The Short Case for Nvidia Stock

As someone who spent ~10 years working as a generalist investment analyst at various long/short hedge funds (including stints at Millennium and Balyasny), while also being something of a math and computer nerd who has been studying deep learning since 2010 (back when Geoff Hinton was still talking about Restricted Boltzmann Machines and everything was still programmed using MATLAB, and researchers were still trying to show that they could get better results at classifying handwritten digits than by using Support Vector Machines), I'd like to think that I have a fairly unusual perspective on how AI technology is developing and how this relates to equity valuations in the stock market.

For the past few years, I have been working more as a developer, and have several popular open-source projects for working with various forms of AI models/services (e.g., see LLM Aided OCR, Swiss Army Llama, Fast Vector Similarity, Source to Prompt, and Pastel Inference Layer for a few recent examples). Basically, I am using these frontier models all day, every day, in about as intense a way as possible. I have 3 Claude accounts so I don't run out of requests, and signed up for ChatGPT Pro within minutes of it being available.

I also try to keep on top of the latest research advances, and carefully read all the major technical report papers that come out from the major AI labs. So I think I have a pretty good read on the space and how things are developing. At the same time, I've shorted a ton of stocks in my life and have won the best idea prize on the Value Investors Club twice (for TMS long and PDH short if you're keeping track at home).

I say this not to brag, but rather to help establish my bona fides as someone who could opine on the subject without coming across as hopelessly naive to either technologists or professional investors. And while there are surely many people who know the math/science better, and people who are better at long/short investing in the stock market than me, I doubt there are very many who are in the middle of the Venn diagram to the extent I can claim to be.

With all that said, whenever I meet with and chat with my friends and ex colleagues from the hedge fund world, the conversation quickly turns to Nvidia. It's not every day that a company goes from relative obscurity to being worth more than the combined stock markets of England, France, or Germany! And naturally, these friends want to know my thoughts on the subject. Because I am such a dyed-in-the-wool believer in the long term transformative impact of this technology— I truly believe it's going to radically change nearly every aspect of our economy and society in the next 5-10 years, with basically no historical precedent— it has been hard for me to make the argument that Nvidia's momentum is going to slow down or stop anytime soon.

But even though I've thought the valuation was just too rich for my blood for the past year or so, a confluence of recent developments has caused me to flip a bit to my usual instinct, which is to be a bit more contrarian in

outlook and to question the consensus when it seems to be more than priced in. The saying "what the wise man believes in the beginning, the fool believes in the end" became famous for a good reason.

The Bull Case

Before we get into the developments that give me pause, let's pause to briefly review the bull case for NVDA shares, which is basically now known by everyone and his brother. Deep learning and AI are the most transformative technologies since the internet, and poised to change basically everything in our society. Nvidia has somehow ended up with something close to a monopoly in terms of the share of aggregate industry capex that is spent on training and inference infrastructure.

Some of the largest and most profitable companies in the world, like Microsoft, Apple, Amazon, Meta, Google, Oracle, etc., have all decided that they must do and spend whatever it takes to stay competitive in this space because they simply cannot afford to be left behind. The amount of capex dollars, gigawatts of electricity used, square footage of new-build data centers, and, of course, the number of GPUs, has absolutely exploded and seems to show no sign of slowing down. And Nvidia is able to earn insanely high 90%+ gross margins on the most high-end, datacenter oriented products.

We've just scratched the surface here of the bull case. There are many additional aspects to it now, which have made even people who were already very bullish to become incrementally more bullish. Besides things like the rise of humanoid robots, which I suspect is going to take most people by surprise when they are rapidly able to perform a huge number of tasks that currently require an unskilled (or even skilled) human worker (e.g., doing laundry, cleaning, organizing, and cooking; doing construction work like renovating a bathroom or building a house in a team of workers; running a warehouse and driving forklifts, etc.), there are other factors which most people haven't even considered.

One major thing that you hear the smart crowd talking about is the rise of "a new scaling law," which has created a new paradigm thinking about how compute needs will increase over time. The original scaling law, which is what has been driving progress in AI since AlexNet appeared in 2012 and the Transformer architecture was invented in 2017, is the pre-training scaling law: that the more billions (and now trillions) worth of tokens we can use as training data, and the larger the parameter count of the models we are training, and the more FLOPS of compute that we expend on training those models on those tokens, the better the performance of the resulting models on a large variety of highly useful downstream tasks.

Not only that, but this improvement is somewhat knowable, to the point where the leading AI labs like OpenAI and Anthropic have a pretty good idea of just how good their latest models would be even before they started the actual training runs— in some cases, predicting the benchmarks of the final models to within a couple percentage points. This "original scaling law" has been vitally important, but always caused some doubts in the minds of people projecting the future with it.

For one thing, we seem to have already exhausted the world's accumulated set of high quality training data. Of course, that's not literally true— there are still so many old books and periodicals that haven't yet been properly digitized, and even if they have, are not properly licensed for use as training data. The problem is that, even if you give credit for all that stuff— say the sum total of "professionally" produced English language written content from the year 1500 to, say, the year 2000, it's not such a tremendous amount in percentage terms when you're talking about a training corpus of nearly 15 trillion tokens, which is the scale of current frontier models.

For a quick reality check of those numbers: Google Books has digitized around 40mm books so far; if a typical book has 50k to 100k words, or 65k to 130k tokens, then that's between 2.6T and 5.2T tokens just from books, though surely a large chunk of that is already included in the training corpora used by the big labs, whether it's strictly legal or not. And there are lots of academic papers, with the arXiv website alone having over 2mm papers. And the Library of Congress has over 3 billion digitized newspaper pages. Taken together, that could be as much as 7T tokens in total, but since much of this is in fact included in training corpora, the remaining "incremental" training data probably isn't all that significant in the grand scheme of things.

Of course, there are other ways to gather more training data. You could automatically transcribe every single YouTube video for example, and use that text. And while that might be helpful on the margin, it's certainly of much lower quality than, say, a highly respected textbook on Organic Chemistry as a source of useful knowledge about the world. So we've always had a looming "data wall" when it comes to the original scaling law; although we know we can keep shoveling more and more capex into GPUs and building more and more data centers, it's a lot harder to mass produce useful new human knowledge which is correct and incremental to what is already out there. Now, one intriguing response to this has been the rise of "synthetic data," which is text that is itself the output of an LLM. And while this seems almost nonsensical that it would work to "get high on your own supply" as a way of improving model quality, it actually seems to work very well in practice, at least in the domain of math, logic, and computer programming.

The reason, of course, is that these are areas where we can mechanically check and prove the correctness of things. So we can sample from the vast universe of possible math theorems or possible Python scripts, and then actually check if they are correct, and only include them in our corpus if they are. And in this way, we can very dramatically expand our collection of high quality training data, at least in these kinds of areas.

And then there are all the other kinds of data we could be training AI on besides text. For example, what if we take the entire whole genome sequencing (around 200 GB to 300 GB uncompressed for a single human being) for 100 million people? That's a *lot* of data obviously, although the vast majority of it would be nearly identical between any two people. Of course, this could be misleading to compare to textual data from books and the internet for various reasons:

- Raw genome size isn't directly comparable to token counts
- The information content of genomic data is very different from text

- The training value of highly redundant data isn't clear
- The computational requirements for processing genomic data are different

But it's still another large source of diverse information that we could train huge models on in the future, which is why I included it.

So while there is some hope in terms of being able to capture more and more additional training data, if you look at the rate at which training corpora have grown in recent years, it quickly becomes obvious that we are close to hitting a wall in terms of data availability for "generally useful" knowledge that can get us closer to the ultimate goal of getting artificial super-intelligence which is 10x smarter than John von Neumann and is an absolute world-class expert on every specialty known to man.

Besides the limited amount of available data, there have always been a couple other things that have lurked in the back of the mind of proponents of the pre-training scaling law. A big one of these is, after you've finished training the model, what are you supposed to do with all that compute infrastructure? Train the next model? Sure, you can do that, but given the rapid improvement in GPU speed and capacity, and the importance of electricity and other opex in the economic calculations, does it even really make sense to use your 2 year old cluster to train your new model? Surely you'd rather use the brand new data center you just built that costs 10x the old data center and is 20x more powerful because of better technology. The problem is, at some point you do need to amortize the up-front cost of these investments and recoup it with a stream of (hopefully positive) operating profit, right?

The market is so excited about AI that it has thankfully ignored this, allowing companies like OpenAI to post breathtaking from-inception, cumulative operating losses while garnering increasingly eye-popping valuations in follow-up investment rounds (although, to their credit, they have also been able to demonstrate very fast growing revenues). But eventually, for this situation to be sustainable over a full market cycle, these data center costs do need to eventually be recouped, hopefully with a profit, which over time is competitive with other investment opportunities on a risk-adjusted basis.

The New Paradigm

OK, so that was the pre-training scaling law. What's this "new" scaling law? Well, that's something that people really just started focusing on in the past year: inference time compute scaling. Before, the vast majority of all the compute you'd expend in the process was the up-front training compute to create the model in the first place. Once you had the trained model, performing inference on that model— i.e., asking a question or having the LLM perform some kind of task for you— used a certain, limited amount of compute.

Critically, the total amount of inference compute (measured in various ways, such as FLOPS, in GPU memory footprint, etc.) was much, much less than what was required for the pre-training phase. Of course, the amount

of inference compute does flex up when you increase the context window size of the models and the amount of output that you generate from them in one go (although researchers have made breathtaking algorithmic improvements on this front relative to the initial quadratic scaling people originally expected in scaling this up). But essentially, until recently, inference compute was generally a lot less intensive than training compute, and scaled basically linearly with the number of requests you are handling— the more demand for text completions from ChatGPT, for instance, the more inference compute you used up.

With the advent of the revolutionary Chain-of-Thought ("COT") models introduced in the past year, most noticeably in OpenAI's flagship O1 model (but very recently in DeepSeek's new R1 model, which we will talk about later in much more detail), all that changed. Instead of the amount of inference compute being directly proportional to the length of the output text generated by the model (scaling up for larger context windows, model size, etc.), these new COT models also generate intermediate "logic tokens"; think of this as a sort of scratchpad or "internal monologue" of the model while it's trying to solve your problem or complete its assigned task.

This represents a true sea change in how inference compute works: now, the more tokens you use for this internal chain of thought process, the better the quality of the final output you can provide the user. In effect, it's like giving a human worker more time and resources to accomplish a task, so they can double and triple check their work, do the same basic task in multiple different ways and verify that they come out the same way; take the result they came up with and "plug it in" to the formula to check that it actually does solve the equation, etc.

It turns out that this approach works almost amazingly well; it is essentially leveraging the long anticipated power of what is called "reinforcement learning" with the power of the Transformer architecture. It directly addresses the single biggest weakness of the otherwise phenomenally successful Transformer model, which is its propensity to "hallucinate".

Basically, the way Transformers work in terms of predicting the next token at each step is that, if they start out on a bad "path" in their initial response, they become almost like a prevaricating child who tries to spin a yarn about why they are actually correct, even if they should have realized mid-stream using common sense that what they are saying couldn't possibly be correct.

Because the models are always seeking to be internally consistent and to have each successive generated token flow naturally from the preceding tokens and context, it's very hard for them to course-correct and backtrack. By breaking the inference process into what is effectively many intermediate stages, they can try lots of different things and see what's working and keep trying to course-correct and try other approaches until they can reach a fairly high threshold of confidence that they aren't talking nonsense.

Perhaps the most extraordinary thing about this approach, beyond the fact that it works at all, is that the more logic/COT tokens you use, the better it works. Suddenly, you now have an additional dial you can turn so that, as you increase the amount of COT reasoning tokens (which uses a lot more inference compute, both in terms

of FLOPS and memory), the higher the probability is that you will give a correct response— code that runs the first time without errors, or a solution to a logic problem without an obviously wrong deductive step.

I can tell you from a lot of firsthand experience that, as good as Anthropic's Claude3.5 Sonnet model is at Python programming— and it is indeed VERY good— whenever you need to generate anything long and complicated, it invariably ends up making one or more stupid mistakes. Now, these mistakes are usually pretty easy to fix, and in fact you can normally fix them by simply feeding the errors generated by the Python interpreter, without any further explanation, as a follow-up inference prompt (or, more usefully, paste in the complete set of detected "problems" found in the code by your code editor, using what something called a Linter), it was still an annoying additional step. And when the code becomes very long or very complicated, it can sometimes take a lot longer to fix, and might even require some manual debugging by hand.

The first time I tried the O1 model from OpenAI was like a revelation: I was amazed how often the code would be perfect the very first time. And that's because the COT process automatically finds and fixes problems before they ever make it to a final response token in the answer the model gives you.

In fact, the O1 model used in OpenAI's ChatGPT Plus subscription for $20/month is basically the same model as the one used in the O1-Pro model featured in their new ChatGPT Pro subscription for 10x the price ($200/month, which raised plenty of eyebrows in the developer community); the main difference is that O1-Pro thinks for a lot longer before responding, generating vastly more COT logic tokens, and consuming a far larger amount of inference compute for every response.

This is quite striking in that, even a very long and complex prompt for Claude3.5 Sonnet or GPT4o, with ~400kb+ of context given, generally takes less than 10 seconds to begin responding, and often less than 5 seconds. Whereas that same prompt to O1-Pro could easily take 5+ MINUTES before you get a response (although OpenAI does show you some of the "reasoning steps" that are generated during the process while you wait; critically, OpenAI has decided, presumably for trade secret related reasons,to hide from you the exact reasoning tokens it generates, showing you instead a highly abbreviated summary of these).

As you can probably imagine, there are tons of contexts where accuracy is paramount— where you'd rather give up and tell the user you can't do it at all rather than give an answer that could be trivially proven wrong or which involves hallucinated facts or otherwise specious reasoning. Anything involving money/transactions, medical stuff, legal stuff, just to name a few.

Basically, wherever the cost of inference is trivial relative to the hourly all-in compensation of the human knowledge worker who is interacting with the AI system, that's a case where it become a complete no-brainer to dial up the COT compute (the major drawback is that it increases the latency of responses by a lot, so there are still some contexts where you might prefer to iterate faster by getting lower latency responses that are less accurate or correct).

Some of the most exciting news in the AI world came out just a few weeks ago and concerned OpenAI's new unreleased O3 model, which was able to solve a large variety of tasks that were previously deemed to be out of reach of current AI approaches in the near term. And the way it was able to do these hardest problems (which include exceptionally tough "foundational" math problems that would be very hard for even highly skilled professional mathematicians to solve), is that OpenAI threw insane amount of compute resources at the problems— in some cases, spending $3k+ worth of compute power to solve a single task (compare this to traditional inference costs for a single task, which would be unlikely to exceed a couple dollars using regular Transformer models without chain-of-thought).

It doesn't take an AI genius to realize that this development creates a new scaling law that is totally independent of the original pre-training scaling law. Now, you still want to train the best model you can by cleverly leveraging as much compute as you can and as many trillion tokens of high quality training data as possible, but that's just the beginning of the story in this new world; now, you could easily use incredibly huge amounts of compute just to do inference from these models at a very high level of confidence or when trying to solve extremely tough problems that require "genius level" reasoning to avoid all the potential pitfalls that would lead a regular LLM astray.

But Why Should Nvidia Get to Capture All The Upside?

Even if you believe, as I do, that the future prospects for AI are almost unimaginably bright, the question still remains, "Why should one company extract the majority of the profit pool from this technology?" There are certainly many historical cases where a very important new technology changed the world, but the main winners were not the companies that seemed the most promising during the initial stages of the process. The Wright Brothers' airplane company in all its current incarnations across many different firms today isn't worth more than $10b despite them inventing and perfecting the technology well ahead of everyone else. And while Ford has a respectable market cap of $40b today, it's just 1.1% of Nvidia's current market cap.

To understand this, it's important to really understand why Nvidia is currently capturing so much of the pie today. After all, they aren't the only company that even makes GPUs. AMD makes respectable GPUs that, on paper, have comparable numbers of transistors, which are made using similar process nodes, etc. Sure, they aren't as fast or as advanced as Nvidia's GPUs, but it's not like the Nvidia GPUs are 10x faster or anything like that. In fact, in terms of naive/raw dollars per FLOP, AMD GPUs are something like half the price of Nvidia GPUs.

Looking at other semiconductor markets such as the DRAM market, despite the fact that it is also very highly consolidated with only 3 meaningful global players (Samsung, Micron, SK-Hynix), gross margins in the DRAM market range from negative at the bottom of the cycle to ~60% at the very top of the cycle, with an average in the 20% range. Compare that to Nvidia's overall gross margin in recent quarters of ~75%, which is dragged down by the lower-margin and more commoditized consumer 3D graphics category.

So how is this possible? Well, the main reasons have to do with software— better drivers that "just work" on Linux and which are highly battle-tested and reliable (unlike AMD, which is notorious for the low quality and instability of their Linux drivers), and highly optimized open-source code in popular libraries such as PyTorch that has been tuned to work really well on Nvidia GPUs.

It goes beyond that though— the very programming framework that coders use to write low-level code that is optimized for GPUs, CUDA, is totally proprietary to Nvidia, and it has become a de facto standard. If you want to hire a bunch of extremely talented programmers who know how to make things go really fast on GPUs, and pay them $650k/year or whatever the going rate is for people with that particular expertise, chances are that they are going to "think" and work in CUDA.

Besides software superiority, the other major thing that Nvidia has going for it is what is known as interconnect — essentially, the bandwidth that connects together thousands of GPUs together efficiently so they can be jointly harnessed to train today's leading-edge foundational models. In short, the key to efficient training is to keep all the GPUs as fully utilized as possible all the time— not waiting around idling until they receive the next chunk of data they need to compute the next step of the training process.

The bandwidth requirements are extremely high— much, much higher than the typical bandwidth that is needed in traditional data center use cases. You can't really use traditional networking gear or fiber optics for this kind of interconnect, since it would introduce too much latency and wouldn't give you the pure terabytes per second of bandwidth that is needed to keep all the GPUs constantly busy.

Nvidia made an incredibly smart decision to purchase the Israeli company Mellanox back in 2019 for a mere $6.9b, and this acquisition is what provided them with their industry leading interconnect technology. Note that interconnect speed is a lot more relevant to the training process, where you have to harness together the output of thousands of GPUs at the same time, than the inference process (including COT inference), which can use just a handful of GPUs— all you need is enough VRAM to store the quantized (compressed) model weights of the already-trained model.

So those are arguably the major components of Nvidia's "moat" and how it has been able to maintain such high margins for so long (there is also a "flywheel" aspect to things, where they aggressively invest their super-normal profits into tons of R&D, which in turn helps them improve their tech at a faster rate than the competition, so they are always in the lead in terms of raw performance).

But as was pointed out earlier, what customers really tend to care about, all other things being equal, is performance per dollar (both in up-front capex cost of equipment and in energy usage, so performance per watt), and even though Nvidia's GPUs are certainly the fastest, they are not the best price/performance when measured naively in terms of FLOPS.

But the thing is, all other things are NOT equal, and the fact that AMD's drivers suck, that popular AI software libraries don't run as well on AMD GPUs, that you can't find really good GPU experts who specialize in AMD GPUs outside of the gaming world (why would they bother when there is more demand in the market for CUDA experts?), that you can't wire thousands of them together as effectively because of lousy interconnect technology for AMD— all this means that AMD is basically not competitive in the high-end data center world, and doesn't seem to have very good prospects for getting there in the near term.

Well, that all sounds very bullish for Nvidia, right? Now you can see why the stock is trading at such a huge valuation! But what are the other clouds on the horizon? Well, there are few that I think merit significant attention. Some have been lurking in the background for the last few years, but too small to make a dent considering how quickly the pie has been growing, but where they are getting ready to potentially inflect upwards. Others are very recent developments (as in, the last 2 weeks) that might dramatically change the near-term trajectory of incremental GPU demand.

The Major Threats

At a very high level, you can think of things like this: Nvidia operated in a pretty niche area for a very long time; they had very limited competition, and the competition wasn't particular profitable or growing fast enough to ever pose a real threat, since they didn't have the capital needed to really apply pressure to a market leader like Nvidia. The gaming market was large and growing, but didn't feature earth shattering margins or particularly fabulous year over year growth rates.

A few big tech companies started ramping up hiring and spending on machine learning and AI efforts around 2016-2017, but it was never a truly significant line item for any of them on an aggregate basis— more of a "moonshot" R&D expenditure. But once the big AI race started in earnest with the release of ChatGPT in 2022 — only a bit over 2 years ago, although it seems like a lifetime ago in terms of developments— that situation changed very dramatically.

Suddenly, big companies were ready to spend many, many billions of dollars incredibly quickly. The number of researchers showing up at the big research conferences like Neurips and ICML went up very, very dramatically. All the smart students who might have previously studied financial derivatives were instead studying Transformers, and $1mm+ compensation packages for non-executive engineering roles (i.e., for independent contributors not managing a team) became the norm at the leading AI labs.

It takes a while to change the direction of a massive cruise ship; and even if you move really quickly and spend billions, it takes a year or more to build greenfield data centers and order all the equipment (with ballooning lead times) and get it all set up and working. It takes a long time to hire and onboard even smart coders before they can really hit their stride and familiarize themselves with the existing codebases and infrastructure.

But now, you can imagine that absolutely biblical amounts of capital, brainpower, and effort are being expended in this area. And Nvidia has the biggest target of any player on their back, because they are the ones who are making the lion's share of the profits TODAY, not in some hypothetical future where the AI runs our whole lives.

So the very high level takeaway is basically that "markets find a way"; they find alternative, radically innovative new approaches to building hardware that leverage completely new ideas to sidestep barriers that help prop up Nvidia's moat.

The Hardware Level Threat

For example, so-called "wafer scale" AI training chips from Cerebras, which dedicate an entire 300mm silicon wafer to an absolutely gargantuan chip that contains orders of magnitude more transistors and cores on a single chip (see this recent blog post from them explaining how they were able to solve the "yield problem" that had been preventing this approach from being economically practical in the past).

To put this into perspective, if you compare Cerebras' newest WSE-3 chip to Nvidia's flagship data-center GPU, the H100, the Cerebras chip has a total die area of 46,225 square millimeters compared to just 814 for the H100 (and the H100 is itself considered an enormous chip by industry standards); that's a multiple of ~57x! And instead of having 132 "streaming multiprocessor" cores enabled on the chip like the H100 has, the Cerebras chip has ~900,000 cores (granted, each of these cores is smaller and does a lot less, but it's still an almost unfathomably large number in comparison). In more concrete apples-to-apples terms, the Cerebras chip can do around ~32x the FLOPS in AI contexts as a single H100 chip. Since an H100 sells for close to $40k a pop, you can imagine that the WSE-3 chip isn't cheap.

So why does this all matter? Well, instead of trying to battle Nvidia head-on by using a similar approach and trying to match the Mellanox interconnect technology, Cerebras has used a radically innovative approach to do an end-run around the interconnect problem: inter-processor bandwidth becomes much less of an issue when everything is running on the same super-sized chip. You don't even need to have the same level of interconnect because one mega chip replaces tons of H100s.

And the Cerebras chips also work extremely well for AI inference tasks. In fact, you can try it today for free here and use Meta's very respectable Llama-3.3-70B model. It responds basically instantaneously, at ~1,500 tokens per second. To put that into perspective, anything above 30 tokens per second feels relatively snappy to users based on comparisons to ChatGPT and Claude, and even 10 tokens per second is fast enough that you can basically read the response while it's being generated.

Cerebras is also not alone; there are other companies, like Groq (not to be confused with the Grok model family trained by Elon Musk's X AI). Groq has taken yet another innovative approach to solving the same fundamental problem. Instead of trying to compete with Nvidia's CUDA software stack directly, they've developed what they

call a "tensor processing unit" (TPU) that is specifically designed for the exact mathematical operations that deep learning models need to perform. Their chips are designed around a concept called "deterministic compute," which means that, unlike traditional GPUs where the exact timing of operations can vary, their chips execute operations in a completely predictable way every single time.

This might sound like a minor technical detail, but it actually makes a massive difference for both chip design and software development. Because the timing is completely deterministic, Groq can optimize their chips in ways that would be impossible with traditional GPU architectures. As a result, they've been demonstrating for the past 6+ months inference speeds of over 500 tokens per second with the Llama series of models and other open source models, far exceeding what's possible with traditional GPU setups. Like Cerebras, this is available today and you can try it for free here.

Using a comparable Llama3 model with "speculative decoding," Groq is able to generate 1,320 tokens per second, on par with Cerebras and far in excess of what is possible using regular GPUs. Now, you might ask what the point is of achieving 1,000+ tokens per second when users seem pretty satisfied with ChatGPT, which is operating at less than 10% of that speed. And the thing is, it does matter. It makes it a lot faster to iterate and not lose focus as a human knowledge worker when you get instant feedback. And if you're using the model programmatically via the API, which is increasingly where much of the demand is coming from, then it can enable whole new classes of applications that require multi-stage inference (where the output of previous stages is used as input in successive stages of prompting/inference) or which require low-latency responses, such as content moderation, fraud detection, dynamic pricing, etc.

But even more fundamentally, the faster you can serve requests, the faster you can cycle things, and the busier you can keep the hardware. Although Groq's hardware is extremely expensive, clocking in at $2mm to $3mm for a single server, it ends up costing far less per request fulfilled if you have enough demand to keep the hardware busy all the time.

And like Nvidia with CUDA, a huge part of Groq's advantage comes from their own proprietary software stack. They are able to take the same open source models that other companies like Meta, DeepSeek, and Mistral develop and release for free, and decompose them in special ways that allow them to run dramatically faster on their specific hardware.

Like Cerebras, they have taken different technical decisions to optimize certain particular aspects of the process, which allows them to do things in a fundamentally different way. In Groq's case, it's because they are entirely focused on inference level compute, not on training: all their special sauce hardware and software only give these huge speed and efficiency advantages when doing inference on an already trained model.

But if the next big scaling law that people are excited about is for inference level compute— and if the biggest drawback of COT models is the high latency introduced by having to generate all those intermediate logic tokens before they can respond— then even a company that only does inference compute, but which does it

dramatically faster and more efficiently than Nvidia can— can introduce a serious competitive threat in the coming years. At the very least, Cerebras and Groq can chip away at the lofty expectations for Nvidia's revenue growth over the next 2-3 years that are embedded in the current equity valuation.

Besides these particularly innovative, if relatively unknown, startup competitors, there is some serious competition coming from some of Nvidia's biggest customers themselves who have been making custom silicon that specifically targets AI training and inference workloads. Perhaps the best known of these is Google, which has been developing its own proprietary TPUs since 2016. Interestingly, although it briefly sold TPUs to external customers, Google has been using all its TPUs internally for the past several years, and it is already on its 6th generation of TPU hardware.

Amazon has also been developing its own custom chips called Trainium2 and Inferentia2. And while Amazon is building out data centers featuring billions of dollars of Nvidia GPUs, they are also at the same time investing many billions in other data centers that use these internal chips. They have one cluster that they are bringing online for Anthropic that features over 400k chips.

Amazon gets a lot of flak for totally bungling their internal AI model development, squandering massive amounts of internal compute resources on models that ultimately are not competitive, but the custom silicon is another matter. Again, they don't necessarily need their chips to be better and faster than Nvidia's. What they need is for their chips to be good enough, but build them at a breakeven gross margin instead of the ~90%+ gross margin that Nvidia earns on its H100 business.

OpenAI has also announced their plans to build custom chips, and they (together with Microsoft) are obviously the single largest user of Nvidia's data center hardware. As if that weren't enough, Microsoft have themselves announced their own custom chips!

And Apple, the most valuable technology company in the world, has been blowing away expectations for years now with their highly innovative and disruptive custom silicon operation, which now completely trounces the CPUs from both Intel and AMD in terms of performance per watt, which is the most important factor in mobile (phone/tablet/laptop) applications. And they have been making their own internally designed GPUs and "Neural Processors" for years, even though they have yet to really demonstrate the utility of such chips outside of their own custom applications, like the advanced software based image processing used in the iPhone's camera.

While Apple's focus seems somewhat orthogonal to these other players in terms of its mobile-first, consumer oriented, "edge compute" focus, if it ends up spending enough money on its new contract with OpenAI to provide AI services to iPhone users, you have to imagine that they have teams looking into making their own custom silicon for inference/training (although given their secrecy, you might never even know about it directly!).

Now, it's no secret that there is a strong power law distribution of Nvidia's hyper-scaler customer base, with the top handful of customers representing the lion's share of high-margin revenue. How should one think about the

future of this business when literally every single one of these VIP customers is building their own custom chips specifically for AI training and inference?

When thinking about all this, you should keep one incredibly important thing in mind: Nvidia is largely an IP based company. They don't make their own chips. The true special sauce for making these incredible devices arguably comes more from TSMC, the actual fab, and ASML, which makes the special EUV lithography machines used by TSMC to make these leading-edge process node chips. And that's critically important, because TSMC will sell their most advanced chips to anyone who comes to them with enough up-front investment and is willing to guarantee a certain amount of volume. They don't care if it's for Bitcoin mining ASICs, GPUs, TPUs, mobile phone SoCs, etc.

As much as senior chip designers at Nvidia earn per year, surely some of the best of them could be lured away by these other tech behemoths for enough cash and stock. And once they have a team and resources, they can design innovative chips (again, perhaps not even 50% as advanced as an H100, but with that Nvidia gross margin, there is plenty of room to work with) in 2 to 3 years, and thanks for TSMC, they can turn those into actual silicon using the exact same process node technology as Nvidia.

The Software Threat(s)

As if these looming hardware threats weren't bad enough, there are a few developments in the software world in the last couple years that, while they started out slowly, are now picking up real steam and could pose a serious threat to the software dominance of Nvidia's CUDA. The first of these is the horrible Linux drivers for AMD GPUs. Remember we talked about how AMD has inexplicably allowed these drivers to suck for years despite leaving massive amounts of money on the table?

Well, amusingly enough, the infamous hacker George Hotz (famous for jailbreaking the original iphone as a teenager, and currently the CEO of self-driving startup Comma.ai and AI computer company Tiny Corp, which also makes the open-source tinygrad AI software framework), recently announced that he was sick and tired of dealing with AMD's bad drivers, and desperately wanted to be able to to leverage the lower cost AMD GPUs in their TinyBox AI computers (which come in multiple flavors, some of which use Nvidia GPUs, and some of which use AMD GPUS).

Well, he is making his own custom drivers and software stack for AMD GPUs without any help from AMD themselves; on Jan. 15th of 2025, he tweeted via his company's X account that *"We are one piece away from a completely sovereign stack on AMD, the RDNA3 assembler. We have our own driver, runtime, libraries, and emulator. (all in ~12,000 lines!)"* Given his track record and skills, it is likely that they will have this all working in the next couple months, and this would allow for a lot of exciting possibilities of using AMD GPUs for all sorts of applications where companies currently feel compelled to pay up for Nvidia GPUs.

OK, well that's just a driver for AMD, and it's not even done yet. What else is there? Well, there are a few other areas on the software side that are a lot more impactful. For one, there is now a massive concerted effort across many large tech companies and the open source software community at large to make more generic AI software frameworks that have CUDA as just one of many "compilation targets".

That is, you write your software using higher-level abstractions, and the system itself can automatically turn those high-level constructs into super well-tuned low-level code that works extremely well on CUDA. But because it's done at this higher level of abstraction, it can just as easily get compiled into low-level code that works extremely well on lots of other GPUs and TPUs from a variety of providers, such as the massive number of custom chips in the pipeline from every big tech company.

The most famous examples of these frameworks are MLX (sponsored primarily by Apple), Triton (sponsored primarily by OpenAI), and JAX (developed by Google). MLX is particularly interesting because it provides a PyTorch-like API that can run efficiently on Apple Silicon, showing how these abstraction layers can enable AI workloads to run on completely different architectures. Triton, meanwhile, has become increasingly popular as it allows developers to write high-performance code that can be compiled to run on various hardware targets without having to understand the low-level details of each platform.

These frameworks allow developers to write their code once using high powered abstractions and then target tons of platforms automatically— doesn't that sound like a better way to do things, which would give you a lot more flexibility in terms of how you actually run the code?

In the 1980s, all the most popular, best selling software was written in hand-tuned assembly language. The PKZIP compression utility for example was hand crafted to maximize speed, to the point where a competently coded version written in the standard C programming language and compiled using the best available optimizing compilers at the time, would run at probably half the speed of the hand-tuned assembly code. The same is true for other popular software packages like WordStar, VisiCalc, and so on.

Over time, compilers kept getting better and better, and every time the CPU architectures changed (say, from Intel releasing the 486, then the Pentium, and so on), that hand-rolled assembler would often have to be thrown out and rewritten, something that only the smartest coders were capable of (sort of like how CUDA experts are on a different level in the job market versus a "regular" software developer). Eventually, things converged so that the speed benefits of hand-rolled assembly were outweighed dramatically by the flexibility of being able to write code in a high-level language like C or C++, where you rely on the compiler to make things run really optimally on the given CPU.

Nowadays, very little new code is written in assembly. I believe a similar transformation will end up happening for AI training and inference code, for similar reasons: computers are good at optimization, and flexibility and speed of development is increasingly the more important factor— especially if it also allows you to save

dramatically on your hardware bill because you don't need to keep paying the "CUDA tax" that gives Nvidia 90%+ margins.

Yet another area where you might see things change dramatically is that CUDA might very well end up being more of a high level abstraction itself— a "specification language" similar to Verilog (used as the industry standard to describe chip layouts) that skilled developers can use to describe high-level algorithms that involve massive parallelism (since they are already familiar with it, it's very well constructed, it's the lingua franca, etc.), but then instead of having that code compiled for use on Nvidia GPUs like you would normally do, it can instead be fed as source code into an LLM which can port it into whatever low-level code is understood by the new Cerebras chip, or the new Amazon Trainium2, or the new Google TPUv6, etc. This isn't as far off as you might think; it's probably already well within reach using OpenAI's latest O3 model, and surely will be possible generally within a year or two.

The Theoretical Threat

Perhaps the most shocking development which was alluded to earlier happened in the last couple of weeks. And that is the news that has totally rocked the AI world, and which has been dominating the discourse among knowledgeable people on Twitter despite its complete absence from any of the mainstream media outlets: that a small Chinese startup called DeepSeek released two new models that have basically world-competitive performance levels on par with the best models from OpenAI and Anthropic (blowing past the Meta Llama3 models and other smaller open source model players such as Mistral). These models are called DeepSeek-V3 (basically their answer to GPT-4o and Claude3.5 Sonnet) and DeepSeek-R1 (basically their answer to OpenAI's O1 model).

Why is this all so shocking? Well, first of all, DeepSeek is a tiny Chinese company that reportedly has under 200 employees. The story goes that they started out as a quant trading hedge fund similar to TwoSigma or RenTec, but after Xi Jinping cracked down on that space, they used their math and engineering chops to pivot into AI research. Who knows if any of that is really true or if they are merely some kind of front for the CCP or the Chinese military. But the fact remains that they have released two incredibly detailed technical reports, for DeepSeek-V3 and DeepSeekR1.

These are heavy technical reports, and if you don't know a lot of linear algebra, you probably won't understand much. But what you should really try is to download the free DeepSeek app on the AppStore here and install it using a Google account to log in and give it a try (you can also install it on Android here), or simply try it out on your desktop computer in the browser here. Make sure to select the "DeepThink" option to enable chain-of-thought (the R1 model) and ask it to explain parts of the technical reports in simple terms.

This will simultaneously show you a few important things:

- One, this model is absolutely legit. There is a lot of BS that goes on with AI benchmarks, which are routinely gamed so that models appear to perform great on the benchmarks but then suck in real world tests. Google is certainly the worst offender in this regard, constantly crowing about how amazing their LLMs are, when they are so awful in any real world test that they can't even reliably accomplish the simplest possible tasks, let alone challenging coding tasks. These DeepSeek models are not like that— the responses are coherent, compelling, and absolutely on the same level as those from OpenAI and Anthropic.

- Two, that DeepSeek has made profound advancements not just in model quality, but more importantly in model training and inference efficiency. By being extremely close to the hardware and by layering together a handful of distinct, very clever optimizations, DeepSeek was able to train these incredible models using GPUs in a dramatically more efficient way. By some measurements, over ~45x more efficiently than other leading-edge models. DeepSeek claims that the complete cost to train DeepSeek-V3 was just over $5mm. That is absolutely nothing by the standards of OpenAI, Anthropic, etc., which were well into the $100mm+ level for training costs for a single model as early as 2024.

How in the world could this be possible? How could this little Chinese company completely upstage all the smartest minds at our leading AI labs, which have 100 times more resources, headcount, payroll, capital, GPUs, etc? Wasn't China supposed to be crippled by Biden's restriction on GPU exports? Well, the details are fairly technical, but we can at least describe them at a high level. It might have just turned out that the relative GPU processing poverty of DeepSeek was the critical ingredient to make them more creative and clever, necessity being the mother of invention and all.

A major innovation is their sophisticated mixed-precision training framework that lets them use 8-bit floating point numbers (FP8) throughout the entire training process. Most Western AI labs train using "full precision" 32-bit numbers (this basically specifies the number of gradations possible in describing the output of an artificial neuron; 8 bits in FP8 lets you store a much wider range of numbers than you might expect— it's not just limited to 256 different equal-sized magnitudes like you'd get with regular integers, but instead uses clever math tricks to store both very small and very large numbers— though naturally with less precision than you'd get with 32 bits.) The main tradeoff is that while FP32 can store numbers with incredible precision across an enormous range, FP8 sacrifices some of that precision to save memory and boost performance, while still maintaining enough accuracy for many AI workloads.

DeepSeek cracked this problem by developing a clever system that breaks numbers into small tiles for activations and blocks for weights, and strategically uses high-precision calculations at key points in the network. Unlike other labs that train in high precision and then compress later (losing some quality in the process), DeepSeek's native FP8 approach means they get the massive memory savings without compromising performance. When you're training across thousands of GPUs, this dramatic reduction in memory requirements per GPU translates into needing far fewer GPUs overall.

Another major breakthrough is their multi-token prediction system. Most Transformer based LLM models do inference by predicting the next token— one token at a time. DeepSeek figured out how to predict multiple tokens while maintaining the quality you'd get from single-token prediction. Their approach achieves about 85-90% accuracy on these additional token predictions, which effectively doubles inference speed without sacrificing much quality. The clever part is they maintain the complete causal chain of predictions, so the model isn't just guessing— it's making structured, contextual predictions.

One of their most innovative developments is what they call Multi-head Latent Attention (MLA). This is a breakthrough in how they handle what are called the Key-Value indices, which are basically how individual tokens are represented in the attention mechanism within the Transformer architecture. Although this is getting a bit too advanced in technical terms, suffice it to say that these KV indices are some of the major uses of VRAM during the training and inference process, and part of the reason why you need to use thousands of GPUs at the same time to train these models— each GPU has a maximum of 96 gb of VRAM, and these indices eat that memory up for breakfast.

Their MLA system finds a way to store a compressed version of these indices that captures the essential information while using far less memory. The brilliant part is this compression is built directly into how the model learns— it's not some separate step they need to do, it's built directly into the end-to-end training pipeline. This means that the entire mechanism is "differentiable" and able to be trained directly using the standard optimizers. All this stuff works because these models are ultimately finding much lower-dimensional representations of the underlying data than the so-called "ambient dimensions". So it's wasteful to store the full KV indices, even though that is basically what everyone else does.

Not only do you end up wasting tons of space by storing way more numbers than you need, which gives a massive boost to the training memory footprint and efficiency (again, slashing the number of GPUs you need to train a world class model), but it can actually end up improving model quality because it can act like a "regularizer," forcing the model to pay attention to the truly important stuff instead of using the wasted capacity to fit to noise in the training data. So not only do you save a ton of memory, but the model might even perform better. At the very least, you don't get a massive hit to performance in exchange for the huge memory savings, which is generally the kind of tradeoff you are faced with in AI training.

They also made major advances in GPU communication efficiency through their DualPipe algorithm and custom communication kernels. This system intelligently overlaps computation and communication, carefully balancing GPU resources between these tasks. They only need about 20 of their GPUs' streaming multiprocessors (SMs) for communication, leaving the rest free for computation. The result is much higher GPU utilization than typical training setups achieve.

Another very smart thing they did is to use what is known as a Mixture-of-Experts (MOE) Transformer architecture, but with key innovations around load balancing. As you might know, the size or capacity of an AI model is often measured in terms of the number of parameters the model contains. A parameter is just a

number that stores some attribute of the model; either the "weight" or importance a particular artificial neuron has relative to another one, or the importance of a particular token depending on its context (in the "attention mechanism"), etc.

Meta's latest Llama3 models come in a few sizes, for example: a 1 billion parameter version (the smallest), a 70B parameter model (the most commonly deployed one), and even a massive 405B parameter model. This largest model is of limited utility for most users because you would need to have tens of thousands of dollars worth of GPUs in your computer just to run at tolerable speeds for inference, at least if you deployed it in the naive full-precision version. Therefore most of the real-world usage and excitement surrounding these open source models is at the 8B parameter or highly quantized 70B parameter level, since that's what can fit in a consumer-grade Nvidia 4090 GPU, which you can buy now for under $1,000.

So why does any of this matter? Well, in a sense, the parameter count and precision tells you something about how much raw information or data the model has stored internally. Note that I'm not talking about reasoning ability, or the model's "IQ" if you will: it turns out that models with even surprisingly modest parameter counts can show remarkable cognitive performance when it comes to solving complex logic problems, proving theorems in plane geometry, SAT math problems, etc.

But those small models aren't going to be able to necessarily tell you every aspect of every plot twist in every single novel by Stendhal, whereas the really big models can potentially do that. The "cost" of that extreme level of knowledge is that the models become very unwieldy both to train and to do inference on, because you always need to store every single one of those 405B parameters (or whatever the parameter count is) in the GPU's VRAM at the same time in order to do any inference with the model.

The beauty of the MOE model approach is that you can decompose the big model into a collection of smaller models that each know different, non-overlapping (at least fully) pieces of knowledge. DeepSeek's innovation here was developing what they call an "auxiliary-loss-free" load balancing strategy that maintains efficient expert utilization without the usual performance degradation that comes from load balancing. Then, depending on the nature of the inference request, you can intelligently route the inference to the "expert" models within that collection of smaller models that are most able to answer that question or solve that task.

You can loosely think of it as being a committee of experts who have their own specialized knowledge domains: one might be a legal expert, the other a computer science expert, the other a business strategy expert. So if a question comes in about linear algebra, you don't give it to the legal expert. This is of course a very loose analogy and it doesn't actually work like this in practice.

The real advantage of this approach is that it allows the model to contain a huge amount of knowledge without being very unwieldy, because even though the aggregate number of parameters is high across all the experts, only a small subset of these parameters is "active" at any given time, which means that you only need to store this small subset of weights in VRAM in order to do inference. In the case of DeepSeek-V3, they have an

absolutely massive MOE model with 671B parameters, so it's much bigger than even the largest Llama3 model, but only 37B of these parameters are active at any given time— enough to fit in the VRAM of two consumer-grade Nvidia 4090 GPUs (under $2,000 total cost), rather than requiring one or more H100 GPUs which cost something like $40k each.

It's rumored that both ChatGPT and Claude use an MoE architecture, with some leaks suggesting that GPT-4 had a total of 1.8 trillion parameters split across 8 models containing 220 billion parameters each. Despite that being a lot more doable than trying to fit all 1.8 trillion parameters in VRAM, it still requires multiple H100-grade GPUs just to run the model because of the massive amount of memory used.

Beyond what has already been described, the technical papers mention several other key optimizations. These include their extremely memory-efficient training framework that avoids tensor parallelism, recomputes certain operations during backpropagation instead of storing them, and shares parameters between the main model and auxiliary prediction modules. The sum total of all these innovations, when layered together, has led to the ~45x efficiency improvement numbers that have been tossed around online, and I am perfectly willing to believe these are in the right ballpark.

One very strong indicator that it's true is the cost of DeepSeek's API: despite this nearly best-in-class model performance, DeepSeek charges something like 95% less money for inference requests via its API than comparable models from OpenAI and Anthropic. In a sense, it's sort of like comparing Nvidia's GPUs to the new custom chips from competitors: even if they aren't quite as good, the value for money is so much better that it can still be a no-brainer depending on the application, as long as you can qualify the performance level and prove that it's good enough for your requirements and the API availability and latency is good enough (thus far, people have been amazed at how well DeepSeek's infrastructure has held up despite the truly incredible surge of demand owing to the performance of these new models).

But unlike the case of Nvidia, where the cost differential is the result of them earning monopoly gross margins of 90%+ on their data-center products, the cost differential of the DeepSeek API relative to the OpenAI and Anthropic API could be simply that they are nearly 50x more compute efficient (it might even be significantly more than that on the inference side— the ~45x efficiency was on the training side). Indeed, it's not even clear that OpenAI and Anthropic are making great margins on their API services— they might be more interested in revenue growth and gathering more data from analyzing all the API requests they receive.

Before moving on, I'd be remiss if I didn't mention that many people are speculating that DeepSeek is simply lying about the number of GPUs and GPU hours spent training these models because they actually possess far more H100s than they are supposed to have given the export restrictions on these cards, and they don't want to cause trouble for themselves or hurt their chances of acquiring more of these cards. While it's certainly possible, I think it's more likely that they are telling the truth, and that they have simply been able to achieve these incredible results by being extremely clever and creative in their approach to training and inference. They

explain how they are doing things, and I suspect that it's only a matter of time before their results are widely replicated and confirmed by other researchers at various other labs.

A Model That Can Really Think

The newer R1 model and technical report might even be even more mind blowing, since they were able to beat Anthropic to Chain-of-thought and now are basically the only ones besides OpenAI who have made this technology work at scale. But note that the O1 preview model was only released by OpenAI in mid-September of 2024. That's only ~4 months ago! Something you absolutely must keep in mind is that, unlike OpenAI, which is incredibly secretive about how these models really work at a low level, and won't release the actual model weights to anyone besides partners like Microsoft and other who sign heavy-duty NDAs, these DeepSeek models are both completely open-source and permissively licensed. They have released extremely detailed technical reports explaining how they work, as well as the code that anyone can look at and try to copy.

With R1, DeepSeek essentially cracked one of the holy grails of AI: getting models to reason step-by-step without relying on massive supervised datasets. Their DeepSeek-R1-Zero experiment showed something remarkable: using pure reinforcement learning with carefully crafted reward functions, they managed to get models to develop sophisticated reasoning capabilities completely autonomously. This wasn't just about solving problems— the model organically learned to generate long chains of thought, self-verify its work, and allocate more computation time to harder problems.

The technical breakthrough here was their novel approach to reward modeling. Rather than using complex neural reward models that can lead to "reward hacking" (where the model finds bogus ways to boost their rewards that don't actually lead to better real-world model performance), they developed a clever rule-based system that combines accuracy rewards (verifying final answers) with format rewards (encouraging structured thinking). This simpler approach turned out to be more robust and scalable than the process-based reward models that others have tried.

What's particularly fascinating is that during training, they observed what they called an "aha moment," a phase where the model spontaneously learned to revise its thinking process mid-stream when encountering uncertainty. This emergent behavior wasn't explicitly programmed; it arose naturally from the interaction between the model and the reinforcement learning environment. The model would literally stop itself, flag potential issues in its reasoning, and restart with a different approach, all without being explicitly trained to do this.

The full R1 model built on these insights by introducing what they call "cold-start" data— a small set of high-quality examples— before applying their RL techniques. They also solved one of the major challenges in reasoning models: language consistency. Previous attempts at chain-of-thought reasoning often resulted in models mixing languages or producing incoherent outputs. DeepSeek solved this through a clever language

consistency reward during RL training, trading off a small performance hit for much more readable and consistent outputs.

The results are mind-boggling: on AIME 2024, one of the most challenging high school math competitions, R1 achieved 79.8% accuracy, matching OpenAI's O1 model. On MATH-500, it hit 97.3%, and it achieved the 96.3 percentile on Codeforces programming competitions. But perhaps most impressively, they managed to distill these capabilities down to much smaller models: their 14B parameter version outperforms many models several times its size, suggesting that reasoning ability isn't just about raw parameter count but about how you train the model to process information.

The Fallout

The recent scuttlebutt on Twitter and Blind (a corporate rumor website) is that these models caught Meta completely off guard and that they perform better than the new Llama4 models which are still being trained. Apparently, the Llama project within Meta has attracted a lot of attention internally from high-ranking technical executives, and as a result they have something like 13 individuals working on the Llama stuff who each individually earn more per year in total compensation than the combined training cost for the DeepSeek-V3 models which outperform it. How do you explain that to Zuck with a straight face? How does Zuck keep smiling while shoveling multiple billions of dollars to Nvidia to buy 100k H100s when a better model was trained using just 2k H100s for a bit over $5mm?

But you better believe that Meta and every other big AI lab is taking these DeepSeek models apart, studying every word in those technical reports and every line of the open source code they released, trying desperately to integrate these same tricks and optimizations into their own training and inference pipelines. So what's the impact of all that? Well, naively it sort of seems like the aggregate demand for training and inference compute should be divided by some big number. Maybe not by 45, but maybe by 25 or even 30? Because whatever you thought you needed before these model releases, it's now a lot less.

Now, an optimist might say "You are talking about a mere constant of proportionality, a single multiple. When you're dealing with an exponential growth curve, that stuff gets washed out so quickly that it doesn't end up matter all that much." And there is some truth to that: if AI really is as transformational as I expect, if the real-world utility of this tech is measured in the trillions, if inference-time compute is the new scaling law of the land, if we are going to have armies of humanoid robots running around doing massive amounts of inference constantly, then maybe the growth curve is still so steep and extreme, and Nvidia has a big enough lead, that it will still work out.

But Nvidia is pricing in a LOT of good news in the coming years for that valuation to make sense, and when you start layering all these things together into a total mosaic, it starts to make me at least feel extremely uneasy about spending ~20x the 2025 estimated sales for their shares. What happens if you even see a slight

moderation in sales growth? What if it turns out to be 85% instead of over 100%? What if gross margins come in a bit from 75% to 70%— still ridiculously high for a semiconductor company?

Wrapping it All Up

At a high level, NVIDIA faces an unprecedented convergence of competitive threats that make its premium valuation increasingly difficult to justify at 20x forward sales and 75% gross margins. The company's supposed moats in hardware, software, and efficiency are all showing concerning cracks. The whole world— thousands of the smartest people on the planet, backed by untold billions of dollars of capital resources— are trying to assail them from every angle.

On the hardware front, innovative architectures from Cerebras and Groq demonstrate that NVIDIA's interconnect advantage— a cornerstone of its data center dominance— can be circumvented through radical redesigns. Cerebras' wafer-scale chips and Groq's deterministic compute approach deliver compelling performance without needing NVIDIA's complex interconnect solutions. More traditionally, every major NVIDIA customer (Google, Amazon, Microsoft, Meta, Apple) is developing custom silicon that could chip away at high-margin data center revenue. These aren't experimental projects anymore— Amazon alone is building out massive infrastructure with over 400,000 custom chips for Anthropic.

The software moat appears equally vulnerable. New high-level frameworks like MLX, Triton, and JAX are abstracting away CUDA's importance, while efforts to improve AMD drivers could unlock much cheaper hardware alternatives. The trend toward higher-level abstractions mirrors how assembly language gave way to C/C++, suggesting CUDA's dominance may be more temporary than assumed. Most importantly, we're seeing the emergence of LLM-powered code translation that could automatically port CUDA code to run on any hardware target, potentially eliminating one of NVIDIA's strongest lock-in effects.

Perhaps most devastating is DeepSeek's recent efficiency breakthrough, achieving comparable model performance at approximately 1/45th the compute cost. This suggests the entire industry has been massively over-provisioning compute resources. Combined with the emergence of more efficient inference architectures through chain-of-thought models, the aggregate demand for compute could be significantly lower than current projections assume. The economics here are compelling: when DeepSeek can match GPT-4 level performance while charging 95% less for API calls, it suggests either NVIDIA's customers are burning cash unnecessarily or margins must come down dramatically.

The fact that TSMC will manufacture competitive chips for any well-funded customer puts a natural ceiling on NVIDIA's architectural advantages. But more fundamentally, history shows that markets eventually find a way around artificial bottlenecks that generate super-normal profits. When layered together, these threats suggest NVIDIA faces a much rockier path to maintaining its current growth trajectory and margins than its valuation implies. With five distinct vectors of attack— architectural innovation, customer vertical integration, software

abstraction, efficiency breakthroughs, and manufacturing democratization— the probability that at least one succeeds in meaningfully impacting NVIDIA's margins or growth rate seems high. At current valuations, the market isn't pricing in any of these risks.

I hope you enjoyed reading this article. If you work at a hedge fund and are interested in consulting with me on NVDA or other AI-related stocks or investing themes, I'm already signed up as an expert on GLG and Coleman Research.