Python and AI Power-Up Program Offline Class-20250918_113502-Meeting Recording

September 18, 2025, 6:05AM
1h 13m 27s

Ajay Patel started transcription

TJ **Tarun Jain** 0:04
Important details, then probably we'll proceed. One important thing I wanted to share is so when you are creating any report generation thing, right? How long is the response?
is the response.
So let's suppose I'll just Scroll down. You ask a question, you're getting the response, right? Is it very short or is it too big?

Hardip Patel 0:27
I guess you know.

TJ **Tarun Jain** 0:29
OK, yeah, it's fine, right? Then mostly what will happen is in some of the use cases, whenever you want detailed responses, what you can do is you can have different styles. So how many of you have used Claude? Have you noticed this before? So if you see here in Claude when you click on this icons and once you click on this use style.
You have concise, you have explanatory, you have formal. So basically whenever you're building any applications right when it is related to that, most of the prompt templates will have concise. That is, it will answer in one or two liners.
Right. If you want to be detailed, what you can do is you can use a keyword called moderate. So this is medium and be detailed.
And if you need very large responses, let's suppose one or two paragraphs or at least two or three paragraphs, then what you can do is you can add a word called prolonged.
Prolonged or what did they use? Explanatory. So these are the three keywords you can use if in case you want to extend the output format. So you need the consent response. You can say answer.

Precisely.
And concise.
So this can be your last line of the prompt. If not, what you can do is you can just answer keep the answer moderate. Moderate is nothing but medium and if you need a very large response you can keep it as keep the response very prolonged.
So these are nothing but various styles. You have concise, you have moderate and you have prolonged. So this keywords you can keep it different as well. Concise is mainly used everywhere. Instead of moderate you can also keep it formal.
And then prolonged is nothing but explanatory, so you can refer to Claude. Is this clear if in case you need detailed responses?
Concise, formal, moderate and prolonged. All right, so just to revisit quickly on what we did last few weeks, it was vector databases. Let me open vector database.
OK, so there are three different approaches. One you can use one directly first create the client, which is first create the client and after you create the client you will have collection name.
And after you have collection name, the next thing is you need to save your data. So whenever you are saving the data, the first thing is you need to have your metadata and this metadata needs to go in payload and then you need to have main content which is your page content.
Which will be passed inside your vectors.
And then you will have ID till here everyone are clear metadata page content and ID.
Did you guys try this approach in the project that you're doing?

Hardip Patel 3:41
Yes.

TJ **Tarun Jain** 3:44
Cool, so that will and when you're doing inference you just have to use prefetch and if you're using hybrid search you will have both parts and as well as dense you can give certain limit. Once you give this limit in your query points you can set how much maximum documents you need. I hope till here.
No one has any doubts or you're not facing any errors in this approach.
Cool. And then we had filtering technique as well. Filter technique basically if in case you want any field name. So field name is nothing but if there is any key which is already existing in metadata and if user is providing the value, how can you filter it so

that whatever documents you're getting it is from the.
Particular source, right. So this is the second approach that we looked. First is hybrid search, then we had filtering and after we had filtering we had something called as reordering. So for reordering basically we use something called as MMR which is mainly like search technique for.
Diversity and as well as relevance. So if you keep it as zero, it is diverse. If you keep it as one, it is relevant. If you keep it as 0.5, it is both diverse.
And relevant. So this is nothing but your Lambda value in MMR. So always it is ideal to keep value to be 0.5.
All right. Tilia, no one has any doubts, right?
OK, so I'll open this particular book. So there is one book called EI Powered Search and in this EI powered search always if you have anything related to search right, whether it is knowledge graphs or what are the different search techniques we looked into TFIDF.

Hardip Patel 5:18
No.

TJ **Tarun Jain** 5:36
Cosine similarity beyond 25 and then also dense vectors, right? So just like that, if you want to know what are the different search techniques or if you want to improve the performance, there is this book called EI Power Search. I'll also share it this PDF and now if you look at the search engines, so far what we did was there are only three things.
The first thing is ingest content which is your indexing which we already did. You have a vector database, you bring in your own data and data plus embeddings. You're saving it in a database vector database which is indexing. We already completed first step.
And second step is it has to return the content matching based on the incoming queries, which is nothing but matching. This is what the inferencing is about. We used cosine similarity, dense vectors. If not, we use hybrid search to get the relevant documents. So if you see return content matching for the incoming queries.
So this was the second step which is nothing but search which is also completed and the last part which was pending is how do you reorder it or how do you sort based on the relevance. So this is where you have something called as re ranking.

So for re ranking again there are two different approaches which we will do today.
What you guys can do is you can open the rag notebook that we had.
Rag notebook where uh vector store was from Lang Chin. Do you guys remember this one?
So this quadrant vector store is coming from line chain.
Where did we import that?
Yeah, this one. So from lines in quadrant import quadrant vector store. I hope you have this notebook. If not, I'll share it again.

Hardip Patel 7:25
That.

Tarun Jain 7:36
So you guys understood. Now that we have retriever, we will only do the inference.
Remunking is applied, whatever.

Hardip Patel 7:44
Could just confirm. I don't have one doubt.

Tarun Jain 7:49
OK.

Hardip Patel 7:53
But you know, like I worked on the the same one.

Tarun Jain 7:53
Oh yeah.
Just give me one moment. I'll I I'll get my microphone. There is bit of disturbance here. Just one minute.

Hardip Patel 8:01
Yeah, it might be because of me.
No.
Yeah, it's.
No, no.

Tarun Jain 9:15
Yeah, we can repeat your question.

Hardip Patel 9:19
OK, audible now I tried to try to use.

Tarun Jain 9:22
Yeah.

Hardip Patel 9:38
After I get a lot of prompting and I'll do this.

Tarun Jain 9:39
Is it just me or is it? There is some background disturbance. I thought it was from my end.

Hardip Patel 9:43
Is there? OK, OK, let me now. How is it now?
Let me.
No. How is it now?

Tarun Jain 9:51
OK, I thought it was from my end.

Hardip Patel 9:53
OK. Uh, is it better and any better?
So just give me a second.

Tarun Jain 10:08
OK, meanwhile what we can do is let's open the same drag notebook where we had quadrant vector store and just add this new line. One is pip install sentence transformers and then you can also add pip install langing.
Line thing.
Go here. So there are two different approaches. One I'll show cross encoders on how

to use re-ranking.
Using open source model.
And then you have closed source.
And the only player when it comes to re ranking in close or Cisco year. Now where do we actually re ranking right? So when it comes to typical rag, rag in the sense if you are just working with question and answers during that time LLM are better now so you don't need re ranking if it is related to QA.
You don't need.
Re ranking because LLMS are much better. Previously LLMS you had something called as curse of not curse of dimensionality. You have a test called needle in the haystack. Needle in the haystack. Did I mention this word before?

Tirth 11:27
Yes, you did.

Tarun Jain 11:28
So basically needle in the haystack is a test technique. So let's suppose you have 2,000,000 context window for Gemini. After how much question are you losing that particular context? So what we usually do is let's suppose you have 10K first.
Then you have 50K, then you have next number. I'll take 100K then just like that you will take 150K. Now what you will do is from this 10K context you will take one query and you will see if LLM is able to generate the response or not within the 1st under 10.
Now if it is able to generate the response, that means in first 10K context your LLM is able to retain that particular knowledge. So now what you will do is you will place a second needle. Needle is nothing but your query.
Haystack is nothing but your context. So now you will place this needle in between 10K to 50K and you will see if you are able to get the response or not. So earlier when you had needle in the haystack technique, most of the models were not good enough after 64K.
Some models which were good enough was Mistral.
And it was open AI.
Right. But now in the recent days, most of the models are able to sustain knowledge up to 200 K as well. So now when do we have to use re ranking? In most of the cases, what I've seen is if you're using Q&A, this course doesn't improve whether it is

ranking or re ranking. That's the reason why I said you don't have to.
to do re-ranking when it comes to just personal and such chatbot, but when it comes to report generation.
Usually how much documents are you picking? You're taking your K value as seven or you're taking K value as 10. So this is a very huge number, right? And in most of the cases if you have a query, let's suppose you have query like.
It's something related to sales. Now this particular context is there in four context, four to five chunks, and what you are doing is you are keeping your K value to be 7. So in this case what will happen? You will have two chunks which have very low relevance score. So what re ranking does is basically coyere. Coyere will add a score which is relevant score.
For each chunk, whether it is relevant or not, and based on that LLM will pick the particular response. So this is very important, the relevance for which you come from re-ranking and you use re-ranking only when your K value is more than 7 and your K value is more than 7 only when you are working with.
Code generation this is 1 use case and 2nd use case is if you have multi document chatbot.
Report generation then multi document.
Rag. So now what is this multi document rag? Let's suppose you have your data in PDF.
You have it in PDF and there is also a document and there is also your data coming from Microsoft SharePoint.
And if you also have your data in Confluence, if your data is coming from multiple sources, during that time also you can add re-ranking. Re-ranking basically will add relevance for which will help the model while you're doing inference. So this relevance for is very important, right? And in most of the cases re-ranking is not.
Every time it's just for experimentation. If your scores are improved then it's fine. If scores are not improved, you can skip the reranking part, but for report generation sometimes it will be useful.
But again, this is optional.
I'll just paste those two things here. One is report generation.
And we basically use it for RFP. So I added report generation and second thing is for multi document.
Rag and you need to have a field called relevance 4.
As a metadata, so this scoring is coming from the V ranking model.

So reranking is a model you have to use. Same like embeddings, right? So far where are we using models? We are using it for LLMS, we are using it for embedding model.
And now we'll also use a model for re-ranking. Since this add additional cost, that's the reason why most of the people avoid adding re-ranking that pipeline, but you will see re-ranking in the advanced rack courses.
So I'll show you both the approaches on how you can use the closed source ones and also the open source. For open source you have cross encoders. If not, you can use something called as a flag re-rank.
As of now, you only have cross encoders and the.
It should be free than.
Lankshin.
Sorry, Flash time. Sorry, sorry.
This is flash, not flag. One is cross encoders and one more is flash redone.
Is this clear? Sentence transformers will be using it for open source, which is by again face.

Hardip Patel 17:31
It.

Tarun Jain 17:35
So whenever I say anything that is related to open source and if I'm referring to the model, typically it's coming from mugging face.
And for the closures directly, we'll use foyer. So let's install this.
We don't need this five PDF because we are just doing the inference.
Uh, you guys have this codebook, right? The line chain quarter one?

Tirth 18:23
Yes, we do. Yes, we do.

Tarun Jain 18:26
Well.
So the only two new lines that we have added is these two things, one is sentence transformers and one more is lines in for here remaining others were same.
And if it asks you to restart the session, don't do it as of now. Let every package be

installed, then we'll do it. And when are you supposed to use re-ranking? After the search is done. So first step is you index it. Second step is you use the search and once you're satisfied.
Or if you're not satisfied with your search queries, then you will proceed with the re-ranking. So for re-ranking you will use the same inference engine. The only thing is you will have a model to it which will add the relevance score. So the index is same, the collection name is same.
OK, this is done.

Tirth 19:26
I'm so answer the question from the chat. One was from Pardeep and one was from my side.

TJ **Tarun Jain** 19:29
Oh.
But not getting anything workable with 27P.
Oh, which model is this?

Hardip Patel 19:57
Same.

TJ **Tarun Jain** 20:00
OK, this is the embedding model right 270 M.

Hardip Patel 20:02
No, no, no. It's a check model, inspect model.

TJ **Tarun Jain** 20:11
Uh, which model 270M or twenty-seven billion?

Tirth 20:19
He's sleeping now.

TJ **Tarun Jain** 20:24
OK, when I was trying to generate a policy that was around 20,000 to 40,000 or more

token output, no LLM being such.
So.
OK, so for this policy generation like 20,000 tokens for one single subbedding.

Tirth 20:50
For single policies.
For one policy, like the policy was pretty big, we were making it IASO of type 2 compatible, but it would not give appropriate policy. If it was a small policy it would be detailed and good, but if it was a bigger policy.
It just, you know, only the sections with very small content within the important parts.

TJ **Tarun Jain** 21:17
OK, so uh, did you also pass any context or is it just plain LLM?

Tirth 21:22
Context. Lot of context actually.
I was passing a lot of content, yeah.

TJ **Tarun Jain** 21:29
Uh, so usually.
ohh Basically when it comes to report generation where you have the context, again we use two approaches basically. One is chunk level. So chunk level what we do is whatever input you have, you will divide that into separate chunk. That chunk will go as a context and then using that context, how
How much output are you supposed to generate? So this can be section wise or it can be detailed, but it's better to keep it section wise itself because that way it's more logical.

Tirth 22:01
But then in in section wise, uh, the major challenge that I faced in section wise was there was a lot of reputed content in each section.
So you know, like if it's it's not aware of what content was already there in different section, it would repeat itself.

Tarun Jain 22:20
OK, the repetition thing, right? Uh.

Tirth 22:23
Yes.
So I'm still working on it and I'm trying to give a summary like you only focus on this section, nothing more with prompting. But if there is a better way of doing it, I just wanted to know that.

Tarun Jain 22:34
Yeah, yeah, that that issue is there like the reputation part. I'm not sure how well the memory will handle because even if you use memory, it will just add your application will become too slow.

Tirth 22:48
Correct, correct.

Tarun Jain 22:50
OK, reputation thing. I'll come back. I'm not sure if you tried with.
One second. OK, I'll come back on the reputation part. I'm not able to recall.
Google Gemma 7-P This is embedding model. This is not an LLM.

Hardip Patel 23:26
I think there's a chat now.

Tarun Jain 23:37
Are you?
No, this is an embedding model.
So if you see Gemma 3.
Uh.

Hardip Patel 23:49
That is the instruct version of it.

Tarun Jain 23:49

27.

Here if you see the model files are different, can you see the model files? Now if you look at this model files, this is mainly for embeddings.

Hardip Patel 23:56

Yeah, I think you can answer us properly.

And it is very, very, very concerned.

Tarun Jain 24:25

Oh.

Hardip Patel 24:29

Do you see?

Tarun Jain 24:45

OK, so for this one, neither you'll have to fine tune. If not, 1 billion parameter models won't answer well.

Hardip Patel 24:57

One million does answer properly.

Tarun Jain 25:02

Did you try Jhemat Riyaan? They do have this origin parameter model.

Hardip Patel 25:06

No, I didn't.

I will, I guess.

Tarun Jain 25:15

Because it just felt like it was an embedding model.

Yes.

I don't think this will work well with rag.

Hardip Patel 25:49
Yeah, yeah, it doesn't. It doesn't work well.

Tarun Jain 25:53
Because here also if you see there are not much of the model files, but if you see for other model files will have multiple safe tensors.

Hardip Patel 26:00
OK.

Tarun Jain 26:03
So for 1 billion, so there should be more if you check with the other files.

Tirth 26:04
One model dot safe sensors.

Hardip Patel 26:07
Right.

Tarun Jain 26:14
Uh, where is it?
Gemma 3 if I ease a select this 4 billion IT.
Here you have multiple files for safety inserts because the model is trained well for the given data, but when it comes to 270 billion there are very limited data and it won't work well. So probably they built it only for fine tuning purpose.

Hardip Patel 26:36
OK.
OK, so if I find you right then.

Tarun Jain 26:45
If anyone have very less resources and if they want to fine tune the model then they can fine tune. So for that you can use anslug.

Hardip Patel 26:54
OK.

Tarun Jain 26:54
Probably they might have added new model inside them. So onslaught every time a new model is released, they'll add a script on how usually you can find on it and Google is partners with them.

Hardip Patel 27:04
Got it.
OK.

Tarun Jain 27:08
We can just search for.

Hardip Patel 27:12
I will.

Tarun Jain 27:17
OK.
So is this done? What I'll do is I'll come back on the report generation repetition issue.
Better solution for you.
OK, so let us run this cell because we need Google API key for the LLM inference.
And here what we typically need is we just need a fast embedded embeddings, then it quadrant vector store, then quadrant client and then this parameter. This is also not required so.
When do we use this? Basically distance and vector params.

Tirth 28:04
We use it for querying purposes and when we are creating embeddings.

Tarun Jain 28:09
For query purpose do we use this too?

Tirth 28:15
When we are creating, uh, you know the model.

Hardip Patel 28:18
Sparse vectors V of distance distance for sparse vectors.

Tirth 28:21
It was.

Tarun Jain 28:26
Distance. No distance is for dense vectors, not sparse.

Tirth 28:26
1.

Hardip Patel 28:28
Probably.
OK. Sorry.

Tirth 28:31
Distance is for a dense lecture.

Tarun Jain 28:33
So you use distance dot you use cosine.
So this will come in dense.
So for sparks you will have beyond 25 or you can use TFRDF.

Hardip Patel 28:49
Yeah.

Tarun Jain 28:52
So basically we use distance and vector params when you are defining or initializing.
The collection name.

Which is nothing but creating collection.
So do we need this particular liner when you're doing inference?

Hardip Patel 29:11
No thank you.

Tarun Jain 29:13
No, so we don't need this line as well. Is this clear? We just need these three lines which is fast embed embeddings, then quadrant vector store and client. Why client? Client is just to initialize where your current data is saved right or your collection name is saved somewhere whether it is on quadrant vector store.
On cloud or whether it is local host or it is in memory which is local memory since we saved our data inside of cloud. So what we can do is we can define the client. I don't have to run this URLs because data is already saved.
So I will avoid this thing. I will also avoid something embeddings I will done where we are using Gina embeddings.
And here you have URL and API for quadrant. So I'll just run this and the collection name is Web pages.
So you guys understood right? We are just doing the inference part. So for inference part we don't need documents. That's the reason why I didn't run the document part.
So if you see documents were skipped and the chunking part was also skipped.
If you are running it on VS Code, in VS Code you already have your inference script.
Is this clear?

Hardip Patel 30:37
Yeah.

Tarun Jain 30:37
So we are just running the inference. If anyone has done it on VS code, we can just reuse that particular script and here we are using this quadrant vector store. So for quadrant vector store we need client, we need collection name and we need embedding. So these are the three things which are very important when you define a vector store.
And if you want to add the data, you can just use add documents and add data. This is just one time and that's the reason why it's commented.

I will just run this part client equals to client, collection equals to collection, embeddings equals to embeddings and user queries. How do I contact Atyantik and K value? What I will do is instead of three I will make it 7.
And I'll include the prompt as well framework use. So just make sure K value are giving high value instead of very small because if you keep 3K equals to three.

Hardip Patel 31:28
Outlook.
Hey.

Tarun Jain 31:39
There is no meaning in doing re-ranking because LLM will definitely pick better response if your K is just three, but if your K is 7 or if your K is 10 and out of those 10 if you want to pick the right relevant chunks.

Hardip Patel 31:40
Yes.

Tarun Jain 31:54
That's the reason we're we're using the re ranking.
Is it done?
Let me know if you got the response when you do retriever dot invoke and if you ask any query you should get the retrieved chunks. So if I just do query and if I want to check the length of this.
What will be the length of this one?
What will be the length?

Hardip Patel 32:35
The 7th.

Tarun Jain 32:38
It is 7, so you can just come from till here if you have done.
We just have to define the DB, the user query and if you normally want to check it, you can use the search which is maximum marginal relevance. Then what we are supposed to do is when we are doing rag, the first step is retriever and retriever is

nothing but whatever.
Variable you have used here. Just define it as as retriever search type is MMR then search keyword arguments K equals to 7.
Till here is this done? We are just repeating things from the previous rag node that we had.
And below that what you can do now is you can just add a sub wedding called reranking and below reranking if you see I've created cross encoders which is the slowest approach. But here what are we doing? We are using open source.
Reranking models.
Which is from mugging phrase.
And then you have coir. Coir basically is for your closed source.
Reranking model.
So here what you can do is above cross encoders you can define one line from line chain dot retrievers you have to get contextual.
Contextual compression retriever. So this is common for both open source and as well as closed source. So what you're trying to do is let's suppose you have 7.
Chunks that was retrieved.
From the base retrievers.
So now what are we trying to do here? We are trying to create a new retriever. That new retriever is re ranking retriever, right? So when you're defining re ranking retriever, you need to have a base. So that base is already defined which is your vector database. Vector database dot as retriever you have seven. Now what I'm trying to do is I want.
I want to get the relevant score and that relevant score I need only top four, so I will define K to be Fourier.
So now this K will be the top values which are most relevant to the given user query. This is where re ranking is applied. So this approach of compressing your 7 chunks to 4 chunks, it's called contextual compression retriever which is basically for re ranking. and here we have to define the model. So the model is from line chain dot Retriever dot.
Document compressors. You have cross importers here.
Rahul Sankhoda I Rankar.
And then the model is from mugging face, so from line chain.
Community dot.
Ross encoders.

So the model is coming from Uggingface and then you're just reloading that into the retriever which is provided by line chain. So these are just the three lines that we have used so far.
Is it done?
So this will be common for both Coier and as well as cross encoders.
I'll just define model equals to. I'll copy this.
Has to be VGE.
Re Lankers.
I will copy this model.
And yeah.
Model name is equal to B.
So first thing is we'll define the model name. What this will try to do is it will download the model locally which will take some time. If in case you switch to GPUs it will be bit faster but we forgot to switch into GPU but that is fine all these models.
Are capable to run on CPU as well.
We just copy the sign and send it.
So this is a common file. Whether you use any embedding model or LLM file, you will see safe tensors common every single time and then you will also find tokenizers and I hope you remember what tokenizer is and you recall this keyword.
Can anyone tell me what is the full form of this?
Hello.

RamKrishna Bhatt 39:27
I think something related to bite or.

Hardip Patel 39:29
Yeah, database noise.

Tirth 39:31
Hi there.

Tarun Jain 39:31
Bye.
So this is basically our tokenizer. So whatever open AI uses right? So we tested this open AI tokenizer here if I just write open AI. So open AI is basically two different

tokens. So how did we get these tokens using VP? So whatever encoding models you're using which is the.
Cross encoder even that has a model and tokenization which they're using. So the model name is BG Reranker, but the tokenizer says BP which is your byte pair encoding similar to what open AI is using. Is this clear the tokenizer part and as well as the model?
So this model is VGE re-ranker and the tokenizer is nothing but the BP which you just train based on the given data that you have.
Let me know if you are able to download this.
And then we we will define.

Ronak Makwana 40:28
Yeah, it is in progress.

Tarun Jain 40:31
This will take time. It should take at least one minute. Then you will just define the compressor. The compressor is nothing but your re-ranker.

Ronak Makwana 40:35
OK.

Tarun Jain 40:41
So here I'll just define cross encoder re-ranker and define the model. Model is nothing but this hugging face model. And then what is the top K value unit? Here top K is nothing but top N top N I'll just copy this and I'll make it 4.
And once the compressor is defined, we have our second retriever. I will keep it as retriever 2 equals to.
Copy this contextual compression retriever and here you need to just define 2 variables. The first variable is which is your compressor. Compressor is nothing but your re anchor.
Compressor. I'll keep this as compressor 1.
Compressor one and then you need base retriever.
Base retriever is nothing but the retriever which you are getting from vector database.
That's it. The first thing is we define the retriever where we have to combine our

reranking technique and now you need the model. So the model we are using from open source. Whenever I mention open source blindly the first keyword will be hugging face. So from hugging face what I need is I need to pick this BTE model which is my model.
And once I have the model, I have to reuse this model for re-ranking, which is our compressor and it's called cross encoder reranker. Define the model name, pass your key value and then just define the retriever. So when you define retriever, we need two things. One is your re-ranking and one more is your base retriever. That's it.
And now how do we get the relevant documents? What is the function?
Vrang documents equal to. What's the function that we use here? Just invoke.
So it is defined retriever to dot invoke and user query.
And now if I do the length of V rank doc.
It should be 4.
And this is slow.
Now this is 4 and the format is same. Here you'll have page content.
dot 0 dot page content.
And you also have metadata.
The only issue with cross encoders it it doesn't have the relevance for in metadata, but typically it does use this relevance for to have all the four relevant documents which it will sort. But what we need is when we are passing this inside the payload.
I need to have the relevance score of free ranking which is not basically added by the retriever from cross encoders. This is where many people use Coyar and Coyar is also directly supported in AWS Bedrock and also Azure. So if anyone is using AWS Bedrock and Azure, we can use Coyar directly from there for very less.
Or.
And it's also fast. It's it takes hardly 0.1 seconds or 0.2 seconds if you're using coir.
But if you're using cross encoders, you'll have to hose this because this is a model.
But if there is any requirement where you have to use everything open source, so during that time you use hugging phase cross encoders.
Let's add the texture.
If it is open source.
If it is closer.
So which LLM will you use here if it is open source for? Let's suppose the text tag is relevant to rag.

Which are the top two LLMS you will use if your use case is something relevant to you have to use only open source text tag. So what?

Tirth 45:16
Yeah, Mike.
Demo.

Tarun Jain 45:22
Not Gemma.
I wrote three names when we discussed the ugging phase.

45:29
Uh, Gina.

Tarun Jain 45:30
When we influence Ravikas about three names.
Can you come here?

Ronak Makwana 45:34
DM 20 songs.

Tarun Jain 45:37
No LLM. I'm talking about LLMS, large language models.

Tirth 45:42
Told about Mixture AI, right? Mixture AI, Mixture AI is Mixture AI, yeah.

Tarun Jain 45:43
Zena is not related then Kumar.
There are two more IDFD names.
Does anyone recall?

Tirth 45:57
When you said what it is Chinese model.

Tarun Jain 46:02

Correct when, when and.

Tirth 46:04

Yeah.

Tarun Jain 46:09

That's me.

Tirth 46:09

Uh, Uh is open source, right?

Tarun Jain 46:12

Right. Yeah. Deepshik.

So.

This is on the LLM side. Now coming to the embeddings.

Tirth 46:22

In embeddings, you said Gina.

Tarun Jain 46:25

Uh Gina, where are we using it from? We are directly using it from fast number or. HF is nothing but hugging face. It's very short form to use it. Now when it comes to re-ranking, what are the two approaches?

Tirth 46:42

Put me around the one that you need.

Tarun Jain 46:45

No Courier will come in closed source. So here what I'm trying to do is if you guys have a tech stack where the requirement is you should not use any closed source LLM so that all your confidential data is safe. That is when I'll be using Mistel as my LLM, fast embed as my embeddings and cross encoder as my.

Tirth 46:49
Oh, open.
Hello.

Tarun Jain 47:04
Uh, what do you call re-ranking? So here your entire text stack is local.
No API keys involved.
Completely local.

Tirth 47:19
It will still be performant, right?

Tarun Jain 47:21
Oh, what happened?

Tirth 47:24
It will still be performant.

Tarun Jain 47:26
Yeah, yeah, correct. So that's the reason I wrote Mistel. If not, I could have used Gemma. So Mistel is good.

Hardip Patel 47:33
Ha ha.

Tarun Jain 47:33
Which is on most of the closer cell elements and then in embeddings also you can use Xena. I'll let this specific Xena and there is also someone called as Numi.

Tirth 47:34
OK.

Tarun Jain 47:46
So these are good players when it comes to embedding models and Google also

have good embedding model which they recently released, but I have not benchmarked it, so I'm not writing it.
And for reranking, one is cross encoder which is again coming from hugging face.
And one more technique is flash rerun.
And now when it comes to flow source, if you if in case you are free to use any API keys, then for LLM you can go with open AI 4O and 4O1 are good.
4.1 and then you have Gemini 2.5 Pro.
What about embeddings?
I need this specific embedding model name we have discussed.

Tirth 48:38
Open AI embeddings as well.

Hardip Patel 48:40
Next time getting small and large.

Tarun Jain 48:41
What is that temporary name?

Tirth 48:42
We have open AI embeddings as well that was large, next large, something something to do with that.

Tarun Jain 48:51
Text and a text small and text large.

Hardip Patel 48:53
OK.

Tarun Jain 48:54
So you can use open AI. It's very cheap compared to the other elements, I mean embedding model then for re ranking.

Tirth 49:04
Then go here as we discuss.

Hardip Patel 49:06
Oh yeah.

Tarun Jain 49:08
So is this clear?
We just take this and I will add it in the tag.

Tirth 49:18
Yeah, that's a good question. Like you know we have VG large ENB 1.5. It came on the MDE board which said it is better than Gina embedding. So do you have any experience with VG large or Gina is still better?

Tarun Jain 49:32
So the major, so the only reason why we usually don't prefer BGE in embedding is sometimes we are not sure of what you call the data that they have used. BGE is what you call Chinese embedding models, so they actually overfit.
Their models overfit in the sense whatever is used in leaderboard, right? The data set, they use that data set to train their model. So you got the point here.

Tirth 50:01
That's the easy.

Tarun Jain 50:03
So you have overfitting concept, overfitting in the sense you train the model and then you test the model on some data set. So whatever you use for testing, it is not supposed to be used for training. So Chinese people, they merge both of them and then train the model.
So this is overfitting. So most of the BG models which is from Alibaba. So BG is from Alibaba Cloud.

Tirth 50:24
That's fine.

Tarun Jain 50:32

And you can trust them, but again, it's not that much of useful. Not many people use that in embedding models when it is from Alibaba cloud.
Because most of the tokens are associated in Chinese. You love Chinese tokens as well. It's not pure English tokens.

Tirth 50:50
Yes, Sir.

Tarun Jain 50:57
But here you have no other choice when it comes to what you call reranking. So if you look at reranking in cross encoders, there are very few models.
Can you see this for BE Re ranker base?
Hello.

Tirth 51:15
Yes, yes.

Tarun Jain 51:16
If I check cross and border models.
Most of them are only Chinese when it comes to encoders.

Tirth 51:29
This Zeera is also built on top of Kuen and Kuen is Chinese, right?

Tarun Jain 51:33
No, there are different. So Jina also have their own base models. So if you look at Jina's version one, version two and version three, those are their base models. So if they're using Quen, those are about that.

Tirth 51:44
OK, version 4 is already.

Tarun Jain 51:49
Jino Version 4.

Tirth 51:49
OK.
And we are using version.

Tarun Jain 51:55
We are using version 2.

Tirth 51:57
Show.

Tarun Jain 52:06
Zena Embeddings 4 They're using it on base model. Then there is Zena Embeddings V3. These are their base models.
So here they're not mentioned it. So till V3 they have their own base models.
One is Jina and one more is Naomi.
So uh, do you remember I told these two keywords? One is iterate and MRR.
Evaluate. So you usually use this tool to evaluate your embeddings as well. So understanding iterate and MRR, I'll send you one Pallab notebook. What you can do is you can just change the model name and you can run the rerun the script. So once you rerun the script you'll get iterate and MRR.
Whichever model has both more, you can use that model and I'm pretty sure Zena and Nomik will have good scores.
I'll send you this collab notebook. You just have to change the model name.
Is this clear? Iterate and MRR. So MRR here is nothing but mean reciprocal ratio which is basically used to evaluate your retrievers, which I guess Hardip was experimenting some time before.

Tirth 53:32
OK.

Tarun Jain 53:41
Uh, everyone got the results here.
Uh, anyone is still pending?
OK, so let's proceed with the Coier one. This will be the last part when it comes to

Lantern. After that we'll just have eval. This one is already done. We have already installed Coier, so I will just comment this out. And since we are using closed source, every time we are using closed source before surface we need to insert the.
API key. So you have the API key is for your API key. So let me let me run this first. Then what you can do is you can rerun it from your end as well. I'll will give you my API key.
This is the Courier API key. What you can do is you can just define OS dot environment Courier under score API under score key and then you can directly paste that key. And now the first step is same. The first step is you need to have compressor retriever which is already defined. Now what you need to do is.
You don't need to separate things because here if you see your model is coming from cross encoders which is from mugging face and then you're reusing it for reranking. But when it comes to coier you can directly use the model.
So you just have to combine both of this, combine both of this and then you have from.
Linechain foyer import foyer rerun.
So this Courier rerank itself has both model and as well as the retriever. So I'll just define the compressor 2 directly. So earlier if you see when you define compressor, what are the two parameters you need?
What are the two parameters we need for a compressor?
One is model and one more is the top N value here. Where are you getting the model? You are getting it from mugging phase. Here you can directly define the model. So just define compressor equals to coyrerank.
And then you have model. Copy this model equals to.
It gave an auto correct. It is version 2, but I guess they have version 3 now V rank Courier V3.
This is the model rerank English V3.
I'll just copy this.
You don't have to define four years. Rerun English V30.
And then just define retriever. That's it. Retriever 3 equals to contextual compression retriever. Your base compressor is your compressor 2.
Here it was compressor one, it is compressor 2 and your base retriever is retriever.
Now this is very fast.
Rerank dot 3 equal to.
Retriever 3 dot invoke user query.

I hope you understood the syntax. The syntax is very much same. The first thing is you have to define compressor. In compressor you have two parameters, one is the model and one more is top pen which is top pen equals to four. And once you define your compressor you need to define your retriever.
So in order to define retriever you have contextual compressor retriever where you have to pass your re ranker which is nothing but compressor and the base retriever which is your vector database. So this is nothing but your DB vector DB which has top K to be 7.
Now if you are wondering why they have kept it as top N, that depends on the framework providers. Typically you can call it as top K itself, but as per how they define their function, they have kept that function as top N right? Which doesn't matter, you can just take this syntax.
If you just come here, you will see that parameter.
Top end I hope you observed it. One is you have client. Client is already defined inside coy re rank and then you have top end. By default it is 3 which we are making it as four and then you have model which is nothing but re rank English V3.
And API key it is saved in environment variable. If you see here you have an or operator either you can define it as a keyword variable which is for your API key equal to you can add your key. If not what you can do is you can save it inside for your API key.
So this is how you define functions when you're creating your own library.
And now what I'll do is I'll just run this cell.
And rerank docs 3 zeroth index dot metadata.
This score is very important, relevant score which will be passed as a context and if it is less than 65%, don't even consider that sum.
You should have this parameter. Just cross check if you have relevant score.
Oh, and all of you don't run it at same time. There might be chances that same API key, different IP address they might block.
How many of you are able to run it?

Hardip Patel 1:00:24
It will take some time for me. I'm using separate.

Tarun Jain 1:00:24
Oh, everyone got the results.

Oh, OK. OK.
But you understood the syntax, right? It's the same thing. Define compressor retriever, then re ranker and the base retriever.
OK, I'll also be sharing the updated slides. So what I'll do is I'll create a new slide here duplicate.
And I'll attach the screenshot.
And whatever we have done so far is we started with base.
Traditional drag.
Then we looked at binary quantization. So binary quantization, when are we defining this? We are defining this when you are creating a collection name.
So this is for fast rag, faster rag performance.
With memory utilization with.
Memory utilization.
And then what did we use? We use hybrid search.
So for ivory search you have dense embeddings which is your dense vectors and then you have sparse. So for sparse we use VM25 similar to what Uber are using and then today we have looked into pre-ranking.
So this is the beginner one which is traditional drag where you're not using any binary quantization, no hybrid search. So whatever you did so far in this particular collab notebook, it was traditional drag. After traditional drag, we just added re ranking.
So if I come back, one is base, then binary quantization for faster drag performance. Then you have hybrid search, then you have re-ranking. So re-ranking can be applied anywhere. It can be applied after traditional drag. It can also be applied after binary quantization or also after hybrid search. So this doesn't matter. This can be applied. Once you have your search parameters right, so these are search parameters.
What do you spelling?
I just make it beyond 25. So anyone has any questions in this four approaches based traditional drag, binary quantization, then hybrid search and then re ranking?
So with this we completed line chain. I'll just add it in new slide.
And now comes to the research topic. If in case anyone is interested to learn the next steps, then the first thing is you can learn something called as self reg. Now why self reg really important? Most of the times your retriever might not give you the right results, so you need to improvise your retriever.
If you want to self improvise it, you have a framework for that where you can just use

self drag which is 1 and then there is also called as late interaction drag. This is mainly used for multimodals, multimodal drag which we will have one more example for multimodal drag.
So during that time you can use late interaction drag. So here you have something called as cold but.
So let me tell you what this is about. We have dense vectors.
We have sparse vectors.
So for sparse we have BM25. For dense it is common all the embeddings models that you have it is dense vectors. Then now you have two research areas, one is late interaction.
So for late interaction you have something called as cold bird. Just like VM25, it is an algorithm. Even cold bird is an algorithm and the 2nd is you have.
Mini spurs.
Which is also called as sparse neural. Here you have something called as coil.
And mini coin.
And in your information this is very simple how you define sparse embeddings. Just like that you have to define late interactions and mini sparse and if in case you get stuck I do have blog articles on it.
So you can just search for advanced retrieval and evaluation. So when you click on this you will see minicoil. So minicoil is nothing but this algorithm.
Where you're getting the relevance score at the end. At the end of the day, you just need better relevance score. That's the only purpose of RAG, right? So whatever research is going on, they mainly focus on how you can improvise the relevance score. And if you see her, can you see this word called BERT?
This BERT is everywhere. Even in late interaction you have BERT, even in sparse you have BERT.
So this techniques, does it guarantee the performance of rag? It doesn't guarantee.
So this may work or may not work.
May work in some use cases.
May work in some use cases.
And in some use cases it might be flop. That's the reason why not many people use a slate interaction in every approaches. So hybrid search is the safest option when it comes to that.
I'll just add it as safest.
So you have self reg, then you have late interaction. Late interaction is similar to

dense sparse, but the only thing is model is different. Then you have sparse neural. Again the model is different, but the way you define it is same. So if I show you the code.
So do we know this step web-based loader?
This line is same then you have chunking so far same code and now if you see here you have dense embeddings. So for dense embeddings I'm using a model which is GTE large and for sparse embeddings what am I using?

Ronak Makwana 1:07:36
This.

Tarun Jain 1:07:47
I'm using mini coil and mini coil. I'm just defining quadrant mini coil V1 right? And now if you see I'm defining the client till here we are still doing the same thing and now if you look at what you call.
The create collection, create collection. Also you have vectors configuration, you have sparse vector configuration. So what are we trying to do? When you're defining this mini sparse, you have to use the same sparse vector store, but the model is different. Instead of BM 25 you are using foil and mini foil.
Is this clear? You're defining minicoil and then you index your data. So when you index your data you are using points, same 3 keywords, one is ID, vector and after vector you are using payload. In payload if you want to add your metadata you can add it and then observe.
This is the same code that we have done in the previous session as well and then prefetch prefetch. You are defining your NLP at the E lab, then mini coil and once you have mini coil you are doing the query point and same steps will be repeated for late interaction as well, but for late interaction this will be different.
So you can just search for late interaction.
So if you see the first keyword, late interaction models, you will see cold, cold work and quadrant is a vector database. Deviate is also a vector database. Then if you see Gina, this is there everywhere Gina.
Then elastic is also vector search. Most of the vector search or vector database you have will have these things.
And these are advanced techniques which you can explore and it's very simple. It's not that much of complex as well. So self RAD, late interaction and last is you have

something called a CRAD.
So CRAG is some kind of controllability that you are providing to self rag. So first you have to learn self rag and then you see rag.
So yeah, pretty much that's it. So what we will be doing is since most of you have Postgres account, I'll show you one session on how you can use Postgres SQL as a vector database.
So this is the only thing that we are currently left in Lancen, but apart from that we are pretty much covered most of the details that we need to cover. So I'll also take the screenshot of this and I will add it in the slides.
And.
Advanced.
Anyone has any doubts so far in lunching?
I'll come back to that repetition part of which Tirsat mentioned.
Copy and yeah.
Oh, any other questions? Anyone? Do try to share this across the last two slides because those last two slides are very important.
Any questions in RAG?

Mitesh Rathod 1:11:52
No, no.

Tarun Jain 1:11:52
Cool. So do let me know once you have this particular program RFP or if you have any issues in RFP thing you can let me know and if you have already completed you can send me an e-mail of the source code or if you have deployed it on stream it even that works.
So you can just send me the link, I'll check it out and I'll provide the feedback.
OK.

Mitesh Rathod 1:12:19
For me.

Tarun Jain 1:12:21
Oh, what happened?
All right, so on the RAG and the line chain part, this is it. Probably we'll go with the

vals part and then we'll also try to see Lama index, which is one of the similar library to Lang chain and once that is done, the only left.
Part is the agents part. So for agents again we'll do in person sessions. So I'll be coming back to Vadodara on October 3rd. So hardly we need four or five more sessions for agents and then agents will also be done.
Cool. Oh, that's it, right?

Ronak Makwana 1:13:06
Yeah.
Yep.

Tarun Jain 1:13:11
OK, let me know once you have the project ready, you can send me the e-mail.

Ayush Makwana 1:13:16
Thank you.

Ronak Makwana 1:13:16
Yeah, sure.

Tarun Jain 1:13:18
Hello.

Mitesh Rathod 1:13:18
OK, sure.

Tarun Jain 1:13:20
Thank you everyone. Thanks.

Ronak Makwana 1:13:22
Thank you you.

RamKrishna Bhatt 1:13:22
Thank you.

◉ **Ishan Chavda** stopped transcription