

Don't tell me RAG is easy

A nighttime aerial photograph of a city skyline, likely Boston, featuring the Zakim Bridge, the Prudential Center, and the John Hancock Tower. The city lights reflect off the water in the foreground.

Major Hayden | major@mhtx.net | DevConf.US 2025



Agenda

0. Quick refresher
 - a. Large language models (LLMs)
 - b. Retrieval-augmented generation (RAG)
1. The Fellowship of the RAG
2. The Mines
3. The Road Goes Ever On



Large language models (LLM)

An LLM is:

A very sophisticated auto-completer trained on many, many text examples

Something that takes text input and returns text output

A compressed representation of human writing patterns

An LLM is not:

A knowledge base or API
(no access to real-time information*)

Comprehending what you asked or what it is writing; it only performs sophisticated pattern matching and statistical analysis

Reliable nor deterministic**
(ask twice and get two different answers)

* Although some LLMs respond with up to date information, they're almost always doing this by calling tools that return information to them, and then they process that information as input (just like your original query).

** Under some conditions, LLMs will reply deterministically, but this does require configuration (random seed, other parameters) that might not always be set.



Robert McNees

@mcnees@mastodon.social

My students are often surprised to learn that LLMs aren't answering their questions. Rather, an LLM answers the question "what would a reply to this look like?" It's one of the first things I explain in the "Should I use LLMs?" portion of my syllabus.

But that's not the only problem. Interactions with LLMs feel like a dialog, so it's natural to think the usual rules of conversation apply. You ask a question and expect the response will be an answer *to that question*. It's important to understand that this is not what's happening. An LLM is designed to generate statistically likely responses to the question "What would an answer to this query *sound* like?" This is not the same thing as answering the question. It might produce what you are looking for, or it might not. This is one reason why output from an LLM will sound authoritative even when it's wrong, and apologetic when mistakes are pointed out. It isn't authoritative or apologetic, and it isn't "thinking" about the question. These are just the sorts of responses that best fit a very complicated set of likelihood criteria.

Hide

ALT

Sep 08, 2025, 11:54 PM · · Ivory for iOS

772 boosts · 0 quotes · 803 favorites

[Robert McNees on Mastodon](#)



**Imagine hiring an intern
with an unknown set of biases,
that will never learn anything new,
was trained by people you never met
on a broad, unknown set of topics,
and then you put that intern in front of your most
valuable users.**

That's an LLM.

How do you raise the odds of a quality response with your own information that changes over time?

Your LLM needs a tool! 
RAG is one of those tools that can help.

What is RAG?

Retrieval

Retrieving relevant information from an external source that helps the LLM answer a question better.

Augmented

Enhance the LLM's ability to answer the question accurately and completely with these new pieces of knowledge.

Generation

The LLM can now use its internal knowledge (from training) along with the new input from RAG to generate a better answer.

RAG flow



API

A user sends a question
to an API expecting a
valid answer

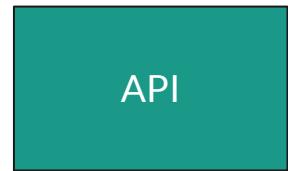


RAG system



LLM

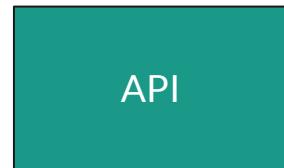
RAG flow



The API asks the RAG system for additional relevant context based on the user's query



RAG flow



The user's query, the RAG context, and the prompt are sent to the LLM to generate a response



RAG flow



The user receives a
correct and complete
response



RAG is much like an open-note exam at school

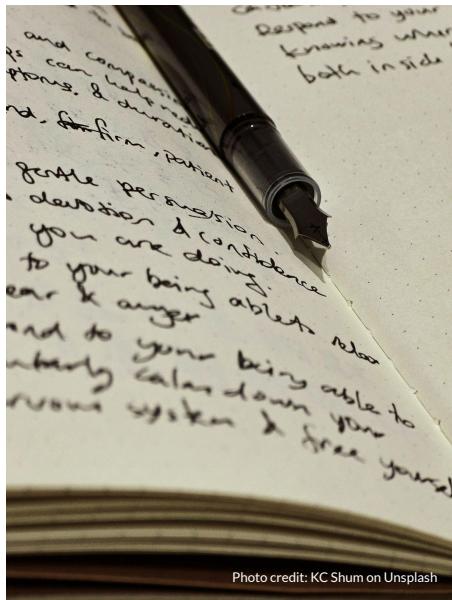


Photo credit: KC Shum on Unsplash

If you know the concepts and material that you're tested on, but you forgot important names, dates, or formulas, **you can quickly refer to your notes and continue with the exam.**

However, if you don't understand the concepts and you're unfamiliar with the material, **no notes are going to help you.**

LLMs must understand the language and concepts for RAG to work. An LLM trained on English documents can't read RAG content in Portuguese.



SOMEONE SAID RAG WAS EASY

BUT I'M WORRIED THAT IT'S NOT

**BETTER
BE CAREFUL**

Act I:

The Fellowship of the RAG

In a hole in the ground lived a documentation system...



“Let’s take all of the documents we have, toss them into a RAG system, and then have LLMs answer questions from our employees and customers!”

...but it turned out to be a difficult path.

That's why you need a fellowship (and a plan).



The senior engineer



“Is it secret? Is it safe?”

- Do some of the documents contain sensitive information or data that has compliance requirements?
- How do you prevent someone from getting access to data that they shouldn't be able to access?
- Can you ensure that data in the RAG system was not tampered with by anyone, internal or external?

Overwhelmed junior developers

Where are the documents stored?

In which formats are documents stored?

Are the documents accurate?



"I don't know half of you half as well as I should like; and I like less than half of you half as well as you deserve."

Is the content up to date?

Are the documents written for the same audience?

How often do these documents change?

Quality engineer

"I'm going to see it through to the end!"

- AI systems aren't deterministic, so a new approach is required
- Need to follow development closely to track results from changing LLMs, fine tuning, RAG content, embedding models, prompts, random seeds, maximum token lengths, AI frameworks...



"There's some good in this world, Mr. Frodo, and it's worth testing for!"

The AI enthusiast



“We needs it to be perfect, precious. We saw 20 useful, very useful, posts on HackerNews this week. Yes, we did! You will read them all!”

- LLM capabilities change constantly
- RAG content changes constantly
- Customer questions change constantly
- AI system strategies change constantly
- HackerNews changes constantly



**The perfect
solution is the
enemy of progress**

Act II: The Mines

Start with a user story

Build a story with this simple format:

- As a *(role of a person using your RAG system)*
- I want to *(something they can do)*
- So I can *(benefit they get)*

Make it achievable, measurable, and meaningful.
This becomes your project's north star.

Internal: As a customer support engineer, I want to search internal knowledge about known issues with specific hardware configurations so I can help customers faster.

External: As a sysadmin, I want to search for troubleshooting steps when my system shows specific error messages so I can resolve issues without opening support tickets.

Align the stakeholders

Who are the end users?

What information is the most useful for a first release?

How do we know when the first release is good enough to ship?

When does it need to be ready?

What is the budget for infrastructure?

How will we measure our success?



The first stumbles



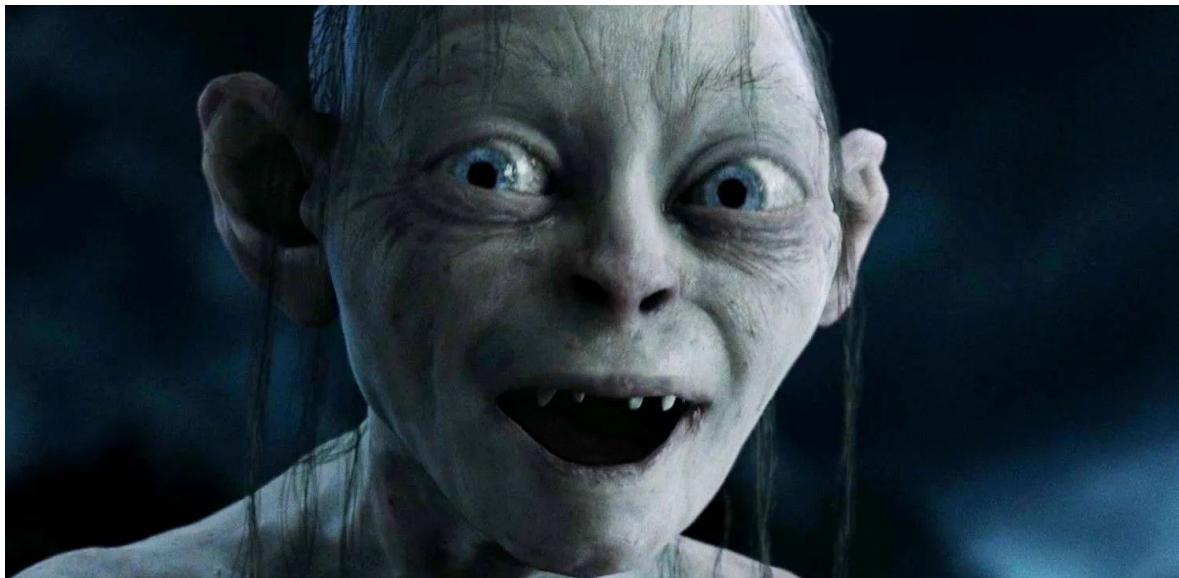
Documents are easy for humans but difficult for computers. Parse your documents consistently for every format. [Docling](#) is helpful here.

Choose an initial method for searching your documents (more on the next slide) while leaving the door open for other methods in the future.

Score your results using an objective method, such as [LLM-as-a-judge](#), to check the search result quality as you develop.

Keep your changes small and do team demos frequently.

“So bright, so beautiful, a new HackerNews article.”



Stay on the path!



RAG search strategies

Keyword
(lexical)

Very fast traditional text search looking for phrases/keywords

Doesn't look at meaning/semantics; would miss truck/lorry or car/automobile

Vector
(semantic)

Uses vectors to find similarities with more context and meaning

Requires embedding text first into special databases; less precise matching

Graph
(relationship)

Uses knowledge graphs or doc relationships to find results

Must build these relationships and graphs first if they do not exist in your documents

Hybrid

Combines multiple strategies; usually keyword + vector

Still requires expensive embedding and also usually requires a re-ranking mechanism

Splitting and chunking for vector search



Embedding models have a maximum context length, so split and chunk your lengthy documents:

- **Split:** Break documents up via chapters, subchapters, sections, topics, or other boundaries that a human would recognize
- **Chunk:** Break up splits into smaller pieces that fit in your embedding model's context window

Scoring and judging is critical here!

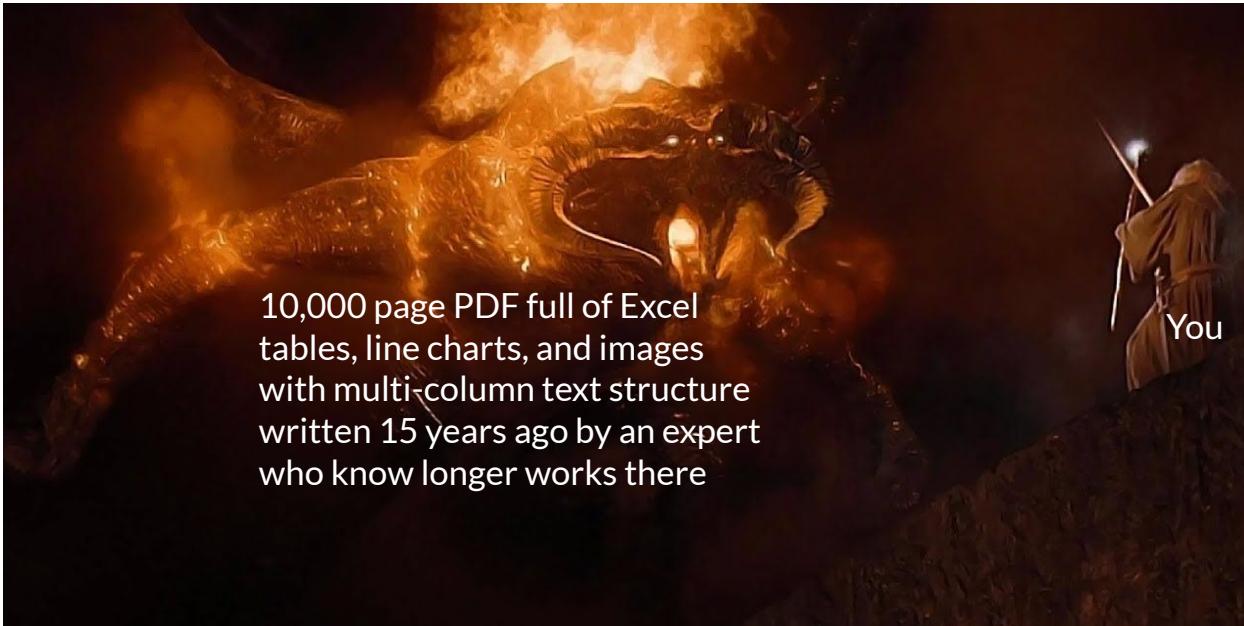
You can check the score and then adjust chunk sizes, chunk overlaps, and embedding models. Then, check the score again.

“Put the PDFs in the bucket, we says. RAG, the precious, is easy.”



Stay on the path!

Some documents will not bend to your will



YOU SHALL NOT PARSE!





Keep the end goal in sight for tough documents

Find owners for the document who can revise it, update it, or otherwise make it easier to parse.

Put the document into a list of documents to address once you're further along.

Docling* offers some powerful OCR and image extraction tools that can extract really tough PDFs.

Keep in mind that some content should likely stay buried where you found it.

Budget more time for processing documents than anything else. **Garbage in, garbage out.**

Consider methods for pre-extracting content from markup (XML, JSON, etc) into a friendlier text format, and then ingest the document.

* There's a [great research paper](#) from the docling team about parsing the most difficult documents.



Consider common failure scenarios

Incorrect response

The RAG search simply returned the wrong information. Try searching for strings you know are in your documents and ensure you're using the same embedding model for search as you are when creating vectors.

Correct & incomplete

A recipe for beef wellington was returned, but only steps 4-8 of 20. Consider expanding RAG context to logical (human recognizable) boundaries, especially when dealing with lists.

LLM Hallucination

RAG searching was fine, but the LLM does not know enough about the topic to use the information that you provided. Re-evaluate your selection of LLM (biggest impact) or provide more context via RAG (lowest impact).

Correct & irrelevant

"How do I reset my password" returns best practices for making a new password instead of the steps to reset it. Classify intent or use LLMs to rewrite the query before doing the RAG search.

Hallucinations

LLMs hallucinate when they generate information for topics that are not in their training data

If your RAG results are accurate and complete, but the LLM response remains incorrect, your model might need more fine tuning (expensive) on your data

It may be easier to switch to another LLM which was trained on material more relevant to your documentation



Model size matters

Smaller models are trained on less data (fewer parameters) and they often need more context from RAG

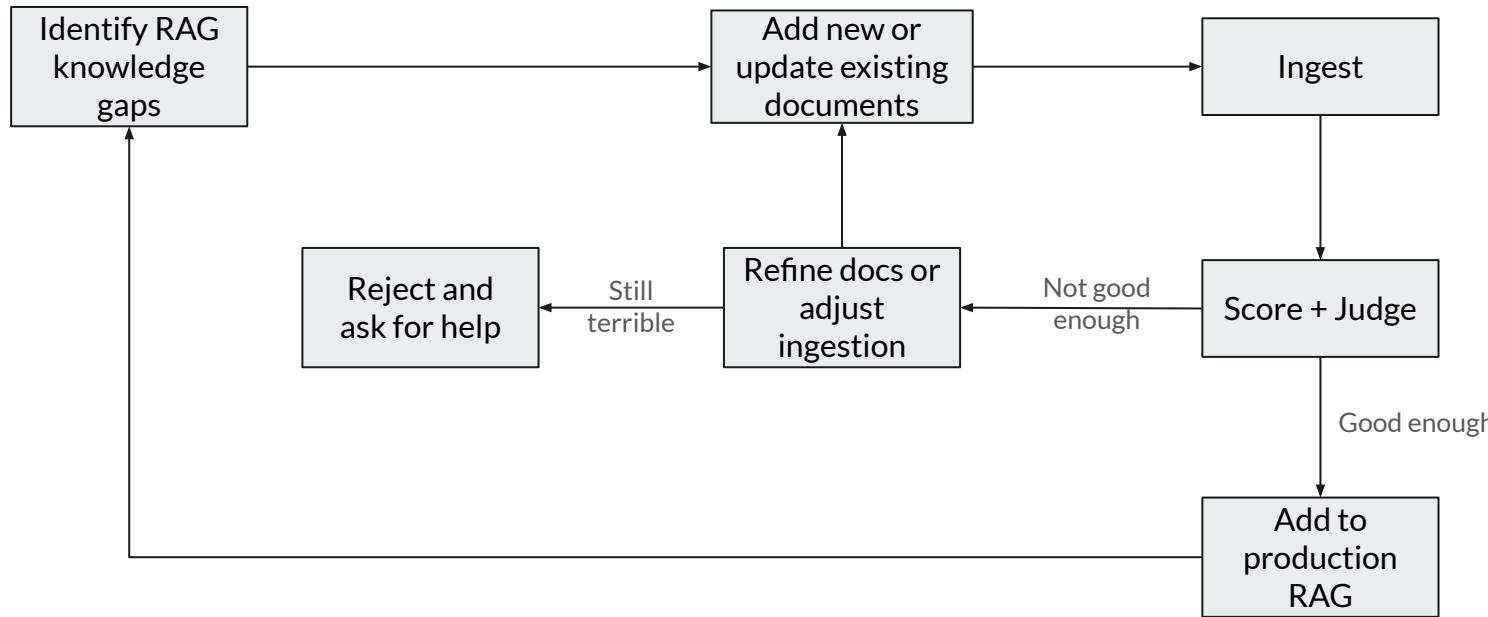
Larger models (Llama 4 Scout/Maverick) or frontier models (Claude Opus, GPT-4o) do not require as much context as their training is extensive and tool usage capability is stronger

You can get away with lower accuracy RAG with the larger models, but not with the smaller ones



Act III. The Road Ever Goes On

Develop a RAG pipeline for improving context



Arriving at “done enough”

RAG is a continuous journey, but eventually you’re ready for launch.

Remember that accuracy will always fluctuate and edge cases will always appear, but look for broad topics which are missing from the RAG database that should be added.

Identify troublesome topics where documentation or RAG ingestion should be modified.





First production deployment

Optimize your RAG database configuration for high concurrency and identify data to be indexed for fast querying.

Add logging of queries received and the responses provided from the RAG system along with any similarity scores so you can track down anomalies or poor results.

Find internal testers who are willing to provide feedback on the answers they receive.

Build a method for testers and team members to log bugs about problems with RAG responses in an organized way.

"But late in the night the watchmen cried out, and all awoke. The moon was gone. Stars were shining above; but over the ground there crept a darkness blacker than the night. On both sides of the river it rolled towards them, going northward."

The Two Towers by J.R.R. Tolkien

RAG lessons from building RHEL Lightspeed

RAG is not the destination, but an ongoing quest

Your fellowship matters more than your
technology stack



"I will take the Ring, though I do not know the way." - Frodo

RAG lessons from building RHEL Lightspeed

Start small, fail fast, and learn constantly

Measure everything and optimize what matters



"I will take the Ring, though I do not know the way." - Frodo

RAG lessons from building RHEL Lightspeed

Documentation quality matters
much more than RAG strategy

Garbage in,
Garbage out



"I will take the Ring, though I do not know the way." - Frodo

Thank you!

Major Hayden | major@mhtx.net | DevConf.US 2025