LLM Eval For Text2SQL

51 MIN · YOUTUBE · HTTPS://WWW.YOUTUBE.COM/WATCH?V=UGMENKJGXQM
https://www.youtube.com/watch?v=UGmenkjGXqM

SUMMARY

The presentation discusses the development and evaluation of AI models, particularly focusing on the creation of effective evaluation (eval) processes using Brain Trust, a startup that specializes in AI tooling. It emphasizes the importance of systematic evaluation through three key components: data, task functions, and scoring functions, and demonstrates a practical example of building a text-to-SQL application.

- *Brain Trust originated from the need for better internal tools to evaluate AI models effectively.*
- *Evals are crucial for understanding the impact of changes made to AI models, allowing for quick identification of improvements or regressions.*
- *The eval process consists of three components: data (which can be hardcoded or sourced), task functions (to process input and generate output), and scoring functions (to assess the quality of outputs).*
- *The demo showcased building a text-to-SQL application using NBA data, illustrating how to generate SQL queries and evaluate their correctness.*
- *The importance of creating a "golden data set" was highlighted, which consists of validated examples that can be used for future evaluations.*
- *The presentation emphasized the iterative nature of model development, where continuous improvements can be made by refining data, task functions, and scoring methods.*
- *Brain Trust supports both offline evaluations and online observability, allowing users to gather feedback and improve models in real-time.*
- *The platform is designed to facilitate easy experimentation and exploration of different models and configurations, enhancing the overall development workflow.*

I'm going to do just like a few slides to sort of set the stage on you know some Brain Trust stuff and then just talk about how we think about like how how to build really good evals and then we'll jump into a demo where we just kind of walk through step by step of how to do a bunch of the kind of classic flows in building good evals and then I'll share the notebook afterwards so

yeah just a little bit about Brain Trust we're a startup compy we come from a variety of different backgrounds actually prior to Brain Trust I led the AI team at figma and before that I started a company called impira and at both companies we were shipping AI software and imp Pura we were you know in the stone ages pre chat GPT and you know training our own language models and then figma we were building on top of

you know open Ai and other models that were pre-trained but the problem was was the same it was just really hard to make changes to our code or our prompts and know you know whether we had broken stuff and what we had broken and so we ended up building internal tooling at both companies that basically helped with evals

and that is what turned into Brain Trust so you know you'll see some of that today the only other thing I'll mention

we're you know really fortunate to work with some really great folks as you know team members investors and customers I think one of the really fortunate things for us is we've had the opportunity to build Brain Trust alongside the AI teams at some of the best product companies in the world that are building AI software and so you know something I take a lot of personal pride in is that a

lot of the workflows and UI and just little details in the product are informed by you know some of the engineers at these companies who are giving us feedback all the time okay enough propaganda about Brain Trust let's talk about evals so I you know I think everyone in the course probably already knows this you you've already learned about evals but just like a very quick recap why do evals in the first place I think some of the key takeaways that you want

to have when you're doing an evl and maybe the bar that you should hold yourself or the tools or whatever to is you know can you actually solve these goals when you do an eval do you want can you figure out very quickly did the change that I make improve or regress things and if so what let me like quickly and clearly look at good and bad examples and just you know use my human intuition and stare at them

so I can learn stuff let me understand what the differences are between what I just generated and what I generated before so I can you know build some intuition about maybe what changed to The Prompt led to that and then you know very importantly we don't want to play wacka like we don't want to improve some things and break other things and then you know not really know what we broke or why we broke it and so it's important

to actually systematically improve the quality of a system so you know the way that we suggest thinking about evls whether you use Brain Trust or or not is to think about it as basically you know there's three components that go into an eval the data which honestly when you're starting out and it's what we're going to do in a second in the in the notebook is you should just like hardcode it

there's all this fancy stuff you can do you know you can generate data you can Source it from your logs you know all this other stuff but when you start you might as well just hardcode it and sort of get get going a task function which is you know the thing that takes some input and generates some output you know this is obviously a silly example but it could be a single call to an llm it could be a rag pipeline it can

be you know an army of agents that are talking to each other for a while and then coming up with an answer but at the end of the day it's you know taking some input returning some output and then one or more scoring functions and there's so many different ways you can score things actually in this example that we walk through we're going to like handr write some scoring functions just to really understand how they work but there's a whole world of

using you know llm base scoring functions fancy heris and so on and so again like honestly the if there's one thing I could leave you with it's you know evals are you know equal equal equal to just these three things and it's important because sometimes I I talk to people who are sort of Lost in analysis paralysis about you know where to start how do I you know I

ran an eval doesn't look great what do I do you know it's important to just remember it's just these three things so if you run an eval and it doesn't look good yes it's going to be challenging to sort of figure out what to do next but it is literally just one of these three things either you improve the data you change the task function or you change the scoring functions and there's actually a few different things

you can do in each of these I'm happy to send these slides out as well afterwards but we're going to we're going to actually go through a good chunk of these things in the demo today but you know on the data side there's there's you can handwrite cases you can generate cases and you can get them from your users that's pretty much it there's you know maybe some other things you can do but generally those are the three methods to to follow

we're actually not going to spend too much time on the task function part of it I think there's probably other material in the course that covers how to do various fancy things with prompts we're going to do some basic things in today and then on the scoring side again there's really only three things you can do you can write heris functions which will do you can use an llm to sort of compare and do some reasoning about an output or an

output versus an expected value and then you can use your human eyeballs to sort of look at something and and and get an understanding of whether it's good or bad and then take some action on it so with that I'm going to switch over to walk through a notebook let me just see if there's any questions so far there's no sound yes there is okay I hope the AV is okay so sorry AV

is okay so you've got the three panelists we can talk and we've got 230 people who can't can't talk okay but at some point you'll see there's a bunch there they're dropping questions in like a Q&A thing so we we'll come to those questions in a bit okay perfect so let's run through a real use case of of doing some text tosql stuff I don't know if other people are watching the NBA Playoffs but I'm a

big fan so I found some you know basic NBA data on hugging face and we're going to use that to play with some text tosql stuff today so nowadays it's actually really easy to just grab some data I think I actually saw a blog post this morning and now it's even easier to load hugging face data into duck DB but I think it's a great way to to play with these kinds of use cases duck DB is a very expressive SQL

Tool and it's easy to run inside of a notebook and then obviously hugging face has so many good data sets that are available to play with for free so you know this is the flavor of the data it looks like it's probably one row per game and I I I looked at this before and I think it's 2014 to 2018 so you know here's here's the data we're going to do something super simple to actually do the text

tosql stuff so just a little bit about what's going on here this is the standard open aai client I guess haml sort of alluded to this earlier but you know in the spirit of staying simple we we really believe in sort of writing framework free simple code that just uses the open AI client directly you can use all kinds of amazing tools as you get more and more into a use case but almost always we see people

start simple and so this is just the standard opening eye client we did a little fun stuff here this Bas URL thing it's totally optional but we're we're actually going to use this proxy that's kind of like a free thing that Brain Trust has it lets you cache llm calls so you know when you're doing repetitive stuff like this it's very helpful and then this thing we'll see it in a second but basically it adds a little bit of

special fairy dust into the client so that it traces the you know the llm cult in a way that makes it easy to debug but it's just the standard you know open AI client we're going to grab the schema from the table and then we're going to use GPT 40 for at least most of of today and and that's it you know here's the simple prompt nothing too fancy asking it to write a SQL query

and let's give it a go cool so here's the query that it generated I don't know Dan does that look right we'll we'll find out in I guess in a few minutes but you know maybe let's you know let's run it okay well as a big Warriors fan I definitely believe that that's right so looks like the Golden State Warriors

won the most games in this period of time and again you know that's probably right so as we talked about an eval is it's literally just three things data task function and scores we've pretty much implemented our task function you know this thing can take input and then generate a query and a result so we're pretty much there on the task function we just need to figure out the data in the

scores okay so to create the initial data set I just wrote five questions honestly I'm too lazy to write the SQL queries and the answers and that's okay I would encourage you to be lazy as well in certain cases and so let's just just use these questions and what we're going to do is actually to start we're not going to evaluate whether the model generates the correct answer or not we're just going

to evaluate whether it generates an a valid SQL query and then we're going to do some human review to actually bootstrap a data set from there so we'll just use these questions as I mentioned we basically already wrote the task function so this thing just you know you know takes those two calls and puts them together into one function with a little bit of error handling so that we can you know distinguish between a valid and invalid

query and then you know the scoring function let's just keep it very simple we don't really know what the correct answer is to these SQL queries yet so let's just see you know it's let's say it's a good output if there's no error and it's a bad output if there is an error again to Ham's Point earlier like very little sort of framework stuff in here just this is literally just plain python code let's keep it very

simple and and now we're we're just going to glue this stuff together into an eval call and you know we give it a

project name the experiment name is actually optional but it helps you know keep it helps us keep track of some stuff easily and then we'll give it the questions the task function and the score and we can run that there we go okay so let's take a

look at this okay so welcome to Brain Trust let me go into let me actually just make my thing light mode there we go okay welcome to Brain Trust so you can see you know really quickly it looks like three out of the five past this no error thing we can just quickly look at these to get a sense looks

like there's a binder error and looks like a binder error over here if you recall I mentioned that open AI wrapping stuff it sort of nicely captures the ex llm call that ran so you can see you know this is how we formatted The Columns that looks about right to me if you want to mess around with it you can actually you know do that you can rerun the query you can try out different models and stuff we're

not going to do that right now but just to give you a sense of some of the exploration you can do and then let's actually look at these ones that were correct so let's see who led the league in three-point shots pretty sure Houston is correct that was James Harden's sort of Golden Era okay great so what we're going to do is actually again we were lazy earlier but but the model has very

kindly for us generated a good piece of data and so we can actually save this as something that we use as a reference example and so I'm just going to add it to a data set let me make sure I cleared this out properly from earlier yeah perfect so I'm going to add it to that data set called golden data and we'll use that back in our code in a moment but let's just look at these other ones so

which team won the most games in 2015 that's an empty result so that that's clearly not the right answer and you know my Spidey Sense tells me that this is probably not the right format for the day and it's filtering out rows that it shouldn't so you know I don't know Dan probably we should give the model a preview of what some of the data looks like so that it knows what the right date format is for

example so let's just let's just keep that in our heads for for a second and then this one again big Warriors fan we know that this is correct and so we're going to add this to the data set as well awesome let's just quickly look at the data set so it's kind of the same thing as an experiment you know it has some columns we can play with the data but now what we've done is we've

actually bootstrapped some golden data and this is data that not only do we have a question but we have a reference answer both the SQL query and the output that we believe to be correct and because we we have that we can actually make some stronger assertions when we're running an evl to compare the query to the correct query and the generated answer to the correct

answer so let's go back and what we're going to do is rework a few things to take advantage of the golden data that we have and also try to fix some of the issues that we saw when we were poking around so I'm going to

change how the data Works a little bit now I'm going to read the data set from Brain Trust and I'm going

to return those golden questions and then the remaining questions that are not in the golden data set I'm just going to have the question itself so now you can see for the for the things that are in the data set we have this kind of fancier data structure which has the question it has the expected answer it has some other stuff that is Brain Trust will take advantage of we don't need to bother with it right now but it's really

those those things that matter okay cool and now let's change the prompt a little bit so instead of just looking at the columns we're going to get a row of data and then sort of inject it into this table I really hope I formatted this correctly we can double check that I did in in a second but you know it doesn't matter too much it's okay to be wrong every once in a while and so let's do that looks like okay that

looks like a much better thing you know it's probably taking advantage of the date format that it that it knows about now and then let's improve the scoring stuff a little bit so we're going to actually pull a few fancy scoring functions from Auto evels which is an open source Library that we and our sort of community of customers and users U maintain and we're going to add two more scoring function so this correct

result function is going to get all the values from the query we don't care too much about the column names so we're just going to look at the values and then it's going to use this Json diff method to just sort of recursive compare them there's probably some better stuff we can do but let's just start here and then we're also going to use a an llm based scorer called SQL and this actually asks a model to look at the

reference query and the generated query and try to determine if they're semantically answering the same question or not so we'll just save these is that score is that score Bing how what's the score mean or what does the score look like let's let's look at it in a second it's a good great question so let's just you know plug it together so again we have this eval thingy we haven't CH we redefined this function now we're using this load data

thing to get the data and now we have these three scoring functions so we'll just run this awesome okay so let's start by answering your question so anytime you run a squaring function in Brain Trust we actually capture a bunch of metadata about it we really really

believe that debugging a scoring function is as important as debugging the task function itself so let's let's let's look at like exactly what the model sees so this is the prompt that the model saw it said you're comparing this here's the question here's the expert SQL here's a submitted SQL this is the criteria and then it's actually doing a function call under the hood and the

function call is just picking one of these two values correct or incorrect and so this is a binary sore basically that's asking it to do that and and sort of explain its reasoning which you can also debug right here great so let's sort of analyze this as a whole the first thing you'll see is that luckily we did not break those two that we know to be correct like we didn't break the SQL

queries or the results it looks like we improved on the no error front so there's one example that did not return an error that did return an error before we can actually if we want to zoom in on that we can just look at that example it looks like you know before it returned an error now it didn't if we turn off diff mode we can kind of look at this determine whether this is the correct answer or not I actually

don't think it's the correct answer because it's looking at the you know which team had the biggest difference in consecutive years and it's like hardcoded 20 and9 so I I I don't think this is the right this is the right thing so you know that's one for us to improve one thing you could do actually at this point is you could add it to the data set and if you want you could try to handr write the correct SQL query and

the answer I'm not going to do that but you could and let's just poke around a little bit more so that's still an error which team won the most games this one was still correct again I I like really really like to manually look at data because it allows me to double check all of my assumptions I I don't trust anything and so I spend a lot of time actually doing this when I'm eving stuff which team won the most

games in 2015 okay this is the one that had an incorrect answer before for and now it looks like it's actually correct we can turn on diff mode and you can call that it returned an empty results before and now it returns something so again you know as a Warriors fan I know 100% in my heart that this answer is correct and so I'm going to add this to the golden data set awesome so now we're you know

we're building up a data set I think we're like 15 minutes in in my experience honestly 50 good golden examples that have been lovingly curated by a passionate small team of people is really all you need to have a really good AI application we we're already like a good chunk of the way there just sort of playing around with this demo but you know hopefully it's it's it's kind of clear it's not it's not crazy rocket science to do this stuff I think

it just requires some care and like you know look looking at the data closely Okay cool so you know there's a bunch of stuff we could do from here we could try to play around to The Prompt a little bit more while I was building this I was thinking about maybe like feeding the errors back to the prompt in a new message and asking it to generate another one but just to take it in a different direction

like I said there's like three things you could do you can change the data you can change the task function you could change the scoring function let's keep playing with the data and what we're going to do is actually try to generate more data than than we currently have and we're going to use a slightly different method so the first method we kind of handw wrote some really good questions that we thought pinpointed at the at the task that we're working

on what we're going to do now is more of like a coverage style thing where we actually use a model to generate questions for us and this code again it's not that much code so I'll just walk through it really quickly we're going to use function calling to sort of force the model to generate stuff that we can easily parse and so in this case we're going to have it generate a list of questions and each question is

going to contain SQL and a question I'm specifically asking it to generate the SQL because my intuition is that if I I have the model generate SQL and then generate an explanation about the SQL it's more likely that the explanation is correct then if I have a model generate a question and then a SQL query that answers the question it feels less likely that it would generate the correct SQL query and so I'm kind of

asking it to to do both at once and really focus on the SQL then I'm doing my you know same sort of prompt as up above honestly a fun thing you could do we're not going to do this right now but you could try using a non-open aai model here just to sort of avoid some built-in biases that one vendor's models may have I'm I'm not going to do that here but you you could or you could do both and

yeah I'm just going to ask it to generate some queries give it the schema and that's it so we'll run this here's an example of a query and the question what do you think Dan this looks about right to me having trouble are they asking for 82 or are they asking for the 82 times the number of teams there are oh they're doing a group by team

okay yeah yep yep y Co cool okay and the last thing we're going to do is my guess is that some of the queries that it generated are bogus like they just not not valid SQL queries so we're just going to try them and then the ones that succeed we're going to add them to list so let's do that okay cool looks like a few of them didn't work and then here's an example of one that did and

because we generated it we have this really rich data structure that that we can use so we we have the the query or the question we have the the expected results because we run the query and we have the SQL so that that's awesome now we're going to add this to our load data thing as well we're going to kind of do the same thing where we load the golden data set and then we get all of the questions and

generated data that are not in the golden data set the reason I wrote the code this way is that now as we keep iterating on this each time we add something to the golden data set if it overlaps with the generated data or the original questions we don't need to sort of worry about that we don't really need to change our scoring functions because we already have scoring functions that test the you know correctness of the SQL whether

it's a valid query or not and whether the results are correct so let's just reuse the scoring functions that we had before and run another experiment awesome great so you can see this experiment is automatically compared with the sample one that we did we could compare it to the initial one if

we want you it's very easy to play around with this looks like we didn't regress anything there's not a lot of overlap between the two so this makes sense we didn't change the task function at all and we just added data and so makes sense that we didn't improve or regress anything compared to the previous experiment and it looks like we have a fairly Rich set of values here so you

know there's some examples like this one looks like it was one of the generated queries and looks like although

we have an expected answer here maybe the generated SQL query is a little bit different and so you know we should dig into this and understand which one is correct maybe the generated answer is incorrect maybe the original

data generation process yielded something incorrect but at least we have something to diff and sort of understand that you can also sort of break this down here so if you want to if you want to look per category and try to understand what the differences and scores are that's really easy to do or if you want to break it down this way and actually see like of the three categories how did we do it's easy to

sort of explore the data and just look at the subcomponents that you want and so you know from here again we' we've just made our eval process richer now we have more data we could go and generate even more data just literally run that code again and generate 10 more examples try a different model we could explore some of these cases and try to understand why is essentially the model disagreeing with itself here because we use GPT 40 in both cases

so this is probably a really good example to dig into or we could just try changing the task function maybe provide it two rows of data or maybe provide it some summary statistics about the minimum and maximum year or something to help with some of the questions there's so many different directions that we could take it however just for fun we're going to take it in kind of a simple Direction which is let's just try GPT 4 and see

how that does compared to GPT 40 so I'm just going to change this Global variable which I referenced above and give it a go interesting okay so it doesn't

look like it was a slam dunk you know we actually it looks like regressed across the board with gp4 versus GPT 40 I I would personally be curious I guess we can sort of look at this quickly like does this how does it vary between yeah it looks like we even regressed on the golden data so the golden data we know is correct the generated data were a little bit less certain on but it looks like it even

regressed on on this I i' actually be I haven't looked at this myself yet but be kind of interesting to understand yeah it looks like you know for example gbt 4 messed up the the date syntax once more we should make sure that we gave it the right prompt yeah it looks like we looks like we gave it the right sample I mean this formatting could definitely be improved it doesn't look like my new lines maybe made it

through correctly but yeah this just kind of gives you an idea there's there's so much stuff to do in in digging through data and and understanding it and you know literally starting from nothing just a data set I think we've we've kind of iterated our way into and sorry no data set no no eval data set we just had like an NBA data set we've iterated our way into a pretty rich application here where we have you know a non-trivial prompt

that's generating queries we have a pretty rich data set with that attack kind of different parts of the problem and then we have three scoring functions that help us diagnose systematic things like is the query even

succeeding in the first place from you know some more Nuance things like does it return the correct result or is it semantically the right sequel so you know there's a lot of different places to take it from here and that's I just I just want to just

point out that for students kind of watching the class this is actually very similar to the honeycomb example that we've been talking about in many ways like you know my workflow was actually very similar to this we didn't have any we barely had any data to start with we had to synthetically generate lots of data you've seen that and so I think it's really interesting here like you can see like what what workflow might look like if you use a tool and how you

know it might automate some things and help you organize all of this information so it's actually like this is not this is not necessarily like a toy example like this is actually like very closely mirrors like something that I've done in real life and I think it's great that that enor is like you know kind of doing the SQL use case because it Maps really nicely to the one that you may have been practicing with already awesome yeah and I'm very happy

to share we'll publish the The Notebook right after this so that folks can play around with it and you know tweak it and use it in their own environment we're gonna have a bunch of questions but I have one right now so there is a bunch of functionality built in if you're generating SQL that is really like cool and is sort of taking advantage of the specifics of knowing that you're using SQL what are the other

type I think we saw something for generating Json what are the other types of use cases or generated text where you've got some magic built in yeah I mean the you know the only thing that we use that was built in here that's SQL related is that one scoring function that Compares two SQL queries we have in Auto evals we we sort of optimize for quality over quantity so I think there's about 20 different scoring functions that help

you out with things like rag and you know in within rag assessing the answer the generated answer the you know retrieved context relative to the answer relative to the query and so on there's a really popular framework called ragos we have the actually an implementation of the RAS metrics built into Auto evals with a few bug fixes and you know uses function calling to improve the accuracy and so on we

also have a bunch of tools for measuring the output of tool calls and we see people do increasingly like complex agentic tool calling workflows where you're measuring both the output of individual tool calls as well as kind of like the endtoend flow did you accomplish the task that you initially wanted to set out to do and then honestly we have a bunch of just building blocks that are they sound really simple but having written a good

chunk of the code myself they're like they're just a pain in the butt like comparing two lists of strings in today's world is really hard first of all like normalizing to that comparison to a number between zero and one is not trivial even if you're just comparing the string but doing it in a way that is semantic is even harder and so we have one of the most popular scoring functions is a list of strings comparator and you can plug in the

comparison function so you can do like GPT pawise comparison you can do embedding comparison you could do you know Leen Sheen and so on so just kind some building blocks like that that I think are very painful to hand write but quite applicable cool awesome yes I see some questions in here do do you guys want to MC through the questions or should I sort of

I me happy either way if you're gonna do it yourself you should sort by most upvotes okay why don't you MC them just so because make it more interactive great we got two from Wade Gilliam brain tust looks a lot like L Smith wondering what the feature differences are or pros and cons from experienced people if you were saying like when should someone prefer one or the other what are the I don't feel pressured to answer this question I know certain I

know that there's some culture around hey let's not you know try to attack or try to compare ourselves to other vendors so don't feel free I mean you don't have to answer it directly yeah honestly I think I will hold up I I like I have a lot of respect for the team at Lang chain and and all the amazing stuff that they've built so I I you know I'm not going to say anything bad about about their

product both products are pretty easy to try like you can just sign up on the websites and try them there are a number of differences in the workflow and the UI and a bunch of other things so I would just if I were you I just try them and sort of get a sense of what feels right for what you're trying to do I know H you tried like 25 different tools or 30 different tools if he can try

30 you know you can try two and so you might as well more like a dozen I I'm not that crazy okay okay well sure I felt like 30 but yeah cool next question how would you run open source or open weights models in Brain Trust yeah great question let's let's actually answer this sort of specifically so again I I I know I sound like a you know like a a

dead horse or whatever a broken record but evals are they're just three things data task function and scoring function so let's talk about each of them so data you know we just handw wrote these questions there's no doesn't matter it's no open weights or closed weight model you know it's just these you could do the data generation however you want it's just a script later here as well the task function so we have our generate query thing

you know none of this is open AI specific this is in in all Brain Trust cares about is that you write a function that takes some input and returns some output what happens inside of this function it doesn't matter if you use the open aai client library then you get some nice tracing features like you get inside of Brain Trust you get you know this kind of like really nice formatted thing but it doesn't it

you don't have to do that and most of the open weight models hosted on platforms like together for example they actually they actually have open AI compatible apis and so you could you could do that and and get sort of like all the tracing goodies and stuff too I know someone asked about caching the proxy is actually an open source project that we maintain that's deployed on cloudflare and you can deploy it in AWS or you know your

own

cloud flare wherever you want as well it in each place that you can deploy it it it just uses like you know whatever the easy durable key value store is so Cloud flare has its own key Value Store it endtoend encrypts the payload and response in terms of the API key or rather a hash of the API key and so it's you know very safe from that standpoint and the proxy is also pluggable and so you can point it at

self-hosted models or you could even point it to a model that's running on your laptop I when I'm on Long flights and I'm working on this stuff I run AMA locally and then point the proxy at it and so you can also use kind of that abstraction with an open weight model and many of our customers to we've got a couple questions about sharing the link to the notebook I think that may have already been shared in the

Discord but at some point I assume that you'll are you comfortable sharing a link of course yeah yeah yeah I'll just we actually have this section on our website called The cookbook and there's a bunch of use cases here I can share this in the Discord Channel I haven't published this new one yet but publish it in like 2 minutes after we after we wrap up this call and then you'll be able to sort of scroll through it like this and

then access the The Notebook here cool how about this one so we've got from someone an anonymous attendee wrote I love this workflow are there examples or tasks that you'd say do not lend themselves to data generation of the kind that you showcase today or maybe are there tasks that you sometimes see people ask you about it you're like we're not a great fit for that yet that's a good question let me try

to think of some recent examples you know I I think one thing that comes to mind is if you're doing classical ml I think the shape of your data is quite a bit different so let's say you're doing like a fraud classification problem and you're training like a boosted decision tree and you are you know you have like 1 million examples that you want to test which is totally

reasonable because it takes less than 5 Mill seconds to run on one example I think that the workflow in Brain Trust theoretically will work however as you probably saw from me clicking around we we really believe that it's important to stare at individual examples and build a lot of intuition around them and so a lot of Brain Trust is looking it's it's helping you wayfind to

individual examples that you can look at in more detail someone asked about the difference between us and other products and you know spiritually I would actually say that something that is very informed by my personal experience working on evals for like eight years it's just that's just the workflow that I've really done and refined and I think that's actually pretty unique to Brain Trust and so you know in llm land I think wayf finding is really important in classical

ml I think looking at aggregate statistics with multi-dimensional visualizations and stuff is actually often a more useful way to analyze the data and so that's probably a use case where I would would recommend using a

different tool and I assume two questions I assume that you're looking for entirely Text data and that you don't you are not used or someone submitting images or getting

images back is that no we actually support images natively we also support them in our promp playground and you know we visualize images llm calls show images and everything cool and then do people when they're using Brain Trust is that purely during model development or do they have some Brain Trust call to collect data on a deployed model yeah so we have there's two major use cases for Brain Trust one is

what we would call offline evals which is what we talked about today and the other it's sometimes called online evals or observability or logging and pretty much all of our customers do both and they integrate really nicely together in Brain Trust so I can actually show you just really quickly maybe we can kill a few birds with one stone here but here's an example

where it's actually not just generating an image it's generating HTML and you can render you know the HTML that's interactive right here in Brain Trust there's a Sandbox built into into the product because a lot of our customers are generating UI using AI and this is the logs view so it's actually not different not very different from the eval view it's another thing that makes Brain Trust I think quite special and different is kind of the

very tight integration and identical data structure between logs and evals and you can capture user feedback right in the UI this you can capture it through the API from your users if your users leave comments you can actually capture those too through the API or you can just save stuff in the UI and then you can also do the same thing where you add stuff to a data set from here cool

I got a couple more top one right now is often when we develop text tosql applications for a SQL database there are a bunch of challenges is due to lack of familiarity with the database in our case there's no documentation available and the database is large and complex this complexity makes it difficult to determine the right question to ask because you don't have a clear understanding of the databases structure do you have any suggestions

for operating when you've got a complex database and you don't actually understand the structure of it yeah I mean I think that is a great question because first in practice things are much hairier than what we can talk about in like a 20 minute demo session but honestly I wouldn't over complicate it like in those scenarios I think the most useful thing you can do is you know let's say that you're doing that in the context of like an internal tool that

your teams can use to to ask questions on some data I would instrument that tool from day one to generate logs like this and capture user feedback and then try to categorize the questions either using a model or asking your users to categorize them or categorizing them manually you know it's not the end of the world to do that every once in a while you can tag them or you know do whatever is easy and then what what I

would personally do if you remember over here we sort of built these different subcategories and then we

looked at the scores within the subcategories I think you want to sort of get to a point where where you can successfully classify the different types of questions whether it has to do with like the nature of the question or the tables or subset of the schema targeting but try to build an understanding of you know what kinds of

questions you can answer successfully and which kinds you can't and that's not going to happen overnight you need to sort of iterate towards it but getting to a good taxonomy is very very valuable it's a little bit of an art form so that's where a lot of your creativity can can come in and then and then you then you know exactly what to do like either you say hey there's a set of questions like I have my super complex

data and none of the financial questions or auditing questions work maybe we focus on improving those for a while or we accept that the data set that we are running the text tosql app on is inherently too messy to do fin auditing questions so let's do some schema work maybe let's create some views or something that make the life of the model a lot easier and see if we can improve that category yep it's also nice this

seems like a place where you see the connection between what you were showing for observability on a deployed model and maybe you even have like thumbs up or thumbs down coming back and looping back to the experimental yeah exactly flow exactly yeah there's some stuff to you know for example I'll show you really quickly like you we also make it really easy for you if you hit the r key from anywhere in the product you can

actually enter this sort of human review mode where you can just really quickly use keyboard shortcuts and actually like rate things and this is really good especially in a use case like that if you have maybe an analyst audience who could look through a bunch of questions and answers and just rapidly rate them I think it's it's really powerful to do that and you don't have to like prepopulate a queue or do anything like that you literally just hit the r key

from anywhere in the product log view experiment Etc and you can sort of enter this workflow cool all right thanks so much thanks for having me all right see you