Python and AI Power-Up Program Offline Class-20250922_120853-Meeting Recording

September 22, 2025, 6:38AM

1h 35m 51s

Tirth started transcription

Ronak Makwana 0:04
OK.

Tarun Jain 0:07
So is it started the recording or?

Tirth 0:09
Yes, yes, I started it.

Tarun Jain 0:13
OK, So what we'll do is we'll start with the doubt solving part and then we'll proceed with the evals. So what I've done is I've put together some of the resources which are pretty much important. So if you remember last time I said I have a collab notebook using which you can evaluate your embeddings and retrievers.
So the iterate and MRR that we spoke about. So this is the collab notebook and apart from that the previous points are pretty much the same. Like if in case you don't have any restriction of open source or closed source, you can go with open AI embeddings and make sure you keep the embedding size as 1536. The performance is good.
Good with this size and apart from that if in case you want to go with open source one, the best two models are Gina and Omic embeddings, BGE and Chinese models. They're good as a base model which can be further fine-tuned. So Lama index helps you fine-tune the embedding models, but if you are.
Using BG models directly probably won't be that great because of the Chinese tokens and this is where you have the leader board as well which we have discussed. And if I open this collab notebook, I'll do a quick walkthrough of how usually this works.
So this is a llama index code, but it's pretty much similar to Langchen. We will cover

llama index tomorrow after we complete eval today. So I'll start with the first step. Have I ever showed these two lines of code earlier?

Tirth 1:43
No, not that I remember.

Tarun Jain 1:44
OK, so basically whenever it comes to Collab notebook, right, it doesn't support asynchronous programming. So this is where we have to use nested essential. So what nested essential will do is in Collab notebook you just have to apply so that if there is any asynchronous.
Synchronous operations like we have these two keywords right? Async and await. So whenever we are using async and await, this will not directly work in Collab. So we have to use these two lines of code so that we can use asynchronous programming within Collab notebook.

Tirth 2:09
Yeah.

Tarun Jain 2:20
And there are some operations, some functions that utilizes async programming. That's the reason why we need to use this. If not, we can directly run this on VS code. And the first step is the same thing. What we need to do is we need to create a directory reader. So here directory reader is nothing but your loader.
And sentence splitter is nothing but your chunks. And once you have chunks, what is the next step? You need to create the embeddings.
So here you have ugging face embeddings and once you have ugging face embeddings, what you're supposed to do, you need to create a index. So when you create an index, you have your vector store and as well as the embedding model and your chunks. In llama index chunks is referred as nodes.
OK, so we don't have to worry about the syntax. Here the major thing what we need to focus is how to evaluate the embedding model and get this two score. One is iterate and one more is MRR. So once you do this, what I'm trying to do is I'm loading the embedding model.

Tirth 3:24
So so we are not using Langen community over here for whatever we did, we are using Lama index.

Tarun Jain 3:26
Yeah.
Yeah, the only major goal is to evaluate the embeddings. That's it.

Tirth 3:37
OK, OK.

Tarun Jain 3:39
What you can do is Llama index supports line chain wrappers. We can also add line chain wrappers, read that document from line chain and then use this metrics even that is possible.

Tirth 3:49
OK. No, no, just wanted to know, yeah.

Tarun Jain 3:51
So in this, yeah, we can do that. And once you have the model, this is the major focus. What you need to do is whenever you are creating any RAG model and if you want to evaluate anything, first thing is get your data. And once you have your data, whatever model you have in MTB leaderboard, just change this one line. That's it. You only have to change this line, the model name, remaining code will be the same. So then you have embedding model and then Gemini is not required. The only reason why we need this cell is when you run index internally it uses an engine.

Tirth 4:11
Mhm.

Tarun Jain 4:26
So what typically happens in lama indexes? So do you see this entire block query

embedding model, vector DB, retrieve context LLM. This is happening in just one line of code, this entire thing.

Tirth 4:32
Mhm.

Tarun Jain 4:41
Which is your vector store index. So vector store index has multiple commands. One is as retriever. Tomorrow when we start llama index you will use as query engine. Right, since we are using the same component vector store index, this will consider the entire block and in this block you have LLM and that's the reason why we are using LLM in that.
Where is that code? This line LLM equals to Google Gen. AI because by default in Lama index embedding model is open AI, LLM is also open AI and when you run this vector store index you only need.
Embedding model because can you see this line? What are you doing here?
Vector index dot as retriever. So do you think in retriever you need any LLM?

Tirth 5:34
No.

Tarun Jain 5:35
All right, so when you run this line, there is no LLM call that has been triggered. So till here, right, whenever you are evaluating, there is no LLM call required. But the only thing is when you run vector store index, you need an LLM and that's the reason why you're just overriding the default LLM to Gemini so that you don't have to.
You open your API key. So once that is done, what you have to do is here you have to create a folder. You have to create a folder for data.
New folder.
Then just name it data and whatever data you have right, whether it is PDF, document, CSV, everything you can upload an inside data that you want to evaluate and then you can directly just read that particular folder and then you have chunk embedding. All these things are same.
This is very important. Here you need to define what evaluation you need. One is MRR and one more is head rate. And once you define this, if you want to generate

the data set, you can use LLM where you have notes which is your chunk. For every chunk you need to generate 2 portions.
Once you have this, you will have a dictionary. Inside dictionary you will have queries, corpus and relevant docs. So queries is basically like this. You have one unique identity and you have a question which the LLM will generate and then relevant documents is a.
Code. So if I see this relevant documents, you have the same query. For this query you have certain code. Inside this code you have your corpus of data which is your relevant documents. Just a second.
So whatever you have here right this will contain your relevant documents which is nothing but your corpus. And once this is done you directly just have to run this eval results. So once you run the eval results you are passing your data set and here if you see a using await which is happening asynchronously.
And then you're just printing the table. Now you have your iterate and MRR. This should be high and most of the cases this will be around 80% or 90%, but GPE large it is around 66%.
So you just have to change this thing, which is your embedding model.

Tirth 8:04
So can you help me understand? Once we Scroll down a bit below, we are asking the LLM to create the queries.

TJ **Tarun Jain** 8:12
Data set. Yeah, this data set.

Tirth 8:13
Yeah, yeah. And then this data set also has the reference to the chunk that should be received.

TJ **Tarun Jain** 8:19
That is corpus.

Tirth 8:23
OK, OK.

Tarun Jain 8:24

So when you print this corpus right, this will have the relevant documents, relevant documents which is node.

Tirth 8:31

For each query.

Tarun Jain 8:33

So now let's suppose you have this query, right? Here are two questions for your upcoming quiz and examination. For this particular thing, which corpus or which node was responsible to generate this question?

Tirth 8:37

Mhm.

Mhm.

Open.

Tarun Jain 8:50

That is this so I've actually removed it, but.

Tirth 8:52

So we are doing it reverse, we are doing it with reverse. So we already know the data, we are creating the questions out of it and then we are testing out from the embedding. If we ask the same question, is it giving that data back or not?

Tarun Jain 9:06

So when you create this data set right, one thing is you can have your own data set. Second, we are directly using our own data and creating data set. But the major thing is you need diversity.

So diversity will have single term questions and multi term questions.

Tirth 9:23

Hmm.

Tarun Jain 9:24

So Singleton is like how directly you're asking question and how your embedding or evaluate retriever is performing. And if you're asking double questions in the same question, if there is double, then how is your retriever? And this is mainly based on that.

Tirth 9:39

Oh, how do you get the diversity from the original question? Like LLM does it or how does it work? I'm just trying to OK.

Tarun Jain 9:44

Hi LLM this is prompt. This is prompt. So in prompt you will define what is the diversity unit. I'll talk about it. I have that slide here. How should your data contain and how to build one?

Tirth 9:55

OK.

Understood. Understood. Welcome.

Yes.

Mhm.

Tarun Jain 10:13

Because here you are not evaluating LLM right? So you are only generating questions and you need context. So basically it's like Q of probability with C which is question and context. There is no answer in it because we are not evaluating grad.

The only purpose is to evaluate the retriever which has the embedding model which is question and context.

Tirth 10:30

Um.

Me.

OK, OK.

Tarun Jain 10:41

And the reason why they have done this is because they can map it. So if you check the data type of this, probably I have to rerun this. I'm not sure if I have that code available.
OK, I'll tell you what will be the data type of this. So when you print this right your QA data set.

Tirth 11:00
Mhm.

Tarun Jain 11:06
Huh.
The data type of your data set will be something called as embedding turn fine tune data set.
Let me just cross check this.
API reference.
OK, uh, let's do one thing. Uh, I'll go with the slides. Until then, I'll also run this code.

Tirth 11:51
OK. OK. Yeah.

Tarun Jain 11:54
So that I can show the data type. Data type I'm pretty much sure it is something like embedding turn fine tune data set. So when you run this right, this particular thing is in a dictionary.

Tirth 12:03
Mhm.

Tarun Jain 12:06
But this dictionary is closed by this particular object.

Tirth 12:11
OK.

Tarun Jain 12:12

So only if you see I'm doing dict. After dict I'm trying to add Q data set and then keys. So queries it's visible. You have certain ID and then you have question. Corpus is your relevant documents. This whatever you have right relevant documents is just a code.

Tirth 12:18
Thank you.

Tarun Jain 12:29
This code is repeated in corpus.

Tirth 12:30
Yeah.

Tarun Jain 12:33
I'll show that. Uh, let me just run this.
The only thing is this takes time. The I've added this as T4 GPU.
And I've already pushed this code as well in the collab that we are using. I mean the GitHub.

Tirth 12:48
OK.

Hardip Patel 12:51
And we started the cloud.

Tarun Jain 12:54
Oh, what happened?

Hardip Patel 12:55
Yeah, yeah.

Tarun Jain 13:01
Hello.

Tirth 13:01

Are the female not able to hear him?

I think it OK.

Tarun Jain 13:10

OK, no worries.

OK, let's proceed next. So is this clear? These were some of the questions and also in the previous class we had this. Some of you had left already. So what we actually covered is the traditional rack that we had covered. And then we have also looked at binary quantization so that we can have faster rack performance. Then we have. Agreed search. So if you see this word that we added here safest, the reason why we added the safest keyword was I also mentioned some of the research techniques that one can explore while working rag. One is self rag. Then we have late interaction. I'll repeat what late interaction is. So we have dense vectors, right?

We use any embedding model from MTB leaderboard. Then we also add sparse vectors. So for sparse we're using VM25. Just like that there is also one more research concept which uses late interaction. It is also an embedding model, but it uses an actual model which is cold but.

So when do people use code word when they're working with multimodal RAG? So I'll give one example of multimodal RAG. Let's suppose you have data. In your data you only have images, right? And these images are not like scanned PDF.

So what happens in scanned PDF is you have images in your PDF but it is text. But in multimodal drag this images can be anything. It can be food recipes right? It has only images of food recipes and.

Tirth 14:42

Yeah.

Tarun Jain 14:46

One more example, it can have only tables like research papers. So if you want to build anything related to multimodal RAG, people usually prefer late interaction, but there is also alternative approach that we will be taking and apart from that there is also something called a CRAG.

Which is self-controllable RAG. So why do we need this in first place which is self

RAG and CRAG? Most of the times what is RAG? Basically you have retriever, you have generator. So what are the issues with retrievers? You might not find relevant document for the given user.
That is a drawback. So if you want to fix that automatically, you have a research concept called self drag. This takes too much of time and it's also not properly used in most of the research purpose. I mean in the production application, it's just research.
But if anyone wants to experiment, you have multiple cookbooks available on Langtin. You just have to search for self frag Langra, right? So these are the three techniques that I usually said, but the safest one which most of the people are using is hybrid search.
That's the reason why safe X was added. And then there is also something called as mini coil. Mini coil. It's like a subversion of sparse vectors. So when you take the code of mini coil instead of beyond 25, you're just replacing with mini coil.
So the code is same. Whatever code you wrote, whenever you are loading VM 25, instead of VM 25 you are using minicol. So that is called sparse neural retriever instead of sparse vectors. It's just a subversion of VM 25.

Tirth 16:12
OK.

Tarun Jain 16:26
And this is the article for it, which is written by me only. Like you just have to change beyond 25 to mini coil. Remaining code is same whatever you saw in hybrid search, right? Mini coil is also used in hybrid search. The memory utilization of mini coil is much better than beyond 25 and this was released in 2025.

Tirth 16:37
Mhm.

Tarun Jain 16:46
this year only, this technique.

Tirth 16:49
Sure.

Tarun Jain 16:51
So it's under research as of now.
All right, so while while I was discussing this in the yesterday last class, this concept was missed. So this was one question and second question was the question which Steve had like how to avoid reputation for long context generation. So I found one article which was working on the similar use case. So what they're trying to do is. When you're creating the chunks, for every chunk you're creating summary and you're passing that for every single chunk or LLM call that you're doing. So this way what you're trying to do is you're avoiding the reputation. So if you see the second line, use iterative refinement with context preservation. So this context.

Tirth 17:24
M.
Yeah.

Tarun Jain 17:34
Is nothing but the summary where every single interactive refinement is over the chunk. And then there is also one more article. In this article what they're doing is they're using schematic chunking. So what happens is before you can even.

Tirth 17:41
M.

Tarun Jain 17:49
Do rag. You're doing semantic chunking where you're grouping all the similar chunks and once this happens then you're looping through the chunks. So here you have your chunk.
This is the second step. First step is you load the data. Second step is you perform semantic jumping. Third step is you have something called as MapReduce. So if you just search for MapReduce and if you perform summarization, this is mainly for creating summary.

Tirth 18:19
Mhm.

Tarun Jain 18:20
So MapReduce is nothing but it will utilize less tokens.
And if you open this article, you will see the diagram as well. So if you see what is the problem statement here, you are you ask a prompt, GPT generates a response, then you ask continue, then again you have response, continue response. So while you do this, there are times LLM will hallucinate. There are times where.
You will have repeated tokens. So what they did was once you have user input you will generate the content and then you will create the summary for each section.

Tirth 18:46
Mhm.
OK.

Tarun Jain 18:55
And once you have that each section, you will have the final response and then you have to pass it to the LLM.

Tirth 19:03
Yeah.

Tarun Jain 19:04
So this is 1 approach which again if you're using LLM it's too partially. If not the another alternative approach what I found out was using multi agent architecture. So this is the same thing what even you mentioned that you're creating subbeddings right?

Tirth 19:19
Mhm.

Tarun Jain 19:20
So each subbedding will be an agent which will have its own context.

Tirth 19:26
OK.

Tarun Jain 19:28
But for time being you don't have to experiment with this. You can experiment with the summary one. You can use NLTK for summary.

Tirth 19:33
M.
OK.

Tarun Jain 19:36
I hope we guys remember NLTK.

Hardip Patel 19:40
I guess my.

Tarun Jain 19:42
So NLTK has some libraries, I mean some functions where you can use for creating summaries.
And again, this is one time approach, right? This is happening on the offline document processing. Once you have saved the data, you don't have to worry about the answer generation, which is your real-time report generation.
So is this clear here? So summary where do where would you add summary?

Tirth 20:05
Yes, that is very clear.
Three chunks.

Tarun Jain 20:14
It should be metadata.

Tirth 20:16
Yep.

Tarun Jain 20:17
Chunks will be added as a metadata. Sorry, the summary chunks.

Will be added as metadata.

This one is clear, right? So this is your actual chunks, whatever you say in schematic chunking, right? So if you look at the approach, you start with raw data.

And once you have raw data, you will perform chunking. So when you do this chunking process, what do you have? You have page content and you have metadata and the approach that you're taking for chunking is schematic chunking instead of.

Recursive character text splitter.

And before you save it in Vector DB, before you save it in Vector DB.

For each chunk that you have.

You can add a metadata.

So this metadata is nothing but your summary.

Oh, is this clear?

You can try this approach. You can let me know if it works or not. If it doesn't work then probably we'll try with multi agent and these are the two resources which are working on same problem statement.

Tirth 21:31

Yes.

Yes.

Perfect.

Tarun Jain 21:47

OK, so does anyone have any other questions before we go with Ivald?

Hardip Patel 21:59

No.

Tarun Jain 22:00

OK, so in evals, probably we just have to focus on three things. The first thing is we have to prepare our own data set. If we are not able to prepare our own data set, we can use any libraries that will help us to build a synthetic data set. So let us first discuss how your data set should look.

Tirth 22:00

No.

TJ **Tarun Jain** 22:20

Look like right? So whenever you're creating your data set, you have to focus on two things. One is you need to have scenarios. So this scenario is like let's take an example of you're building an Atlantic chatbot. So what are the single answer? Single answer is like what framework is Atlantic using that is a single.

I'll answer then you will have multi answer where you have multi term questions. So multi term questions is like how do I contact at and I wanted to know what frameworks they're using. So this is like multi questions for which you have multi answers.

And then you also have no answer. Let's suppose if I ask questions like who is Virat Kohli? Now Virat Kohli is not at all relevant to your website, so this is no answer. And then you have invalid data provided. So there might be chances that most of the times.

Let's suppose you have authentic chatbot and you have anything related to SAS, right? And related to SAS itself, there are multiple data source which is available, which data is relevant to your website. So those kind of scenarios you are supposed to generate. So when I say scenario.

Because it is just prompt, you have to define what are the negative edge cases that your data set needs to handle and 3rd you have to keep it in terms of personas. So this personas is like is it relevant to technical folks?

Is it relevant to non-technical folks or is it relevant to the experts? So based on what data set you need to create, you also have to define the personas. So let's let us take an example of how this looks like.

So first what you have to do is you have your data and once you have your data you will define your prompt. So in this prompt you will define all the scenarios, possible scenarios which will handle the negative edge spaces.

And once you have the negative edge cases, you need to tell now generate.

The data set.

With distribution.

As follows.

So now it will be like for technical person you need 10% of questions or 20% of question. Then for non-technical person it will be 20% of data distribution. So what do you mean by this technical 20% distribution?

In simple words, whatever questions you generate.

How will a technical person ask the question? It are these are those questions like how will a technical person answer? How will a tech?
Person ask a question.
So this will be 20%, then non tech 20% just like that you have expert and you also have other roles. I'll just open this blog.
So if you see you have new user, expert user, non-native speaker, busy professional, then techno probe, then elderly user. When you're writing prompt, you can just copy this particular what you call this particular descriptions.
Is this clear? One is defining scenarios and one more is defining personas. So when you have questions, you have different diversity. So that is the main purpose. When you're creating synthetic data set, the major thing what you have to focus is the data set has to be diverse.
It has to handle multiple edge cases and how are you generating it? You are using LLM to create the realistic user inputs. So this user inputs is nothing but questions. And in most of the cases, how many questions are you supposed to generate? Hardly. You can have around 30 to 50 questions to begin with.
So first step we start with the data set and once you have the data set here as of now you only have questions. Now once you have the questions, this is where you need to have a domain expert. This domain expert what you will do now is whatever questions are generated.
It should tell whether it is pass or fail. So I'll tell you what this pass or fail judgment is. So let's suppose you generated the data set. Now in this data set you have only two things. One is you have questions, second you have context.
One second.
So.
Yeah, sorry.
So if you look at these different scenarios, right in different scenarios there are times when you have no answer. So for that question you have no answer, but when you look at the relevant documents, you have some relevant documents. So now what you need to do is you have to tell.
Whether this pair of question, context and answer, is it true or is it false, right? And with whatever you say, whether it is pass or fail, you have to give certain judgment. Why did you make it pass? Why did you make it fail? So this is where domain expert comes very important, which will add the judgment. Now why do we need to add?

Pass or fail? Let me tell you some of the use cases here. So can you see this left hand side? There is a image which says 4.04.94.93.5. Are you able to see this?

Hardip Patel 28:11
religious seems to be good here.

Tarun Jain 28:13
Yeah, so basically if you remember when we did prompt engineering, I told you one example which was Rubrics based evals. So now what happens in Rubric based evals? I gave you 2 examples.
So the first example was product documentation use case.
This was code product documentation and second example was.
AI resume hiring. So now let's suppose we'll take this example first, which is product document UCS. Now you have a code.
Which is related to LLM and let's suppose you built your own library and you have total 10 LLM. Let me also open it open Ajay.
So this is the folder that you want to prepare the documentation. So you have Azure, you have Cerebus, you have Claud, you have Coir, you have Gemini, and then you need to write a documentation.
OK, so this is how the documentation needs to look like. You have certain text and then you have the modeling and how to use this model. So what is your input? Your input here is a code which is this particular code file and what is your output? Output is a documentation.
So this documentation is nothing but this is this should be your final output. So how do you evaluate whether whatever document was created if it is right or wrong? So this is where you define certain rubrics.
So this rule will be example. If example was generated, view file marks.
If use cases was generated, give 5 marks and for technical details, technical details, give 5 marks and what else is there in this document? One is heading.
Title example then installation guide. So if there is any installation guide.
Give 5 marks. So if you want to understand this flow, you have the input which is your code or GitHub and the output is the documentation. Once you have the documentation you're running the LLM. So this LLM has to evaluate based on rubrics. So this will give you scores for example use case, technical and technical guide and at the end it will give you 18 out of 20. So now for this 18 out of 20 it will also give you

breakdown. That example was 5, use case was 3, technical was 5, installation guide is 3.
So this is how usually you define Rubik's Rubik's based device. Same goes for AI resume hiring. So for AI resume hiring what will you do? Your input will be a PDF and your output will be.
A score. So this score will be how is that person in technical? How is this person in terms of leadership and how is this person in terms of projects? If the projects are relevant to the JD, so one will be your PDF and the other input will be.
The job description if this person technically is he good enough as per our JD, is he technical good enough projects as per our JD and then you'll have this code. So this is mainly for Rubik's based eval second when it comes to RAD.
You should.
You should never use.
Binary based scoring. Sorry, not binary. This is you should never use.
Rating base scoring.
Rather use binary. So binary is nothing but yes or no.
Or pass or fail?
And this is what we meant here. So when you're creating the label for the data set for that, make sure for every single question you have, you have such certain judgment. So this judgment is nothing but pass or fail along with critique. So critique is nothing but why did you made made it pass? Why? What was your reason to make it fail?
So this is what is defined here. So if you open this particular link directly, it is redirecting you to that particular blog. So this article is pretty long, but you can only focus on 2 subsection which is mentioned in this slide. One is.
If I hover on this, can you see this particular URL?
Let me copy this.
Paste it here.
OK, wait.
You just have to check with this one. Don't stray from binary pass fail judgment when starting out. So this is 1 sub adding you have to check out and 2nd sub adding is Step 2 which is create a data set remaining. If in case you are interested you can learn but the major part is the creating data set.
And the third thing is.
This part.
If I directly click on this you will be redirected which says don't stray from binary pass

fail judgment when starting out. Here he also mentions like why do you don't have to have 125 scale. So the reason being is most of the time when you're building RAG application.
If you have truthfulness as four, there is not justification for why was it 4, why was it 3? LLM will not be able to generate it and this is the reason why it's better to use true or false or pass or fail, which is binary.
Oh, is this clear?
One is Rubix based and the second one for Rag which is LLM as a judge.
We have to use pass or fail but with critique.

Tirth 34:39
Yeah, yeah.

Tarun Jain 34:42
And the article resources is here only one you can click on this link which will tell you regarding the binary what is the purpose and 2nd is this one how to create data set.
If you directly click on this you also have certain examples on how you need to have that.
Data, so scenarios, persona and then the question.

Tirth 35:04
OK.

Tarun Jain 35:05
So you'll also have examples on how usually it will be. If you see if you have the user an answer, what was the judgment? It is past and then the critique. What is the reason to make this particular judgment as past? So this is where you have domain expert.
So if you check this article, the first thing is you need to find a domain expert, then create data set and then pass or fail which will be decided by the domain expert.
Is this clear the first two steps?
This particular thing, the second slide is where the most important and time consuming part of your entire code base because this will take too much of time.
This one is easy. You can create data set using LLM within few hours.

Hardip Patel 35:41
So.

Tarun Jain 35:52
But this might sometimes take days because most of the time domain expert will not give critiques. He'll be like, hey, just pass it to LLM, we'll see it later. So this part, most of the time it is skipped, this part some of the time takes weeks and.
This is where most of the evals. If it works, it works fine. If it doesn't work, this is the step where it will fail. So this has to be treated very properly, especially how you label the data set. Because if you label properly, there are also ways you can self-improve your pipeline using a library called.
DS5.
So you can just search for DS5 self-learning. So if you see you have DS5 self-learning pipeline. So this will actually help you on how you can self-improvise your entire rack pipeline if you have critique because when you use DS5 you also have to give what you call a data set URL.
So using the data set URL, it will train that particular LLM model which will improvise your entire pipeline and you can use DS5 with Gemini. And let's see if we have some time probably I'll tell you some of the cookbooks where you can find that exact code.
But in order to use this library it it.

Hardip Patel 37:18
We lost him.

Tirth 37:21
Got disconnected.

Ajay Patel 37:23
Maybe.
Yeah, I'll.

Tarun Jain 37:39
Hello, sorry, I actually accidentally clicked on exit.

Tirth 37:40
It's back.

Ajay Patel 37:44
OK.

Tarun Jain 37:46
So is this clear? I'll also add DS5 here so that we don't forget it DS5.
Later purpose.
Only when something goes wrong.
This is for creating self-improvement pipelines. This is actually not required, but if anything is not working properly, if you have created your evals pipeline and if you want to improvise it only, then you'll be using DSI. If not, you can just skip this.
Is this clear? This will be actually very tricky because not every time we can label the data set.

Hardip Patel 38:28
Yes.

Tarun Jain 38:33
Is it clear anyone has any questions?

Tirth 38:37
Notice of now.

Tarun Jain 38:40
Cool. So these are the two learning resources. One you can click on this link directly and second one is this one.
And now coming to the last part which we will implement today and that is called LLM as a judge, which I've already said on how we have to build one. So when it comes to building LLM as a judge for RAG, we can use tracing and monitoring which is our OPIC. And since we have three major components, one is question, then you have retrieve content.
Text C which is our context and then answer.

So if you have worked with Langraf, which we have done, we have used these three variables in our rack state which is QC and DA. And using these three things what we can do is we can have context relevance. So now if you see this particular quotation which is C.
Then standing pipeline queue. What this is telling is context relevance will tell you is the question relevant to the context or not? And then you have something called as faithfulness and groundedness. So in groundedness what you're trying to do is answer relevant to the context or not.
And then you have answer 11 is answer relevant to the question or not. So these are the three major parameters that you will use and this is called RAG trial.

Tirth 39:54
Yes.

Tarun Jain 39:56
So this is referred as.
Rag.
Ryad evals. So if you just search for this, you will find multiple frameworks which will help you build this particular thing. But at the end of the day this is just prompt, so you can open any libraries that you need, whether it is Rulence, you have something called as.
Rulents. Then you have DP valves.
And then you have Ragas. So all these three frameworks supports Rag Rad, which will help you build context relevance, faithfulness and answer relevance. And for every single thing they're just using prompt. You use a prompt and you will tell whether it is true or false. And what I've done is I've actually copied that prompt.
From the Langsmith. So Langsmith is a product of Langchain which is mainly used for evals. So I've copied the prompt from there and we will use that for context relevance and answer relevance and for faithfulness we will directly use Ragas.
And there is one article called. There is only 6 rag evals. So if you open this as you can see you only have 3-4 components, question, relevant, context and answer and then it breaks down. You have context relevance where.
Is context relevant to the question, then faithfulness, then answer relevance, then remaining thing are same. It's like you're just repeating the combinations, but these are three majors.

Context relevance, groundedness and answer relevance, which is your RAG triad.
OK, so this is it. What I'll do is I will. You already have the slides, but I will still share it.
And let's proceed with the code.
All right, this is done. Let me show you the data type.
So if I hover on this, probably it should show me the data type. So this is the data type.
If you see.
Embedding QA fine tune data set.

42:54
Hey.

Tarun Jain 42:57
Embedding QA fine-tuned data set. So this data type is nothing but your QA data set which will have query corpus and relevant documents. Corpus is your actual data which is extracted from your chunks and it is pretty long and that's the reason why you're not able to see it. It's not output was visible just like how other.

Tirth 42:59
Hmm.

Tarun Jain 43:17
Output visible.
You can try it. So if anyone is using any other embedding model, right, just replace this, add your data inside a data folder and then just run the entire cell. You don't have to worry about the code. Once we learn llama index, most of the code cell will be similar.

Tirth 43:21
Yeah.
If this might not be the right call, I can take it offline with you, but I'm still a bit confused. Like when this QA fine tune data set is created, does it create with diversity already or?

Tarun Jain 43:47
Yeah, it will create with diversity. So that is that is returning the prompt.

Tirth 43:49
OK, OK.
OK, OK, OK.

Tarun Jain 43:54
But I'm not sure if we can modify that prompt, because most of the time obviously we'll need our own prompt. Some keywords will be there.
But the diversity.

Tirth 44:04
Where where is this prompt? Where is this prompt?

Tarun Jain 44:07
I have to check if they have. OK you can edit it. Can you see this default way generate from template?

Tirth 44:14
Yes, before you were.
Mhm.

Tarun Jain 44:18
You can pass that as an input if in case you have it.
You do you saw that there. So the notes you have LLM and then you have number of questions for some to be two. If in case you have your own template you can define a parameter called QA generate prompt template equals to then you can define your prompt.

Tirth 44:24
Yeah, yeah, yeah.
Thank you.

Tarun Jain 44:41

If you're not defining your prompt, it will take the default one.

Oh, it's this one.

And if I click on this view source.

This is the prompt.

Given the context, information or knowledge generate on the question base.

See, can you see this? The question should be diverse in nature.

So this is a common keyword across whenever we are generating the data set. But if in case you need very specific like persona and all, instead of using this QA generate prompt template, you can have your own prompt template, but you have to make sure these two input variables are there.

Tirth 45:19

Thank you.

OK.

Tarun Jain 45:31

And your do you see this keyword teacher and professor? This is something that we usually use for context relevance and answer relevance as well, which we'll do it now. So if you see you are a teacher reading a quiz, let me share this collab notebook.

Tirth 45:39

Yeah.

Tarun Jain 45:46

This is today's collab.

Iterate and MRR was just based on what previous question we had.

So this is the template code. Some of the code is already available, some of the code we will write it now, which is mainly for faithfulness and context precision.

And regarding this generate synthetic data set, how to run evals, I'll come to this one.

Oh, is this clear, Kia?

Tirth 46:23

Yeah, yeah.

Tarun Jain 46:25
Create an MRR. So this is already available in slides also and also on the GitHub repo.

Tirth 46:26
Yeah.

Tarun Jain 46:34
It should be under notebook and under notebook you have evals, you have it rate and MRR.
OK, so let's start with this. Make sure you create the copy of this notebook instead of editing it in this Polar notebook.
And this four lines are safe. What are we trying to do? Whatever we built last less, which was the influence of quadrant, we are using the same thing. Fast embed, quadrant, client, Langraph, Langchain and then Langchain, Google Gen. AI. And if in case we need to use OPIC, let's use OPIC today I want to show.
One such detail. So do you guys remember in OPIC we had three things. One is we had trace, then we had feedback score and then we had metadata.
Uh, did you guys observe this?

Hardip Patel 47:36
Yes.

Tarun Jain 47:40
Oh, this one is clear, right? The Rubik's based eval and LLM as a judge. Today we are working with LLM as a judge.

Tirth 47:44
Mhm.
Mhm.

Tarun Jain 47:48
Rubrics based device probably there will be some agent used here.
So let me add this in the notes.
So in terms of notes, one is Rubix based device is pending, then Postgres is still

pending and then Langchen prompt template is still pending. So these are the three things.
So these are the three things. So Ruby based device, it's not actually pending. This is something just I've added as a note which needs to be covered.
When I do agent PostgreSQL and lengthened from template, this is nothing but the visualization part. Last time I told not to use this, but I didn't give the reason for not using it. So this one is still pending. Let's see when we will cover this, but yeah.
So as of now, when you use opaque, what we have done is we have only experimented with input and output. We never used feedback score. So today when we are doing evals, what we will do is we'll try to add our feedback scores that we get from LLM evals and.
Considering the metadata, metadata will only use when we are working with MC PS and as well as tool calling, function calling. Until then we'll not get the metadata part. I'll the reason why we need metadata is let's suppose if you know agent, if you remember we work with tool calling.
So what happened in function calling? Let's suppose I ask anything real-time question and I'm using a tool called tably.
And then LLM is generating the response. So how is LLM generating the response? I want to know what tably return. So if you want to track and trace what was the input and output that was passed in a tool, that is where we'll be using feedback metadata. And feedback and scores will be used for reverse. Input and output is very common. This we have already seen. So that's the reason why we are using OPIC again today. And then you have ragas. Ragas is widely used when you have to evaluate.

Tirth 50:22
M.

Tarun Jain 50:30
RAG agent and LLM and this was also showcased in most of the.
Keynote that was done in open AI. So even open AI team is widely using this. They have also shared it in one of their there was 12 series of open AI. If anyone remembers 12 series of open AI during that time one of them had ragas in it.
So this is widely used. It's an open source tool itself. We can just search for Raga's framework. So this is the repo.
And what we'll do is we'll be using this repo and there are multiple metrics that they

provide faithfulness, context, relevance, precision, recall, we'll just use those.
So this libraries we have done these two we need to install today and then this code is same.
We start with the embedding model where we have Gina and Quadrant. Make sure it's the same that you have used for hybrid search, then the API keys configuration, then you have client and once you have the client we need to create node since we are using Langraff.

Tirth 51:34
Thank you.

Tarun Jain 51:45
System prompt, user prompt, then drag state how many variables we need to have in drag, query, context and answer. So if you check the data type of context, it is list STR. This is very important today because when you are using ragas when you have to pass context.
Contact should have list STR.
Which we have it in list STR. Usually what is the format of lancet? It will be list. Then inside list you have page content.
And metadata, which we don't need. What we need is we directly need the page content of the each chunk inside a list. So this logic is already written in search. So if you see it for info in relevant points dot points, you're only getting the document. And you're saving it in a context. So the data type of context is list STR. This is the same thing again.
I hope you remember this logic what prefetch does.

Tirth 52:50
Yes, yes.

Tarun Jain 52:51
And then you have query points which is taking three, so RK value is 3 and then you have answer generation.
And once we have the answer generation we are creating the node. So we start with start and after start we have search context, search to answer then answer to end.
I hope till here everyone are done. So you can just run the cells and check this output

and once this is done then we'll start with LLM as a search because we need the three variables for evals.
So those 3 variables are context equals to.
Result.
Of context.
Or should we use these variables as per this blog which is C?
You.
Andy.
Just define these three variables CQ and A.

Tirth 54:01
Mhm.

Tarun Jain 54:10
C is for context, then question and answer, and we have context relevance, faithfulness and answer relevance. Sometimes faithfulness is also referred as groundedness.
And let me add this inside LLM as a judge.
Let me know till here if it is done.

Tirth 54:45
In a minute.

Tarun Jain 54:56
Until then, I'll just show you from where I got this from.
Lance Smith.
It was taken from here.
So here, if you see you have correctness, then you have relevance.
Inside relevance you have instructions.

Tirth 55:35
Sure.

Tarun Jain 55:37
And then you have groundedness. Groundedness is nothing but your faithfulness.

Then retrieval, retrieval relevance is nothing but your context relevance. So if you see retrieve docs, So what is retrieve docs?
Retrieve dogs is nothing but.
See.
And input is nothing but QC slash Q is nothing but your context relevance, which is tagged as retrieval relevance in this blog.
And here if you see relevance, you have response vexes input. So response is nothing but TA input is nothing but Q. So this relevance is nothing but your answer relevance.
So if you search for rag triad.
So these are the three inputs we need to take context relevance, answer relevance and groundedness.
Copy.
OK.

Tirth 56:57
Question over here I have is whenever we are doing this LNMS judge, would it not need all the context details before judging it?

Tarun Jain 57:08
For LLM as a judge, no. If it is not there, then it will say no. If it is there, it will say yes.
So this is regarding you only after you have the response.

Tirth 57:21
Yeah, so we first got the response from the query embeddings right. Now we are using LLM as a judge to whether see whether the retrieved context is right or not.

Tarun Jain 57:21
If you want to.
Yeah.
Yeah.

Tirth 57:33
Right.
So would they not need all the context to judge it properly?

Tarun Jain 57:40
Oh, so can you give me a reference what you mean by all the context?

Tirth 57:44
And so like we have a whole document of, you know, our PDF file and when we are asking it, yeah.

Tarun Jain 57:48
OK, let's.

Tirth 57:52
Yeah.

Tarun Jain 57:52
20 chunks is referred as whole document, OK.

Tirth 57:55
Great. And uh, we are usually passing limited turns to the LLN.

Tarun Jain 58:00
Three, which is your context.

Tirth 58:02
Right now when we have to use LLM as a judge, when we ask like OK, was the question answered properly? Does it not need to know all the 20 chunks whether to evaluate properly?

Tarun Jain 58:15
No, no, no. So that that won't serve the purpose, right? So how? Let's suppose you're acting as a judge.

Tirth 58:22
Mhm.

Tarun Jain 58:24
So now you had a query, so you want to judge the answer, right? You're judging the answer based on the given query. So this is 1 metric and the second metric what is happening, which is our question, which is the context.

Tirth 58:26
Thank you.
Mhm.
Hmm.

Tarun Jain 58:40
Now in this regard, the context what judge will do is it will only refer to three, so there is no need for.

Tirth 58:46
And then it will tell whether it was good or not. But now let's say you know I'm getting the context which is partially true or it does answer the question. OK, this is just for my understanding. So it is partially true or it does answer the question in a way.

Tarun Jain 58:50
No.

Tirth 59:05
But there could have been a better answer because there are different chunks and you know, for example, for example, I have a document and if I ask it, how many levels of software engineers are there in at Atentic?
OK and or and explain me a brief introduction about each one. So now in different chunks there are different answers to it and we have you know two short questions or it is multi step so it will tell me 5.
And then it will give me some details which is not true. 5 is the right answer, but then some details is not true. Wouldn't the judge or the LLM require the whole document to judge it properly whether the answer given by that context is true or not?

Tarun Jain 59:55
No. So in this regard, when it comes to judging it, it will only look at what was the context used to come up with this solution. So it will only judge for the given thing.
So this evals whatever you have, it is post process.
So when you do post process, you only have these three variables with you. Old document itself is not there when you're doing answer generation, right? So do you think when you do answer generation you have these 20 documents with you?

Tirth 1:00:18
OK.
No, but that is happening from the vector database.

Tarun Jain 1:00:31
Uh, but you don't have those 20 documents with you, right?

Tirth 1:00:34
Great.

Tarun Jain 1:00:35
So it's like you'll have to give your K value to be 20 then.
So you understood the problem. So basically when you're making this evals, right, let's look at this code only. Somewhere in this code, do you think we have those documents, the 20 chunks data that we have obviously if I.

Tirth 1:00:44
What will you tell?
Mhm.
Mhm.
When we load it.

Tarun Jain 1:01:00
But that loading will not happen real time, right?

Tirth 1:01:04
Right, right.

Tarun Jain 1:01:05
So if I need all the 20 documents, what is the safest option? Here I'll keep it 20. So this will give me all the chunks. But do you think for judgment all 20 is required in terms of evals? It makes sense if it's well and good, but what is the purpose there?

Tirth 1:01:10
Yeah.

Tarun Jain 1:01:23
The purpose for you to have all 20 documents is to have a feedback mechanism.
So that is the old purpose you need, right? So here the judgment in your case is like you need a feedback.

Tirth 1:01:39
OK.

Tarun Jain 1:01:40
So feedback in the sense what I meant is let's suppose you give 20 documents.
20 documents.
And then you ask judgment or alum. Hey, this is the context which is your 20 documents.
And then you have question which is XYZ.

Tirth 1:02:00
Mhm.

Tarun Jain 1:02:01
So LLM doesn't know how to do schematic meaning schematic search. So do you think there is any search happening? LLM will just see is this question relevant to the 20 documents or not? And if in case you want any feedback then it can give you the feedback message. But feedback is different than evals.

But in most of the cases, I don't think 20 documents is even needed for feedback.
Also, this will just add too much of redundant data.

Tirth 1:02:23
OK.
OK.

Tarun Jain 1:02:31
Rather we can just increase the K value so that we can have critique. So what I'll do is I'll make it K equals to 8 and then I will pass that as context and then what you can do is when you build this evals you can just make sure to tell which chunk.

Tirth 1:02:38
M.

Tarun Jain 1:02:49
Was most relevant to this query and you can improvise your critique based on that. If you need feedback you can define hey give me the feedback which we are also doing it here.
So if you see when you have the final response, this is the only thing that I need, but I'm also telling it to check step by step for each chunk and provide me the critique.
So how you want this critique you can define.
But in most of the cases, we don't need all the 20 documents, it's just the search documents relevance.

Tirth 1:03:27
Open.
Make them so far.

Tarun Jain 1:03:30
But we can increase the number of K value in that case instead of the whole document.

Tirth 1:03:36

Mhm.

OK.

Tarun Jain 1:03:40
So everyone have the CQ and A ready.

Hardip Patel 1:03:44
I'm having a problem.

Tirth 1:03:45
Oh.

Tarun Jain 1:03:46
So you can just try to print this print C.
C should be list and it should be length of three. Length of C should be 3 and then Q is nothing but whatever you asked. Then A is your final response. This cross check if it is available.
So when you saved your database right, what name did you give here?

Hardip Patel 1:04:32
Well, I forgot it. It is different, so that's why it's the problem.

Tarun Jain 1:04:38
Yeah, so if you see the error message, the vector with the name dense is not configured. In our case it is dense. In your case, if you gave like this right the embedding model name, it will be different.

Hardip Patel 1:04:49
OK, so if I haven't.

Tarun Jain 1:04:53
So this name was defined when we created collection.

Hardip Patel 1:04:57
OK, let's see I I might have given this.

Tarun Jain 1:04:59
So if I do vector.
So you can just.
So where did we create collection?

Hardip Patel 1:05:08
Yeah, it's.

Tarun Jain 1:05:12
So check this code. When you did create collection, what name did you give here?
That name should be matched with the inference.
For others, is it done till here? Should we proceed?

Ronak Makwana 1:05:30
Yes.

Tarun Jain 1:05:31
OK, so we'll start with context context relevance. So if you look at the input, what are we doing for context relevance? You have C and then you have Q, so context and then question and you have a prompt. So the prompt is very simple. You are defining LLM as a judge.
Where you are telling it to just based on the facts. So every single context that you have it is acting as a facts. So if you look at this line eval set up and you have an input variable called facts. So this facts is nothing but your relevant documents which is the context and the question is.
This is nothing but the user input and then what are you saying? This particular LLM? If in case it is there in the fact, just keep it as true. If it is not there, keep it as false. This is nothing but pass and fail. Along with this I also need a critique which is nothing but reasoning.
In a step-by-step manner which will provide the conclusion for it. So this line over here you can edit it As for your choice like what is your end goal, how your critique needs to be.

Tirth 1:06:38
Mhm.

Tarun Jain 1:06:39
At the end of the day, what you need is you just need true or false. Apart from true and false, how you need the critique to be that you will define it here this line or you can also define it in the criteria.
So this particular thing I've just copy pasted exact same it is how it is used in most of the frameworks. So ragas also have the same thing. If you see this teacher and student guiding system, even when I showed iterate and MRR it was teacher professor thing.

Tirth 1:07:08
Hmm.

Tarun Jain 1:07:10
So this is a standard template which is used in this frameworks.

Tirth 1:07:16
No.

Tarun Jain 1:07:17
And if you notice here when I'm doing this relevant context, what is the data type of context?

Tirth 1:07:26
Yes.

Tarun Jain 1:07:26
Data type of context.

Hardip Patel 1:07:27
Play stop song.

Tirth 1:07:29
List options.

Tarun Jain 1:07:29
Is this? But what I need to do when I pass that into the prompt, my data type should be string. So now this relevant document.
Is in string.
Data type.
Is STR and once you do that, what are the input variables we have? One is C and one more is Q and then you just have to pass that as a prompt. You have the system prompt. System prompt is nothing but your entire instruction and your user prompt is nothing but where you have your input variables.
So how many input variables we have? Just two relevant documents and question. So this is user prompt. So instead of eval setup, I'll keep it as user prompt. I'll copy this and I'll paste it here and then you can directly use the LLM LLM dot invoke, give the prompt and then generate the content.
So either you want to keep in markdown, we can keep in markdown. If not, you can keep structured responses.
So if I want structured parser, how will the syntax be?
So let's suppose I want to make this a structured response. So how will you define it?

Tirth 1:08:54
We'll first define the identity class, so we will have.

Tarun Jain 1:08:58
Correct. So from pygon tick import model then field.

Tirth 1:09:00
Bish model.

Tarun Jain 1:09:06
And then you have to define your schema. So this schema will be for context relevance.

Tirth 1:09:10

But.

And we need the output in three formats, three things, right? So we will have that accordingly. So we will.

Tarun Jain 1:09:21

Three or two?

Tirth 1:09:24

Oh, we just need the steps and.

What is the output that you're requesting?

Tarun Jain 1:09:27

So one will be critique.

And one more will be.

Tirth 1:09:33

Relevance. Relevance is boolean. It is boolean.

Tarun Jain 1:09:35

Relevance. So relevance. What is the data type of this will be both and this will be string.

Tirth 1:09:41

And.

String or list of string, yeah.

Tarun Jain 1:09:43

And that's it.

And then we can pass the description.

Tirth 1:09:51

Yeah.

Tarun Jain 1:09:53
In description.
And then what will we do here?
From Langchain.
4 dot.
Output parses import.
Pied and tick output parser and then.
What should we change in system prompt?

Tirth 1:10:23
We will need to add the format in which we need the output.

Tarun Jain 1:10:27
Format instructions.
And once you add format instruction, there are two ways you can edit it directly here.
If not, you can define it in the what is that called language agent. Sorry LCL language.
Hey, LCEL Lang Chain Expression Language. Sorry Lang Chain Expression Language.
Is this clear right? If in case we need it in Jason, this is the output format.

Tirth 1:10:58
Yes.
Right basically.

Tarun Jain 1:11:01
So this is for answer context elevance. Now what will you do for answer elevance? So here what we'll do is for context just pass C and for questions just pass Q.

Tirth 1:11:19
Yeah.

Tarun Jain 1:11:22
And here also what I'll do is I'll for question I'll pass Q for answer I'll make it here because we have already defined those variables.

Tirth 1:11:31
But.

Tarun Jain 1:11:34
And same goes for answer relevance. If it is true or false, you need to define for relevance and then how you need your critique to be. I'm just telling it to explain step by step. So whatever reasoning it is doing to make it true or false, I just need that manner.
But if in case we need any specific then we can change this line and with the criteria.

Tirth 1:11:56
M.

Tarun Jain 1:12:08
Now one question I'll ask here. So let's suppose I want to run for a different question. Don't you think this is a very tedious task? Like have to run this again? Like let's suppose I want to change this question.
And once I change this question, I will have my result answer result question. So don't you think I'm repeating this?

Hardip Patel 1:12:33
Look, yes.

Tarun Jain 1:12:33
So now let's suppose I want to change the question. I want to change the question as who is Virat Kohli?

Hardip Patel 1:12:36
OK.

Tirth 1:12:41
You have to change everything now.

Tarun Jain 1:12:44

So I have to run this user query then again this invoke. Once I run this invoke then only this is updated.
So now if I run the entire thing.

Tirth 1:12:51
Mm.

Tarun Jain 1:12:55
I just have to run this variable.
So now it should be false.
It is false so.
Can anyone suggest what is the best approach that we can take here and we have already discussed.

Tirth 1:13:12
Ticket from user input.
We can take it from user input now input.

Hardip Patel 1:13:19
Thank you.

Tarun Jain 1:13:21
OK, this one then.

Tirth 1:13:23
It becomes from input, so you know the user can enter the question himself and then we can make it as a whole function which you're passing it.

Hardip Patel 1:13:29
Can we can we add? Can we add it to bug flow?

Tarun Jain 1:13:32
OK, you mean, uh, this entire thing will?
Now one second, can you repeat? Let's suppose I enter a question, then what function will I run?

Tirth 1:13:37

Can become.

Yeah.

Mhm.

So you can make this whole thing function, you know, like a run function or the main function.

Tarun Jain 1:13:47

OK, you mean?

Got it. So it will be like you run def main.

Tirth 1:13:53

8.

Tarun Jain 1:13:54

So this this main will take the query.

And inside this query we'll have what will my chain response be, response or chain.

Then after I have the response or chain I will get the relevance of context.

Tirth 1:14:04

Right.

Tarun Jain 1:14:14

Which is context relevance.

I'm getting the query whatever the input variables are. Same goes for answer relevance. OK, this is one approach, but which one is the most optimized?

Tirth 1:14:24

In this, uh, yeah, Lengra.

Hardip Patel 1:14:26

Can we add it to work? We can add it to work.

Tarun Jain 1:14:28

Correct Lang graph. So what will we do here? So let's suppose how is our graph currently? How is the graph so?

So you have start, then you have search.

Then you have answer, then you have end. Now how will our graph needs to be like?

Tirth 1:14:50

We'll start. Yeah, we'll start.

Tarun Jain 1:14:51

So how should I modify this start then?

Hardip Patel 1:14:52

No, after answer there should be.

Tarun Jain 1:14:56

Then search then.

Tirth 1:14:56

Same search will be there. We'll also have the answer and then there will be the LLMS search for all the three-part, all the two-part, both parts will first verify.

Tarun Jain 1:14:59

Answer them.

But we have.

Hardip Patel 1:15:07

Yeah, but it we will end it and then in the.

Tarun Jain 1:15:11

So if you see here, what order is this?

So if you see this is going in sequential order and here I have two more function. So when I say function, typically this is a node now, correct? This is a node and this is also a node. So I have two nodes now.

Tirth 1:15:23

Correct.

Mhm.

Tarun Jain 1:15:29
So how will my ads be like? So total I have search.
I have answer, I have context relevance, I have answer relevance.

Tirth 1:15:41
Correct.

Tarun Jain 1:15:42
So I have total 4 nodes. How will the edge be like?

Hardip Patel 1:15:46
After answer we can end and uh uh in the parallel we can have the.

Tirth 1:15:47
So.
Context relevance.

Tarun Jain 1:15:54
Then answer relevance.
Then end. Should it be like this sequential order?

Tirth 1:16:00
Well, context relevance and answer relevance can be parallel, so you know once we have the answer.

Hardip Patel 1:16:02
Me.

Tarun Jain 1:16:05
Correct. So what we need to do it will be parallel. So if you see context relevance and answer relevance, whatever input variable these two functions need, it is already there in answer failure. So these two can go parallel.

Hardip Patel 1:16:05
Like.

Tirth 1:16:20
Right.

Tarun Jain 1:16:22
So now what we'll do is let's add this as a node and define directly inside a workflow.
Is it clear? So instead of being sequential, we'll make it parallel.

Tirth 1:16:32
Yep.

Tarun Jain 1:16:37
Till here is it done?

Tirth 1:16:40
Dish.

Tarun Jain 1:16:42
Langra.
Parallel node.
So how will this change like now function? I have to change this function.
What changes should I make?
I will copy this entire function OK and I will copy it here inside parallel node. What changes should I make here now?

Tirth 1:17:07
Good.
We first need to have the state that is the rack state that will hold the.

Tarun Jain 1:17:17
Right, so I will change this.

Hardip Patel 1:17:19
right a.

Tirth 1:17:21
Yeah.

Tarun Jain 1:17:21
And I have to define state and what is the shift?
State is rack state.

Tirth 1:17:29
Right.

Tarun Jain 1:17:37
Then.

Tirth 1:17:38
We'll have to define the rat state. Do you already have the rat state?

Tarun Jain 1:17:45
Yeah, is already there.
It is defined.

Hardip Patel 1:17:49
But we need to update the next statement.

Tirth 1:17:52
Right, we lost the head.

Tarun Jain 1:17:52
Correct. So what will you, what will you update? What two new things will be added here?

Tirth 1:17:57
The answer relevance, context relevance.

Tarun Jain 1:18:00
Answer relevance.

Hardip Patel 1:18:01
Content and.

Tarun Jain 1:18:04
This will be string then context relevance.
Will be string.
OK, we are up to the time. Let's complete this context 11 sensor relevance. This is state, state and what am I supposed to do here? This will be state of context.

Tirth 1:18:20
M.
No.
But.

Tarun Jain 1:18:27
And this will be state of.
Query. This is clear.

Tirth 1:18:35
Yeah.

Hardip Patel 1:18:36
Yes.

Tarun Jain 1:18:36
And now I'll just copy this answer relevance.
This will be state.
Of right state.

Date of.

Is it very?

And then you have state of answer.

And also one more thing what we have to change is we have to change this into dictionary.

So what will be your answer? This will be.

Tirth 1:19:13

the right state. We should return the right state at the end.

Tarun Jain 1:19:15

Contacts.

Context relevance.

And what are we supposed to do? Grade dot content?

So I'm updating the state.

Hardip Patel 1:19:28

Don't we need to? Don't we need to return the state itself?

Tarun Jain 1:19:31

Same thing? What am I supposed to do?

Tirth 1:19:35

Yeah, Douglas.

Tarun Jain 1:19:36

That also we can do. What we can do is you can. How do we do it?

Hardip Patel 1:19:38

Yeah.

Tirth 1:19:40

We can add this to the. We can add this to the right state and then we return the right state, yeah.

Tarun Jain 1:19:48
Correct.
Hello.

Hardip Patel 1:19:49
OK.

Tarun Jain 1:19:52
State of context relevance equals to.
Grade dot content then return state.

Hardip Patel 1:20:15
I can't.

Tarun Jain 1:20:22
And this will be return state. Is it clear? Still here? Everyone not clear?
This is the same thing we have done. We just change the input variables and then when we are returning we are making sure that we update our answer relevance and context relevance. So these two variables should be similar to what we have defined in the attributes.

Tirth 1:20:28
Yes.

Hardip Patel 1:20:34
Thank.

Tarun Jain 1:20:47
I'll run this. I will run this.
And now I'll run this again I have to define the workflow, so I will define workflow.
Equals to.
State.
Should we stay at the graph, Sir?
I will copy this.

I'll paste it here. So what is the first thing? You have workflow which is state graphs which is nothing but track state. We need to add node which is answer relevance, right search answer generation. We need to add two more nodes.
Two more notes.
Workflow dot add node.

Hardip Patel 1:21:37
Answer that.

TJ **Tarun Jain** 1:21:39
I'll use the same names.
Get context relevance comma get context relevance workflow dot add node.
Get answer relevance.
Get answer relevance till here everyone are clear. I'll remove this code cell.
So you define your state graph, then you add nodes. Now what you need to do is once you add node, you have to make sure that you create a empty node which is mainly used for joining. So I will define join node.

Hardip Patel 1:22:00
Yes.

1:22:01
Mhm.

Tirth 1:22:03
Mhm.

TJ **Tarun Jain** 1:22:17
And I will return a empty state.
Return a state and then what I will do is we have to define a parallel node.
Route to parallel, so I'll just keep it as parallel node.
So what are the two nodes we need in parallel?

Tirth 1:22:40
Get answer relevance and get context relevance.

Tarun Jain 1:22:40
You have to pay.
So one is whatever you use the names here, use the same names, context relevance and answer relevance.
This is rack state.
So one is to join the nodes and one more is to run the parallel nodes. So what are the parallel nodes we have? One is context relevance and one more is answer relevance. And then what you are supposed to do, just define the join node.
Workflow dot add node.
Which is nothing but your join node.
Join notes.
Till here, is it clear?

Tirth 1:23:33
Yes.

Hardip Patel 1:23:35
Yes.

Tarun Jain 1:23:35
So now whatever you have this as a parallel node, this is your edge and if you see this is already added in the node. So you don't have to define your parallel node when you add your nodes because these two things are already defined. You only need to define join node as empty Y because you have two parallel nodes.
So once you have those two parallel nodes, you have to join them. So that's the reason why after you have two parallel nodes, I'm defining the join nodes. So this will be added as your edge. Is this clear?

Hardip Patel 1:24:05
OK.

Tarun Jain 1:24:07
Cool. So now what you need to do is once you have your add node, the next thing is

we have to define our edge workflow dot add edge. So what is the starting point start?

Tirth 1:24:17
It's.

Tarun Jain 1:24:23
Which is nothing but search context.
Then workflow dot add edge.
Search context comma.
Answer generation. Now here is where you have to define a conditional node. So conditional node is another functionality of land graph which is typically used whenever you have human in a loop or you have a parallel node.
So you have something called as add conditional edges.
So I'll write a comment here. Add conditional.
Edge is mainly used for parallel node.
And human in a loop.
So when you add conditional node, first thing is you have to define your answer generation from where you have to create that. So I will just copy this answer generation.
And after answer generation, what is the node you need?
What is the note we are supposed to pass?

Tirth 1:25:35
The main node.

Hardip Patel 1:25:36
Join node.

Tarun Jain 1:25:38
The parallel nodes no join node is already added. So how you want to see after answer generation what is the condition you want to add. So the condition is after answer generation I have a parallel node which has context relevance and answer relevance. So you're adding that and then define the root of it.

Hardip Patel 1:25:42
OK, OK, OK.

Tarun Jain 1:25:59
Which is a dictionary. Sorry dictionary. So in conditional node we have to add dictionary here. Just pass this context relevance whatever you have added.
Where should this route to? This should route to get context relevance. I'm using the same variables.
So whatever answer relevance is there.
I'm defining it to the answer relevance. Is it clear what is happening here? So this parallel node, how many variables does it has? Get context relevance and get answer relevance. Now this two I've added and then I'm routing it to the same thing. I'm not changing any variables to avoid any confusion.
Hila, is it clear?

Hardip Patel 1:26:42
I am.

Tarun Jain 1:26:45
And once this is done, you just have to use this as a join node workflow dot added.
So whatever auto complete is there, that is correct.

Hardip Patel 1:26:48
Oh.

Tarun Jain 1:26:55
So you start with search. After search you go to answer generation. After answer generation you need to create a parallel node. So when you create a parallel node you need to have a condition which is your conditional edge. What is the starting point? The starting point is answer generation.
Then you have a parallel node. So this parallel node has two variables which is get context relevance and get answer relevance. For each of these two things you have the node already available and I'm not using any new variables, just avoid the confusion.

Once this is done, just combine it with your join nodes and then complete the end.

That's it.

Tirth 1:27:34

Can you? Can you? Yeah.

Tarun Jain 1:27:34

And then you just have to define your graph graph.

Equals to.

Workflow compile.

Then now let's print graph.

From ipython.

dot display import image.

The image of.

What was the function graph dot?

Hardip Patel 1:28:26

I guess you can just put in the graph. Uh, it will give the.

Tarun Jain 1:28:26

Get.

Now there is some mermaid error. Are you able to print this? How many of you are able to print it?

I'm getting mermaid.

Hardip Patel 1:28:37

I'm not following, but yeah.

Tarun Jain 1:28:41

It's an API error. Can anyone try this? Just copy this line.

Hardip Patel 1:28:42

Oh.

Tarun Jain 1:28:57

If I do get grass, I'm getting the grass.

Then draw mermaid PNG.

It's in the mermaid.

Let me copy this.

Hardip Patel 1:29:17

You can install the package Mermaid 5 now.

Tarun Jain 1:29:23

Let's just use Claude.

Yougraf.

So how this should look like is you have start then search answer. After answer you need to have join node. Join node will have parallel node which is context relevance and answer relevance at same time. Then you have end.

Are you able to see this?

Hardip Patel 1:29:54

Yes.

Tarun Jain 1:29:57

So if I do copy this and go to mermaid, if I paste it here.

So you have start, then answer generation, context, then answer generation, get context relevance, get answer relevance, then join nodes and then you have end. So this is how the mermaid will look like.

And I use the same code that I got here.

Is this clear how to create a parallel node?

Only this is the new thing. Remaining is same.

Oh, you understood the syntax. OK, so now what you can do is just print the same user query response equals to graph dot invoke.

Tirth 1:30:31

Yes, it is clear, yes.

Tarun Jain 1:30:46

Very.

User query.

And here I'll just print answer relevance.

OK, where am I defining user query?

Are we using any variable which is not defined state query state answer?

Hardip Patel 1:32:10

I think uh get context relevance needs to have a.

Tarun Jain 1:32:17

Which one?

Hardip Patel 1:32:18

But but it will it work right?

Tarun Jain 1:32:21

No, it's as a return statement. This also has return.

Hardip Patel 1:32:24

No, the great answer relevance does not have written type.

Tarun Jain 1:32:28

Oh.

This join notes also needs to. This is fine. Let me remove it.

Hardip Patel 1:32:46

Yeah.

It.

Tarun Jain 1:32:59

Search search context translation.

Did anyone get the result or is it the same error?

Hardip Patel 1:33:23

I am not a but don't join not need to return state because it is part of.

Tarun Jain 1:33:36
OK, let me check this error and we'll come back again.

Hardip Patel 1:33:44
OK.

Tarun Jain 1:33:46
OK, we are already time up as well. Till year everyone understood right the context relevance and answer relevance and also how the data data set should look like. So faithfulness is pending and after faithfulness we also have some of the metrics based device.

Hardip Patel 1:33:56
Yes.

Tarun Jain 1:34:03
Which is basically the precision and recall.
Which we'll probably discuss tomorrow. And once that is done, you also have output running the batches. Obviously you'll not have single user query, right? You'll have multiple user query. So using Ragas itself, you can also generate synthetic data set and probably they have the best prompt that they've used. So we'll use the same package which is Ragas.
And then generate synthetic data set. The only thing is only Open AI model are best suited to generate this. If you use Gemini after some time it will start giving you rate limits. But what we'll do is we'll use simple LLM which is Gemini 2.5 Pro, sorry Gemini 2.5 Flash.
And we will generate 10. But if anyone wants to generate more than 10, I'll also show you the example using Open AI embeddings and Open AI LLM. So only for this particular use case I'll be using Open AI. For remaining I'll only use them.
Only for generating data set. If anyone has Azure in the team then we'll also use Azure instead of open AI.
Yeah, this will cover tomorrow.

Tirth 1:35:14
OK, OK.

Hardip Patel 1:35:17
Thank you.

Tarun Jain 1:35:20
You can also check out the GitHub repo. I've updated the repo and you have the slides available as well.

Hardip Patel 1:35:24
OK.

Ajay Patel 1:35:29
OK.

Tarun Jain 1:35:33
Cool. Uh, any other questions?

Ajay Patel 1:35:34
Nope, not as of now.

Tarun Jain 1:35:38
OK, I will continue from this error tomorrow.

Ajay Patel 1:35:42
Thank you, Tarun. Thank you.

Tarun Jain 1:35:44
Yeah, thanks.

Ishan Chavda 1:35:44
Thank you.

Hardip Patel 1:35:44

Thank you.

◉ **Margi Varmora** stopped transcription