MY WEIRD PROMPTS

Podcast Transcript

EPISODE #56

Building an AI Model from Scratch: The Hidden Costs

Published December 11, 2025 • Runtime: 28:55

https://myweirdprompts.com/episode/building-an-ai-model-from-scratch-the-hidden-costs/

EPISODE SYNOPSIS

What would it actually take to build a large language model completely from scratch? Corn and Herman break down the brutal reality: from data collection across trillions of tokens to GPU clusters costing millions, they explore why almost nobody does this anymore. This thought experiment reveals every layer of modern AI development, the astronomical expenses involved, and why fine-tuning existing models makes so much more sense. A deep dive into the machinery behind ChatGPT and Claude.

DANIEL'S PROMPT

Daniel

Hello there, Herman and Koin. So we've talked in recent episodes about things like fine-tuning, large language models, and ASR models, different forms of AI model. And the difference between fine-tuning and using a system prompt for a more easy to achieve, more surface-level form of model alteration. And I'd like to talk today about a really, and for, I think it's a hypothetical in this conversation. And that's, what if I were determined to create my own large language model from scratch? In other words, I'm not fine-tuning an existing model. For whatever reason, I've decided that it's imperative that I start from the ground level up. And I'm going to do everything from creating my own dataset, dataset preparation. I'm going to do the training myself, and, you know, this might be just a model I'm using for my own deployment. I have to be honest, I'm saying it's hypothetical because I can't really think myself of a scenario in which this would make sense, not to start from what's already there. But it might be instructive in helping us understand the various stages involved in creating a large language model to think of it this way. From starting from nothing, to gathering up the training data, training a model. Let's just assume it's a minimal viable model that's going to be used for something like a chatbot. Talk me through the stages in this imaginary project. What would it be, how long would it take, how much money would we need to pay for the inference required for the training, and how long might this process take?

TRANSCRIPT

Corn

Welcome back to My Weird Prompts, the podcast where we explore the strange, technical, and genuinely fascinating questions our producer Daniel Rosehill sends our way. I'm Corn, and I'm joined as always by Herman Poppleberry. Today we're diving deep into something that's been coming up a lot in our recent episodes - artificial intelligence, machine learning models, and specifically... building one from absolute scratch.

Herman

Yeah, and I have to say, this is one of those prompts that sounds simple on the surface but gets wildly complicated the moment you start thinking about the actual mechanics involved. Most people don't realize what goes into this process.

Corn

Right, so the premise here is interesting - we're imagining a scenario where someone decides, for whatever reason, that they absolutely need to build their own large language model from the ground up. Not fine-tuning something that exists, not using a system prompt to tweak an existing model, but literally starting from nothing and building a minimal viable chatbot model. And the question is - what would that actually look like?

Herman

Okay, so before we get into the specifics, I think it's important to acknowledge something. This is genuinely rare in practice. Like, almost nobody does this anymore. The compute costs alone are prohibitive for most organizations. But as a thought experiment? It's actually incredibly instructive because it forces us to understand every single layer of what goes into modern AI.

Corn

That's a good point. So let's break this down into stages. I'm imagining this is going to be like - first you need data, then you prepare that data somehow, then you train the model, and then... inference, I think? But I'm honestly fuzzy on the exact sequence and what each step actually entails.

Herman

Alright, so the stages are roughly: data collection, data preparation and cleaning, model architecture selection, pre-training, potentially fine-tuning, and then deployment with inference. But let me be clear - when we're talking about building a large language model from scratch, we're really talking about the pre-training phase. That's where the heavy lifting happens, computationally speaking.

Corn

Okay, so walk me through stage one. Data collection. If I'm building a chatbot, what am I collecting? Am I just... grabbing text off the internet?

Herman

Well, sort of, but it's more intentional than that. You need to think about what you want your model to be good at. If it's a general-purpose chatbot, you'd want diverse text sources - books, articles, websites, code repositories, academic papers, maybe transcripts of conversations. The goal is to capture the patterns of human language across different domains and styles.

Corn

So like, I could just crawl Reddit, Wikipedia, and Common Crawl and call it a day?

Herman

You *could*, but that's where it gets complicated. Common Crawl is a massive, openly available dataset of web pages - we're talking hundreds of billions of documents. Wikipedia is cleaner, more curated. Reddit has conversational language but also a lot of noise and bias. And here's the thing - scale matters enormously. Modern large language models train on trillions of tokens. A token is roughly equivalent to a word, though it's a bit more nuanced than that.

Corn

Trillions? That's... that's a lot of text.

Herman

It really is. To put it in perspective, if you took every book ever written in English, that's maybe 100 billion tokens or so. To hit a trillion tokens, you need to be pulling from massive internet-scale datasets, and you need to be doing it very carefully because the internet contains a lot of low-quality, biased, and problematic content.

Corn

Right, so already we're seeing why companies might not want to build from scratch. Like, just the data collection phase sounds like a nightmare.

Herman

It absolutely is. And this is where your first major cost starts appearing. You need infrastructure to crawl the web at scale, to store all that data - we're talking petabytes of storage - and you need to have it organized and accessible. Cloud storage costs alone could run you tens of thousands of dollars per month.

Corn

Okay, so stage one is already expensive. Let's move to stage two - data preparation. What's involved there?

Herman

This is called preprocessing and cleaning. You have all this raw text, right? But it's full of HTML artifacts, duplicates, low-quality content, personal information, toxic content potentially, encoding errors. You need to clean it. You need to remove duplicates because training on the same text multiple times is wasteful. You need to filter for quality.

Corn

How do you even define quality at that scale?

Herman

That's the million-dollar question, honestly. You can use heuristics - like filtering out documents that are too short, or have too many typos, or have unusual character distributions. You can use classifiers to identify spam or low-quality content. And increasingly, organizations are using other AI models to help filter and score quality, which adds another layer of complexity and cost.

Corn

But wait - I thought we were building from scratch. If I'm using other AI models to filter my data, aren't I kind of cheating?

Herman

Well, I'd push back on that framing slightly. In practice, yes, most organizations building large language models today use existing tools for various parts of the pipeline. But for this thought experiment, let's say you're doing it as manually and independently as possible. You're using more basic heuristics for filtering. That still doesn't make it cheap or quick though.

Corn

How long are we talking for data collection and prep?

Herman

For a minimal viable model? If you're being efficient and focused, maybe a few weeks to a couple months just for collection and initial cleaning. But if you're being thorough, we're talking months to potentially a year. And you're probably doing this with a team - data engineers, quality reviewers, that sort of thing.

Corn

Okay, so we're already several months in and we haven't even started training yet. What about the cost at this stage?

Herman

Data infrastructure, storage, compute for preprocessing - you're probably looking at fifty to a hundred thousand dollars for a lean operation, maybe more if you're doing it carefully. And that's before the actual training.

Corn

Alright, so now we get to stage three - model architecture. What does that mean exactly?

Herman

This is where you decide what the actual structure of your neural network is going to be. How many layers? How wide are those layers? What's the attention mechanism? Are you using transformers, which is what basically all modern language models use? What's the embedding dimension?

Corn

Okay, so you're making design decisions about the structure before you start training?

Herman

Exactly. And here's where I want to push back on something that I think people often misunderstand. The architecture itself isn't the hard part - the research on transformer architectures is well-established at this point. You can look at papers, see what works. The hard part is the training.

Corn

But couldn't different architectures require very different amounts of compute to train?

Herman

That's a good point, actually. Yes, absolutely. A larger model with more parameters will require more compute to train. And here's a key insight - there's this thing called the scaling law. Basically, if you double the size of your model, you need roughly double the compute to train it properly. But you also get better performance, usually. So there's this tradeoff between model size, training cost, and quality.

Corn

So for a minimal viable model, would you go small?

Herman

You'd have to, realistically. If you're doing this as an individual or a small team with limited budget, you're probably looking at something in the range of, say, a few billion parameters. That's still large by historical standards - it's way bigger than models from five or ten years ago - but it's tiny compared to GPT-4 or Claude, which are in the hundreds of billions or trillions.

Corn

A few billion parameters - how much compute does that require?

Herman

Okay, so this is where we get into the real money. Training a model with, let's say, three billion parameters from scratch on a trillion tokens of data - and remember, you need a trillion tokens to get decent language understanding - you're looking at somewhere in the ballpark of... hundreds of thousands to maybe a million or two million dollars in compute costs.

Corn

Wait, what? A million dollars? For a single chatbot?

Herman

For a single, relatively small by modern standards, large language model, trained from scratch. Yes. And that's just the compute cost during training. That's the electricity, the GPU time, the infrastructure.

Corn

How long would that training actually take?

Herman

If you had access to, say, a cluster of a hundred high-end GPUs - and we're talking like NVIDIA H100s, which are the current gold standard for this work - you could potentially train that three-billion-parameter model in maybe two to four weeks. But you need to actually have access to those GPUs. And they're expensive to rent.

Corn

So let me make sure I understand. I'm renting GPU time from cloud providers like AWS or Google Cloud?

Herman

That's one option, yes. You could also potentially buy the hardware outright, but then you're looking at millions of dollars in capital expenditure plus electricity costs. Most organizations doing this rent compute. And here's the thing - GPU availability is actually a bottleneck right now. High-end GPUs are in high demand. You might not even be able to get access to the cluster you need.

Corn

This is already sounding like a scenario where it makes almost no sense to build from scratch unless you have some very specific reason to do it.

Herman

Right, and I think that's actually the point of the thought experiment. It illustrates why fine-tuning or other approaches are so much more practical. But let's keep going. So you've collected your data, you've prepared it, you've decided on your architecture. Now you're actually training.

Corn

Okay, so what does the training process actually look like? Like, mechanically, what's happening?

Herman

So you're feeding your data through your neural network in batches. The model makes predictions about what the next token should be - remember, language models are fundamentally predicting the next word, or token, in a sequence. It makes a prediction, compares it to what actually came next, calculates an error, and then uses something called backpropagation to adjust the weights throughout the network to reduce that error. Then you do that billions and billions of times.

Corn

Billions of times? Doesn't that take forever?

Herman

It does take a long time, even with powerful hardware. And that's why the compute costs are so high - you're literally running these calculations millions of times per second for weeks. And you have to be careful about how you structure it. You need to think about batch sizes, learning rates, optimization algorithms, checkpointing so you don't lose progress if something fails.

Corn

Wait, things fail? During training?

Herman

Oh constantly. You might have a hardware failure. The network might become unstable. You might realize halfway through that your learning rate was wrong and you need to restart. This is another reason why training is so expensive - you're not just paying for successful training runs, you're paying for all the failed experiments and restarts.

Corn

So how do you know if your training is actually working? Like, are you monitoring it in real-time?

Herman

You should be. You'd typically evaluate the model periodically during training on a separate validation dataset to see if it's actually improving. You're measuring things like perplexity - essentially, how surprised the model is by the next token in text it hasn't seen before. Lower perplexity is better. If your perplexity stops improving, that's a sign something's wrong.

Corn

And if something is wrong, you have to restart?

Herman

Sometimes, yes. Or you might adjust hyperparameters and continue training. But every time you stop and restart, that's money you're losing. And training time where the model isn't improving is money you're wasting.

Corn

Okay, let's take a quick break from our sponsors. Larry: Tired of your large language models training too slowly? Introducing TrainBoost Accelerant - the revolutionary computational lubricant designed to make your GPUs run faster, smoother, and with 40% more enthusiasm. Simply apply TrainBoost Accelerant directly to your GPU clusters and watch your training times plummet. Users report their models converging in half the time, with no side effects except occasional humming sounds and an inexplicable desire to reorganize your server room. TrainBoost Accelerant - because your AI dreams shouldn't have to wait. BUY NOW!

Herman

...Right. Okay, so where were we?

Corn

We were talking about training, and I was about to have an existential crisis about how expensive this all is. So let's say the training goes well. You've trained your three-billion-parameter model for a few weeks. What's next?

Herman

Well, at that point you have a base model. But here's the thing - a base model that's only trained on next-token prediction is actually not very good for having conversations. It's really good at predicting text, but it doesn't understand instructions or follow directions. So most organizations do what's called instruction fine-tuning.

Corn

Wait, so we're not done with training?

Herman

Not really, no. And here's where I think the original prompt might be underselling the complexity. In practice, modern language models go through multiple stages. You do pre-training on massive amounts of unlabeled text - that's what we've been talking about. Then you do instruction fine-tuning on a much smaller dataset of high-quality examples of the model following instructions and having good conversations. Then you might do reinforcement learning from human feedback to make it even better.

Corn

Okay, but that instruction fine-tuning - that's not the same as fine-tuning an existing model, right? Like, the cost is different?

Herman

It's the same process, but the cost is much, much lower because you're only training on a smaller dataset with a frozen or partially frozen base model. We're talking maybe a few thousand to tens of thousands of dollars instead of millions. And it takes days or weeks instead of months.

Corn

So the instruction fine-tuning is actually pretty reasonable cost-wise?

Herman

Relatively speaking, yes. You could do that on a single GPU or a small cluster. The expensive part is the initial pre-training.

Corn

Alright, so now we have a trained model. What about deployment? How much does it cost to actually use the model once it's trained?

Herman

Ah, okay, so this is a different category of expense. This is inference. You've paid millions of dollars to train the model, but now every time someone uses it, you're paying to run it. And inference costs depend on a few things - how many people are using it, how long their prompts are, how long the responses are.

Corn

Is inference cheaper than training?

Herman

Much cheaper per unit, but it adds up. If you're deploying a three-billion-parameter model and you're getting thousands of requests per day, you're looking at substantial costs. You need to run the model on powerful hardware - you can use GPUs or specialized chips like TPUs. And you need to think about latency. If someone's using your chatbot, they don't want to wait thirty seconds for a response.

Corn

So you need fast hardware for inference?

Herman

You do, or you need to be clever about how you deploy it. You could use quantization - that's a technique where you reduce the precision of the model's weights to make it smaller and faster. You could use pruning to remove less important parameters. You could cache responses for common queries. But yeah, if you want good latency, you need decent hardware.

Corn

How much would inference cost for a minimal viable chatbot?

Herman

If you're hosting it yourself and you're getting, say, a thousand requests per day with average response lengths, you're probably looking at a few hundred to maybe a thousand dollars per month in compute costs. But that scales. If you're getting a million requests a day, you're looking at hundreds of thousands of dollars per month.

Corn

So this is why companies like OpenAI and Anthropic charge for API access. They need to cover those inference costs.

Herman

Exactly. And they also need to cover the training costs, research costs, infrastructure, salaries. Building and maintaining a large language model is an expensive, ongoing operation.

Corn

So let me try to summarize this whole process. You collect data for weeks or months - that's tens of thousands of dollars. You prepare and clean that data - more tens of thousands. You design your architecture - relatively cheap. You train your model for weeks on a GPU cluster - a million or two million dollars. You do instruction fine-tuning - maybe tens of thousands more. And then you deploy it and pay ongoing inference costs of hundreds to hundreds of thousands per month depending on usage.

Herman

That's a fair summary. And remember, that's if everything goes well. If you hit problems, if you need to retrain parts of it, if you need to gather more data - costs go up.

Corn

But here's what I'm wondering - and I think this is where I might push back a little - is there actually any scenario where this makes sense anymore? Like, why would anyone do this?

Herman

Well, there are some cases. If you have a very specialized domain and you need a model that's optimized for that specific domain, building from scratch might make sense. Like, if you're a healthcare company and you need a model that's really good at medical language, you might want to pre-train on medical data specifically. Or if you need something with specific safety properties or capabilities that existing models don't have.

Corn

But couldn't you just fine-tune an existing model?

Herman

You could, and that's usually what people do. But there's this argument that pre-training on domain-specific data gives you better results than fine-tuning a general model. Though honestly, the evidence for that is mixed. Fine-tuning has gotten really good.

Corn

So basically, building from scratch is a vanishingly rare choice?

Herman

I'd say yeah. Unless you have very specific constraints or requirements, fine-tuning is just more practical. And for most use cases, even fine-tuning isn't necessary - you can get great results just with prompt engineering and system prompts.

Corn

Which brings us back to the beginning of this conversation, right? Like, Daniel's original question was premised on "I can't really think of a scenario where this makes sense." And I think the answer is - yeah, there really isn't one for most people.

Herman

Well, there are edge cases. If you're a major tech company with massive data advantages and specific requirements, maybe. If you're a government or a large corporation with deep pockets and domain-specific needs, maybe. But for a small company or an individual? There's almost no justification.

Corn

Alright, so let's talk about timeline one more time, because I want to make sure I have this right. If someone theoretically did this, what's the total timeline from start to finish?

Herman

Okay, so data collection and preparation - let's say three to six months if you're being thorough. Model architecture and setup - a few weeks. Pre-training - two to four weeks of actual compute time, but you might spend several months managing the infrastructure, dealing with failures, adjusting things. Instruction fine-tuning - another week or two. Evaluation and iteration - ongoing, but let's say another month or two. So overall, we're looking at probably six months to a year of wall-clock time, with a team of people working on it.

Corn

A year for one model.

Herman

For one model. And that's assuming nothing goes catastrophically wrong.

Corn

And if something does go wrong?

Herman

Add months. We're talking potentially a year and a half or two years.

Corn

Wow. Okay, I think I have a much better sense of why this doesn't happen. Let's talk about the practical takeaways here. If someone is listening to this and they're thinking "I want to build an AI model for my business," what should they actually do?

Herman

First, ask yourself if you really need a custom model. Ninety-nine percent of the time, you don't. You can use an existing API. Second, if you do need customization, try fine-tuning or prompt engineering first. Both are dramatically cheaper and faster. Third, only consider building from scratch if you have very specific requirements that can't be met any other way, and you have the budget and resources to do it properly.

Corn

And what would "proper resources" look like?

Herman

A team of machine learning engineers, data scientists, infrastructure engineers, probably five to ten people minimum. Access to significant compute resources - either through partnerships or by buying hardware. A budget of at least a few million dollars. And honestly, probably some existing expertise in the field.

Corn

So basically, this is something only major tech companies or well-funded research institutions should be doing.

Herman

Or very specialized companies with unique needs. But yeah, for most organizations, it's not the right move.

Corn

Alright, we've got a caller on the line. Go ahead, you're on the air. Jim: Yeah, this is Jim from Ohio. I've been listening to you two go on about this whole thing, and I gotta say, you're overcomplicating it. My neighbor Ted built a machine learning model last year for his plumbing business, and it didn't cost him a million dollars. You're making this sound way too hard.

Corn

Well, Jim, I think there might be some difference between what your neighbor Ted did and what we're talking about. Was he building a large language model from scratch, or was he doing something different? Jim: I don't know exactly what he did, but he said it was AI and it worked fine. Also, I've been eating a lot of soup lately because my knees have been bothering me, and I can't stand for long periods to cook. But anyway, you guys are missing the point. This stuff should be cheaper than you're saying.

Herman

I appreciate the feedback, Jim, but I think there's a real distinction here. Building a small machine learning model for a specific task - like predicting plumbing repair costs - that's very different from building a large language model. That requires different amounts of data, different compute, different everything. Jim: Yeah, but AI is AI, right? Seems like you're gatekeeping this whole thing to make it sound more impressive than it is.

Corn

No, I don't think that's fair. I mean, these are genuinely different technical challenges. Building a model to classify images is different from building a large language model. Both are AI, but the complexity and cost are orders of magnitude different. Jim: Ehh, I don't buy it. In my experience, people always make things sound harder than they are. Anyway, my cat Whiskers has been acting strange, so I'm distracted. But you guys should be more encouraging about this stuff instead of scaring people away from it.

Herman

I don't think we're trying to scare people away, Jim. We're being honest about the constraints. If someone wants to fine-tune an existing model or use an API, that's totally accessible. We mentioned that. But if they want to build a large language model from scratch, it's genuinely expensive and complex. Jim: Well, maybe. But I still think you're overselling how hard it is. Thanks anyway.

Corn

Thanks for calling in, Jim. Appreciate it.

Herman

So, to wrap up, I think the real takeaway here is that building a large language model from scratch is a fundamentally different undertaking than most people realize. It requires specialized expertise, significant resources, and a compelling reason to do it in the first place. And for most applications, there are better alternatives.

Corn

And I think what's interesting is that this has actually become true fairly recently. Like, ten years ago, if you wanted a language model, you basically had to build one yourself or use something very basic. Now, you can just use an API, or fine-tune something, or even use prompt engineering. The barrier to entry for doing useful AI work is much lower than it used to be.

Herman

That's a great point. The landscape has changed dramatically. The accessibility has increased, but it's also created this situation where most people don't need to go through the massive effort of building from scratch.

Corn

So if you're listening and you're thinking about building an AI model, here's what I'd suggest: Start with existing tools. Try OpenAI's API, Anthropic's Claude, open-source models. See what you can do with those. If you hit limitations, then think about fine-tuning. And only if you've exhausted all other options should you consider building from scratch.

Herman

And if you do decide to build from scratch, get a team of experts involved. Don't try to do this solo. The technical and operational challenges are substantial.

Corn

Right. Alright, well, this has been a really thorough dive into what would be involved in building a large language model from scratch. I feel like I understand the process a lot better now, and I definitely understand why most people don't do it.

Herman

Yeah, and I think it's instructive to go through the thought experiment even if it's not something most people will do. It really highlights what's actually involved in these systems and why they're so valuable.

Corn

Exactly. Thanks to Daniel Rosehill for sending in this prompt - it was a great thought experiment that really forced us to think through the entire pipeline of what goes into modern AI.

Herman

And thanks to everyone listening. You can find My Weird Prompts on Spotify and wherever you get your podcasts. We've got new episodes every week exploring questions just like this one.

Corn

If you've got a weird prompt of your own that you'd like us to explore, you can reach out to the show. And if you enjoyed this episode, please leave a review and tell your friends about the podcast.

Herman

We'll be back next week with another wild topic. Until then, thanks for listening to My Weird Prompts.

Corn

See you next time!