Mihail Eric

@mihail_eric

7h • 1 tweets • 5 min read • Read on X

How Alexa dropped the ball on being the top conversational system on the planet

—

A few weeks ago OpenAI released GPT-4o ushering in a new standard for multimodal, conversational experiences with sophisticated reasoning capabilities.

Several days later, my good friends at PolyAI announced their Series C fundraising round after tremendous growth in the usage of their enterprise voice assistant.

Amid this news, a former Alexa colleague messaged me: You'd think voice assistants would have been our forte at Alexa.

For context, I joined Alexa AI as a research scientist in early 2019. By this time, the Alexa consumer device had existed for 5 years and was already in 100M+ homes throughout the world.

In 2019, Alexa was experiencing a period of hypergrowth. Dozens of new teams sprouted every quarter, huge financial resources were invested, and senior leadership made it clear that Alexa was going to be one of Amazon's big bets moving forward.

My team was born amidst all this with a simple charter: bring the latest and greatest in AI research into the Alexa product and ecosystem. I've often described our group (later dubbed the Conversational Modeling team) as Google Brain meets Alexa AI-SWAT team.

Over the course of the 2.5 years I was there, we grew from 2 to ~20 and tackled every part of the conversational systems stack.

We built the first LLMs for the organization (though back then we didn't call them LLMs), we built knowledge grounded response generators (though we didn't call it RAG), and we pioneered prototypes for what it would mean to make Alexa a multimodal agent in your home.

We had all the resources, talent, and momentum to become the unequivocal market leader in conversational AI. But most of that tech never saw the light of day and never received any noteworthy press.

Why?

The reality is Alexa AI was riddled with technical and bureaucratic problems.

Bad Technical Process

–

Alexa put a huge emphasis on protecting customer data with guardrails in place to prevent leakage and access. Definitely a crucial practice, but one consequence was that the internal infrastructure for developers was agonizingly painful to work with.

It would take weeks to get access to any internal data for analysis or experiments. Data was poorly annotated. Documentation was either nonexistent or stale.

Experiments had to be run in resource-limited compute environments. Imagine trying to train a transformer model when all you can get a hold of is CPUs. Unacceptable for a company sitting on one of the largest collections of

accelerated hardware in the world.

I remember on one occasion our team did an analysis demonstrating that the annotation scheme for some subset of utterance data was completely wrong, leading to incorrect data labels.

That meant for months our internal annotation team had been mislabeling thousands of data points every single day. When we attempted to get the team to change their annotation taxonomy, we discovered it would require a herculean effort to get even the smallest thing modified.

We had to get the team's PM onboard, then their manager's buy-in, then submit a preliminary change request, then get that approved (a multi-month-long process end-to-end).

And most importantly, there was no immediate story for the team's PM to make a promotion case through fixing this issue other than "it's scientifically the right thing to do and could lead to better models for some other team." No incentive meant no action taken.

Since that wasn't our responsibility and the lift from our side wasn't worth the effort, we closed that chapter and moved on.

For all I know, they could still be mislabeling those utterances to this day.

Fragmented Org Structures

—

Alexa's org structure was decentralized by design meaning there were multiple small teams working on sometimes identical problems across geographic locales.

This introduced an almost Darwinian flavor to org dynamics where teams scrambled to get their work done to avoid getting reorged and subsumed into a competing team.

The consequence was an organization plagued by antagonistic mid-managers that had little interest in collaborating for the greater good of Alexa and only wanted to preserve their own fiefdoms.

My group by design was intended to span projects, whereby we found teams that aligned with our research/product interests and urged them to collaborate on ambitious efforts. The resistance and lack of action we encountered was soul-crushing.

I remember on one occasion we were coordinating a project to scale out the large transformers model training I had been leading. This was an ambitious effort which, if done correctly, could have been the genesis of an Amazon ChatGPT (well before ChatGPT was released).

Our Alexa team met with an internal cloud team which independently was initiating similar undertakings. While the goal was to find a way to collaborate on this training infrastructure, over the course of several weeks there were many half-baked promises made which never came to fruition.

At the end of it, our team did our own thing and the sister team did their own thing. Duplicated efforts due to no shared common ground. With no data, infrastructure, or lesson sharing, this inevitably hurt the quality of produced models.

As another example, the Alexa skills ecosystem was Alexa's attempt to apply Amazonian decentralization to the dialogue problem. Have individual teams own individual skills.

But dialogue is not conducive to that degree of separation of concerns. How can you seamlessly hand off

conversational context between skills? This means endowing the system with multi-turn memory (a long-standing dream of dialogue research).

The internal design of the skills ecosystem made achieving this infeasible because each skill acted like its own independent bot. It was conversational AI by an opinionated bot committee each with its own agenda.

Product-Science Misalignment

—

Alexa was viciously customer-focused which I believe is admirable and a principle every company should practice. Within Alexa, this meant that every engineering and science effort had to be aligned to some downstream product.

That did introduce tension for our team because we were supposed to be taking experimental bets for the platform's future. These bets couldn't be baked into product without hacks or shortcuts in the typical quarter as was the expectation.

So we had to constantly justify our existence to senior leadership and massage our projects with metrics that could be seen as more customer-facing.

For example, in one of our projects to build an open-domain chat system, the success metric (i.e. a single integer value representing overall conversational quality) imposed by senior leadership had no scientific grounding and was borderline impossible to achieve.

This introduced product/science conflict in every weekly meeting to track the project's progress leading to manager churn every few months and an eventual sunsetting of the effort.

—

As we look forward, in the battle for the future of the conversational AI market, I still believe it's anyone's game.

Today Alexa has sold 500M+ devices, which is a mind-boggling user data moat. But that alone is not enough.

Here's how I would organize a dialogue systems effort from the ground-up:

Invest in robust developer infrastructure especially around access to compute, data quality assurance, and streamlined data collection processes. Data and compute are the lifeblood of modern ML systems so proactively setting up this foundation is imperative.

Make LLMs the fundamental building block of the dialogue flows. In retrospect, the Alexa skills ecosystem was a premature initiative for the abilities of conversational systems at the time. I liken it to when Leap Motion created and released a developer SDK before the underlying hardware device was stable.

But with the power of modern LLMs, I'm optimistic about redesigning a developer conversational toolkit with LLMs as their primitives.

Ensure product timelines don't dictate science research time frames. Because things are moving so fast in the AI world, it's hard not to feel the pressure of shipping quickly. But there are still so many unsolved problems that will take time to solve.

Of course you should conduct research aggressively, but don't have delivery cycles measured in quarters, as this will produce inferior systems to meet deadlines.

—