Why building eval platforms is hard — Phil Hetzel, Braintrust

25 MIN · YOUTUBE · HTTPS://WWW.YOUTUBE.COM/WATCH?V=_FQ7Z_WFOUK
https://www.youtube.com/watch?v=_fQ7Z_Wfouk

SUMMARY

Phil Hetzel discusses the importance of evaluation (eval) platforms for agents using large language models (LLMs) in a packed session. He emphasizes the need for effective evals to ensure agent performance and mitigate risks, outlining the stages of building an eval platform from basic spreadsheets to advanced systems that integrate observability and analytics.

- *Evals are crucial for ensuring the performance of agents powered by LLMs, which are increasingly expected by users.*
- *The journey of building an eval platform typically starts with simple tools like spreadsheets, progressing to more sophisticated systems.*
- *Collaboration among technical and non-technical team members is essential for effective evals, as diverse expertise enhances agent quality.*
- *Advanced eval platforms should facilitate experimentation, allowing users to tweak agent parameters and compare outputs.*
- *Observability and evals are interconnected; production data can inform offline evals, creating a feedback loop for continuous improvement.*
- *Challenges include managing complex and voluminous data from agent traces, which require robust data platforms for effective analysis.*
- *Future developments in eval platforms should focus on uncovering unknown issues and integrating AI-driven insights for enhanced agent performance.*

All right, it's 11:15. We're going to go ahead and get started. Before we do, everyone say evals. Evals. I was telling my colleague Rose, who is at at the door that I was a adjunct professor for a number of years. And um the first year that I did it, I thought I was going to have this full class of 130 people every single week eager to learn. And then as the weeks went on, 130 became 60, became 30, became 10. So I always tell myself that whenever I give a talk that only about four or five people are going to show up, but I'm going to be really excited to teach those four or five.

Um today is a real blessing because we have we have a a packed house here today. Everyone's excited to learn about evals, and I I'm excited to to teach it. Um here's what we're going to be talking about today. I'll give you a little bit of intro about myself and the company that I work for. An overview of of the problem statement. We'll go into the different stages of when people are building eval platforms.

And then after that we'll we'll talk about, at least in in my opinion, where I think eval platforms are going to go.

Um but yeah, this is this is me. My name is Phil Hetzel. I lead solutions engineering at Braintrust. I'll go into what Braintrust is in a second. Solution engineering that basically means I'm the person and my team are the people that make sure that people are getting the most value out of our platform. And as as quickly as possible.

So I'm fortunate because throughout all of our customers I I see what the state of the art is in in both evals and and agent observability. Prior to BrainTrust, I spent 12 years in consulting and systems implementation. I worked for KPMG for 4 years. I worked for a company called Slalom Consulting for 8 years where I led the global Databricks business unit. And I noticed that as I was helping my my clients with those implementations, they were great they were so good at generating these generative AI proofs of concepts and none of them were getting to production.

And I wanted to be I want to be helpful in making sure that those POCs could get to production. So I actually started using BrainTrust because I knew it helped out in the space. I started using it as a user. And I like the platform so much that I applied for a job and I've been been here for about a year. Outside of work, I I like to play chess but I'm I'm I'm very bad at it.

And I like to spend time with my wife and and my dachshund dachshund is his name Pistol Pete. And he's pictured he's the person in brown. He's not the person in black. Person in black is me. Has anyone heard of BrainTrust before? Anyone? Couple hands. How many people have heard about BrainTrust for the first time this week? Really? Okay, great. Wonderful. BrainTrust for for just a reminder, I think of ourselves as an agent quality platform.

And there's a lot of things that can go into quality. The way that we can get to agent quality two main pillars through evals and through observability. Which we think of as really similar problems to solve. Evals, that's what you're doing with your agent before it gets to production, as you're experimenting, so that you can become confident in your agent. And then observability is really similar, but you're already in production. Your agent is uh in front of real usage from real users, and you want to be confident, you want to remain confident, I should say, that your agent is performing the way that you thought that it it would when you were building it um So, that's BrainTrust. I was specifically told to not make this a sales pitch, so that's like really the last BrainTrust slide that that you'll get today. Uh although, of course, I'm very happy to answer questions about our company this week.

But mainly, I wanted to talk more conceptually about um how people start to mature and and build uh build these platforms, spoken from uh a place where we have a lot of experience in the space. Uh first, why evals are important. Evals are important because um this sounds obvious, but LLMs have extreme variability. Uh we love LLMs because they're highly variable. There are so many different types of problems that LLMs can reason to solve. That's why we're that why we you know, we're so attracted to to them as a technology.

Um agents are also uh of course, agents use LLMs as as the brain of the agent. Agents are becoming the norm in how customers are interacting with companies. People expect an agentic experience now. So, if you combine both of those things together, you really need to be confident in how your agent is going to perform once it is in

production. Without doing so, you're going to incur or you can potentially incur a great deal of risk um from both a a brand perspective, a compliance perspective, uh and even more of a a cost and and and maintenance and systems perspective. So, we want to avoid all of those things happening and make sure that you know, our customers are having a great experience and that our agents are are um acting the way that we thought that they would act.

Um how many people are like they're doing evals right now, but it's just on a Google Sheet or or some spreadsheet is probably There's no shame in that, my friend. Raise that hand high. That's great. Um there's I I and I I think I think that's great. Like just making the step is is really important. It's an acknowledgement of the problem space. And a lot of folks will, you know, they'll they'll come to us and they'll say, "Well, I don't really understand BrainTrust because, you know, I all I need to know is how to loop through my agent with a couple of different inputs and be able to to display some, you know, handwritten notes and scores about that agent." So, the things that I mentioned there, three things, some way to execute your agent, some UI, sometimes it's as simple as a spreadsheet to show those outputs and scores, and then also a way to to gather input examples. What what I mean by input example is the the thing that can initiate a run of an agent, the thing that can invoke an agent, whatever information that's necessary for that.

Um it'll be a really short presentation if if this is all evals was. I would I would thank you for your time and I would walk out the room, but that's not that's not what you're here for. There is a whole other part of the iceberg. It's way more complicated than that. There are a lot of things that you end up having to build when you're really serious about evals. We're not going to talk about every single one of these things today, um but we will touch on on many of them.

And of course, if there's anything here that I don't cover that you're interested in, I'll leave some sometimes time for questions for that. I also see a lot of a lot of phones up. So, I'm going to pause for iceberg pictures. Um, a couple a couple of things while that's happening. Why is this a complicated problem? We already talked about a little bit about how the underlying technology is quite complex. LLMs are are are not a superficial um, engine.

Uh, but it's also a multi-persona problem building these agents. It's not just something that engineers do in isolation. It's something where engineers, whether whether they're product engineers or AI engineers or both, systems engineers to get the thing running. Um, uh, SMEs that have the domain knowledge. All of these people need to be involved. And then lastly, it evals themselves become a systems problem. That'll be the last thing that we that we touch on today.

So, what are the different stages of of building an eval platform? Um, my my my friend over over there that raised his hand proudly about starting out on a spreadsheet. This is this is a great place to start. The most important thing is that you just get started. So, you've got a spreadsheet and you've got a for loop. You've got a bunch of input examples that you can iterate through and you have a way to execute your agent. So, you can say you can see every time you tweak your agent how the outputs are different over time.

Um, while this is a great place to start because there is no barrier to entry here. Everyone has some way to access

some type of spreadsheet technology. Um, the the returns can be diminishing for a couple of reasons. Um, this is more I would call this documenting. It it's not really experimenting. So, while you have this spreadsheet of, you know, a a bunch of input examples. Maybe you keep track across each time you are tweaking your agent that the different output that that emitted. Um, that can become cumbersome to to manage over time, of course. Um, it's really challenging to be able to compare directly experiments over time.

You're probably not doing a lot of analytics across those experiments. And the analytics that you are doing or performing, they're likely coming from some type of human scorer, which is which is really valuable but uh challenging to scale in practice. Um, evals are are a team sport, kind of what I was talking about before. Want to make sure that we're bringing a ton of people into the fold, not just technical folks, but also non-technical folks. They can add a lot of value to uh to your agent because of their domain unique domain expertise and proximity to users. They're probably not coming into the spreadsheet is is my point. Uh, and it's slow. Um, it each time you you eval, um, you have to go through probably a little bit of a cumbersome process to to recreate or append to the spreadsheet.

Um, so uh probably the one of the uh um most fun conversations that uh I have in my job is I'll have, you know, a a very proud product engineer that gets on a call with me and you know, they they puff their chest out and they smirk at me and they say, "Well, I can just vibe code brain trust. It's no problem." Um, and I think for for like like if if if you're just getting started in your journey, it's a really nice step to to go to. So, now instead of being in a spreadsheet land, you're making something a little bit more bespoke for other other people would have bring them into the fold. So, now you've probably got a for loop.

You have a nicer UI now, so it's more approachable. And hopefully you've graduated into some database that that isn't Excel or Google Sheets. You probably um you know use roll roll a a new database in in something like Neon or or something. Um so now you have a a better story around persistence of evals. Uh and because of this, you're bringing more people into the fold, you are um making UIs that are a little bit more bespoke for your specific users.

Um the thing that's that's a problem here is that you're still not really iterating yet. You're still performing work that is a little bit more uh just reporting, just documentation, rather than encouraging a lot of a lot of iteration. So more of a reporting tool here. How many people have vibe coded their own uh UI? Yeah. Makes sense. Uh next step here, so you want to encourage a lot of um experimentation, not just with technical users, but with non-technical users. So um you know, I'm showing this uh image that is more aligned to allowing experimentation for non-technical users. But of course, as you're building these platforms, you want to allow for more SDK driven experience as well. Uh that just doesn't make for for a very nice image in a in a presentation.

So experimentation to me means that you can give a user access to a an agent a a configuration of an agent and a sandbox. And you allow them to tweak certain parameters within within that agent. In my example here, I'm allowing a user in the UI to change the system instructions to an agent running outside of my eval platform, and allowing them to compare two different configurations of that of that of that system prompt. And I'm running evals across those two different agent runs so that I can bubble up scores. You can You can see that in the image

now. I can bubble up different scores to understand both technically and functionally how my agent is behaving. Um so this is like you'll hear about a lot of platforms have a playground feature. You're going to want some type of playground feature both for technical and non-technical users.

This is where the rubber starts meeting the road because the best way to perform evals is to um really think about the failure modes that your agent can fall into um and build scoring functions around those failure modes. The best way to find those failure modes in the first place is to have access to production trace data. I.e. your agent in front of real usage users and in real usage. So the next step here is a really important one. We want to make sure that we can connect what we at least internally we call the flywheel. Observability and evals to us is actually the same problem from a from a systems perspective. Um funny story, we used to be uh 3 years ago when we started we were only an evals platform. And then we noticed one of our customers was running this massive eval like a every hour of every day. So we reached out to this person and they said, "Oh yeah, I'm just piping all of my production traffic into this database and I'm running an eval against it." So we're like, "Okay, we should probably just make make that ability to trace uh and and observe actual traffic and be it and account for that use case without having to cram it into offline evals."

Uh so this is really important. Make sure that we can observe things in production, understand the actual behavior of our agents, also understand the the real lift that the changes that we're making to our agents or or having. Um so we're analyzing that data. Uh we pull that back those actual examples back into an offline environment and then we improve upon those using uh offline evals. This is a loop so it's it's not just a a process.

Uh you're going to be performing this loop hopefully for the lifetime of the agent that that you're pushing to production. You you you should be iterating this loop as many times as possible. That's how you that's how you improve. Um so as a result of that you've you've changed your scope a little. You've widened your scope a lot actually. You are now a tracing plat- platform. You're now a logging platform in addition to being an offline evals platform.

Again, the benefit of that is that you got you're starting to get far higher signal from how users are interacting with uh with your agents and you can use those real interactions uh so you can um almost think about evals almost like you're rerunning production in a safe environment. You're now getting to uh to that point um with uh with with this example. Um you can also perform online evals so you can point scoring functions to your uh to your observability traffic uh and perform things like alerting um all things that you could build in when you're at this phase of maturity for uh for running evals.

Uh the bad here uh if you build it, you have to manage it. So just because you've you know uh vibe coded a platform, guess what? You might get a promotion for it, but also like that's going to be your job now uh is is to is to manage and and continue to grow your eval platform at the pace that the industry is moving um which can be an exciting challenge. That's kind of the bet that that our company's making and we're excited to solve that problem.

The more important challenge though is that agent traces specifically, if you kind of look on the on the screen,

these are really nasty. They're not like normal application traces. Um they are they are really semi-structured. A lot of times they're unstructured. There's just a a ton of text inherent to LLM problems that were that we're solving. Um they're um just like very large in addition to being complicated. So, if you're trying to cram you know, I a 1 GB trace into a Postgres row, that can lead to a lot of performance problems and they're numerous. It's high velocity because there's so much usage happening in production. Hopefully with the with the agent that you've pushed.

Um so, this is how we used to solve this problem. Um just as an example, if if you're at this stage in maturity, you've got traces coming in. You're going to need to account for two query patterns. One, if you're performing observability, you need a way for for folks to instantly be able to see their traces. It's very important to people. So, you'll need a a a very low latency way to ingest data. And then you also need a a second uh a second layer of persistence for the query pattern of I want to be able to analyze in in aggregate these data. So, we used to use an open-source data warehouse for this.

We used to stitch these two sources together through a domain-specific language that we created called BTQL that no one liked and and including us we we we hated it. And then we would perform a like a third level of of aggregation with using DuckDB in the in the browser. Um this worked for us for for a bit and then it it didn't work when I'll just use one one our customer examples. A a customer like Notion for as as an example, just a ton of a lot of of unstructured data that they were sending us. They want to be able to perform things like full text search across a trace. None of these technologies are really equipped to perform text style analytics, which is a challenge with with the LLM domain cuz there's just so much text.

So, that leads us to this, measuring agent quality, performing evals, performing observability. It's actually a systems problem. It's not just a UI UX problem. We recognize that it's quite easy to vibe code the UI of evals, but it's way way way more challenging to create that data layer of of running a successful evals and and observability platform. And not just from a scale perspective, although that matters, mostly from a functional perspective of allowing people to do the things that they would expect to do, like performing full text search across millions of traces in in their platform of choice.

I talked about this a little bit. The reason why this is such a novel problem to solve is across a lot of these dimensions, which I you know, I I won't drain this slide, but the data comes in really fast. The data are are are like just really large when they come in. So, even though traditional spans in a trace, span is just like one part of a trace, traditional span would be like a couple of kilobytes. Here, we've seen spans that are 10, 20 megabytes in size. Just so much context within those spans.

Highly highly unstructured. And then also there are a lot of different types of read patterns. So, you might be performing aggregate types of read patterns, but also you want very low latency types of of read read patterns. So, none of these problems are are individually unique, but together they make a make for a very unique problem from a systems perspective. Uh so what we've done is um and you know, what you all would have to endeavor to do if you were building this yourselves is you really have to think about making the right data platform for traces so that you can perform some of the more functional requirements that that eventually come

down the line.

The example that I have here is that you know, let's say that you want to let a coding agent loose on your evals platform so that you can be a little bit more self-healing with grabbing data in aggregate from your evals platform using a coding agent to grab that into context and change your agent within you know, with with within a coding agent session. That's something that's going to be really challenging to do if you can run a lot of just pure SQL on the data back end of of your evals platform. We've actually noticed a lot of these headless style use cases come up where people aren't in interested in the UI at all. The only thing that they're interested in is how can I perform evals in a way where I can use a codex or I can use a cloud code to to help help increase the quality of my agent for me.

So the the last problem here that that I'll talk about is the so what problem. Um and we'll we'll we'll skip this for now for the sake of time. This is how Braintrust does this. We have a blog about this if if you're interested that that just got released. Um Um but what what kind of comes next here for um what you can expect to build into your evals platform is you want to be able to tell folks the unknown unknowns of your agent. I.E. Don't make me look across a whole bunch of traces. Just tell me how people are are using our our agent. So, you want to be able to uncover those unknown unknowns unknown unknowns through topic modeling techniques so that you know where to spend your engineering time.

Um you want to make sure that you are building your platform not just for humans, but also for agents cuz that's one of the main media for how people are are creating technology now. Um we didn't even talk about the non-functional requirements that go into building these platforms like role-based access control, data masking. It's also something that that's super important that comes up when you want to operate at scale. Uh and lastly, uh a consideration for adding a automatic tracing through some type of AI proxy or gateway so that people don't even have a choice but to trace their their LLMs. Uh you can govern very centrally by adding tracing automatically uh to uh to your eval platform.

Um so, I appreciate the time. Um I've got like a minute and 20 seconds left for for questions. I can probably take two of them. If anyone has any questions. Yes. Okay. Um I'm not sure about BrainTrust, but with BrainTrust and these kind of tools, the problem is often when you create like dynamic prompts and not only only string interpolation, but also like files videos creating LLMs, like solutions struggle and then you often build like custom versions.

How does BrainTrust uh yeah, get around that? So, like how does BrainTrust BrainTrust specifically handle multimodal outputs and inputs and traces? Yeah, we uh like just very technically, we uh put them in some object storage, reference them, and then um display them directly into the trace. So, if you have like an audio file or a video file, you can play it in the trace when someone's reviewing the the trace itself. We don't want people to have to exit the platform for that. And the prompt management is in Brainstorm? It could be, yeah. The question was, is prompt management in Brainstorm? It it it could be or it doesn't have to be.

Yeah. Okay, perfect. Thank you so much for your attention today.