Creating, Curating, and Cleaning Data for LLMs

54 MIN · YOUTUBE · HTTPS://WWW.YOUTUBE.COM/WATCH?V=HEGAEI7K0ZE
https://www.youtube.com/watch?v=HEGaei7k0zE

SUMMARY

The session focuses on strategies for creating, curating, and cleaning data for fine-tuning large language models (LLMs). Daniel vanreen and David from Argilla share insights on finding existing datasets, adapting them, and generating synthetic data, emphasizing the importance of data quality and the iterative process of refining datasets.

- *Existing datasets can be found on platforms like Hugging Face, with a variety of community-contributed options available.*
- *Adapting classic NLP datasets for LLM fine-tuning can be beneficial, especially if organizations already possess relevant data.*
- *Preference datasets can be created using user feedback, either through direct ratings or inferred from user interactions.*
- *Synthetic data generation is a powerful tool for building datasets, allowing for the creation of diverse training examples without extensive manual effort.*
- *The structure of datasets for supervised fine-tuning often follows a question-answer format, with variations for reinforcement learning approaches.*
- *Data quality is crucial; better data leads to improved model performance, and iterative processes can enhance dataset quality over time.*
- *Tools and frameworks like Distill Label and various annotation platforms can streamline the data generation and evaluation process.*
- *Collaboration and sharing within the community can foster better dataset creation and refinement practices.*

So the plan in this session is to kind of do a little bit of a high level overview focusing on this topic of creating curating and cleaning data so I think this is already been discussed quite a lot in the course and I think it's come up various points so there's probably some stuff that will say that you're be familiar with but the idea is to hopefully give you some ideas for how to approach building data

sets for fine-tuning large language models so I'll just quickly introduce myself and then let David introduce himself and then we can kind of dive in so my name is Daniel vanreen I work as a machine learning librarian at hugging face I can talk more about what that means at the end of the session if anyone's interested but my background as the kind of name implies is very much in libraries so I

kind of fell into machine learning and large language models in the same way probably a lot of other people did via the fast a course that I took couple of years ago now but I think the library background is kind of nice for this topic because one thing that you will do a lot in libraries is look at lots and lots of data and think about how to organize data and how to to structure things in a systematic way

and I think that is a big part of working with data in the context of large language model so I think those kind of skills can be quite useful even though they're a little bit outside of what you'd expect for people work in this domain David do you want to do a quick intro and then kind of dive in sure so I'm David I'm doing ML and and Def at Argilla and Argilla is this data collaboration platform for AI

engineers and domain experts for the ones that aren't familiar with it and yeah I kind of started off with with NLP and these kind of things by by initially my masters and then working in private intelligence working on knowledge CLS and also yeah custom domain n models and that's also I thing where data quality and model quality is very very important so yeah that's kind of my my background and I will be covering the the synthetic

data part and hopefully give some some pointers on that for you guys okay that sounds good so I'll I'll jump in then so basically the the kind of structure of this presentation is to start from like the ideal case in which you might already find some existing data and then work to the probably more realistic case that you're going to have to build some of your own data for fine tuning if you're not working in a topic that's already

kind of well worked on so I just talk very quick about some ways in which you might find existing data so obviously I'm going to pitch hugging face as a good place to to look for data sets and I think that the question is a little bit what kind of data sets you need to look for so thing with hugging face there's quite a diversity of data sets that are shared on the Hub but a lot of the ones that

will be trending and very visible tend to be focused on like a very kind of specific type of use case so an example of that here is this fine web data set which probably quite a lot of youve seen through the rounds on social media so this is a kind of data set focused at pre-training large language models so those it's like a very interesting data set it's probably not something you're going to find particularly useful unless you're

actually training base models yourself the other thing that I would say is that there are a lot of research data sets that are associated with papers that can be really interesting but I think there's also a growing number of community contributed data sets that are made by people doing slightly more kind of bespoke and sometimes weird stuff but those data sets can actually be really valuable both for using directly but also for getting a bit of a better sense of how

people are approaching building data sets so one of the the ways I would suggest if you're not already familiar with finding data sets on the Hub is to use this kind of tags feature so I think that's not super well used at the moment but there's kind of growing usage of it and particularly for things like DPO data sets it can be a really useful way of finding data sets that kind of match a specific format and I'll talk a

little bit more about DPO in a little while there is also this full teex search so if people have done a kind of bad job of naming their d set is can often be quite useful way of finding data sets because it's looking in the the full data set C to to find things and then once you found the data set ideally someone would have documented it really well and explained exactly what it's for and what the

limitations are and how it was made but that often doesn't happen but one thing that I think is really nice with the Hub is that you have this data sets viewer that gives you a kind of preview of the data set and I think that can be a really good way of doing this kind of vibe check on whether a data set is going to be useful for your application or not so there's a kind of example

of what that might look like here so there's this data set for doing function calling and as you can see it's like very badly documented there's no information needed I'm not trying to pick out on this person but it's just quite common that people leave that documentation till later and one of the things that you get in this preview some kind of metadata about the data itself and in this case you have this conversation which is this kind of chat ml format

a lot of people are familiar with where you have basically a list of dictionaries and one of the things that might be interesting for you to know is like how long those conversations are and what the distribution of those looks like depending on what kind of domain you're working in you might expect the kind of chats be very short or you might expect to have longer chats and then having a sense of what this looks like can be

quite useful but the other thing that you can then start to do is to dive into okay like what are the actual conversations in there so you can look at actual example Rose and one of the things I noticed with this data set even though some of the chats messages are quite long some of the responses or the the kind of final turns are just basically the person being like oh thank you and then the model being oh oh

you're very welcome so probably that kind of response is not actually that useful so if a lot of the longer messages are like that then maybe you can't kind of see this as a a good data set for those kind of long conversation yeah fine tuning use cases so I said that was probably the ideal scenario that you find something that you can either adapt or kind of directly use but I think in practice often you're going to have to create

your own data set and I looked a little bit through the Discord and I think people are like tend to be quite creative about this already but I think one of the things that probably is like a little bit under you still is just to adapt existing data sets for large language model fine tuning so there's a bunch of data sets from kind of classic NLP that can be restructured or reformatted to work with llm fine

tuning and particularly if you're already an organization that's kind of been using machine learning for a while probably you have some of those data sets already around that you potentially could adapt the other part is like whether you already have some feedback from users so there's a lot of discussion about preferences and preference data sets and you can set about creating those very deliberately either by using human annotations or llm judges but quite often you will already have some

indications either very direct so people have like a thumbs up or a thumbs down but there might be other ways in which you can kind of get at this question of like was a user satisfied with a particular interaction and that might be a source of preference data that you can kind of create without actually having to to reather all that preference data manually and then the kind of final one is this synthetic data which I think

can go hand inand with these other bits but I think it's also very powerful for kind of jump starting the process of building these data sets I won't kind of labor this point too much but I think there's also often quite a lot of work to get your data into kind of format you can actually use for large language model training and I think there's some nice tooling for this but often it will be geared at a kind of very specific use

case but yeah one thing I wanted to kind of mention about that already is that I think a lot of the the work that you do in prepping this data might also end up being very useful for when you get to the stage of deploying a language model in production so I think you want to have that in mind that the kind of format that you're gathering this data from should be quite close to what you're going to actually be be using

that language model for in practice so some of this kind of pre-processing code and work will have a lot of overlap with what you're actually going to be deploying later on so jumping a a little bit more into the kind of data that you need for fine-tuning so I think in some of the earlier lessons this was already discussed a little bit but I think in terms of the the kind of I guess the the question of like what needs to be in

your data set and I know that's a little bit vague but I think one of the the kind of considerations which I think is slightly different when you find tuning for a more specific use case is being okay okay with the the kind of model losing certain abilities so often when you're kind of fine-tuning chat models which I think is what a lot of the literature ends up being about you want to make sure that you have a very

diverse data set and that it kind of captures all the things that users might discuss but in practice quite often the the kind of scope of your problem or the application you're trying to develop is much more narrow and then I think you don't have to worry about the diversity of the data in quite the same way and you have to really think about diversity in terms of what your large language model is actually likely to get us

input and I'll talk a little bit about these common data set genres so unfortunately it's a little bit the Wild West in terms of like the actual way in which data sets are structured so there's a a lot of variation in this but often there's some kind of similarities so for the supervised fine tuning you have this kind of question and answer response and this Probably sounds quite obvious but I think for some of these fine tuning use cases and

also when you're thinking about reinforcement learning and some of those algorithms for a lot of people I think they find it easier to understand okay what does the data set for training using this particular algorithm look like and then they can kind of map a little bit more High level what the process actually looks like and possible ways that they could get to a data set that matches this kind of format so very kind of briefly I w't

talk about this reinforcement learning bit too much but if you're kind of following this literature at all there's a lot of algorithms being published some are kind of iterative improvements on existing ones and some are quite different but what I think seen over the last year or so is that a lot of these algorithms end up using very similar data set structure so I think that's one of the kind of nice things in a way that a lot

of the existing data sets can either be kind of lightly adapted or use as they are without having to do do a lot of work to to reformat things so one example of this kind of algorithm that became very popular and it's been discussed earlier is this direct preference optimization and kind of go back to this idea of some people finding this kind of expression of algorithms not that helpful I think it can then be

really useful to go back to what does the model actually expect when it's being trained using this particular approach and this direct preference optimization I think is really nice because it kind of maps quite well to how you you kind of want to nudge a model when you're doing fine tuning so you have some input and that could be whatever your task is related to it could be kind of natural language it could be code it could be a bunch of

different things and then basically you want to nudge your model more to this chosen and rejected and the reason I kind of mention this is I think one of the nice things we've seen quite a lot in the past year or so since this algorithm became really popular is like really interesting and creative ways for people to actually come up with where are these Chosen and rejected so one way you could do it obviously is to like manually

write a good example and then write a bad example but that's really time consuming and tedious so people have done other things so for example with this chosen if you already have some kind of ground truth or gold standard data that a human is generated often that can serve as a really good value for this Chosen and then for example generate this rejected response from a model that you've kind of evaluated that does an okay job but isn't quite there

in terms of what you would like the the response to look and then the final kind of algorithm that I'll mentioned in terms of data sets is this kto algorithm and the nice thing about this is in contrast to DPO where you need this kind of two preference pairs of Chosen and rejected with kto you basically just have a response from the model and then this binary preference so basically a thumbs up or a

thumbs down and that's something that can be quite easy to collect so I think in general people quite happy to say I don't like this or I do like this in an existing system but again you might also have other kind of creative ways in which you might intuitively be able to understand okay well this person like didn't click this link or didn't do something else in this system so probably that was a thumbs down and they

did do something that kind of implies a thumbs up so I think this can be a really useful approach of gathering data if you already have something in the wild that can kind of be interacted with by some kind of end user and then the final kind of one of these algorithms in terms of data sets that I mention is Spin and Oro so spin is a kind of iterative approach so a lot of

the the kind of approaches to doing this reinforcement learning have been focus on trying to reduce the amount of data you requireed because that's kind of one of the big bottlenecks and the idea with spin is that you basically go through these different stages with some starting data and then you kind of synthetically generate new responses and then kind of build a data set on top of that without actually having to have such a large data set and

initially and this Oro algorithm again is a little bit about efficiency so Oro algorithm basically expects the data to be in the same format as the DPO so you have this input and then a response that's chosen a response that's rejected but the difference here is that it doesn't rely on you having done this self supervised fine-tuning step so you can kind of more quickly go directly from a

base model to actually doing this alignment without having to kind of train in two steps which is often what you have to do with these other approaches so that can be both nice in terms of not having to duplicate data for doing the the fine-tuning part and then the preference part but also it means that the actual model training is a little bit more lightweight because you're know having to do that in two stages and I'll hand over to David do

you want me to hand the slides to you as well yeah please so I I believe you have to stop sharing for a while and then I can hide it I think you need to disable it in Zoom itself perfect so just to kind of get back to kind of what what Dan Daniel already

highlighted is that in the end often you don't have your perfect data set and eventually you actually want to work towards like the the perfect data set that you might have so that's a way to actually do that is through synthetic data and synthetic data is yeah data that's generated by llms and that's commonly used for fine-tuning llms so you can actually start your synthetic data for example by generating prompts from scratch by generating completions based on prompts and normally you

basically prompt an llm along with some initial contexts for example in order to also do rephrasing and some of these algorithms that do rephrasing of PRS to make them more higher quality or higher complexity or more Nuance so to say in order to en ensure a higher quality prompt or completion or the evil complexity and evil quality prompts and then there's also this special kind of AI feedback or synthetic data and that's actually judging judging synthetic

data and that's to assign scores to promps and completions or to do preference ranking or to actually provide a rational critique along with your initial score that kind of explains as to why it came up with why the model actually produced the score so we actually would use a promp template prompting a model okay please provide a score for this prompt and the Associated response and can you also provide a r now along with that and that

would be kind of the prompt template so to say that people apply for for these kind of synthetic data things so one of the earlier models that was actually generated trained on synthetic data was the alaka 7B model and what they did was actually prompt the text of inii model along with the self instruct research paper so it was a prompt template that was actually taking some seed instruction so some initial seed data and was prompted to

rewrite that c data to yeah better instructions or more diverse instructions so they started up with 175 instructions then eventually ended up with 52 th000 instructions and actually did supervis fine training along with the L 7B model that meta initially released so that's I would say amazing you can actually use LM data to train LMS and then everything is solved but as you might expect that's not really the case so if you actually look into there report about synthetic data

and how report about the synthetic data and the self- instruct and how the model actually performed you can see that there's a lot of hallucinations toxicity and stereotypes within the model might be due to the fact that they actually used The Meta model which wasn't trained optimally might be due to the fact that they actually use the Dexter Vinci model from open AI which might contain some bias but in the end it yeah would have probably been better

to actually look at the data and and see what's in there so one of the examples from from a completionist this like what's the the optimal seed why why is 42 the optimal seed and the model actually starts hallucinating that that 42 is the optimal seed and that it can be used for for any neural network training so another example would be maybe using more complex pipelines more more better models better data and that's actually what they try to do for

the ultra feedback paper where they initially sourced instructions so a lot of both synthetic and human generated instructions from an instruction pool and actually asked prompted different models to actually provide completions for each one of these instructions and each one of these completions was actually judged by gp4 based on four different criteria so instruction following truthfulness honesty and helpfulness and also one overall criteria just asking a prompting

model whether overall the completion was correct according to the initial inputed pront and as you good thing to take away is that whenever you do this judging what we've seen is that each of these individual criteria when you focusing on on instruction following truthfulness Etc highly correlates on average so to say with one overall rating so if you actually want to reduce some cost and reduce some compute you can also just apply one overall

rating also given the fact that you probably need some some human involvement at the end so also this didn't turn out to work so when we actually started uploading this data in our UI and actually looking at the data by by yeah human judgment we actually saw that some data was sourced from benchmarks that there was like this weird thing happening where there were some proms that completely or some respons that

completely didn't make sense and still ended up getting a very high score and we kind of investigated that and investigated that and looked into the code and it was apparently a coding error leading the for the scores to be converted from a one to a 10 so all of the scores that were actually a 10 could have been been a one so that kind of messed up the entire data set there was like incomplete ratings due to the

fact that open AI API calls filled and there were some ties in the data that were still being processed as Chosen and rejected pairs even though if the same response if the responses would get the same rating you wouldn't expect them to have one preference over another because they would be in a TI and asally what we kind of also showed that whenever you spend some time look at your data get involved and work towards your higher quality data set

that you can actually train a better model and initially the the hugging phase team published the alignment handbook ched this awesome zier model and made the entire research reproducible and then we did that exact

reproduction but then instead we used the the clean Ultra feedback data set and that actually led to a model that perform better on the benchmarks and also in intuitively according to human judgment perform better so then apparently you need yeah better data or better synthetic data but

whenever you actually want to work towards that you there's a lot of things that that come to mind so initially you can kind of start spamming chat gbt with random request but if you actually want to scale that or make it more cost efficient want to avoid Fender lock in licenses that you might not be able be allowed to use all of the different models for actually generating synthetic data to find your models some fault tolerance that

might arise as you have seen in the ultra feedback data set and things like like structural generation where you might want to just output Json output responses in a Json format so overall there's just a lot of complexity involved when actually starting to do this and yeah I've chosen a set of tools that you might kind of look into whenever you start working with your own synthetic data and one of them is the the outlines package for structured text

generation and this can actually produce like structured text generation based on like promps for LMS based on ret RX ad Json penic models that everyone loves and and uses and also actually functions to to be able to do function calling and this actually relies on the modified by modifying the sampling of tokens and yeah this is a very accurate and approach to actually take and it actually also reduces the the inference time due to the fact that the there's a

limited number of tokens that you can actually sample from and one of the interesting block from from huging face actually Dives a bit deeper into this topic another package is DSP that probably or everyone is familiar with but it's focused on programming not prompting so to say and this actually let you define a dpy signature that you use for actually calling an llm where you kind of program the expectations that you might have for for your model

and then whenever you kind of optimize this for for your specific use case the package underneath tries to optimize the The Prompt and optimize or fine tune the model weight slightly so that then eventually end up with a higher or more accurate prediction and one thing that HL actually showed in one of his blog post is that yeah under the hoood the the model or the the ds5 package actually makes hundreds of calls to the to the open AI API when it tries

to optimize these prompts due to the fact that it's Gathering good few shot examples for for your use case and that actually is quite quite costly and it's an interesting block post where haml kind of goes over some of these tools as well so I recommend reading that too and the last one is D label and it's a synthetic framework for synthetic data generation and a AI feedback and it relies on like a direct graph structure

and it's more of a pipelining framework where we kind of worked on serializable pipelines cast intermediary results and actually included structure generation as well based on the outlines package and everything is execute executable in parallel so yeah whenever you really want to scale this up then you you can actually do

that and it's kind of like application developments like the things like Lama index or or deep set or some of these these tools but then

really focus on on the data set engineer and one of the interesting things that I've seen comb by recently was also the this blog post about the rise of the data set engineer so not really having AI Engineers or these kind of people anymore but really people that focus on data and data quality and it's an interesting read as well so when for you're working on improving your data you it's like a iterative process where you have your

original data set and you add some things to assess diversity the data set quantity data set quality you do duplication filtering and these kind of things and you to some like human feedback but also AI feedback in the mix to to make it like more more intuitive and more smart so what you can actually ensure or what you can think of for improving your data set quality is that there is enough quality but enough quantity but nor more

isn't always better so it's also fine to throw away data that's maybe very present within your data set that you might be able to D duplicate or also reduce the simply reduced lesser quality examples within your your data set so one example what some examples that we've gathered from actual public reproductions of these initial funing techniques that Daniel highlighted earlier is that for for sft we've seen great results according to benchmarks

with 10K input samples for orpo 7K DPO 3K and spin as low as 2K data data examples so apparently with with a lot less data and a lot of models are being F sh you can actually do a pretty good job at at achieving High Benchmark standards but probably also to to actually have some good human intuitive models and whenever you do think about data duplication there's a lot

of research being done in in duplication but often the duplication pipelines aren't very D duplication pipelines aren't very reproducible and not very well documented but yeah as we've seen for the fine rep data set that recently been published by by huging face they actually published the entire Pipeline and made it publicly available to their data 12 library and what you can think about when actually doing D duplication is applying intuitive rules

applying existing metadata that you might might use for filtering your data for filtering out some irrelevant examples for I don't know doing something like a topic topicwise D duplication where you might want the most representative examples per topic you can also think about creating your own custom metadata or your own custom features and doing like intuitive similar things like like hashing of of the the strings or doing like filtering based on embeding simul larity

and then actually grabbing the examplars from from like the the embedded data data points that are most representative for your data when you're doing rule based data it can be yes rule based cleaning of your data can be as simple as just writing a recx query so for example querying as a large language model if you don't want that to have that in your data set normally some some models used to respond like this or using the

word Del or using like these quotation marks that might indicate that the model is going to kind of hallucinate random references some random URLs that you might want to at least change within your data set and all of these things are like very intuitive human understandable rules that you can can apply to in order your to in order to improve your data set quality and on top of that what you can also do is yeah apply these more

advanced techniques or more advanced classifications or embeddings or these kind of things where you quite easily can yeah get a zero shot model out of out of hugging phase that's performing quite well and decide some topics that you deem relevant for for your data set and actually start predicting making initial predictions for these topics and actually use that as some some initial ways to to filter your data as well and these classifiers can also be very very cheap so it's you you

can actually use one of the seros shot models you can use a set fit model after annotating only a couple of examples and if you actually want to go more expensive you can yeah look at llms as judges juries or or rationalizers where you also have this rationale like the explanation as to why this this they provide this score and if you want to go a bit bit more simple and a bit more intuitive and explainable then something

like text descriptives like the the package where provided the EUR can be provided a very easy outof the Box analysis of your of your data and all of these methods that we actually that I went over can are actually intended to be used according to to our vision or view along with human annotation and this really helps normally to to make human annotation more fun more engaging and and also more more

effective so that's kind of where you want to be so maybe somewhere in the middle where you can either choose for a very basic simple intuitive approach using Google Sheets or like a completely customized approach that really works for you but for I would say for for a lot of the people something in between where you might use like some some already out of the box tools or web apps really works and and really differs purpose what you what you want

and what you prefer so one of the custom tools that I've seen combined to kind of play around with your data and kind of see what's what's in your data this bul annotation from from vinet warmerdam and it's like a custom tool that he built with bokei and and pendas and some JavaScript to tie everything together where you embed all of your texts and you can kind of explore and kind of play play around with it so this is yeah what it what it

looks like you can do like this lesso movement kind of see what's within that specific cluster and then move from from there to kind of annotating and actually saving your your initial results by the way I have a question about this I saw this too and I thought it was pretty cool does arila have something like this as specific functionality yeah we we actually have something like this where you might be able to attach vectors to your records

and you can do semantic search but we don't have this interactive kind of annotation thing it's a thing that we've considered I believe snorkel has something similar to it but we've not gotten around to actually implementing this an alternative that you might consider is like using something like notebooks where you use notebooks to initially annotate your your data and I think everything in terms of notebook annotation was a request from

from haml to kind of highlight this is like

based on IPI annotations and then from that there's a lot of branches going looking roughly the same and giving the same overview but this is also fully customizable where after you've filled out one annotation you can actually do some callbacks and actually do some postprocessing so you might be be able to do some Active Learning and actually start Gathering data for fing setf classifier on the fly so to say and then run influence with that with

that classification model another thing that's kind of like a little bit more out of the box is like creating repb apps with with radio with streamlet with with shiny with r shiny and you can actually use like all of their components and tools to directly out of the box have a very intuitive basic UI and this can also be fully customized normally so there's one UI of the the S NP hackaton that someone created that we normally host

together with hugging face and this is the the input text where after that someone is able to to kind of correct or say okay this is a right translation or wrong translation can also be used for like the kto example that Daniel mentioned earlier where you might give her a thumbs up thumbs down and kind of go through your records one by one this is once again like a a nice nice example but with gradio streamlet and

shiny you can really customize everything as as far as you might want to do that and if you actually want something a bit more out of the box there's also have these fully fledged cool Solutions with a lot of integrated features one of them is lilac and lilac kind of has this data set overview thing where whenever you select the data set you can actually select different topics and you can actually see some of the the the main clusters within

within your data and it's pretty pretty cool and on top of that they they also have this semantic search feature that I previously mentioned and on top of that I'm curious about lilac by the way do you recommend using arila with lilac for like the different features like using them concurrently or like what's like the stack that you like personally for for me it's kind of the a bit a bit of a mix of both I think it really depends

on what you prefer what you like I find the Lilac UI UI a bit less intuitive personally for example for within within arilla there's less I think features in included and the ux UI is is a bit more intuitive for me things like like Spacey exposion I play played around with but it really depends on on what you want probably if you as a as a AI engineer look at the API that we have or the SDK that we have

compared to the one with lilac if you as a domain exper or labeler kind of go into into lilac and find their features and and uiux better then then go for that but I I think it's the the same tool inherently covering the same topic having a lot of the the same or similar featur and for me it really depends on on that kind of thing well you Daniel do you have do you have a opinion or do you

have a sack that you like yeah I mean I think one thing is like at what point in your kind of process you're trying to look at your data so I think there's this like initial kind of Vibes check where you're trying to look at quite a

lot of data right quickly and maybe looking at a few examples carefully but not necessarily like actually fixing things yet so I think it's good to to kind of separate the the kind of

annotation mindset from the like I'm trying to just understand this data overall and I think for that lilet can be quite nice because you have these different kind of visualizations and I think the other thing that's like super crude but if you can get your data into some kind of data frame even in Google collab and compute some features like string length and token count and very like basic fude features it gives you these plot suggestions which are

actually like sometimes okay and it's just like a really lazy way of being like hey there's like this one really random outlier so it's a little bit crude but I think it can be quite quick way of finding like where something's going wrong and then I think after that kind of trying to dive in a little bit more detail and actually poke around the individual responses yeah and I think with the the the nice thing from from Lilac or any

more full-fledged tools is that yeah you don't need to think about about what you want so to say so of course they they have a lot of these highly opinionated features built in and based on that you I think a lot of people will get away with having like some nice outof the box to kit there's of course always people that are going to want to have some some custom thing built on top of some some custom features and then I

think it's if you really value that and it's also worth spending time on that and actually building some of these custom tools or some of these custom annotation flows out of the box I think both lilac and and arella are open source as well yeah so this was the the other example that that I had arill same kind of idea having a data set overview you can log in and play with a demo as as with

with lilac and also you have this like CTIC search kind of thing you have your like content that you might load into your UI and you can actually start an ating and doing bu annotation and all of these these kind of things so it's same same but different don't I have a preference with for ailla of course and I'm more familiar with that but I guess for for everyone it's up to them to decide which which UI and kind of API

and workflows they they prefer yeah so maybe I'll hand it over to Danielle again and stop sharing okay can you see that yeah perfect so I'm just going to talk

through some example data sets and this might seem a little bit weird like why I'm talking about these very particular data sets but I actually think you can sometimes learn very little but sometimes learn a lot from looking at some examples of data sets people in the community built particularly for like approaches to generating these data sets in kind of creative ways that doesn't just rely on either throwing loads of API credits at the problem or loads of

human effort so I think that's also goes back to this thing I was saying earlier about DPO data sets why they became so kind of popular in the community is because this idea of generating a Chosen and rejected pair can be

done in like a lot of creative ways so there's an example here of this data set is trying to make models large language models better writing and the approach they take is basically to to kind of take some existing books

written by humans and then prompt a large language model to summarize those and then prompt another model to rewrite based on that summary so you know write a a novel or chapter of Novel based on this kind of summary and then they choose the the kind of the original human generated response as the The Chosen and the model generated responses rejected so that might not apply to your use case but I think it's an example of

like how you can get out these things without having to to just rely on either human data or or model data here's another example so this one is actually kind of a vision model model but it has a kind of similar interest in that it's focusing on kind of actually generating this data flywheel so trying to get this thumbs up thumbs down so I mean it really depends

what kind of context you're working in but I think getting these thumbs up thumbs down ratings can be quite useful and the other thing that I found from doing a lot of annotation so I think it's worth thinking a little bit about the the kind of ergonomics so not just as a tool but also what is the task that you're actually trying to do when you're annotating and how does that work better so for these kind of preference data

sets depending on what you're trying to do sometimes it can be really easy to say that's a good response or that's a bad response other times it's really helpful to have like these two generations next to each other and say okay this one is better than this one because like without having a comparison it's quite difficult to actually say which model response is better so I think when you're doing this human annotation I would give like quite a

lot of thought to those kind of questions which I think people kind of don't really think about that much but I think it can actually yeah I think it can be quite important for both your experience of doing The annotation and how pleasant and easy you find it but I also have no evidence for this but I suspect it actually influences the results of the annotations you get quite a bit depending on how you set up the

task so I think it's worth giving a bit of thought and I think what I maybe we'll do is just give like a very high level overview of this kind of project I'm actually working on for this course so that is basically trying to build a large language model summarizer and hopefully a small large language model summarizer so it can actually be deployed easily they will take a data set card for a data set posted on the

Hub and turn this kind of long markdown into a too long didn't read summary that you can kind of very quickly look at and be like what is this data set about and what is it for this is just like a very kind of high level overview of the steps that are kind of taken to to kind of create that data set so you have as I kind of mentioned at the start a bunch of processing that you do with

the markdown and that will also be processing that you'll carry over to the actual deployment so in the markdown you have a lot of stuff that you probably want to even pass to a large language model because it's just

like formatting or content it's not very informative you might want to move some of this yaml and then you kind of get this summary and then in in this particular case like the approach I'm trying to take is to build this preference data

set and there's different ways of approaching this but this is what I'm kind of using this distill label pipeline for and this kind of looks a little bit crazy but I guess the the overall workflow which is actually not that difficult to Define is that you load this data set that has a bunch of these data set cards and you do this kind of initial filtering you format this prompt and then in this case I'm trying

to use open model so kind of compare these three different summaries and then use Ultra feedback which is a kind of approach to doing this language model judging and in this case I'm using llama 3 for that and I guess that the kind of thing I want to say about this pipeline I think with synthetic data generation it really depends on what you're trying to do sometimes just like having a prompt and a bunch of inputs and being

like right just call that API a bunch of time where this prompt will work well and sometimes just kind of more complicated pipeline can work better but the thing I found quite nice about this is that at each of the kind of steps you kind of maybe increase in the number responses you're generating so maybe initially when you're comparing these different models you kind of do like a 100 or something and just kind of very quickly start looking through the

responses and I guess at that point what you're thinking about which is kind of similar to this point that you're often doing when you're kind of thinking about using prompting versus fine tuning is being questioning about okay can I just improve this this fire The Prompt so you might already done a little bit playing around but when you kind of get a few more responses you think is there something that I'm getting in the response that can just be fixed in this

prompt step and you kind of iterate on this pipeline but each of these steps might actually remain each the kind of workflow might remain quite static but you're kind of fiddling around with each of these steps and the other thing that I will say in this kind of particular pipeline I'm sending all of these responses to our Gilla so you kind of looking at two things when you're doing that so you're looking at the model responses and seeing how they look

but the other thing that I'm trying to look at when I'm kind of building this pipeline is this LM judge part because I'm basically assuming with this workflow that this judge is actually going to be able to detect which summaries I'm going to like better so one of the things that I kind of do and there's a kind of notebook in a GitHub that I'll share afterwards but basically rate a bunch of examples and then look at the ratings that the model gives

and whether there's some like average better rating but also like how often do I change the rating from the model and like how big is that difference maybe if it's like only one score different then that's not so bad but if it's consistently getting bad ratings then that's maybe something to worry about in terms of whether it's

reflecting what I want the the llm to do and the other thing I would say about this part and I'm probably a little bit overly

obsessed with this idea but I think when you're looking at these ratings and the responses like thinking about heuristics that you can use to to kind of capture is this response good or bad like maybe it's too complicated to to judge far a simple rule but quite often there might be patterns that you start to see so once you have like even a 100 rating or maybe even like 20 you can start being like okay do I always prefer the

one that's shorter or the one that's longer or is it like random because if it's just always a shorter one then maybe you can already say I just need to like cap the the kind of Maximum tokens or do something a little bit more manual or I can just basically say all the short responses are always better and then skip this kind of LM judge part because this is the kind of expensive part but it's also like

slightly brittle because you in this case like put quite a lot of faith as you kind of scale this in this consistently working well and if you can use some of these Simple Rules either instead of or as well as the the kind of Judge to check okay is this continuing to work well I think that could be quite useful so because we're running out of time I thought I'd just quickly talk through this repo that we

created so there's some notebooks in here here that kind of gives some examples of like different approaches to duplication how to kind of do these these kind of database checks and then an example of using distill label to generate one of these synthetic pipelines and the other thing which I have to admit as a work in progress is trying to basically put all the work I'm doing for this particular project in here I'm not saying it's necessarily

like a good example of what to do but I will at least try and share it openly and kind of communicate what I'm what I'm trying do there and then yeah I'll maybe just quickly point to these other resources that we have here so one thing just wanted to mention is that with the the course credits that you have from hugging face you can use that for spaces but you can also use it for inference end points

so I think between all the course credits you could actually generate quite a lot of synthetic data and I think they quite interesting to see kind of what people could come up with that and then yeah beyond the the kind of notebooks for this session there's also a few other resources here that might be interested including this blog post about fine web which talks a little bit about their approach to duplication and it is kind of for a

slightly different use case but I think there's some kind of nice lessons in there that might still be kind of relevant that you know doing like a really aggressive D duplication doesn't always work better and there's these kind of subtle heuristics which unfortunately like often you have to actually train a model to see what the impact is but if someone else has done that it's quite useful to to see if there's something you can pick up there

so yeah I think maybe Sor go go ahead sorry no sorry I was just rumbling to a close oh no worries we have about two minutes left there isn't too much time for questions but I'll just throw one out there I mean asked how

would you recommend generating a synthetic generating synthetic data if we're fine-tuning on proprietary data sets and human annotation is

expensive proprietary data sets that someone else creates like a vendor of of some sort or yeah I mean I don't really know what he means but maybe it's yeah like fine-tuning on I don't even know if the word proprietary really I mean just like your own company's data just like let's put that way so I I guess it kind of it's kind of the same as whenever you would use fine tune your own model versus fine-tuning or

versus using a proprietary model it's yeah about data ownership privacy about the fact if you value value these kind of things in your company if you need these things in your company and if you really want to be the owner of your your data your model and those are I think the the the main takeaways for for this kind of kind of tradeoff both for models but but also for your for your data sounds good we can probably take

the rest of the questions to the Discord maybe yall if if you want if you David and Daniel if you have time might want to peruse it there's a channel dedicated specifically to the stock where people are in but yeah thanks a lot this is very good and and just to say also like I'm like overly excited about people building data sets so like I'm happy to try and help out like I might not have the answers but I'm happy to like

brainstorm with people if they're interested in building a DAT set as part of this and sharing it then definitely try and give them that under that yeah great all right thanks a lot thanks all right thanks for having us wa