Agents For Healthcare Chart Review

Thomas Charlon

March 6, 2026 – Scale23x Linux Expo

https://thomaschln.gitlab.io

charlon@protonmail.com

“Agents” For Healthcare Chart Review

What this talk is about:

→ Integrating LLMs in traceable pipelines for medical applications

→ Automating high research impact tasks done by clinicians

→ Providing reproducible frameworks to produce NLP and reasoning with LLMs

→ Discussing the usual key steps, public models, open-source software

What this talk is not about:

→ MCP, tools, LLM planning, orchestration, LLM as a judge

(“traditional” agent definition)

Introducing healthcare chart review

Hospital research studies

Clinical research studies in hospital settings use electronic health records (EHR)

→ Codified tabulated data,
as checkboxes for the diagnoses performed,
medications prescribed, labs and procedures

→ Unstructured narrative notes
as clinicians’ visit reports, comments and justifications

Phenotyping

One problematic all clinical research studies will confront with EHR:
“I want to study the effect of xxx on Diabetes,
I can just pull the data of all patients with Diabetes diagnosis right ?”

→ Many “diagnoses” in EHR are closer to billable codes,
For example, your PCP wants you to get a lab test to check a condition,
Diabetes appears in your EHR data, but it was more like a screening

→ Most often, clinical research studies will say,
”I want patients with at least two diagnosis codes”
Or “one diagnosis code and one prescription of the usual medications for that”

Chart review

Some clinical researchers have tackled this problematic to produce more accurate "phenotyping" to select their patient cohorts or measure outcomes, comorbidities

Usually, they will develop a method, and review a small subset of patients to measure accuracy against gold labels, before applying to full hospital population

For example, manually review 100 patients,
"based on his notes, did the patient actually have condition xxx"
Then, if the method has satisfying accuracy, apply it to +10k patients

→ This is called chart review
It's time intensive and requires expert knowledge

2 projects my lab has worked on

<u>Medication history</u>

→ Sometimes medications appear in EHR, were prescribed and/or ordered, but never taken by the patient, maybe for insurance reasons or other

→ Can we use LLMs to provide a more truthful source than the codified data ?

<u>Heart failure phenotyping in rheumatoid arthritis</u>

→ Heart failure is a key contraindication in rheumatology

→ Can we phenotype heart failure more accurately than before ?

2 projects I have worked on

<u>Measuring disease activity in inflammatory bowel disease</u>

→ After a colonoscopy, clinicians will write down procedure details,
as “the bowel was clean, the camera passed through the whole colon,
mild ulcers in left colon, awaiting biopsy results…”

→ We want to derive scores for “low” *vs.* “high” disease activity,
to then use these to measure for example treatment efficacy

2 projects I have worked on

<u>Phenotyping for suicide prevention</u>

→ Suicide attempt codes are notoriously challenging,
Maybe the patient comes in emergency department,
gets an “injury” diagnosis, and only later sees a psychiatrist

→ Plus, we are interested in a specific date, for survival analysis,
Not just “did the patient ever had a suicide attempt ?”
But mostly: “on which exact date did it happen? ”

Agents for
healthcare chart review

“Agents” For Healthcare Chart Review

The problem with the usual agent definition is
it leaves too much leeway to the LLMs

→ in IBD, trying to put several reports close in time in one prompt, and adding the date of each: the LLM would often not get the date right

→ in suicide prevention, LLMs have a hard time understanding the subtleties differentiating intent, lethality, planification, ideation

Traceable AI

Clinicians are excited about AI
but they’re the ones getting sued at the end of the day

→ “Explainable” and “reproducible” AI often tells only half of the story
→ The question is not so much “what are the features most contributing”

We need more details, and model simplicity
→ Cutoff values ? Overfitting ? Generalizability ?
→ “Show me the exact passage in the note, I want to doublecheck”

Traceable agents

Coming back to the IBD example,
“trying to put several reports close in time in one prompt, and adding the date of each: the LLM would often not get the date right”

The LLM is actually doing a good job on understanding synonyms, negations, and performing basic scaling of features (mild = 1, moderate = 2, …)

My agent philosophy:
“The more you can do with classic methods, the better”
→ This means, you don’t rely on the LLM to keep the dates right,
You use it to do the things you can’t do without it, and only that

Safety alignment

You might have heard about my prior work on LLM ontology mapping,
One difficulty I was faced with was,
when trying to map suicide prevention features between ontologies,
the LLM would trigger safety errors, even the local models,

”Can you list examples of suicide attempts by increasing lethality ?”
“No, here’s the hotline”

And commercial models have additional safety layers

My point is, you need a superintelligent overseer

The superintelligent overseer

Tianxi Cai, biostats tenure professor, Harvard Medical and Public health

Katherine Liao, MD group director, MGB Rheumatology

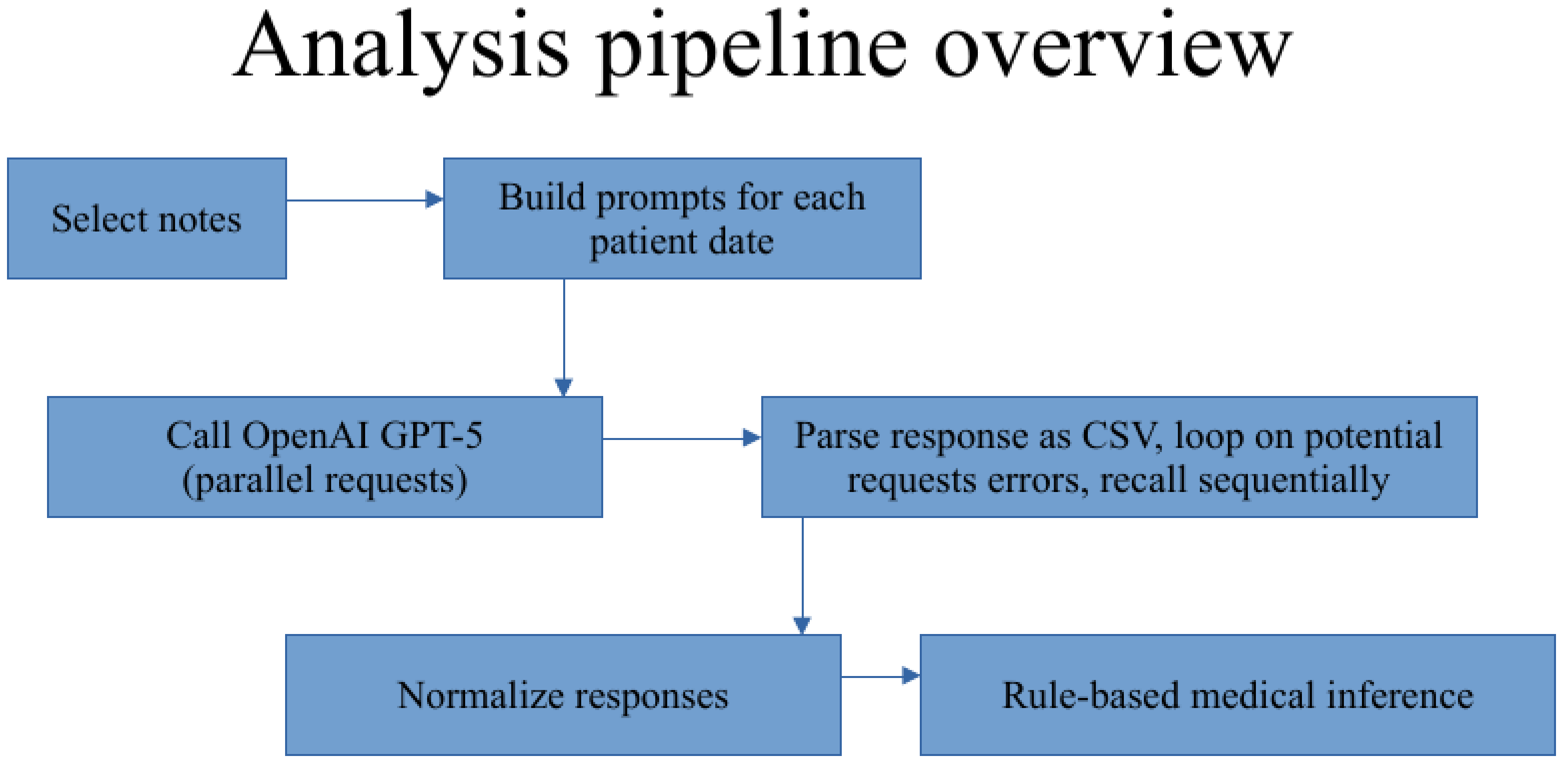
Ashwin Ananthakrishnan, MD associate professor, MGB Gastroenterology

William Meyerson, PsyMD research fellow, MGB Psychiatry

Huaiyuan Ying, LLM PhD student, Harvard Medical and Tsinghua University

Conceptual Architecture

The one figure you might see again in the near future


Analysis pipeline overview
Select notes
Build prompts for each patient date
Call OpenAI GPT-5 (parallel requests)
Parse response as CSV, loop on potential requests errors, recall sequentially
Normalize responses
Rule-based medical inference

Setting the specs

We want to call LLMs on elements we can trace back to

<u>Preferably, on individual notes</u>

So we know exactly the patient and date without having to trust the LLM,
For the IBD endoscopy project, we know precisely which note to analyze,
We want one output per note

<u>When there are too many notes, on individual patients</u>

For the suicide prevention project,
we are interested in detecting as accurately as possible lifetime suicide attempts
On average, for one patient we might have 20 notes per year, and 100k patients,
One note will be on average 10k characters, ~2.5k tokens

Setting the specs

We want the output to be formatted as JSON, to provide detailed
and structured outputs that will be parsed as a table down the road

Suddenly, the number of available public pre-trained models drops drastically,
most will not consistently output well-formatted JSONs

In my (non-exhaustive) experiments, only OpenAI GPT-OSS
and Qwen3 (“parent” of DeepSeek) satisfied this requirement

Some publications mention Mistral-OpenOrca 7b,
and you might be able to output JSON with Llama 7b,
but the real test is once you have a detailed prompt and a 10k character note

Model selection

A few additional considerations include:
→ Context window (maximum length of prompt)
→ GPU requirements (number of parameters in your model)
→ Arguably, country of origin, as it may influence “safety” and deployment

The main reason for dropping Qwen3 was
the small context window for medium size models (~100b)
OpenAI GPT-OSS 20b fits comfortably on a 50 GB GPU (32 should be fine)
For GPT-OSS 120b, with large prompts you’ll need 2 x 50GB GPU

For commercial models,
an additional consideration for medical use is HIPAA compliance (data privacy)

Software selection

I know mainly of 3 softwares to run public models:
→ Ollama → vLLM → Sglang

To perform parallel inference (calling thousands of prompts at once),
Ollama is really not the software of choice,
you should rather go with vLLM and Sglang

For your conversational AI, Ollama might be fine though

I haven’t tested Sglang yet, I hear it’s better than vLLM for multi-GPU settings,
You should see a nice boost when running GPT-OSS 120b on >= 4 x 50 GB GPUs
You would need to tweak vLLM to get similar performances

Primer on reasoning models

You might have heard about reasoning models, it’s like the 2nd gen of LLMs

Basically, all LLMs are trained in the same way (transformers),
but reasoning models then have extensive fine-tuning

Instead of just doing “next token prediction”, reasoning models are fine-tuned to detail step by step the process leading to the answer.

This leads to considerably better answers,
at the cost of considerably more computation.

Primer on reasoning models

So if your task is easy enough you might want to consider using non-reasoning models as they will be much faster

Also, you don’t have the temperature parameter anymore,
and some models enable to choose between minimal and maximal reasoning

For example, Llama 7b is not a reasoning model,
and Mistral OpenOrca 7b is an early version of a reasoning model

In commercial models, GPT 4o is not a reasoning model,
while GPT-5 mini is, and provides 3 levels of reasoning to choose from.
I believe GPT-OSS provides reasoning levels but are not supported by vLLM

Reasoning-heavy *vs.* NLP-heavy

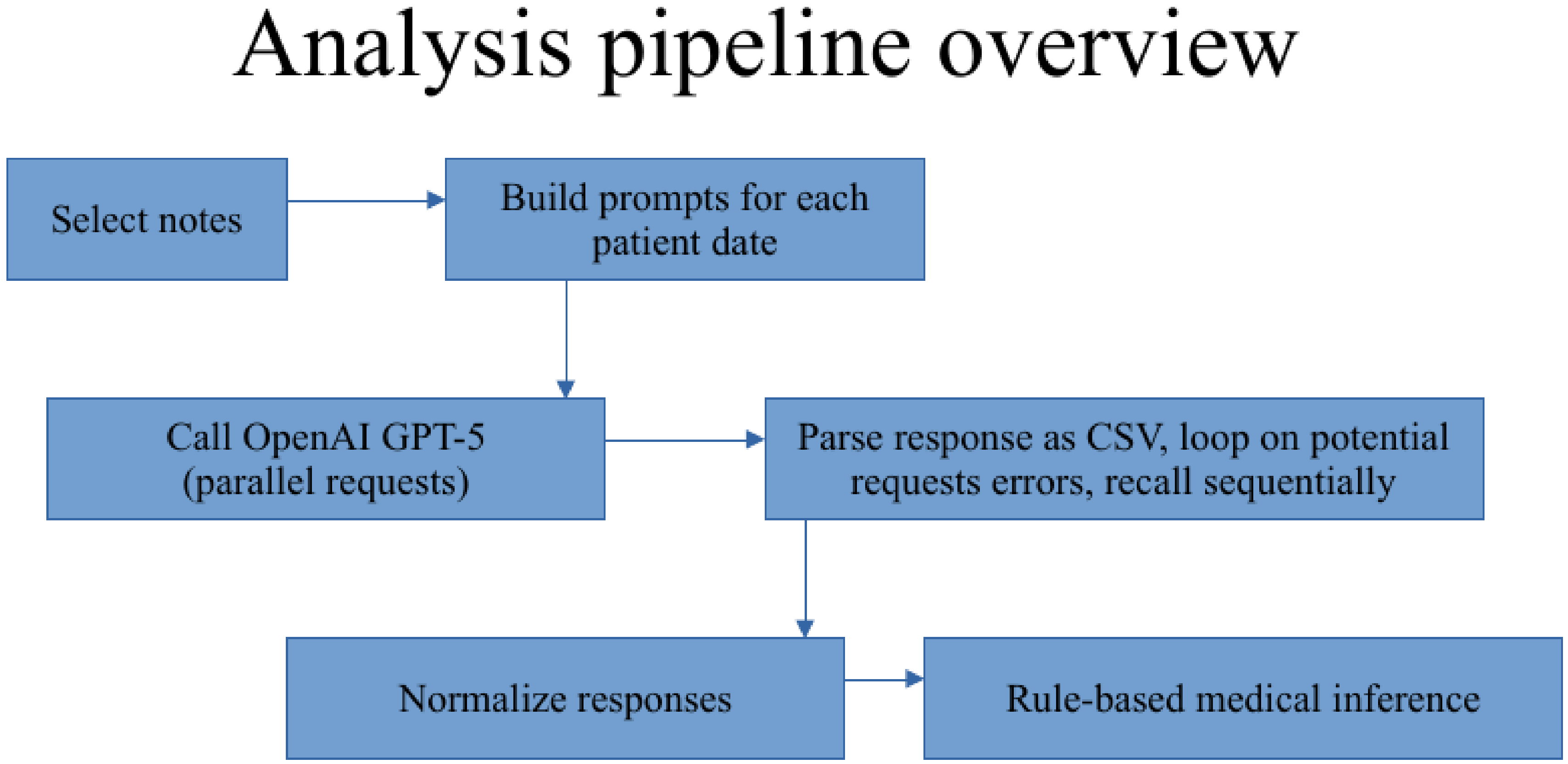
You might want to distinguish two main ways of leveraging LLMs,

→ You have small inputs, are mainly interested by deriving scores,
This is what I call reasoning-heavy, your results will depend heavily on your model, the bulk of your work will be on prompt tuning

→ You have large inputs, need to cross-reference outputs, are mainly interested in information extraction
This is what I call NLP-heavy, your LLM is integrated in a larger pipeline,
your results might be influenced as much by your model as by your pre-LLM input selection and your post-LLM processing

Practical Implementation

Coming back on this one


Analysis pipeline overview

Select notes
Build prompts for each patient date
Call OpenAI GPT-5 (parallel requests)
Parse response as CSV, loop on potential requests errors, recall sequentially
Normalize responses
Rule-based medical inference

Input selection

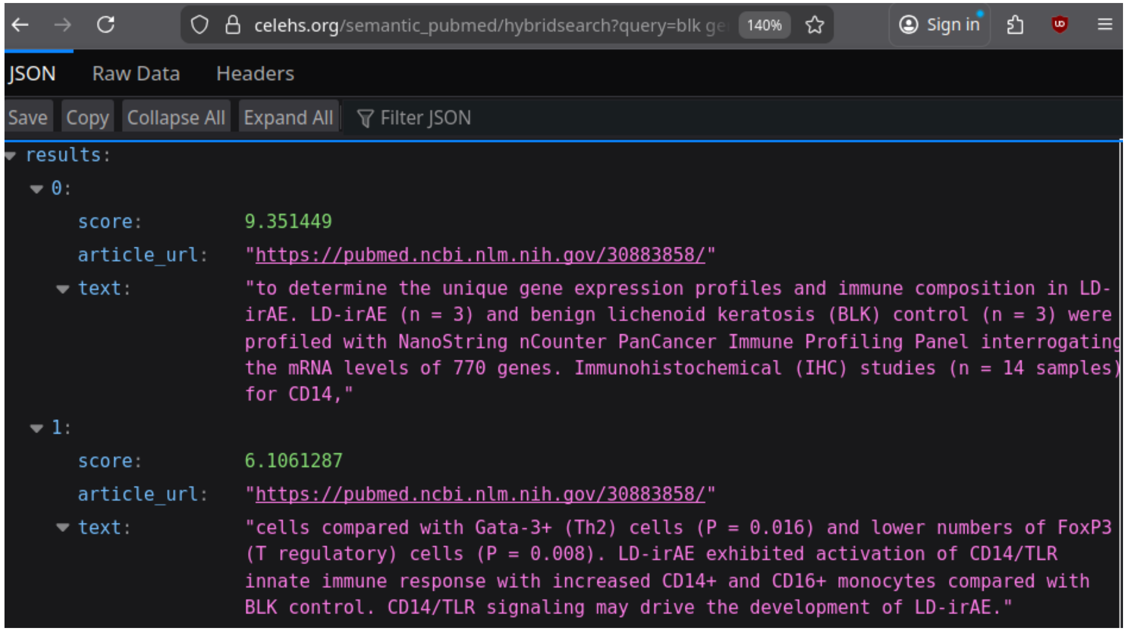
In “NLP-heavy” tasks, to perform input selection,
you might have heard about Retrieval Augmented Generation (RAG),
in which your large inputs are chunked with a small embedding model,
then when the user inputs a query the most similar chunks are retrieved.

This is one way to perform input selection,
modern search engines will prefer combining both semantic similarity
(embedding-based) and “lexical” similarity (exact or approximate string matches)

OpenSearch, the fork of ElasticSearch, provides hybrid search out of the box
Other software choices for semantic search include Milvus and Chroma,
though for hybrid search you’ll need to combine it with lexical search yourself

Hybrid Search

Here's a little API I developed as a testing ground using OpenSearch,
My database is 350k scientific publications summaries,
Each summary (~20 sentences) is chunked in ~3 sentences
https://celehs.org/semantic_pubmed/query=blk%20gene



```
results:
  0:
    score:        9.351449
    article_url:  "https://pubmed.ncbi.nlm.nih.gov/30883858/"
    text:         "to determine the unique gene expression profiles and immune composition in LD-
                  irAE. LD-irAE (n = 3) and benign lichenoid keratosis (BLK) control (n = 3) were
                  profiled with NanoString nCounter PanCancer Immune Profiling Panel interrogating
                  the mRNA levels of 770 genes. Immunohistochemical (IHC) studies (n = 14 samples)
                  for CD14,"
  1:
    score:        6.1061287
    article_url:  "https://pubmed.ncbi.nlm.nih.gov/30883858/"
    text:         "cells compared with Gata-3+ (Th2) cells (P = 0.016) and lower numbers of FoxP3
                  (T regulatory) cells (P = 0.008). LD-irAE exhibited activation of CD14/TLR
                  innate immune response with increased CD14+ and CD16+ monocytes compared with
                  BLK control. CD14/TLR signaling may drive the development of LD-irAE."
```

Building prompts

Aka prompt engineering if you want to get a $300k job
The basics: describe your goal and task, then your expected output,
give a few input and/or output examples (especially outputs),
and maybe a few pitfalls you noticed in previous LLM calls

This is a bit of a dark magic, you might have heard that putting
“If you do it correctly I will tip you $1000” will help accuracy,
also repeating some instructions might help,

Not sure it really helps on complex tasks though.
Personally I just use redundancy on the pitfalls, I rephrase them a few times
“you should not take xxx into account; xxx contributes 0”

DSPy prompting

Worth mentioning, DSPy is a software that tries to abstract out prompt engineering,
Let’s say you have some input e-mails you want to classify as high *vs.* low priority,
You give DSPy ~100 example e-mails with their classifications, and it will build the prompt automatically

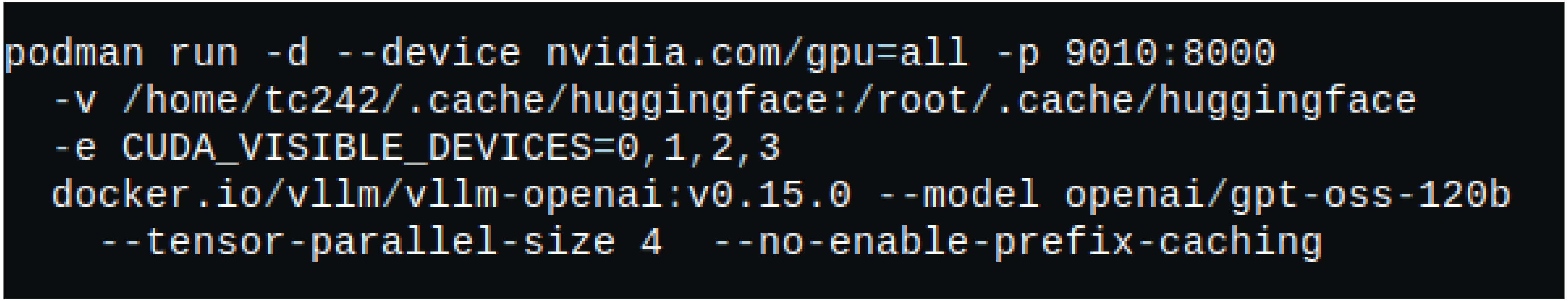
You can also start with a seed prompt you wrote, then ask DSPy to optimize it, and ask it to output a human readable prompt (there’s more to say about this here though it’s not the scope, we can come back to it during questions)

Personally haven’t tested, might be useful for reasoning-heavy tasks, though I doubt it will be able to manage medical considerations effectively, might overfit

Calling the LLM

Here’s an example of how I call vLLM, first start by initiating your vLLM endpoint,

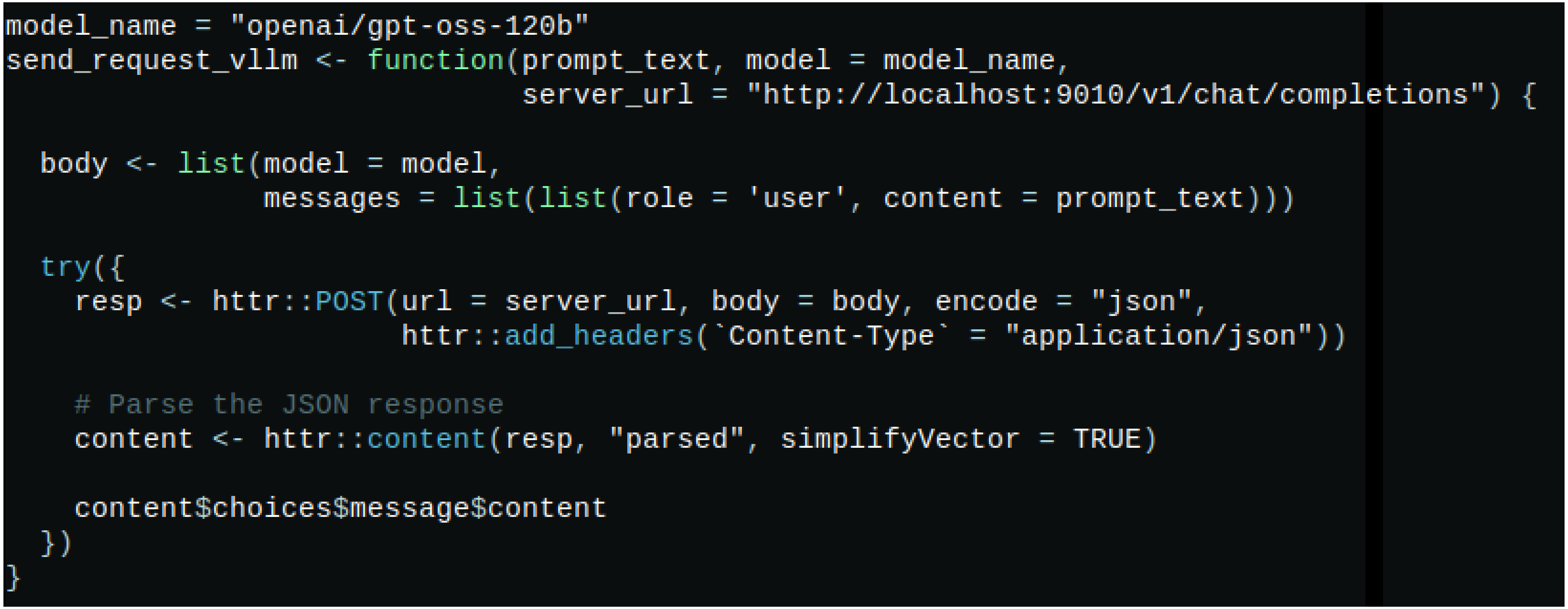
Here we’re using 4 GPUs, GPT-OSS 120b, and using port 9010
Do note the –no-enable-prefix-caching option, you’ll thank me later,
Podman is a non-root Docker, we use vLLM v0.15.0 image, and cache the model



```
podman run -d --device nvidia.com/gpu=all -p 9010:8000
  -v /home/tc242/.cache/huggingface:/root/.cache/huggingface
  -e CUDA_VISIBLE_DEVICES=0,1,2,3
  docker.io/vllm/vllm-openai:v0.15.0 --model openai/gpt-oss-120b
    --tensor-parallel-size 4  --no-enable-prefix-caching
```

Calling the LLM

Then I call port 9010 in my R scripts with a POST



```r
model_name = "openai/gpt-oss-120b"
send_request_vllm <- function(prompt_text, model = model_name,
                              server_url = "http://localhost:9010/v1/chat/completions") {

  body <- list(model = model,
               messages = list(list(role = 'user', content = prompt_text)))

  try({
    resp <- httr::POST(url = server_url, body = body, encode = "json",
                       httr::add_headers(`Content-Type` = "application/json"))

    # Parse the JSON response
    content <- httr::content(resp, "parsed", simplifyVector = TRUE)

    content$choices$message$content
  })
}
```

JSON Normalization

So you called your LLM, it produced a JSON output,
And now you want to build a table out of it

But surprise, you were asking for “If there are ulcers, return { ulcers: 1 }”
And it actually returned 20% of the time “{ ‘presence of ulcers’: 1 }”

You could call the LLM again, aka healing,
“this was your previous output, it does not map to …, please retry”

But you might not want to pay / wait the additional queries / time,
And likely you will need to have a mapping at one point down the road,

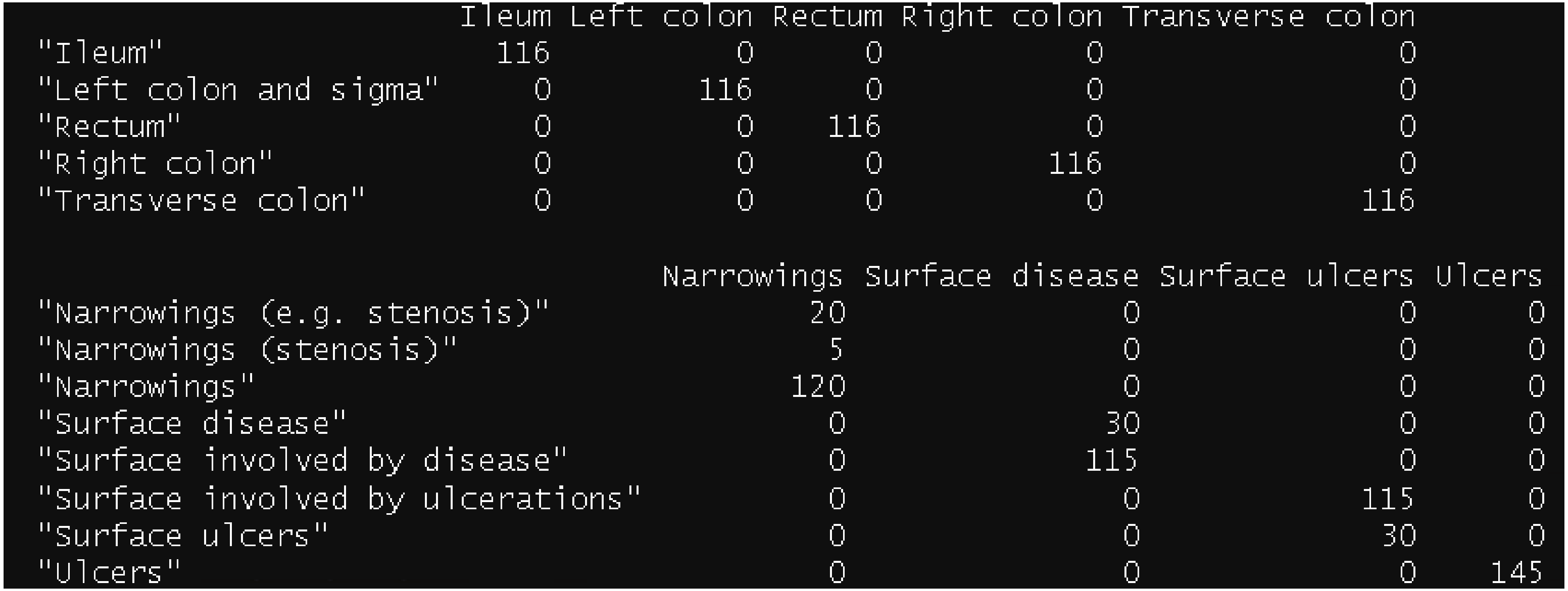
JSON Normalization

So OK, you managed a mapping, maybe via string distance and some regexps, But, that was on your test set, will it work well on your deployment to +10k patients ? Or will you accidentally map apples to oranges ?

For me, this is really where the key efforts happen, because even if you validate your normalization on your 1,000 cases training dataset, you’ll need to find an efficient way to double check your 50,000 deployment dataset

And it will be required in both your reasoning-heavy and NLP-heavy tasks

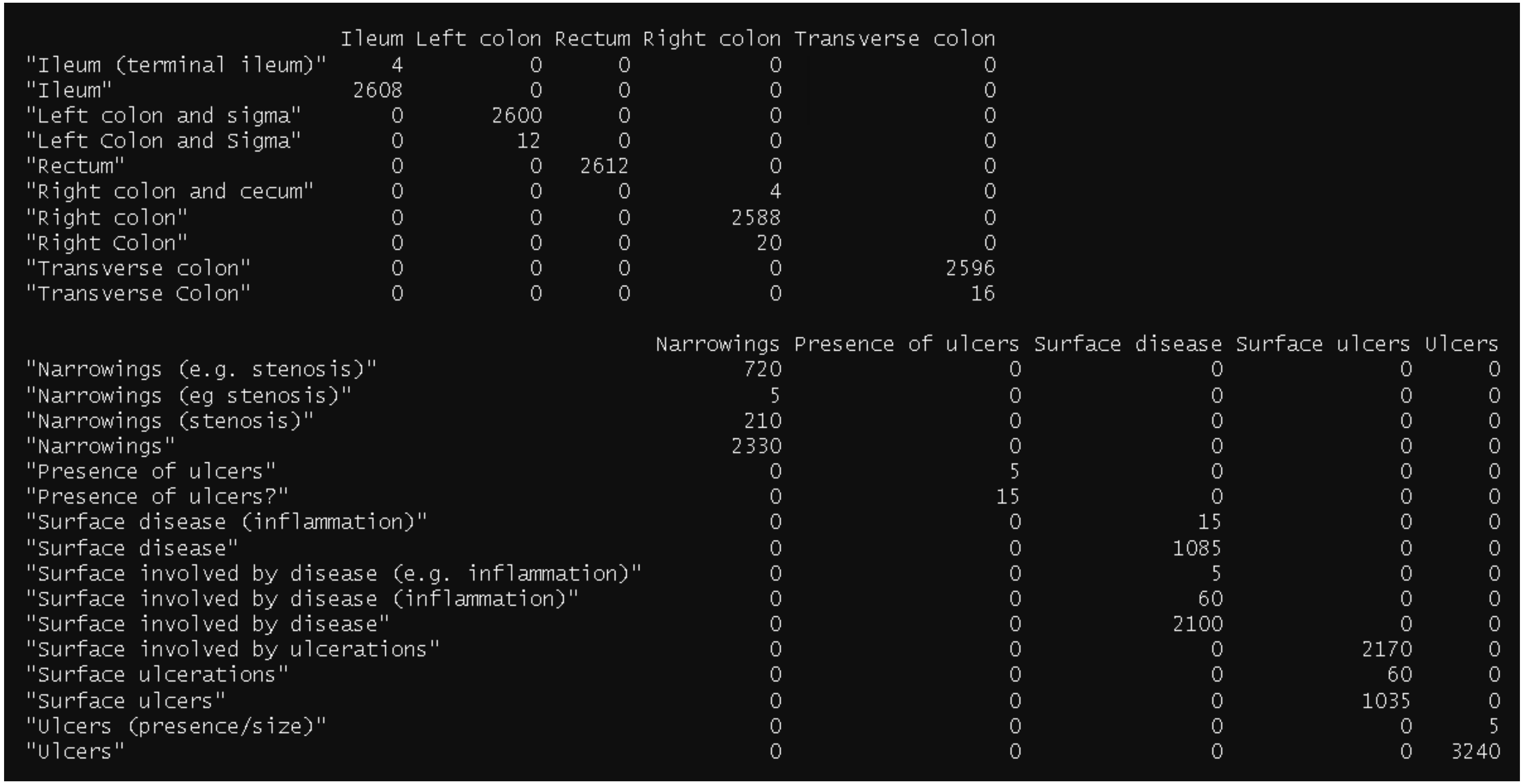
Traceable mappings



```
                        Ileum Left colon Rectum Right colon Transverse colon
"Ileum"                   116          0      0           0                0
"Left colon and sigma"      0        116      0           0                0
"Rectum"                    0          0    116           0                0
"Right colon"               0          0      0         116                0
"Transverse colon"          0          0      0           0              116

                                   Narrowings Surface disease Surface ulcers Ulcers
"Narrowings (e.g. stenosis)"               20               0              0      0
"Narrowings (stenosis)"                     5               0              0      0
"Narrowings"                              120               0              0      0
"Surface disease"                           0              30              0      0
"Surface involved by disease"               0             115              0      0
"Surface involved by ulcerations"           0               0            115      0
"Surface ulcers"                            0               0             30      0
"Ulcers"                                    0               0              0    145
```

Traceable mappings



```
                         Ileum Left colon Rectum Right colon Transverse colon
"Ileum (terminal ileum)"     4          0      0           0                0
"Ileum"                   2608          0      0           0                0
"Left colon and sigma"       0       2600      0           0                0
"Left Colon and Sigma"       0         12      0           0                0
"Rectum"                     0          0   2612           0                0
"Right colon and cecum"      0          0      0           4                0
"Right colon"                0          0      0        2588                0
"Right Colon"                0          0      0          20                0
"Transverse colon"           0          0      0           0             2596
"Transverse Colon"           0          0      0           0               16


                                                  Narrowings Presence of ulcers Surface disease Surface ulcers Ulcers
"Narrowings (e.g. stenosis)"                             720                  0               0              0      0
"Narrowings (eg stenosis)"                                 5                  0               0              0      0
"Narrowings (stenosis)"                                  210                  0               0              0      0
"Narrowings"                                            2330                  0               0              0      0
"Presence of ulcers"                                       0                  5               0              0      0
"Presence of ulcers?"                                      0                 15               0              0      0
"Surface disease (inflammation)"                           0                  0              15              0      0
"Surface disease"                                          0                  0            1085              0      0
"Surface involved by disease (e.g. inflammation)"          0                  0               5              0      0
"Surface involved by disease (inflammation)"               0                  0              60              0      0
"Surface involved by disease"                              0                  0            2100              0      0
"Surface involved by ulcerations"                          0                  0               0           2170      0
"Surface ulcerations"                                      0                  0               0             60      0
"Surface ulcers"                                           0                  0               0           1035      0
"Ulcers (presence/size)"                                   0                  0               0              0      5
"Ulcers"                                                   0                  0               0              0   3240
```

Rule-based medical inference

Finally, your table is all set, and you notice some obvious inconsistencies,
The LLM scored 1 to ulcers but 0 to ‘surface affected by ulcers’

This is what I call rule-based medical inference,
add some final checks to improve medical coherence

Uncertainty estimation

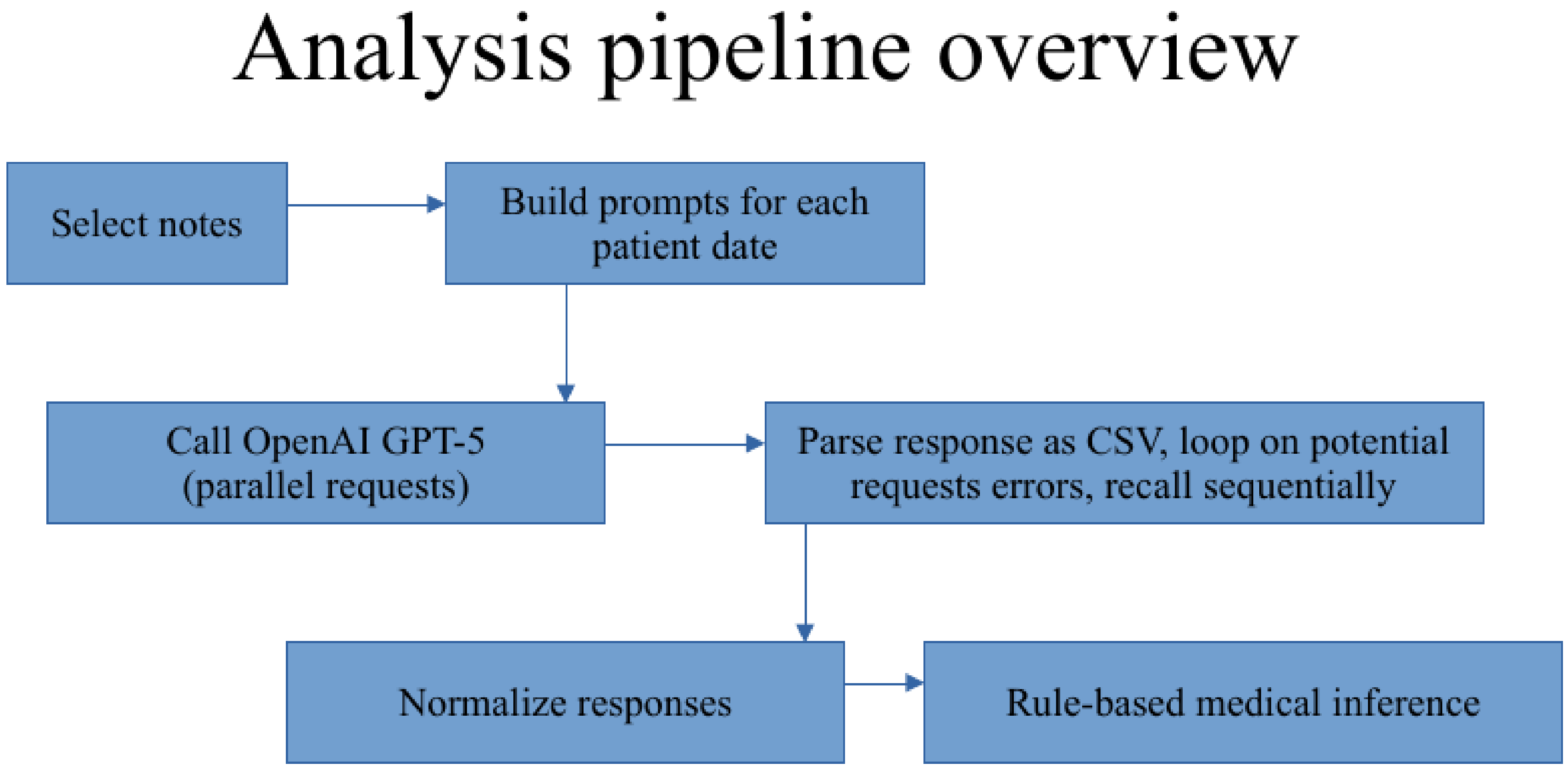
As a final touch, you might want to include some uncertainty estimation,
For example some medical notes will say “mild or moderate disease”,
and you need to make the call

One strategy is to ask the LLM to give you scores about uncertain or contradicting information, then maybe do a second call to reach consensus;
Or you might want to measure your accuracy by including second best guess,

My take on this: ask for low, average and high estimates; use the average estimates generally; and when low and high estimates disagree, flag for uncertainty.
You might also have some leeway in rule-based inference to adapt to estimate level

Concluding superthoughts

This should be clear by now

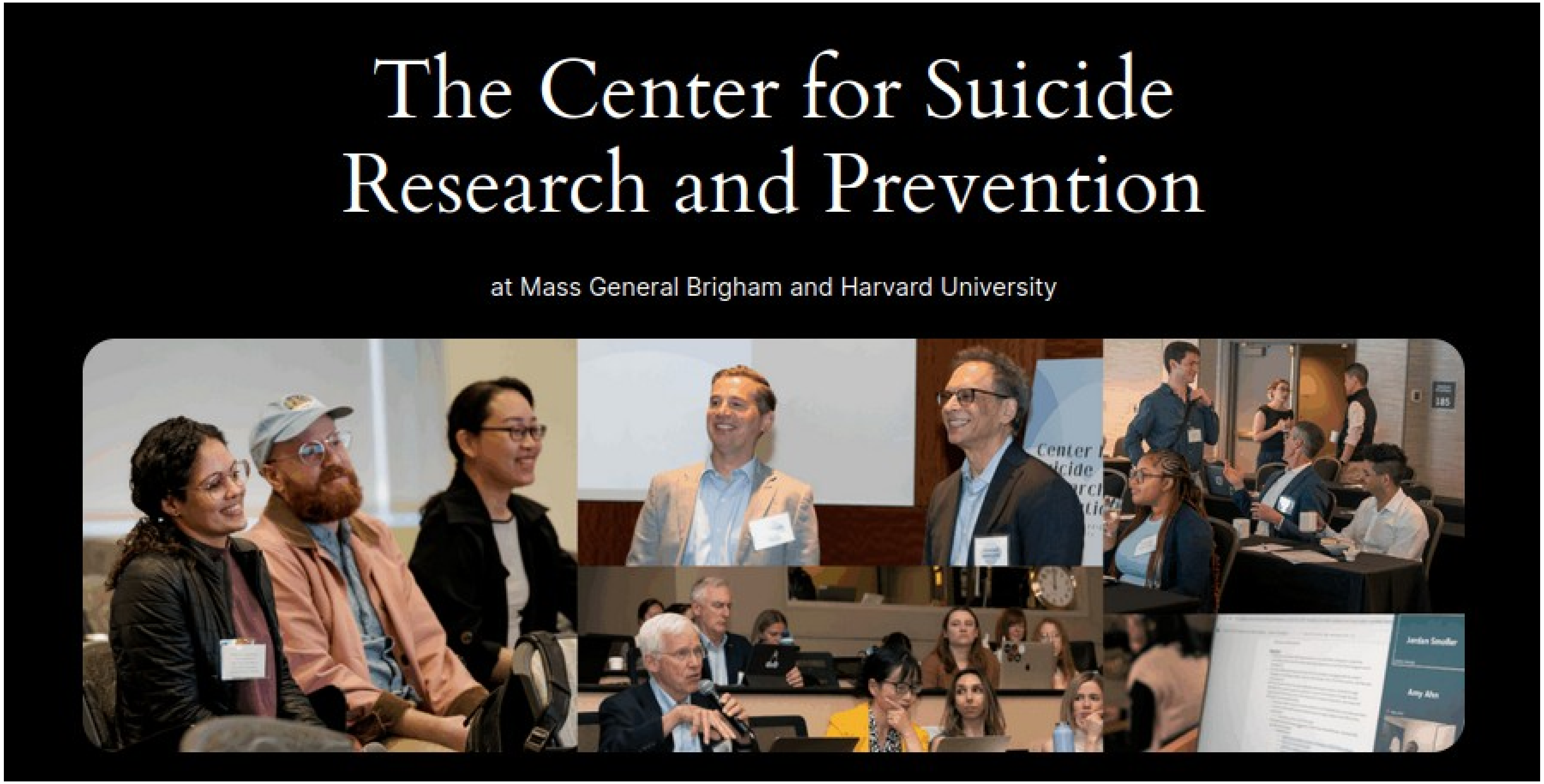

Analysis pipeline overview
Select notes
Build prompts for each patient date
Call OpenAI GPT-5 (parallel requests)
Parse response as CSV, loop on potential requests errors, recall sequentially
Normalize responses
Rule-based medical inference

So now what do we do ?

We can measure IBD treatment efficacy on unprecedented hospital populations
Will cost you around $300 for 50k notes of 20k patients with OpenAI if you don’t have GPUs, should be done in little more than a day

We can revolutionize suicide prevention survival analysis by addressing the long-standing issues of unreliable codified data
Will take you around 2-3 days with 4 GPUs to process 1 year of notes of 3k patients
Likely would cost you around $2.5k on Google Compute
But worry not, we are working on scaling this, we’re now aiming for lifetime of 100k patients (via note selection, chunking, hybrid search, *etc.*), maybe soon 10M

Kudos to the CSRP for their GPUs



The Center for Suicide Research and Prevention

at Mass General Brigham and Harvard University

Next cool ideas

Agentic pipelines for note selection

Currently whenI have a lot of notes to process, I have some simple heuristics to select which ones I want to focus on, and will perform ~5 passes but not necessarily review them all

An agent to dynamically select these heuristics would come in handy

“Derivation mapping” tasks

DSM V provides questionnaires to evaluate personality traits,

e.g. “I am a risk taker” → Very false, Very true, …

We know that impulsivity is a risk factor

Notes contain a lot of diverse information e.g. “jumped over a fire pit at a party”

Promising experiments on deriving personality scores based on notes to model risk

Agents For
Healthcare Chart Review

Thomas Charlon
March 6, 2026 – Scale23x Linux Expo
https://thomaschln.gitlab.io
charlon@protonmail.com