AI and the automation of work ChatGPT and generative AI will change how we work, but how different is this to all the other waves of automation of the last 200 years? What does it mean for employment? Disruption? Coal consumption?

2 July 2023

Pretty much everyone in tech agrees that generative AI, Large Language Models and ChatGPT are a generational change in what we can do with software, and in what we can automate with software. There isn't much agreement on anything else about LLMs - indeed, we're still working out what the arguments are - but everyone agrees there's a lot more automation coming, and entirely new kinds of automation. Automation means jobs, and people.

This is also happening very fast: ChatGPT has (apparently) over 100m users after just six months, and this data from Productiv suggests it's already a top-dozen 'shadow IT' app. So, how many jobs is this going to take, how fast, and can there be new jobs to replace them?

We should start by remembering that we've been automating work for 200 years. Every time we go through a wave of automation, whole classes of jobs go away, but new classes of jobs get created. There is frictional pain and dislocation in that process, and sometimes the new jobs go to different

people in different places, but over time the total number of jobs doesn't go down, and we have all become more prosperous.

When this is happening to your own generation, it seems natural and intuitive to worry that this time, there aren't going to be those new jobs. We can *see* the jobs that are going away, but we can't predict what the new jobs will be, and often they don't exist yet. We know (or should know), empirically, that there always have been those new jobs in the past, and that they weren't predictable either: no-one in 1800 would have predicted that in 1900 a million Americans would work on 'railways' and no-one in 1900 would have predicted 'video post-production' or 'software engineer' as employment categories. But it seems insufficient to take it on faith that this will happen now just because it always has in the past. How do you know it will happen this time? Is this different?

At this point, any first-year economics student will tell us that this is answered by, amongst other things, the 'Lump of Labour' fallacy.

The Lump of Labour fallacy is the misconception that there is a fixed amount of work to be done, and that if some work is taken by a machine then there will be less work for people. But if it becomes cheaper to use a machine to make, say, a pair of shoes, then the shoes are cheaper, more people can buy shoes and they have more money to spend on other things besides, and we discover new things we need or want, and new jobs. The efficient gain isn't confined to the shoe: generally, it ripples outward through the economy and creates new prosperity and new jobs. So, we don't know what the new jobs will be, but we have a model that says, not just that there always have been new jobs, but why that is inherent in the process. Don't worry about AI!

The most fundamental challenge to this model today, I think, is to say that no, what's really been happening for the last 200 years of automation is that

we've been moving up the scale of human capability.

'Barge haulers on the Volga', Ilya Repin, 1870-73. (Note the steam boat on the horizon.)

We began with humans as beasts of burden and moved up: we automated legs, then arms, then fingers, and now brains. We went from farm work to blue collar work to white collar work, and now we'll automate the white-collar work as well and there'll be nothing left. Factories were replaced by call centres, but if we automate the call centres, what else is there?

Here, I think it's useful to look at another piece of economic and tech history: the Jevons Paradox.

In the 19th century the British navy ran on coal. Britain had a lot of coal (it was the Saudi Arabia of the steam age) but people worried what would happen when the coal ran out. Ah, said the engineers: don't worry, because the steam engines keep getting more efficient, so we'll use less coal. No, said Jevons: if we make steam engines more efficient, then they will be cheaper to run, and we will use more of them and use them for new and different things, and so we will use *more* coal.

We've been applying the Jevons Paradox to white collar work for 150 years.

It's hard to imagine jobs of the future that don't exist yet, but it's also hard to imagine some of the jobs of the past that have already been automated away. Gogol's downtrodden clerks in 1830s St Petersburg spent their entire adult lives copying out documents, one at a time, by hand. They were human Xeroxes. By the 1880s, typewriters produced perfectly legible text at twice the words-per-minute, and carbons gave half a dozen free copies as well. Typewriters meant a clerk could produce more than 10 times the output. A few decades later, adding machines from companies like Burroughs did the same for book-keeping and accounting: instead of adding up columns with a

pen, the machine does it for you, in 20% of the time, with no mistakes.

What did that do to clerical employment? People hired far more clerks. Automation plus the Jevons Paradox meant more jobs.

If one clerk with a machine can do the work of 10, then you might have fewer clerks, but you might also do far more with them. If, Jevons tells us, it becomes much cheaper and more efficient to do something, you might do more of it - you might do more analysis or manage more inventory. You might build a different and more efficient business that is only possible because you can automate its administration with typewriters and adding machines.

This process keeps repeating. This is Jack Lemmon as CC Baxter in The Apartment in 1960, using an electro-mechanical adding machine from Friden fifty years after adding machines were new and exciting.

Everyone in that shot is a cell in a spreadsheet, and the whole building is a spreadsheet. Once a week someone on the top floor presses F9 and they recalculate. But they already had computers, and in 1965 or 1970 they bought a mainframe, and scrapped all the adding machines. Did white collar employment collapse? Or, as IBM advertised, did a computer give you 150 extra engineers? 25 years later, what did the PC revolution, and the accounting department in a box, do to accounting?

Dan Bricklin invented the computer spreadsheet in 1979: until then, 'spreadsheets' were paper (you can still buy them on Amazon). He has some entertaining stories about early use: '*People would tell me, "I was doing all this work, and coworkers thought I was amazing. But I was really goofing off*

because it only took an hour and then I took the the rest of the day off. People thought I was a wunderkind but I was using this tool."'

So, what did Excel and the PC do to accounting employment? It went up.

40 years later, do spreadsheets mean you can goof off early? Not really.

New technology generally makes it cheaper and easier to do something, but that might mean you do the same with fewer people, or you might do much more with the same people. It also tends to mean that you change what you do. To begin with, we make the new tool fit the old way of working, but over time, we change how we work to fit the tool. When CC Baxter's company bought a mainframe, they began by automating the way they already did things, but over time, new ways to run the business became possible.

So, all of this is to say that by default, we should expect LLMs to destroy, displace, create, accelerate and multiply jobs just as SAP, Excel, Mainframes or typewriters did. It's just more automation. The machine lets a person do 10x the work, but you need the person.

I think there are two counter-arguments to this.

The first is to say that yes, perhaps this is indeed just more of the same kind of change that we saw with the internet, PCs or computers, and perhaps again it will have no long-term effect on net employment, but this time it will happen much faster, and so that frictional pain will be much greater and it will be much harder to adjust.

LLMs and ChatGPT certainly are happening a lot faster than things like iPhones or the Internet, or indeed PCs. The Apple II shipped in 1977, the IBM PC in 1981 and the Mac in 1984, but it took until the early 1990s before there were 100m PCs in use: there are 100m ChatGPT users today after just six months. You don't need to wait for telcos to build broadband networks, or consumers to buy new devices, and generative AI sits on top of the whole existing stack of cloud, distributed computing and indeed a lot of the machine learning stack itself that was built over the last decade. To a user, it's just a website.

However, your expectations might be different if you think about the implications of these charts, from Productiv again and from Okta (with a different methodology). They report that their typical customer now has hundreds of different software applications, and enterprise customers have almost 500.

And yet, enterprise cloud adoption is still barely a quarter of workflows.

What does that mean for generative AI in the workplace? Whatever you think will happen, it will take years, not weeks.

First, the tools that people use for work, and the tasks that might now get a new layer of automation, are complicated and very specialised, and embody a lot of work and institutional knowledge. A lot of people are *experimenting*

with ChatGPT, and seeing what it will do. If you're reading this, you probably have too. That doesn't mean that ChatGPT has replaced their existing workflows *yet*, and replacing or automating any of those tools and tasks is not trivial.

There's a huge difference between an amazing demo of a transformative technology and something that a big complicated company holding other people's business can use. You can rarely go to a law firm and sell them an API key to GCP's translation or sentiment analysis: you need to wrap it in control, security, versioning, management, client privilege and a whole bunch of other things that only legal software companies know about (there's a graveyard of machine learning companies that learnt this in the last decade). Companies generally can't buy 'technology'. Everlaw doesn't sell translation and People.ai doesn't sell sentiment analysis - they sell tools and products, and often the AI is only one part of that. I don't think a text prompt, a 'go' button and a black-box, general purpose text generation engine make up a product, and product takes time.

Second, buying tools that manage big complicated things takes time even once the tool is built and has product-market fit. One of the most basic challenges in building an enterprise software startup is that startups run on an 18 month funding cycle and a lot of enterprises run on an 18 month decision cycle. SaaS itself accelerated this because you don't need to get into the enterprise datacenter deployment schedule, but you still need purchase, integration and training, and companies with millions of customers and tens or hundreds of thousands of employees have very good reasons not to change things suddenly. The future takes a while, and the world outside Silicon Valley is complicated.

The second objection is that part of the paradigm shift of ChatGPT and LLMs is a shift in the layer of abstraction: this looks like a much more general

purpose technology. Indeed, that's why it's exciting. It can answer *anything,* we are told. So, you could look at that chart of 473 enterprise SaaS apps and say that ChatGPT will disrupt that and collapse many those vertical apps into one prompt box. That would mean it would move faster, and also automate much more.

I think that misunderstands the problem. If a partner at a law firm wants a first draft of a paper, they want to be able to shape the parameters in completely different ways to a salesperson at an insurance company challenging a claim, probably with a different training set and certainly with a bunch of different tooling. Excel is 'general purpose' too, and so is SQL, but how many different kinds of 'database' are there? This is one reason I think the future of LLMs is to move from prompt boxes to GUIs and buttons - I think 'prompt engineering' and 'natural language' are mutually contradictory. But either way, even if you can run everything as a thin wrapper on top of one giant foundational model (and there is very little agreement or clarity about this yet), even those wrappers will take time.

Meanwhile, while one could suggest that LLMs will subsume many apps on one axis, I think it's equally likely that they will enable a whole new wave of unbundling on other axes, as startups peel off dozens more use cases from Word, Salesforce and SAP, and build a whole bunch more big companies by solving problems that no-one had realised were problems until LLMs let you solve them. That's the process that explains why big companies already have 400 SaaS apps today, after all.

More fundamental, of course, there is the error rate. ChatGPT can *try* to answer 'anything' but the answer might be wrong. People call this hallucinations, making things up, lying or bullshitting - it's the 'overconfident undergraduate' problem. I think these are all unhelpful framings: I think the best way to understand this is that when you type something into a prompt,

you're not actually asking it to answer a question at all. Rather, you're asking it "what sort of answers would people be likely to produce to questions that look like this?" You're asking it to match a pattern.

Hence, if I ask ChatGPT4 to write a biography of myself, and then ask it again, it gives different answers. It suggests I went to Cambridge, Oxford or the LSE; my first job was in equity research, consulting or financial journalism. These are always the right pattern: it's the right kind of university and the right kind of job (it never says Anglia Poly and then catering management). It is giving 100% correct answers to the question "what **kinds** of degrees and jobs are people **like** Benedict **likely** to have done?" It's not doing a database lookup: it's making a pattern.

You can see something similar in this image I got from MidJourney. The prompt was "A photograph of advertising people discussing creativity on stage in a panel on a beach at Cannes Lions."

It's matched the pattern almost perfectly - that looks like the beach at Cannes, these people are dressed like advertising people, and they even have the right haircuts. But it doesn't *know* anything, and so it doesn't know that people *never* have three legs, only that it's unlikely. This isn't 'lying' or 'making things up' - it's matching a pattern, imperfectly.

Whatever you call it, if you don't understand this, you can get into trouble, as happened to this unfortunate lawyer, who did not understand that when he asked for precedents, he was actually asking for things that looked like precedents. He duly got things that looked like precedents, but weren't. It's not a database.

If you do understand this, then you have to ask, well, where are LLMs useful? Where is it useful to have automated undergraduates, or automated interns, who can repeat a pattern, that you might have to check? The last wave of machine learning gave you infinite interns who could read anything for you, but you had to check, and now we have infinite interns that can write anything for you, but you have to check. So where is it useful to have infinite interns? Ask Dan Bricklin - we're back to the Jevons Paradox.

This takes me, obviously, to AGI. The really fundamental objection to everything I've just said is to ask what would happen if we had a system that didn't have an error rate, didn't hallucinate, and really could do anything that people can do. If we had that, then you might not have one accountant using Excel to get the output of ten accountants: you might just have the machine. This time, it really would be be different. Where previous waves of automation meant one person could do more, now you don't need the person.

Like a lot of AGI questions, though, this can become circular if you're not careful. 'If we had a machine that could do anything people do, without any of these limitations, could it do anything people can do, without these limitations?'

Well, indeed, and if so we might have bigger problems than middle-class employment, but is that close? You can spend weeks of your life watching three hour YouTube videos of computer scientists arguing about this, and

conclude only that they don't really know either. You might also suggest that the idea this one magic piece of software will change everything, and override all the complexity of real people, real companies and the real economy, and can now be deployed in weeks instead of years, sounds like classic tech solutionism, but turned from utopia to dystopia.

As an analyst, though, I tend to prefer Hume's empiricism over Descartes - I can only analyse what we can know. We don't have AGI, and without that, we have only another wave of automation, and we don't seem to have any *a priori* reason why this must be more or less painful than all the others.