Systematically improving RAG applications

68 MIN · YOUTUBE · HTTPS://WWW.YOUTUBE.COM/WATCH?V=RRDBV6ODPKO
https://www.youtube.com/watch?v=RrDBV6odPKo

SUMMARY

The discussion focuses on systematically improving retrieval-augmented generation (RAG) applications through a structured approach that emphasizes data analysis, user feedback, and iterative improvements. The speaker, an AI consultant, outlines a playbook for enhancing RAG systems, which includes understanding user queries, implementing feedback mechanisms, and using unsupervised learning to identify areas for improvement.

- *Importance of defining clear objectives for RAG applications to understand user needs and satisfaction.*
- *Feedback mechanisms should be simple yet effective, such as thumbs up/down systems, to capture user satisfaction accurately.*
- *Utilizing cosine similarity and ranking scores helps measure the relevance of responses and identify areas needing improvement.*
- *Topic modeling can cluster user queries, allowing teams to prioritize improvements based on user satisfaction and relevance scores.*
- *The distinction between content topics (related to the information being searched) and capability topics (related to the system's ability to answer queries) is crucial for targeted enhancements.*
- *Regular monitoring and analysis of user queries help adapt the system to changing user needs and improve overall performance.*
- *The speaker advocates for the use of synthetic data generation to test system improvements before deploying changes based on user feedback.*
- *Fine-tuning embedding models based on specific user data can significantly enhance the performance of RAG applications.*

today's little topic is systematically improving rag applications and in particular the interesting thing here is this is really a system right so this is a system that you should be able to apply regardless of the applications we're trying to build because the idea is that we're going to you know use this to do a data analysis come up with hypotheses test them and eventually sort of get an understanding of where we need to improve the systems that we're building

before we get started I just want to roughly introduce myself I imagine most people here might be a little bit familiar with me from Twitter but my actual background is that I am currently doing a lot of AI Consulting for companies that do work in the llm space and the rag space and so you know you might notice some of these companies here for example trunk tools narrow like Limitless we they're all rag tools and on top of that we also have a

bunch of AI Hardware companies that again really care about improving things like memory but the issue is

once you deploy this application when things are failing when you are losing users because there's maybe like you know not enough confidence in the responses maybe we are unable to answer certain questions the idea that we have to go and figure out what kind of questions are actually causing the issues and we can't really blank it and say I want to

improve rag instead we need to be able to say things like hey questions that are retrieving documents by file name are 40% of the questions that are being asked but you know there's a 30% user satisfaction and now we can actually go in and identify exactly what segments of the population are asking questions that we cannot serve well and then come up with ideas in the future to make some of these improvements so I'll talk about

that process in a bit but if you you know maybe you had to leave early or anything like that all the stuff that I talk about today I've actually written down before and so if if you're ever interested or have some time to spare you can always take a look at some of my existing writing on the topic and cool this is this is basically the TA contents for today's conversation the idea is I want to cover

the rag Playbook that I run for many companies whether it's small or big the The Playbook really is the same you know I'll cover you know how we should think about the feedback mechanisms understanding that you know feedback isn't something that every single query will have and so we need to also measure things like the cosign and the ranker scores how we can cluster these things and then use that information to really figure out what is the issue and

then near at the end I'll talk about what kind of low hang through that we have in in terms of systems that every rag app should may as well include in order to get the most performance out with it but the first part really is the fact that you know we need to have a feedbacking mechanism we need to actually have an objective that we care about when a company comes to me and they say hey Jason we want to improve

our rag application we actually don't know what that means are you losing users because we don't have faith in the system are we unable to give good citations is it the fact that the things that we're searching for aren't available in the vector databases or is the fact that you know we we're we're using things like file names but because we don't use full text sech and only use vectors we can't actually can't actually map against that so the first thing we

have to ask is what is the actual objective and the easiest way to capture that feedback is you know having a thumbs up and thumbs down for but even there the copy is actually very important I was working with a client recently where the original copy was how did we do thumbs up and thumbs down and what this actually resulted in was the fact that we did not actually know what we were trying to capture it turns out

if the answer was correct but the answer was too long people would thumbs down if the answer was correct and it was concise but it took three or four attempts to get the right answer people would thumbs down but that information isn't actually helpful in understanding what kind of business outcome we're trying to drive what they really cared about is whether the answer was correct and just by changing that copy just by saying did we

answer your question today

versus how did we do we were actually able to 5x the volume of ratings and improve the ratings for the data that we had and what this meant is you know a couple weeks from now we'll be able to look at the things that people thumbs up and thumbs down and have a heris on understanding what exactly are people happy with and what are we doing poorly on but you know even if only like 5% of the population actually

uses these mechanisms we still want relevancy scores across all your search quers and so not only do you want to have a feedback mechanism that measures something like satisfaction you also want a relevancy measurement that is easy to measure right like you could use things like having an llm but in practice it's much more practical and much more cost effective to use something like the cosine distance of your edting score and if you're using a ranker the r ranking scores that might

come out of something like coh here and so now your logging data looks something like this there's a query there's a cosign score and there's a feedback score and that feedback score might be null for 95% of cases but by changing the copy we can improve that quality and improve that the signal that we have is something we're trying to do better on so now you have all this data you have basically a piece of text a couple

of numbers and your goal is to figure out how to prioritize my team to systematically improve the rack application and the most practical thing you can do initially is to do some form of unsupervised learning if you use something like B topic or LDA we can actually use the query strings and generate topics of questions and topics of clusters that we can use to really focus on the segments that we care about and so if you if you

run this topic modeling and we'll we'll cover that later if people interested you can basically have a topic the number of State the number of questions within a topic and now you have both a mean cosine score and a mean feedback user satisfactions score and just by looking at this we're going to give ourselves a lot of ammunition in figuring out how exactly we're failing the user and whether or not it's actually relevant to even improve for

example maybe we find a topic that is very low count very low relevancy and very low satisfaction you know we can at least recognize that that topic is something we might not want to prioritize in the near- term in terms of improving and we might even realize that these kinds of questions are so irrelevant we want to put it into the prompt of our language model to say hey if you're going to ask me questions about this topic this is not something

I'm going to try to answer today whereas you might be in another domain where you find topics with very high counts very high cosign similarity but very low feedback right maybe we can say this is the top cluster that we've discovered our search engine says that we're finding relevant data but the users are extremely unhappy with how we're answering the question now we know we need to be thinking about what kind of answer we're generating whether that's

the question prompt is there something else that's going on that we're not understanding for the users and we can go figure that out other times we might get topics with high counts with low relevancy and high satisfaction and again by looking at this sort of grid of options right higher low account high or low relevancy high or low feedback we can pretty much identify the regions that we can dive into and sort of brainstorm on how we can improve these systems and

when you build these topics there's going to be really two kinds of topics there's going to be the topic where the content that you're searching is the interesting Fe so I'm going to just call that a Content topic there's another kind of topic which is a capability topic so imagine I'm doing a marketing marketing rag app a topic cluster could be questions regarding pricing versus questions regarding like different kinds of case studies right if those kinds of

questions have low relevancy and low feedback usually the solution is to improve the inventory that you have right it might be the case that our customers are coming in and asking about pricing and we're not doing well because we don't have enough pricing documents that would be a topic the other topic we talked about was a capabilities cluster here instead of asking a question like you know what was the pricing that we have instead it could have been who is

the last person who updated the pricing document there the capabilities topic really is around like modified dates and if that information isn't available to the language model if it isn't available in a text Chunk you'll never be able to answer that question but if that does show up we can look at the counts we can look at the coign distance and we can look at the feedback to identify whether or not it's worth fixing today or

tomorrow based on how you prioritize things right other simple examples of a capability might be comparing and contrasting across multiple documents if that's the case you might want to have parallel search if you you know encounter things where we ask about recency or latency a lot s recency or like latest we might want to have date ranges and if we have date ranges now we need now we can recognize that we need to have the current date available in

the prompt for the language model to prepare a search career like that a really funny example that we noticed recently was in different Industries fin like fy2 24 and fy2 don't end on January and so you might discover that Financial year is being referenced a lot and because of that we're matching across the wrong year token these are all capabilities of a model that we need to improve by changing the search system compared to the topic clusters that we

talked about before where the way that we can fix that is to identify like inventory issues and making sure the documents that we have are actually available and then if we have issues there we might build out rules that say you know if there are less than 10 documents that match this question it might be useful to report back to the user or report back to the you know content management team and say hey lots of people are asking about pricing and

this is not available in our Search tool right so now there's different ways of solving the kind of rag Problem by basically looking at this simple topic count score and satisfaction so once you develop these topics then you have to do a little bit of like you know reading the te leads once you identify these topics you're going to have an insight as to what kind of topic clusters and capability clusters actually make sense for your

use case and today we ran this project process offline and so when something changes in the future we might not be able to detect that unless we rerun this process so once you identify the topics and the capabilities one thing that is really useful to do is to actually build classifiers that can classify these questions in real time if we have these question type labels whether it's a type called you know is privacy question or it's you

know a capability like compare and contrast or date range filtering or ownership we can do this classification task sort of asynchronously as data is being processed and then send that to something like amplitude or data dog where we can monitor over time how these things change and what you might notice is today we have some seasonality where you know maybe like sum type questions are being asked like Monday morning at the beginning of a work week but more

importantly when you get new users or new kinds of inventory it's really valuable to understand how the distribution of the these questions and how you prioritize improving these questions change over time a simple example could be you onboard a new Enterprise client and you discover that after the onboarding process the amount of questions that we were ignoring skyrocketed because it turns out this company really cares about comparing contrast questions whereas your previous user base did not and being able to have

this real-time monitoring of saying okay 20% of our questions are document search and all of a sudden it went to 100% maybe we just got like a really big company that you know is using the application differently than it was before and there would be no way to detect that unless you had this real time disability that told you that something was going on otherwise you might just run this clustering job you know a month from now two months from

now and had realized that you know this new customer was very unsatisfied because we deprioritize the type of questions that they really really care about and and generally what happens is when you actually build up these clusters there's usually be like three or four things that stand out and instead of having a goal that is like improving the rag application we can set very simple goals like improving daytime filtering right or improving our met the amount of metadata we include in text

clusters and now you go from this very ambiguous improving this AI system problem to basically figuring out how much like time does your team have and be able to run these very small experiments on top of that because you have these specific topics when you want to evaluate new data sets you can actually use things like synthetic data generation to generate more questions of the topics that you care about and then be able to really quickly test whether

or not your experimental changes has improved some system right and so this is something we're going to talk

about right now around some of the lower hang fruit the general idea really is the fact that once you have this topic information you know what kind of questions people are asking and now you can prompt a language model to generate more question questions like this in order to maybe help you generate better baselines in order to evaluate your

search tools a simple way you can do this is by saying you know what I'm going to randomly sample a text junk I'm going to use AI to generate a question using that text Chunk and my evaluation metric is simply if I search this question in my rag app that text junk must show up in the top three top 10 top 25 results of the search system and I know that whether that whether or not that shows up is really relevant because

I know they come from topics that I care about versus the topics that I don't right and if you you know include better metadata or add better daytime filters you'll immediately be able to recognize whether or not this will improve your Baseline so instead of trying to guess whether something's going to improve your system instead of trying to guess if Day times or if new embedding models or new text chunking models are going to improve you can now just test these by

having synthetic data even before you have a lot of you know user feedback data on something like user satisfaction and that's basically it right like you want to make sure you you do things like having a full Tex search involved right usually when you build these ranker when you use these ranker models we've always found improvements in how we return our our baselines right because people ask questions like who edited the last document what was the title of that

document when was this file last updated show me the latest version of the pricing doc we showed to 500 companies you know that query isn't going to be just an embedding that's going to be a date filter with a category filter with a you know document type filter and by looking at these topics figuring out that these are the things that are important and then building specific Nuance search tools you'll be able to actually identify and and and report back to you

know your executive your manager even yourself and we can say hey for this topic where we have 40% of the questions be in this topic we've been able to meaningfully impact and improve coign distance or user side of St right and this will give you sort of the to me at least the data and the evidence you need to invest more time in improving a system and knowing that improvements in this system is actually to go leave a

better outcome right you can imagine we have things like the count the mean cosine distance that mean feedback we might have you know the average character length or the average average latency and we can now go try and correlate these variables to other business outcomes you know is my rag app actually being able to convert users better are those who have high satisfaction or laded satisfaction using my rag my chatbot once a week twice a week three times a week right

those are the things that you can do once you actually have these metrics and once you have these clusters to identify what exactly is going on and obviously on top of metadata you want to do things like adding type filters

and document filters and date filters I see we have eight minutes left so I'm going to just go jump to the next slide as well the goal of this really isn't to like necessarily show you exactly how to do these things right

that the goal really is to show you that one of the simple ways of going about this is to do this kind of data analysis to identify and prioritize where you want to include your systems and the main goal for me really is to also hear some questions that you guys have on how you guys are running into these kind of issues and the goal is that after this course is done and I will be spinning up a course on really breaking

down this process and understanding how we can improve these applications and so if any of you guys are interested as well I'll be available on Discord to answer any kind of questions if you guys are interested in learning more about the course or you know telling us what you guys are interested about the course this is a c code you can go check out and there's a TI form servation take like two minutes and it'll give it'll

give us a good understanding of what are the concerns and questions that you know your teams have around improving the systems and what kind of outcomes you're trying to drive then do you have anything else yeah we'll drop we'll drop a link to the survey also just a direct link in Discord and then we've got some Q&A so we should use this time and go through some of the the questions yeah do you want to pick the

questions yeah I'll take I'll I've ranked them from most votes to least top question from Simon Willison how many real user queries do you need to have before these techniques start to be worthwhile presumably if you have only a few hundred logged queries you're better off eyeballing the data directly I would say so I think it's still useful toine these clusters because you know like even if you just have 100 questions and the difference was between like regular questions

versus questions that involve like time like an embedding model should be able to at least discern like oh latest recent those are things that show up it's really just useful to have some kind of mechanism you can Group by and then focus on those things independently because because really this topic clustering is just summarization so if you copy pasted all the 100 questions and put them in the Opus you're kind of doing the same thing but once you have

these discrete variables you can then Group by them and say hey are the are the coign distances the same are the thumbs up and thumbs down the same oh it turns out you know every question that is in topic one was asked by this customer of ours let's go let's go reach out and ask and say hey you guys had really low satisfaction what were you looking for what kind of outcomes were you looking for right let me pull up the next

question how do okay I'm gonna actually rephrase it as I talk but someone's asking how to have a standardized data ingestion pipeline when you've got complex data like PDFs PowerPoint flowcharts visual charts and so on which is yeah I think one of the challenges I've seen most frequently is you know especially charts you any quick reactions to

this Jason yeah so if we have an understanding of what kind of questions people are asking what I would I usually prefer to do is I would prefer to commit much more compute upfront you try to pre-answer these questions right like if I have a bunch of images in a PDF is the goal to really pull questions out like do I just want to be finding images or is it usually the case that the paragraphs above

images are also referenc these images themselves and I want to do some kind of question answering system right like for example one of them is like pictures of clothing right so if I have bounding boxes that I run on onto these images maybe what what I'm actually doing is I just want to find like find all images where it contains at least one shirt so understanding what questions people are asking is is usually the first step the

thing that is more General would be around generating summary indices right where like for all images I can generate a text summary what these images look like for charts I can generate text summaries over those charts and again the goal here really is to create synthetic text junks that I can use to look up alongside this like other multimodal data and if I know what kind of questions people are asking I can always say given a chart G given

this chart and questions people ask about charts you know what are some questions that would free this chart and again building a baseline of what exactly is going on I would just add to that so I've worked on PDF that had many tables and there are python libraries that will help you extract those tables into CSV they're really finicky and so if

your tables have for instance common problem that we saw is nested headers so you could have like in the header you'd be like revenue and then subheader would be individual columns would be Revenue broken down by country we never got that to work very well but if your tables are pretty standard and simple we're able to extract those to CSV and then pass the CSV to the llm and we've had not fantastic

results but reasonable results from dealing with tables like that visual charts I imagine that you could use models that are that have a vision component again something I haven't worked on a lot myself but yeah like one specific thing I like to call out in this tables example is you know understanding what kind of questions are being asked about these tables is a is a huge lift right and sometimes when I

have a table I might be trying to do aggregate statistics of that table in which case I might want to then be able to do something like Tak given a CSV file write some kind of textas SQL engine to answer questions or Aggregates that's not always the kind of you know data data analysis we want right other times we might be processing specifications and like large tables we're trying to find like you know certain rows right if we knew that ahead

of time that we knew that we want to do summary statistics versus a scan of the table we can build separate systems that can do that right and that really comes from again looking at the questions and making sure that this is what you're asking for right I think before we were processing like tables with like 20,000 rows and my solution was oh let's put this into a SQL light but it turns out they just wanted to figure out like when

did we order like these kind of hex screws for this kind of you know construction site and we just needed to chunk them like 20 lines each and we'll be able to find the answer because we only looked for figuring out like which row something existed in y here's a question I I really like this one from s any suggestions on capturing feedback when the rag element is hidden away from the user for instance I'm using agents to build a

report my rag tool and my rag is a tool the agent uses to build elements of the report since the user doesn't direct since the user does not directly use or query the rag how do you suggest a capture feedback yeah like in the report context you you kind of still have like fields and inputs and so one piece of direct feedback is you know do we you do we ask the user to ever correct the inputs if

there was a correction that's a piece of feedback right now we know that like oh the fields of this type require a lot of user Corrections What fields are those turns out when the fields are enums of a high cardinality we tend to fail like that could be like a discovery so on a field level there's there's edits that we can do without the feedback we still have this idea of the ranker and the cosine distance right even if we don't

touch anything for every internal rag API call that we make we are getting end taex Trunks and for those n Tech chunks we are getting average distances and the max ranker relevancy the the mean re rank of relevancy right and you know whether it's like 7 or8 that's not really going to tell us much but if certain fields are getting us like. three and other fields are getting us 08 it still guides us to where to look to

come up with a hypothesis to then come up with an experiment if you have a report that you're generating with Rag and you're not able to edit it it seems like really bad ux a really bad product design I think yeah yeah and it really depends like if your report is generated using like a markdown output it's really hard to like assign back what is going on whereas if you have a report that's like a Json

object where there's keys and values we can figure out okay like this key is having lots of edits and we can fix that so there's a little bit of like structured reasoning and how we even design the report experience Dan it's say this also seems to me to be the case and it's probably the one that I have the most experience with this seems to be a case that might be the easiest case if you're extracting

data for a report there might just be a ground truth or the the case I've dealt with there a ground truth now we can literally just write a a set of assertions and say instead of relying on the feedback being excuse me thumbs up or thumbs down we need to extract a certain fact we're going to write a test and the benefit of this compared to one that uses user feedback is user feedback you actually have to deploy a system to

figure out whether it's an improvement or not but if we want to experiment just changing the model or changing something about how we do chunking improve this system if I can write a bunch of tests because I know I'm just pulling out facts to feed into a report now I can iterate very quickly because I can test that in seconds rather than days which you typically need for a deployed system the next most uploaded one I

think is a comment SL question someone says cosine usually isn't an exact measure of relevancy it's relative and to some degree arbitrary why would you recommend including it as a measure anyways yeah the quick answer is it is at least a number to look at or like right what happens is if I wanted to generate like rag to do emails for a sales project the best metric would be like sales conversion if I could capture

that data I would do it and I would try very hard but usually what's going to happen is I'm going to have maybe hundreds of emails before I get a sale and so I don't have enough signal in low volume to really make anything actionable if I did a thumbs up button maybe with a thousand emails I get 50 interactions and so I might need to have like tens of thousands of emails to even be able to figure out satisfaction

whereas with cosine distance it is at least a metric that I can have access to for every single request and the idea is like as I increase my volume I I should be able to rely less and less on that right and so you in this case it was some of up th stand is the best we have cosine is available for everything but really you know we're just trying to have some direction to go and we got to

recognize that like yeah like not all metrics are useful but they could be useful in in sort of a a moment in time Alo the the way we're you the fact that it's relative is fine because we're actually using trying to figure like we're using it in a relative way all right how about this one I'm going to reorder by current upvotes can you speak to the use of graph embeddings in Rag and how to evaluate

that I mean I don't really work too much with knowledge graphs or graphs in general usually if there was a graph like I'm going to be like talk a little bit more technical then maybe I don't I don't know how howal audience is but if there was a graph and I was factoring The adjacency Matrix all I'm really saying is you know I want to train I want to fine-tune my inventing model Beyond something that's like a sence

Transformer or a or a like open Ai call in order to have some outcome right and the question is you know the question I would really be asking is if I switched out open AI edings with graph eddings in the search tool how much would my relevancy metrics show up and the answer is you know depending on the question type it doesn't really matter if the the question was last modified date you know maybe maybe the graph won't be able to

do that whereas the graph embeddings would be able to improve you know the the content topic type clusters but not the capability clusters right and so these kind of questions are really hard to evaluate unless we have these sort of segments and scores that we care about in order to help us prioritize got a question from Peter one that quite a few people hypothesize about how much of what you're talking about today which is a rag will still

be relevant when a hypothetical GPT 5 is released yep I think something like this will be even easier when a gb5 is released because part of it is just understanding how to write queries and recognizing that the embedding isn't the only thing that matters like if we embed a Tex chunk and the question was like can you compare and contrast the pros and cons of these three services right gbd 5 will just be able to write you

know maybe six queries the pros of this service the cons of this service the pros of that service and then be able to synthesize that information and so I don't think we ever get into a place where this is going to be something that fails especially if you recognize that even as long context grows larger it does not mean that search becomes less important right because if you're if you're actually a business it's not just accuracy or user satisfaction you

also have to trade off things like latency for business outcomes right like if gb5 is really good and you can put 10 million tokens in it and I'm building an application that's going to sell products if the response time is 5 Seconds I someone might click the back button I might not not actually buy the product and so there's always going to be trade-offs we have to make that said understanding that the what your customer is actually asking that is a

skill that you develop as like the data scientist and as the business business and those things will be you know de forever if anything I would just love to be able to copy paste all the questions in the tb5 and ask it for the Clusters and ask it for the prioritizations but you know building that thumbs up thumbs down button like that's something that is sort of separate from the analysis that the5 could do all right we've got another one

which I think is again about knowledge graphs I think we've covered that can you talk a bit more about date and dock type filters do you embed something about that in the LM so yeah I think there's someone basically just asking can you talk a little bit about metad metadata filtering even just like what is the idea of it and how does it work yeah if if you see my screen do you see the slid

or do you see a blog post see a blog post okay perfect one simple way of doing this is actually using something like instructor to generate structured outputs right so in this example I have an output model that is a like metaphor query that has a Rewritten query a a published date range and an allowed list of domains and my prompt is effectively says given a request that comes in I want you to structure it in a way that I

can use in a subsequent search system and so now you know for a question that is like whatat are some recent developments in AI I actually get a qu I get a Rewritten query that looks like this it says okay the query should be like novel developments advancements AI AR blah I create date ranges because this is trying to capture what recency looks like and in this specific instance it also recognizes that there are good domains that I want to search against so

in this case the category or type is going to be I only want to return documents created from this time range from Arch right and this is something that you kind of have to customize because you are the one that has to build out the start and end date range right but because you understand that this is something that's important it is something that you can go out and build and deterministically know that this will improve how your search application

runs and so if you have different document types you know maybe this is something that is you know the content in your CRM that is tagged by your marketer and so you know there is a pricing tag and if you you knew that ahead of time you can just make sure you only search documents across certain prices s about pricing great actually let me ask a followup in when you do this Jason so we're getting a query you're

using instructor to map it to to the right metadata to query on do you also when you are putting documents into the vector database use instructor to create metadata a lot of metadata it just is what it is you know when the document was created yeah but historically have you used instructor to create metadata that you add on for each document in the vector database yep so one of the things I do

this is again based on the use case so one really funny example was that for different Industries the financial year does not end in the the fiscal year does not end in the calendar year and so we just had some instructions that said okay if you're in mining fy2 24 is actually in FY 23 and we just had a small prompt that just says for any Financial document convert it to the actual date versus a like the effective fiscal year that's a

really small Pro a really small extraction a more interesting example actually was around pulling complex diagrams and complex tables right in this situation we knew that this table could be like just shoved into the prompt and and generative response but what we did was we said given this table generate for me six or seven questions that would retrieve this table and then when we actually try to we wrote this table in the database we embedded the

questions where the table was metadata rather than bending the table itself right but these are very specific decisions we made because we knew that what people were looking for was we need to just pre-compute an FAQ right and like that was a decision we made because we realized that what we really needed was FAQs across all documents that included tables we got a question here what's your favorite platform for building rag systems Lang chain llama index something

else I think for the most part I've kept things very simple right and so really most of the applications we've built is going to use something like Lance DB and the reason we choose Lance DB is primarily because full teex search SQL and Vector search can be stored in a single database whereas historically if you use like early versions of pine cone and open search you would have to have an open search index a pine cone index

and a postgress table in order to be able to do like a wear clause and a having Clause with a full Tex search with a vector search you have to like get the indices intersect them and then do something else crazy so things like L CB make that very easy in terms of the quer re building you know like basically what I end up doing is I have a lan CB instance I have a processor that converts the user's

question into a pantic object and then I just have that pantic object have a method called search and that makes it very explicit call because I have all this knowledge about what my data looks like I know that this is a CRM I know that it's going to have tags I know that this is going to have a company tag and I want to embed all that knowledge into the query engine I think something like Lin Al Lama index is very easy when

when we just have sort of like very deterministic general data but as data becomes much more specific it's it's just simpler to have the control you need to like turn those knobs right how many more questions you want to handle I got time cool let's keep going then how do you develop an intuition for what questions you have the start of project how does this change if you're building a rag from scratch do you wait

for feedback to come in before you add metrics yes and no so one of the things I do for every project that is at this point in my opinion I must have is once you've chunked the data you should have the first thing you should do is like use something like a language model to generate two or three questions per text Chunk and you might say well how do you know that llm is going to ask

questions that my user is going to ask like we don't but it's going to be something and often times you'll still be surprised at how hard the search problem is for example I had been using Paul gr essay for a long time to demo and so I ran this synthetic data generation process and we found that our search relevancy was like 97% right and then we did bm25 and it was also 97% it turns out for Paul

grandma says you don't need semantic search you just need text search and then I thought okay this is probably too easy let's just move let's just move this to gith issues we do the same thing for any GitHub issue generate a fake query for me and then this time we got 60% full tax got 61 semantic search got 62 semantic search plus full Tech search plus ranker like 65 like oh wow like we actually can't even s even if we cheated

we can't do well on this and now you go and begin the process of understanding what the heck is going on right it turns out for GitHub a question might be how can I get started right and it turns out if you don't condition on repository you'll never do well okay this means I need to build UI to you know condition on these kind of variables because if I don't have these filters I'll never be able to do well and kind of that's just

what you learn as you're spelunking across the data sets and spunking across the kinds of questions right and the idea is like today you have a thousand sythetic questions and no users tomorrow you have a thousand questions and a thousand users and eventually you can wean yourself off of this off of this synthetic data but that's kind of a transition as you deploy to a production system what's the term that you I've been kind of making up a term when I

try to explain it to my clients like so there's you know there's like query expansion I call it Target expansion I just kind of make it up like to you know make each document like into questions and basically make it more likely like the recall happens yeah what what is the right word for that I don't I just feel like I made it up I mean yeah yeah I basically tried to so I think the research is you are building a synthetic

text junk right but I I just see it as like I am eating additional compute cost on insert in order to improve my retrieval right it's like if I'm writing a bunch of like if I have a primary key and it's an integer to a database you're just upsert like inserting a row but behind the seeds it's like building some kind of tree right and like that insert is expensive because I specified like primary key big unique or something

right and you know the insert is slower but I get better read right and that's how I think about it I'm just like doing this pre have to make up new terminology I was just curious if there was yeah I think I think the general consensus is just like synthetic text junks the opposite would be hide yeah coming back to this question it says how do you develop an intership for what questions will be asked at the start of

a project I think that if you don't have at least a theory for that I'd ask like why are you doing this project like most projects they have some goal and you say like oh we want to help users find answer their questions about product reviews and then like it seems to me that I can imagine a hobby project where you're just like I just want to build a rag system but if you've got a business

problem you're trying to solve you should have a intuition about like what are the questions people will come in with oh we do the bm25 one yep I'll do that one when you do something like using synthetic data to generate your question answer pairs to evaluate you'll usually find that bm25 and Vector seit are not that far

off in practice like it might be a 67 68 you know 92 95 but there are specific examples where bm25 is much much more Superior the first one is if the author of the document is also the Searcher it we're much more likely to have exact word matches on how people search for things because I wrote the I wrote the thing I just I just like I had 15 hours of interviews I just want to search and

grass my transcripts I I know how I talk basically so that's one thing the second one is very specific which is often times if you are building like an Enterprise rag application like I would not be surprised that 30% of the users use it as document search like they don't really care about the answer they just got to figure out which document it was and it turns out if again like if you are the author of

these documents you know the file name and bm25 will match the file name and outperform semantic search by you know multiples right because if I know the file name is just like company name 24 report I I do want to just be able to say like company name like 24 report I'd be able to find that document and just matching on file name alone will get me way better results than and then VOR search and these are just small examples

of why it's effectively you know I think a default to be able to include bm25 and in conjunction with vectors we got a question with four up Bo but I think we had to get it at just the right moment in time someone says wait can we Deep dive on that idea what's a good ux for for report editing and I think this is related to HL made a comment about if you're generating reports

they should have editing I don't know that if that's so topical to where we are at this point do have anything else you want to say about that haml the only thing I mean I guess like you have to make your make sure your product is good if you're going to it's not just AI like narly focus on AI so like you have to think about it holistically I mean yeah you can focus on AI all

you want but if the the yeah like you have to think about the product too it's like kind of like being a data scientist too this this is the case in ml as well a lot of times you know as you become more senior you know it often about the product as well yeah people sorry all right so my my Wi-Fi got knocked

off where are we you're back now I got it go ahead yeah so the question was like how do we improve the ux report editing I think the dangerous thing that I would love to avoid is having the report be generated as a

markdown file because if he generat a markdown file you might say okay well if I have headers and action and bullets it's structured up and this is going to be easy to edit but what's

going to H be hard is it's hard to attribute edits with certain regions of a document and so if I can be much more opinionated and do structured outputs right where I have keys and values there's inputs and input field then I can much more easily edit specific parts of a problem right like instead of having action items and summary as markdown if I said acms a list of strings summary is a you know text field

then I can like delete an action item right with a delete action I can instead of going from three I can just add a new one and create a new one those actions are a little bit more straightforward to measure right just if you create three action item if you add another one that means you have bad recall if you delete one it means you had bad Precision right You can again sort of come up with as structure data either

that's just like implementation behind the scenes yeah so yeah okay how do you we actually okay how do you do rag where we want the LM to pinpoint Which documents they got the answer from after getting the topk documents for example after it gets 10 documents from Rag and then the answer for the top 10 document basically how do you do citations yeah so the

easiest way to is to site entire text trunks right so what this H what happens is if you have 10 text trunks you can format The Prompt in a way where you can say here's 10 text chunks each one is an ID and as you generate your answer include like a citation right and maybe the answer is just square bracket chunk text Chunk ID and usually that works right if these text JS are very small then you can still build some UI

that says if I Mouse over like the square brackets 2 I want you to render text Chunk number two that's like a very straightforward way of doing it a much more complex way of doing it could be T like site sighting by spans and whatnot but generally the idea is that because most of these language models are trained to Output markdown if you make the citation look like a markdown URL it works pretty well and so you can say

not only it does every text junk has a as a user ID but you make it look like a URL actually it actually performs much better in practice what do you think about high dimension embedding models as rankers vers cross encoder models yeah I almost always want to use both right again it's mostly like a latency trade-off V databases are are pretty fast in terms of search relative to like you know cross encoders and so

the again the idea is like I might have a million documents Vector search gives me a 100 documents and the ranker get me you know 20 documents and really I'm just making trade-offs between Precision recall and the latency of these systems right it's it's going to be very hard for a vector database to realize whether or not I love coffee and I hate Coffee in Bed to be similar or different but it is something that your cross encoder is

going to be much more comfortable handling next question is the same about just adding citations we talked about using markdown for that can you comment on hierarchical retrievers from llama index fine tuning

embeddings to improve rag how effective this is actually two questions can you comment on hierarchal retrievers and then also

separately can you comment on fine tuning your embedding model to get better measures of relevance for rag yeah so with the haral retriever stuff yeah like funny if I was like talking with Jerry on that kind of stuff I think generally just like as language models get better we we'll definitely need to reason about this like hierarchy less and less because like these hle changes are probably going to be like small percentage Point improvements right in which case like a longer

contact just means we don't have to be as clever to solve these kind of problems like the reason hierarchal matters might be if we had four text jks in a row one two and three were retrieved the idea is like maybe four should like three should have been retrieved too so there's no like missing context I think that can be solved by just having bigger chunks and like better aeval but this is like an opinion I

hold strongly actually mostly because I've been doing like fine-tuning and betting models for search since like 2015 I think any company that is making money is leaving money on the table by not fine tuning inting models if you think about large e-commerce websites you know Netflix trains an inventing model because they know what you watch and what you don't watch and you know we can like have that information be used to build the recommendation system that's better

Amazon has their own customer eding models because they have checkout data and so they know that a user embedding and a product embeding is is very very good and it's just unclear whether or not I believe that open AI has data with question answer pairs for your domain in their training data right so it's like chances are open AI does not have that data available and you have that data available on top of that because of how

smart these inventing models are in general if you go and search like modal inventing fine tuning I actually worked on a project with model where we found that even with 2,000 question answer pairs we will outperform open Ai and coh here right and so in these relevancy tests we got maybe like 84% from AI by default but if you fine-tune a 200 megabyte Bert model you're going to get 86% with 2,000 examples and you're going to get like 89% with 20,000

examples and if you're a company with like real revenue and real users you're going to get 20,000 questions in in weeks right and then you can do much more interesting things and recognize that hey you know before I had to like synthetically generate text junks or synthetically generate text junks from the questions because I know that the question and the text Chunk don't look the same and I'm hoping the eding model can capture this if youd find tun your

eding model it does exactly what you wanted to do coming back to a topic from earlier having a really tough time extracting data from tables and PDFs in rag pipelines any Tri tips to improve that yeah a blanket answer I feel pretty comfortable saying now is actually probably looking at things like llama pars they've put in a lot of good work in blending language models and

existing text extract tools to basically generate tables that are better a simple answer in terms of prompting actually is to just switch from csvs to markdown tables I found that in practice giving language models asking language mods to Output markdown tables is actually higher performance than generating CSV and I have like some recall metrics of like can you find the row in the in in like a 600 L line column it does look like

markdown is is a better tool so for example if you had a PDF in a table that you have a hardtime parsing it might be worth giving this image to gbd 40 or to Opus and asking for it to generate markdown tables back out and then it'll be able to handle things like multiple headers or multiple rows and indices because it can just manage the light space but all right thanks for the informative session do you have any specific

guidelines metrics Telemetry hooks where knowledge extraction involves extraction oh you want to read this one I can also read it but yeah I can try spefic basically the question was like are there any guidelines on you know how metrics Telemetry hooks to figure out where we should instrument our system in order to sort of figure out like how our system is working enough I spent like three years instrumenting recommendation systems if you go back in my writing there's a

blog post called end levels of complexity of rag applications and I kind of just outline like where I like to log and what kind of data I try to log right yeah like you know because it basically looks like a recommendation system right which is a user signed up like signs up I'm going to show them 10 products they're going to scroll and look at the products they're going to click a product they buy the product

right it's very much the same thing like a question came in we use a bunch of text junks right one simple thing that we can do is if I give it 10 text junks and I Jo an answer and sites two of them you could imagine those two are more relevant than the other eight that is still feedback that you can put into an embedding fine team model you can say okay for for any end Tex strunks I

present to the language model I want to find un a model that ranks the ones that were cited higher than the ones that were not cited right and so you can again like use these little micro games in your infrastructure to generate the data you need to F tun a model in the future but this is all kind of like in in in the end levels of rag post and as I build more I I'll have better

information what's your recommended chunking strategy how big should the chunks be should you have overlap between chunks yeah so this is a recommendation that I'm kind of stealing from open ey and thropic and interestingly enough as they are increasing their context lens the recommended chunk sizes have also increased so back in the gd4 32k world I think everyone was recommending like like 500 tokens with a 50% overlap I think now both Opus and Claw are like

100k contexts and now both of them roughly recommend like 800 overlap 5 %. what I've really found is when you actually generate these synthetic data sets right with question answer Pairs and you play around with a chunk size I have not seen too much of a performance change right and so what I yeah what I basically do is I just follow their advice 800 tokens 50% overlap and if I want to improve my systems usually I'll improve them

by

generating synthetic text chunks of whole documents rather than trying to augment individual text junks if that makes any sense is the answer like have evals and test at least test some things is yeah but I also find that like I've never had an eval where when I went from 500 to 800 like something amazing happened right it's so specific like like certain kinds of data have

longer like longer token references right it's like it it it really depends what kind of data you have but that is something you would basically be able to find out if you generated synthetic data sets right like it turns out that you know in legal documents it turns out on page seven it references a date that is a reference to a reference of the sign date which is on page 28 like that might not be like detectable if you use

like 50 token like 500 tokens right it'll just say like well this is due 35 days after signing but signing date is like a different document somewhere else right but that's that's very specific and you'll still be able to uncover that with synthetic data Let Let me refresh this yeah refresh that looks good there's a question yeah what kind of model do you use because I mostly work with Enterprises the model Choice ends up

going down to either Opus Haiku or 40 primarily because we can afford to be less worried about context length limitations and so but just by using one of these models we know it's going to do a good job the only yeah I think that's that's mostly the the thing because we have that context window we have we get to be a little bit less clever with how we do our search and we give much higher resolution instructions on how to

generate outputs that we can use where we can render citations and we can render different interesting uxs right so for example when for Limitless not only do we generate the action items we generate action items with a signes and opis is able to actually generate a URL that we can parse and render like a a profile head or something like that those are the things that we matter and for large scale applications it's really hard to get around the rate limits that

we have on some smaller systems another question is any PI on picking in betting models ultimately again this is about they having those evals right like if you have that synthetic set of a th questions against 10,000 text junks and you have your eval that is just you know when I search this the top three must contain the original text Chunk we just got to try them all and figure out what's whatat and luckily enough if you

if you were in a world where you had you know 2,000 questions and 10,000 relevant text chunks any model you find tun will just outperform whatever base model you have and so you kind of need the evals to make a decision in the beginning but once you actually have production level data data like even 3,000 question answer pairs you'll just find two in which case you're making a different set of constraints right how big is the model how how much

throughput do I have on the model does it fit on an A10 or a100 those end up being the decisions you you have

to make let me think I want to talk about this one just so it's interesting how much do you look at the data and inspect the data up front to understand the questions versus adapting over time in in a bigger company like maybe

not a company with like 10 people a company that's like 50 people I imagine there's going to be someone that's just constantly looking at this data like maybe like once a week like one small thing that I do kind of like the standup process is when we start this initiative it's going to be like you know give me a CSV file of like 40,000 question answer pairs I'm gonna run some clustering algorithm look at the Clusters each have

some ideas rerun it have some ideas and then what I do is I create like very explicit labels okay like this is a marketing like pricing capabilities compare contrast need the haast stack Etc and I log that and basically what I do is I just make sure I have a other key and I monitor the percentage of questions that are other over real time and so I just like check once a week and hopefully other doesn't go from like

10% to like 50% randomly and if it does I can now just come back in and say show me the 10,000 questions that were other run the cluster what did I learn so it's it's going to be a very much a iterative process you know and it's going to be iterative for the rest of your life as long as you're getting new clients and new customers and new users and new inventory to search against and new

types of questions and you know we just do that kind of like during stand up we just like hey you know 10% of questions or other for some reason for questions around categories and date times it has like 30% satisfaction let's go double click and figure out like what the heck is going on how do you teach your clients to look at data feel like it can be hard too oh man if I did I would I

would be out of a job and thank I have a job no I it's really hard like it really is just like the because it's basically about teaching like Engineers a scientific method I think I've been thinking about more as like what is a report you can create what is the data that you can collect and then what are some hypotheses and then once you have these hypotheses for example like I think that we are not doing well because it

looks like all these questions have recent recency filters and we don't have recency filters as a hypothesis the experiment is to add recency filters and the observable is you know that that action score should go up like a simple example was people had feedback on like the transcript summies regenerated and it kind of felt like the issue was just it was too short like a 30 minute meeting and a 2hour meeting had the same length in

transcript so we had that hypothesis so I just plotted like transcript length versus summary length and like the color was satisfaction is it oh it looks like they're correlated and then what we did in practice we just said we just added a line it's like make sure the summary's length is proportional to the length of the transcript we ran an eval that just said for these 50 transcripts without this prompt it was like 1,100 characters on average and then when we

added the prompt it was 2,000 on average okay that's my experiment I'm going to deploy this and see if satisfaction went up and it went up right but it's hard to figure what the system there is the really the ideas are like yeah it's like hypothesis data intervention experiment hypothesis data intervention experiment seems like a yeah seems like data scientist yeah the most popular question right now is just V databases versus cross encoders oh I think we

answered that yeah I think it just comes back down to really it's going to be your evals and your latency constraints as a function of your evals and your business outcomes and in recommendation systems 100 milliseconds of latency could make you 1% higher revenue and so that might just be the difference of I want a top K A th000 Using embeddings and top 100 using ranker versus you know top K 10,000 then top 100 we rank this like that just

might be slower even if it might be 1% better make a SCE less money in the long run yeah so let me let's do this one I had a client that wanted to ask questions of lease contracts such as what is every red flag in terms of late fees this statement would appear 50 times in a document how would you search for this if I knew that UPF front I would just write a synthetic text Chunk

that it's like what are the red flags are on late fees and just have that in explicitly right like if it turns out that you know 20% of the questions are about late fees I might just have Opus look at a single PDF just say give me all the statements on late fees in a single chunk and then have that as something that I can augment my my data set with right you are creating the inventory you

need to kind of sell to the customer that you have like if you were an e-commerce website and never wanted to search shorts but you didn't buy shorts you just got to go and like you know get some shorts and I I think this would be the same example I would just take Opus or hi cou and apply highq on every document and in a situation what you might want to do in practice you can say

okay first first task is for every document that comes in can I predict whether or not it is a lease contract if it is a lease contract I know that my customers care about red flags on late fees for every document that I predict as a lease contract extract out Clauses around late fees and then you put that back in the back database yeah let's do like two more and

then have this up any recommendation for Vector stores and how you decide metadata for different cases again it's just like it's very much like looking at the data like I think the answer kind of look look at the data and like have real examples to work with the only thing I would call out around Vector stores is you don't really need a vector store you need a search engine and that includes more than just vectors right you have bm25

you have SQL right like if I want to do an aggregate a vector database is not going to do aggregate for me right and so V store is only one part of the problem right you might like some companies might just want to use LV some might want to have Dynamo DV and open search and and pine cone in separate places some might be like S3 files those are kind of like implementation details of your business I think all these have one star

ratings so okay I'll do this one as just like a closing statement which is like how do you find your first customers in need of rag for your Consulting Services my back is recommendation systems and one of my core beliefs is that recommendation systems and rag systems are are pretty much identical so the business I worked on before was a business called Stitch fix where a human customer sends an email to a stylist and that email is like a paragraph wrong and

The Stylist uses the AI to turn that paragraph into a bunch of search queries and we we basically show The Stylist like a 100 pieces of clothing and stylist picks the clothing and then writes a note for our customer it turns out that process is the same as an Alm instead of a stylist a human sends a paragraph to the style llm we produce like 100 text Trunks versus 100 pieces of clothes and then the the llm

generates a response whereas you know the sty generates response and so I think I just had a lot of experience instrumenting systems that look like that monitoring and measuring systems that look like that and improving systems that look like that and just been sort of converting all this rexist experience into improving rag applications and yeah that's all I got all right this was awesome thanks Jason yeah problem thank you for going so much over as well but the questions

are good no worries no worries yeah I think a lot of these ones just have like a no thumbs up so I should be good to I'll save this and put them somewhere else