it does collect this kind of thing, but on several different layers of detail, at different intervals, with different definitions, and it changes the definitions every few decades, and stopped collecting ‘elevator operators’ at some point (so it’s not in the current data at all, only the past data), and meanwhile the website has dozens and dozens of different data tools and sources, and it could be an entire profession just to know how to find anything.

At that point, I’d wander back to the intern’s desk and tell them that they should try FRED, and if that doesn’t have it then it would be quicker to type the data in, one year at a time, from scans of the old Statistical Abstracts, and that they’re actually easier to search using the copies in Google Books, that are scanned from random university libraries.

This is a nice illustration of the old joke that a programmer will spend a week automating a task that would take a day to do by hand. It's also a great automation chart and I spent ages typing all of this in by hand, so I'm using it.



How much embodied knowledge is that? Can you get there with a better model? A multi-modal agent? Multi-agent collaboration? Or, is it better to capture all of that embodied knowledge with a GUI, in a dedicated app or service of some kind, where the choices and options are pre-defined by someone who understands data retrieval or taxes or parking ticket disputes? A GUI tells the users what they can do, but it also tells the computer everything we already know about the problem, and with a general-purpose, open-ended prompt, the user has to think of all of that themselves, every single time, or hope it's already in the training data. So, can the GUI itself be generative? Or do we need another whole generation of Dan Bricklins to see the problem, and then turn it into apps, thousands of them, one at a time, each of them with some LLM somewhere under the hood?

On this basis, we would still have an orders of magnitude change in how much can be automated, and how many use-cases can be found for LLMs, but they still need to be found and built one by one. The change would be that these new use-cases would be things that are still automated one-at-a-time, but that could not have been automated before, or that would have needed far more software (and capital) to automate. That would make LLMs the new SQL, not the new HAL9000.

** Visicalc needed an Apple II with 32k of RAM, and including a floppy disk drive, printer and monitor, Apple's list price in 1979 was $2,875 (plus sales tax), which is around $12,000 in 2024 dollars.*

ARTIFICIAL INTELLIGENCE, PRODUCTIVITY19 APRIL 2024

Previous

AI and problems of scale

Next

The problem of AI ethics