

RADONC AI CURRICULUM

Introduction to Deep Learning

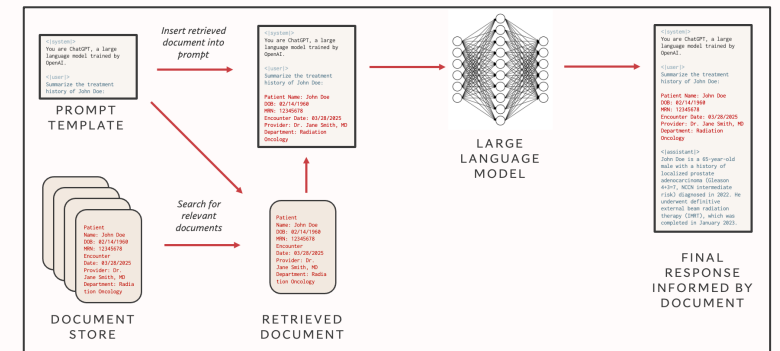
LECTURE 9: Retrieval-Augmented Generation and Agents

ANDREW Y. K. FOONG, PH.D.

June 12th, 2026



Radiation
Oncology
AI & Data Analytics
AIDA



Today's lecture

1. RAG
 - *What is retrieval-augmented generation?*
2. Keyword search
 - *Determining relevance by matching terms*
3. Two-stage RAG
 - *Using LLMs to determine relevance*
4. Vector RAG
 - *Finding relevant documents with embeddings*
5. Agentic AI
 - *Giving LLMs tools*
6. Multiple tool use
 - *LLMs that can take actions*
7. Q&A

What about data we didn't train on?

- Lots of information we care about will *not* be learned during pre-training, SFT or RLHF:
 - Information that arose *after* the LLM was trained.
 - What's today's weather?
 - Can't afford to retrain often
 - Information in private databases, e.g.
 - Electronic health records
 - Emails
 - Intellectual property
- How can LLMs access this information?
 - *Retrieval-augmented generation* (RAG)
 - *AI agents*

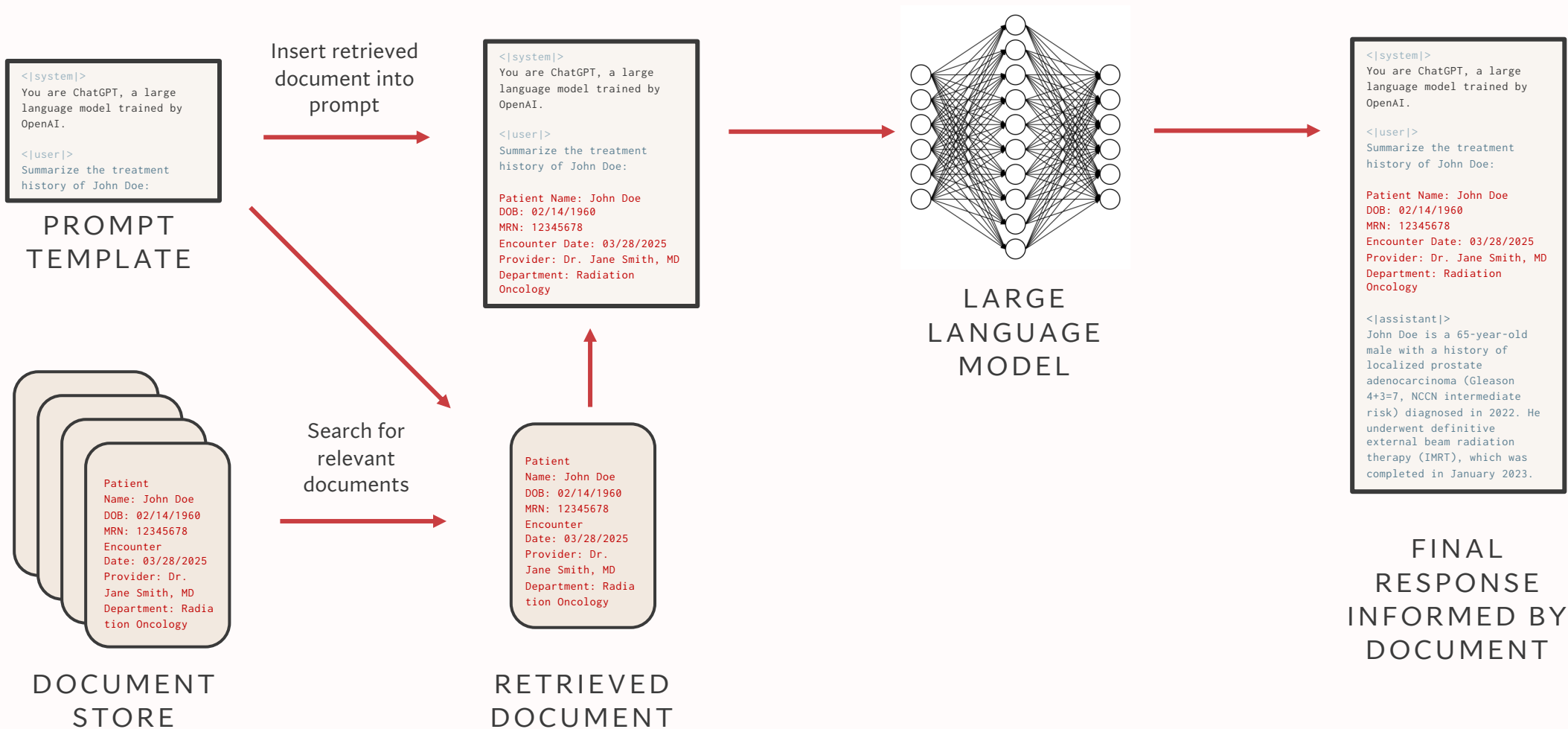
RAG

What is retrieval-augmented generation?

Retrieval-augmented generation

- **KEY PROMPTING PRINCIPLE:** include relevant context.
- Retrieval augmented generation (RAG) is a method that:
 - Automatically retrieves relevant documents from database.
 - Inserts it into the prompt.

Retrieval-augmented generation



Why does RAG work?

- LLMs may not have information to solve a task within their weights
- *But* they have learned how to *repeat* and *summarize* the contents of their prompts.

EXAMPLE TRAINING DATA: supervised fine-tuning to repeat input text

```
<|system|>
```

```
Repeat what the user says
```

```
<|user|>
```

```
John Doe has a history of localized  
prostate cancer
```

```
<|assistant|>
```

```
John Doe has a history of localized  
prostate cancer
```

Why does RAG work?

- LLMs may not have information to solve a task within their weights
- *But* they have learned how to *repeat* and *summarize* the contents of their prompts.

EXAMPLE TRAINING DATA: supervised fine-tuning to repeat input text

```
<|system|>
Repeat what the user says
<|system|>
Repeat what the user says
<|user|>
John: Don't count your chickens before
pros: they even hatch
<|assistant|>
John: Don't count your chickens before
pros: they even hatch
```

Why does RAG work?

- LLMs may not have information to solve a task within their weights
- *But* they have learned how to *repeat* and *summarize* the contents of their prompts.

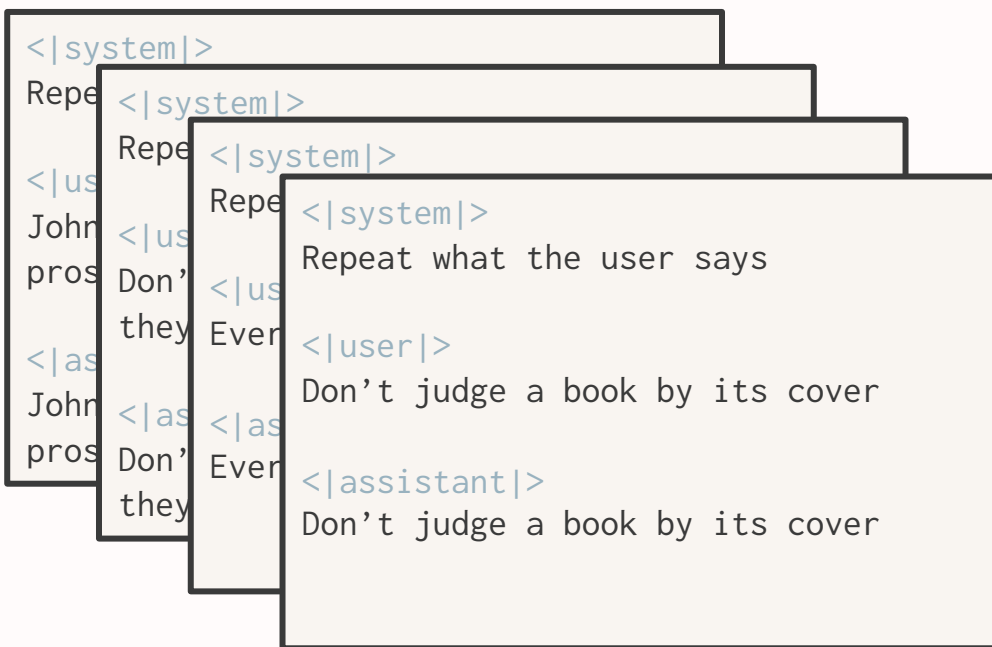
EXAMPLE TRAINING DATA: supervised fine-tuning to repeat input text

```
<|system|>
Repeat what the user says
<|system|>
Repeat what the user says
<|user|>
John: Don't they
prosa: Every cloud has a silver lining
<|assistant|>
John: Don't they
prosa: Every cloud has a silver lining
```


Why does RAG work?

- LLMs may not have information to solve a task within their weights
- *But* they have learned how to *repeat* and *summarize* the contents of their prompts.

EXAMPLE TRAINING DATA: supervised fine-tuning to repeat input text



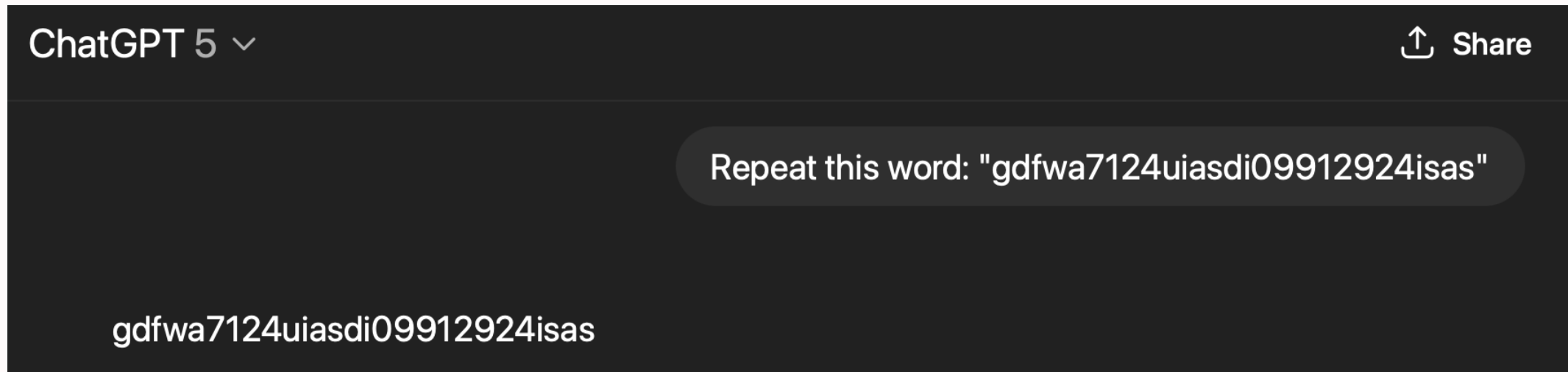
The diagram illustrates training data for supervised fine-tuning to repeat input text. It shows four overlapping boxes, each representing a training example. Each example consists of a system prompt, a user input, and an assistant response that repeats the user input. The examples are as follows:

- Example 1: System prompt: <|system|> Repeat what the user says; User input: <|user|> John pros; Assistant response: <|assistant|> John pros
- Example 2: System prompt: <|system|> Repeat what the user says; User input: <|user|> Don't judge a book by its cover; Assistant response: <|assistant|> Don't judge a book by its cover
- Example 3: System prompt: <|system|> Repeat what the user says; User input: <|user|> Don't judge a book by its cover; Assistant response: <|assistant|> Don't judge a book by its cover
- Example 4: System prompt: <|system|> Repeat what the user says; User input: <|user|> Don't judge a book by its cover; Assistant response: <|assistant|> Don't judge a book by its cover

After seeing enough examples, the model can learn to “parrot” the user input when instructed to, even if the input is brand new

Why does RAG work?

- ChatGPT succeeds perfectly in parroting, even though this random string was not in the training data.
- The neural network learns to output this *by statistical association*, not by using a copy/paste function.
- Earlier versions of ChatGPT struggled with this task.



ChatGPT, accessed November 24th, 2025

Why does RAG work?

- Copying purely by statistical association can lead to unexpected errors—especially when copying large amounts of text.
- *This is wrong!* ChatGPT actually misses out one of the lines of gibberish.

```
Repeat this word:  
*gdfwa7124uiasdi09912924isassa8y284ujsafoja9js9af9u321ir0i-  
0i3r210-ifa-0if-aksfo-12-2o-fi31jf-01ifkcqss0ik0-ai0-i32-1or-rk-  
031-  
Ogdfwa7124uiasdi09912924isassa8y284ujsafoja9js9af9u321ir0i-  
0i3r210-ifa-0if-aksfo-12-2o-fi31jf-01ifkcqss0ik0-ai0-i32-1or-rk-  
031-  
Ogdfwa7124uiasdi09912924isassa8y284ujsafoja9js9af9u321ir0i-  
0i3r210-ifa-0if-aksfo-12-2o-fi31jf-01ifkcqss0ik0-ai0-i32-1or-rk-  
031-  
1gdfwa7124uiasdi09912924isassa8y284ujsafoja9js9af9u321ir0i-  
0i3r210-ifa-0if-aksfo-12-2o-fi31jf-01ifkcqss0ik0-ai0-i32-1or-rk-  
031-  
Ogdfwa7124uiasdi09912924isassa8y284ujsafoja9js9af9u321ir0i-  
0i3r210-ifa-0if-aksfo-12-2o-fi31jf-01ifkcqss0ik0-ai0-i32-1or-rk-  
031-  
Ogdfwa7124uiasdi09912924isassa8y284ujsafoja9js9af9u321ir0i-  
0i3r210-ifa-0if-aksfo-12-2o-fi31jf-01ifkcqss0ik0-ai0-i32-1or-rk-  
031-  
Ogdfwa7124uiasdi09912924isassa8y284ujsafoja9js9af9u321ir0i-  
0i3r210-ifa-0if-aksfo-12-2o-fi31jf-01ifkcqss0ik0-ai0-i32-1or-rk-  
031-  
Oqadqwewfqwwqgdfwa7124uiasdi09912924isassa8y284ujsafoja9  
js9af9u321ir0i-0i3r210-ifa-0if-aksfo-12-2o-fi31jf-01ifkcqss0ik0-  
ai0-i32-1or-rk-031-  
Ogdfwa7124uiasdi09912924isassa8y284ujsafoja9js9af9u321ir0i-  
0i3r210-ifa-0if-aksfo-12-2o-fi31jf-01ifkcqss0ik0-ai0-i32-1or-rk-  
031-0"
```

Here you go:

```
gdfwa7124uiasdi09912924isassa8y284ujsafoja9js9af9u321ir0i-0i3r210-ifa-0if-aksfo-12-2o-  
fi31jf-01ifkcqss0ik0-ai0-i32-1or-rk-031-  
Ogdfwa7124uiasdi09912924isassa8y284ujsafoja9js9af9u321ir0i-0i3r210-ifa-0if-aksfo-12-2o-  
fi31jf-01ifkcqss0ik0-ai0-i32-1or-rk-031-  
Ogdfwa7124uiasdi09912924isassa8y284ujsafoja9js9af9u321ir0i-0i3r210-ifa-0if-aksfo-12-2o-  
fi31jf-01ifkcqss0ik0-ai0-i32-1or-rk-031-  
1gdfwa7124uiasdi09912924isassa8y284ujsafoja9js9af9u321ir0i-0i3r210-ifa-0if-aksfo-12-2o-  
fi31jf-01ifkcqss0ik0-ai0-i32-1or-rk-031-  
Ogdfwa7124uiasdi09912924isassa8y284ujsafoja9js9af9u321ir0i-0i3r210-ifa-0if-aksfo-12-2o-  
fi31jf-01ifkcqss0ik0-ai0-i32-1or-rk-031-  
Ogdfwa7124uiasdi09912924isassa8y284ujsafoja9js9af9u321ir0i-0i3r210-ifa-0if-aksfo-12-2o-  
fi31jf-01ifkcqss0ik0-ai0-i32-1or-rk-031-  
Oqadqwewfqwwqgdfwa7124uiasdi09912924isassa8y284ujsafoja9js9af9u321ir0i-0i3r210-ifa-  
0if-aksfo-12-2o-fi31jf-01ifkcqss0ik0-ai0-i32-1or-rk-031-  
Ogdfwa7124uiasdi09912924isassa8y284ujsafoja9js9af9u321ir0i-0i3r210-ifa-0if-aksfo-12-2o-  
fi31jf-01ifkcqss0ik0-ai0-i32-1or-rk-031-0
```

ChatGPT, accessed November 24th, 2025

Why does RAG work?

- The same is true for *summarization*:

EXAMPLE TRAINING DATA: supervised fine-tuning to **summarize** input text

```
<|system|>
Summarize what the user says

<|user|>
The patient presented with fever,
chills, and a productive cough
lasting three days. Chest X-ray
showed infiltrates consistent with
pneumonia. He was started on
antibiotics and showed improvement.

<|assistant|>
Patient treated for pneumonia with
antibiotics after three-day illness.
```

Why does RAG work?

- The same is true for *summarization*:

EXAMPLE TRAINING DATA: supervised fine-tuning to **summarize** input text

```
<|system|>
Summarize
<|system|>
Summarize what the user says
<|user|>
The patient had a fever, chills,
lasting cough, and showed signs of
pneumonia. The study compared three machine
antibiotic resistance.
<|assistant|>
Patient had a fever, chills,
antibiotic resistance.
Random forest achieved highest
accuracy for readmission prediction
(AUC 0.87).
```

Why does RAG work?

- The same is true for *summarization*:

EXAMPLE TRAINING DATA: supervised fine-tuning to **summarize** input text

```
<|system|>
Summariz
<|system|>
Summariz
<|user|>
The pat
chills,
lasting
showed
pneumon
antibio
<|assis
Patient
antibio
<|system|>
Summarize what the user says
<|user|>
The stud
learning
hospital
forest m
regressi
0.87.
<|assis
Random f
accuracy
(AUC 0.8
<|system|>
Summarize what the user says
<|user|>
The quarterly report shows that
revenue increased by 15% while
expenses remained stable. Growth was
mainly driven by the new
subscription model.
<|assistant|>
Revenue rose 15% due to new
subscription model, with stable
expenses.
```

Why does RAG work?

- The same is true for *summarization*:

EXAMPLE TRAINING DATA: supervised fine-tuning to **summarize** input text

```
<|system|>
Summariz
<|system|>
Summariz
<|user|>
The pat
chills,
lasting
showed
pneumon
antibio
<|assis
Patient
antibio
<|system|>
Summarize what the user says
<|user|>
The stud
learning
hospital
forest m
regressi
0.87.
<|assis
Random f
accuracy
(AUC 0.8
<|system|>
Summarize what the user says
<|user|>
The quarterly report shows that
revenue increased by 15% while
expenses remained stable. Growth was
mainly driven by the new
subscription model.
<|assistant|>
Revenue rose 15% due to new
subscription model, with stable
expenses.
```

After seeing enough examples, the model can learn to summarize the user input when instructed to, even if the input is brand new

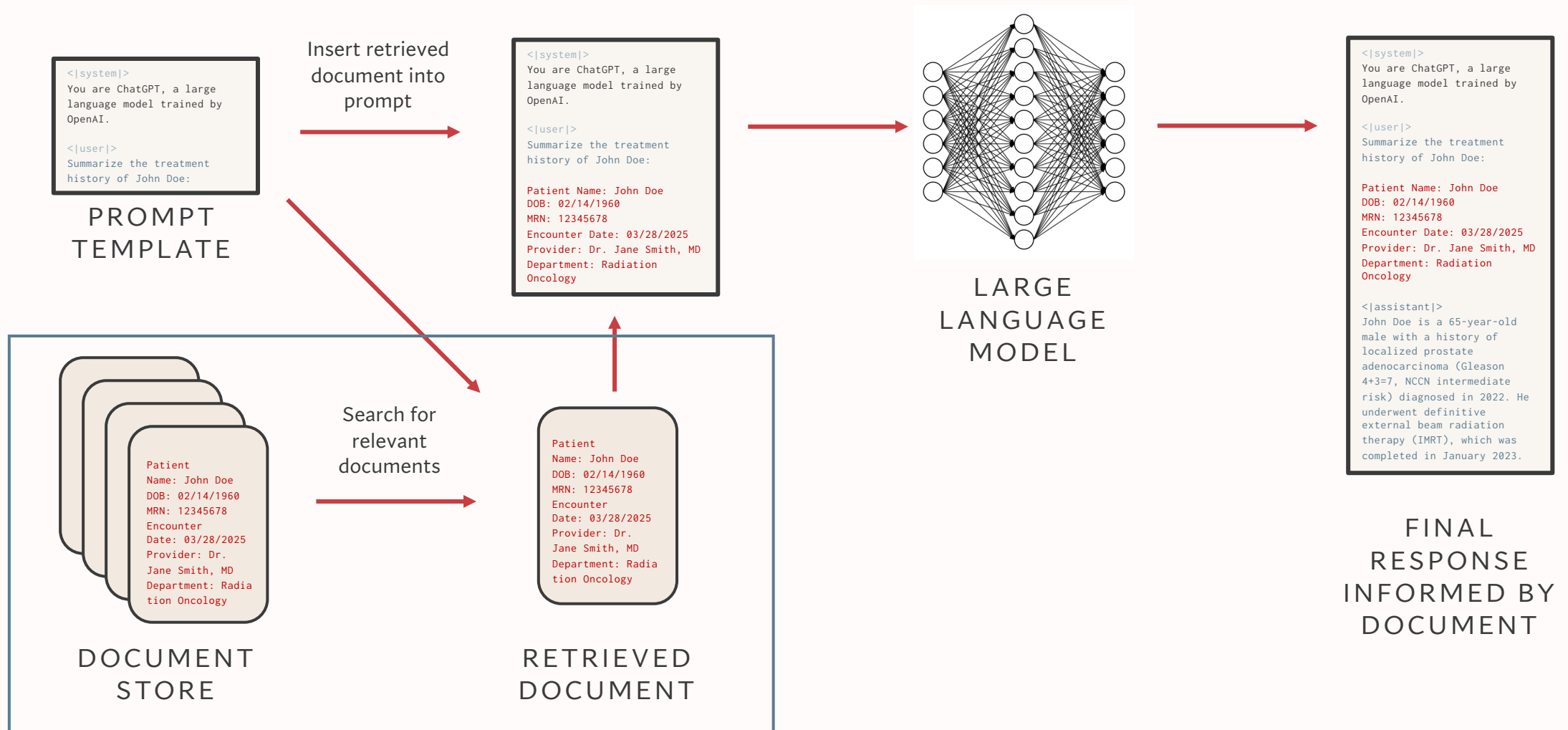
Why does RAG work?

- This goes beyond “parroting” and summarization.
- LLMs are great at performing all kinds of language manipulations of text that appears within their context.
- The key is to get the *right information* into the LLM’s context.
- **FOR EXAMPLE:**
 - An LLM cannot summarize a patient’s history if their history isn’t in the context
 - An LLM cannot say what season we are in if today’s date is not given in the context
 - An LLM cannot write minutes for a meeting if the transcript isn’t in the context
 - Etc.
- **NOTE:** *even if* the right information is present, the LLM *might still fail*.

Keyword search

Determining relevance by matching terms

Document search in RAG



Keyword search in RAG

1. **CHUNKING:** documents split into chunks, where the size of each chunk must be smaller than the maximum context size of the LLM.
2. **INDEXING:** keywords in each document are identified, and database is created of which documents are associated with which keywords.
3. **USER QUERY:** LLM user asks a question.
4. **RETRIEVAL:** keywords are extracted from the question and matching document chunks are retrieved.
5. **GENERATION:** matching chunks inserted into context and LLM answers the user query.

Keyword search in RAG

- **ADVANTAGES:** fast, straightforward to implement. Many reliable keyword search algorithms available online.
- **DISADVANTAGES:** keyword search struggles with *semantic* similarity.

Two-stage RAG

Using LLMs to determine relevance

Care journey can generate hundreds of notes



Care journey can generate hundreds of notes

UROLOGY



Some questions depend on understanding entire timeline:

- Which patients had a recurrence?
- How long between initial treatment and recurrence?

ONCOLOGY

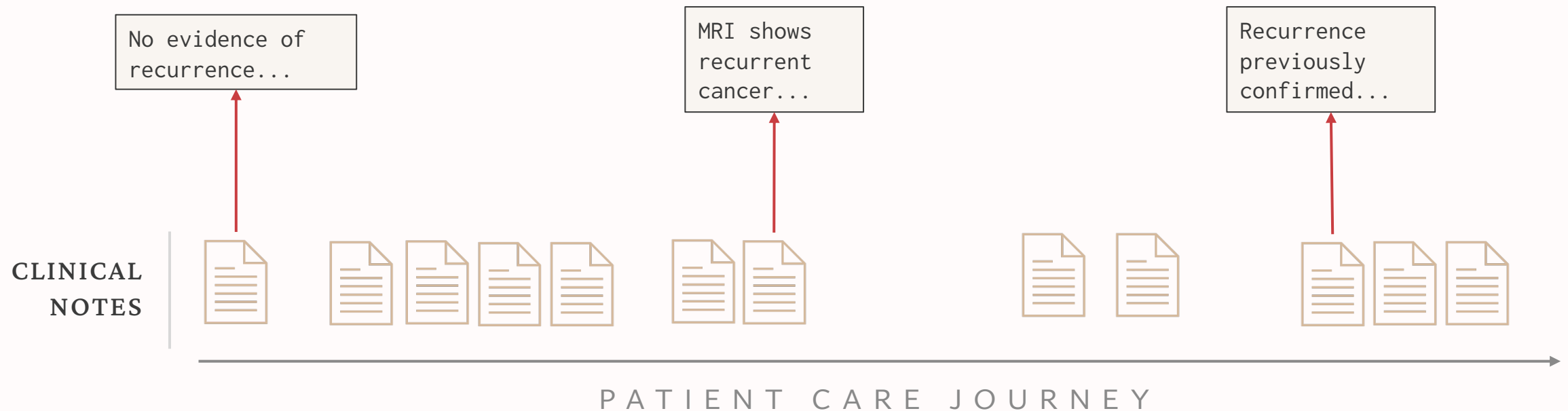


PATIENT CARE JOURNEY

The needle in a haystack problem

- LLM performance tends to **degrade** with longer inputs.
- Often with clinical notes, only small fraction relevant to answer the query: *the needle in a haystack*.

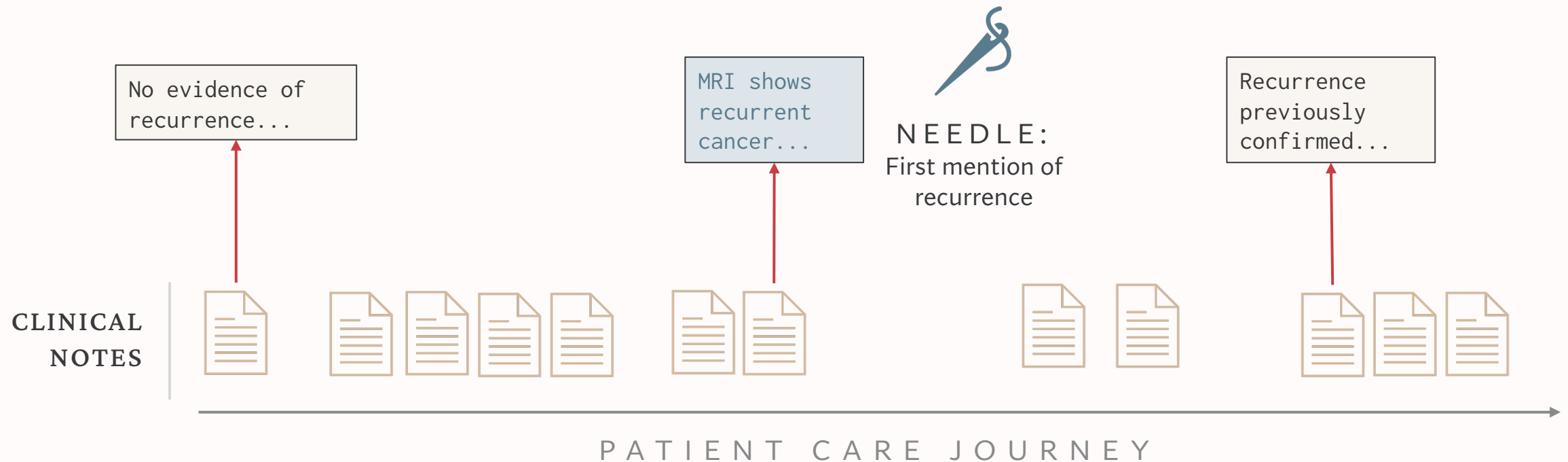
EXAMPLE: RECURRENCE



The needle in a haystack problem

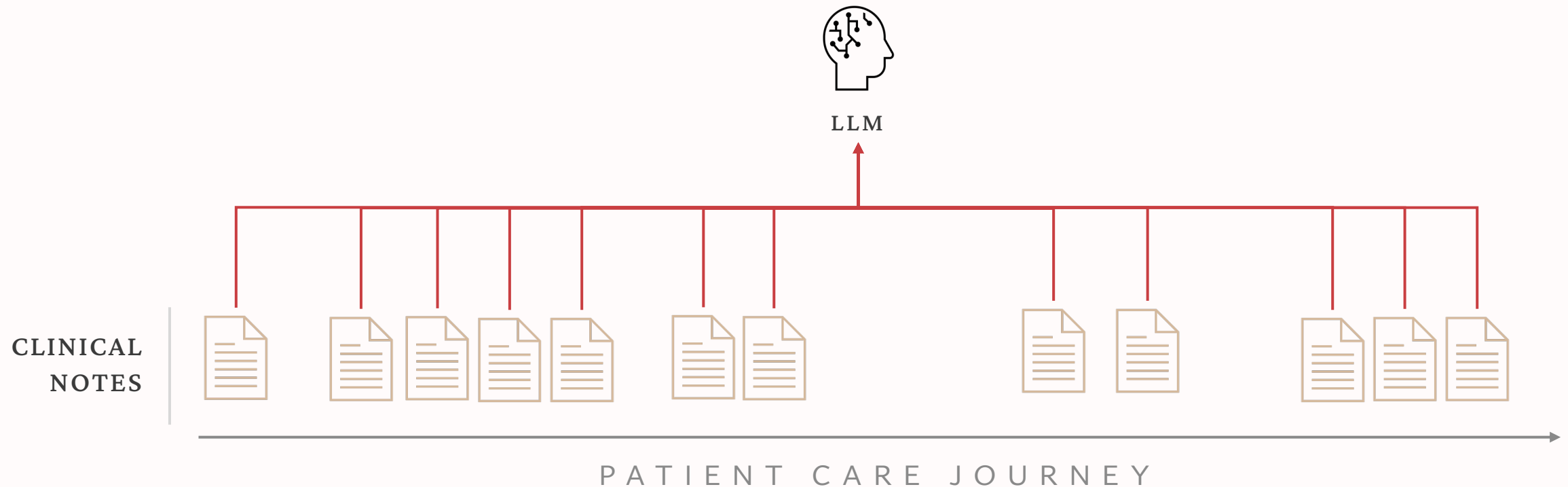
- LLM performance tends to **degrade** with longer inputs.
- Often with clinical notes, only small fraction relevant to answer the query: *the needle in a haystack*.

EXAMPLE: RECURRENCE



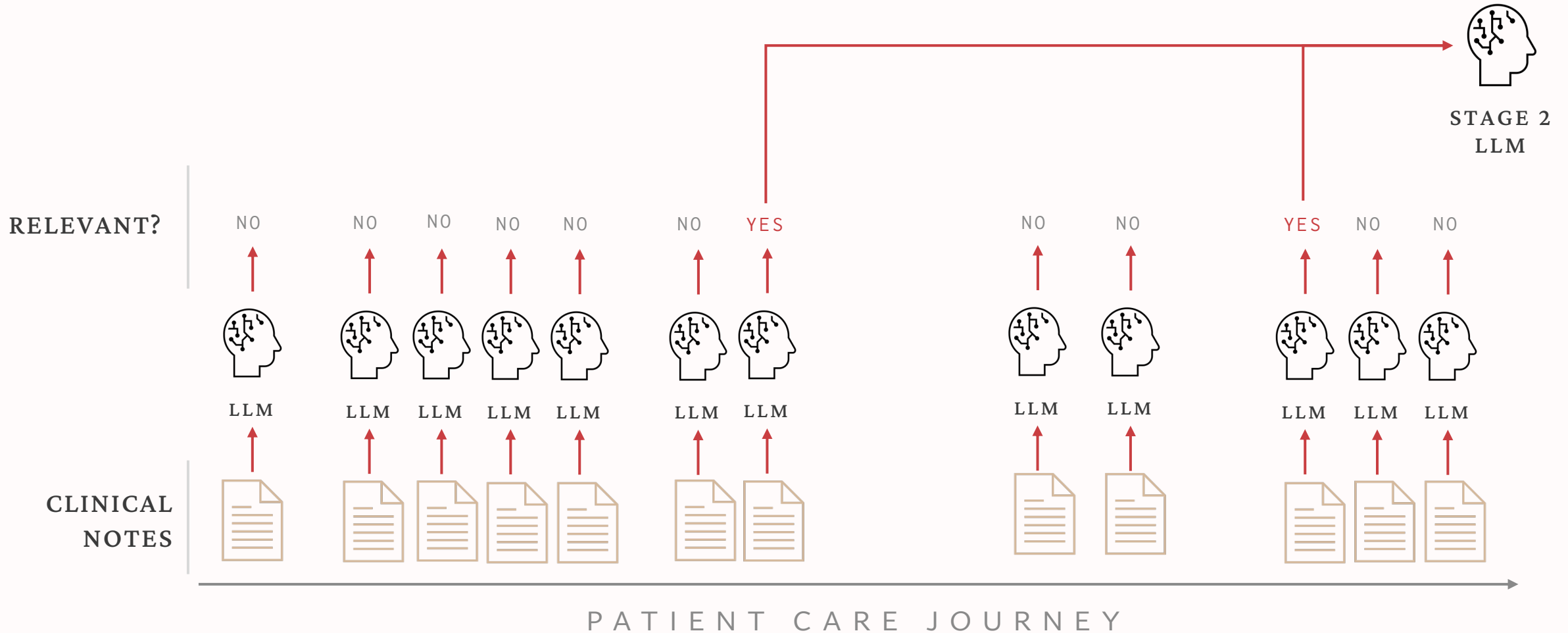
The needle in a haystack problem

- **NAÏVE APPROACH:** Copy all notes into single prompt, ask LLM to find date of first mention of recurrence, if any.
 - Accuracy likely low—needle in a haystack.
 - Expensive/slow—extremely long prompt.



Two-stage approach

- Stage 1: Cheap, fast LLM processes each note in parallel, and flags **relevant** notes.
- Stage 2: **Relevant** notes only read by more advanced LLM. BONUS: get references.



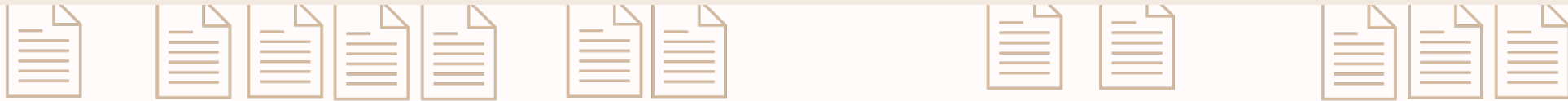
Two-stage approach

- Stage 1: Cheap, fast LLM processes each note in parallel, and flags **relevant** notes.
- Stage 2: **Relevant** notes only read by more advanced LLM. BONUS: get **references**.

Our experience:

- 98% accuracy determining which patients had recurrence.
- Up to hundreds of clinical notes read per patient.
- Approx. \$0.20 per patient.

CLINICAL
NOTES

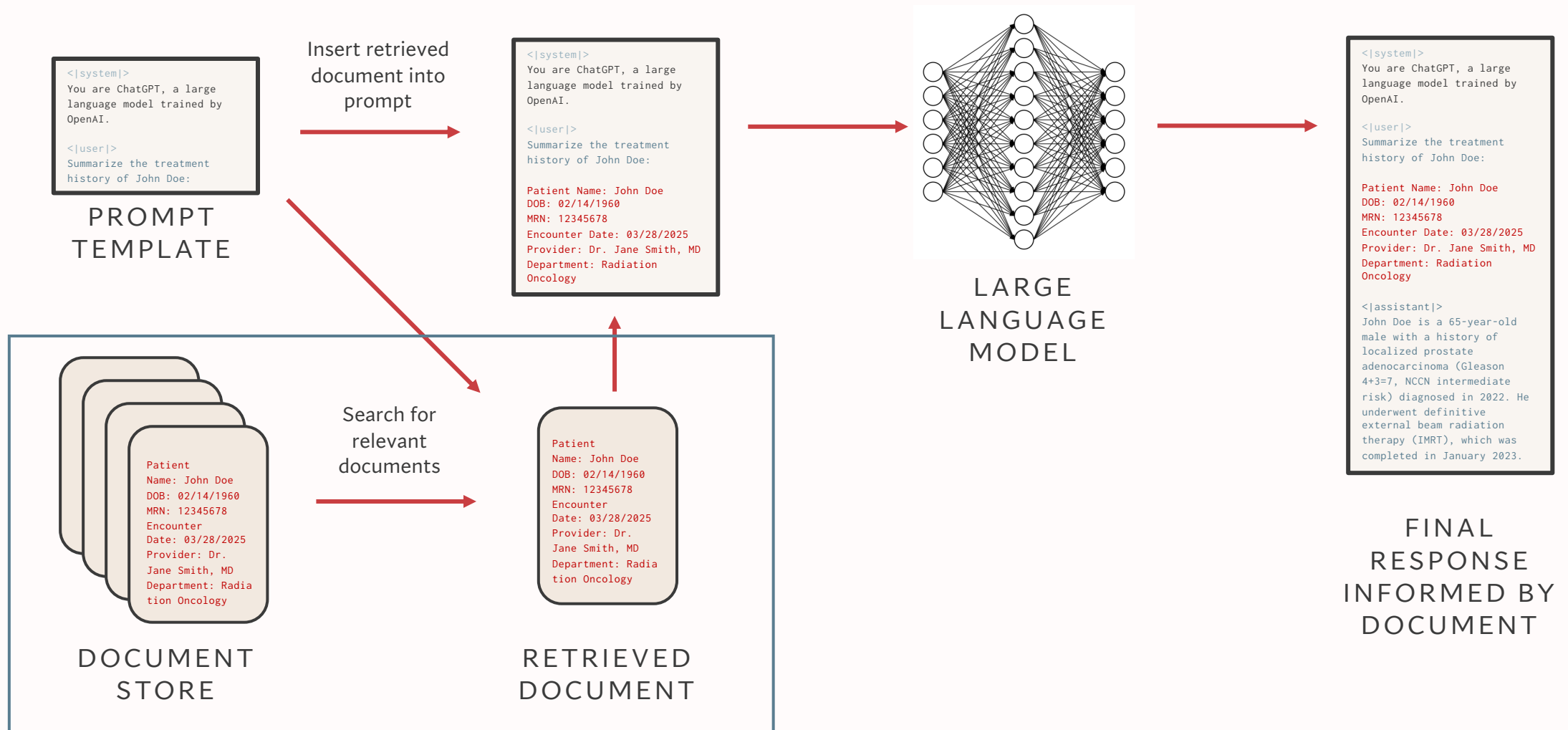


PATIENT CARE JOURNEY

Vector RAG

Finding relevant documents with embeddings

Document search in RAG



RAG recap

1. KEYWORD/TERM-BASED RAG

- Use keyword search to find relevant chunks
- Fast and easy to implement
- **Struggles with understanding synonyms/abstract concepts**

2. TWO-STAGE RAG

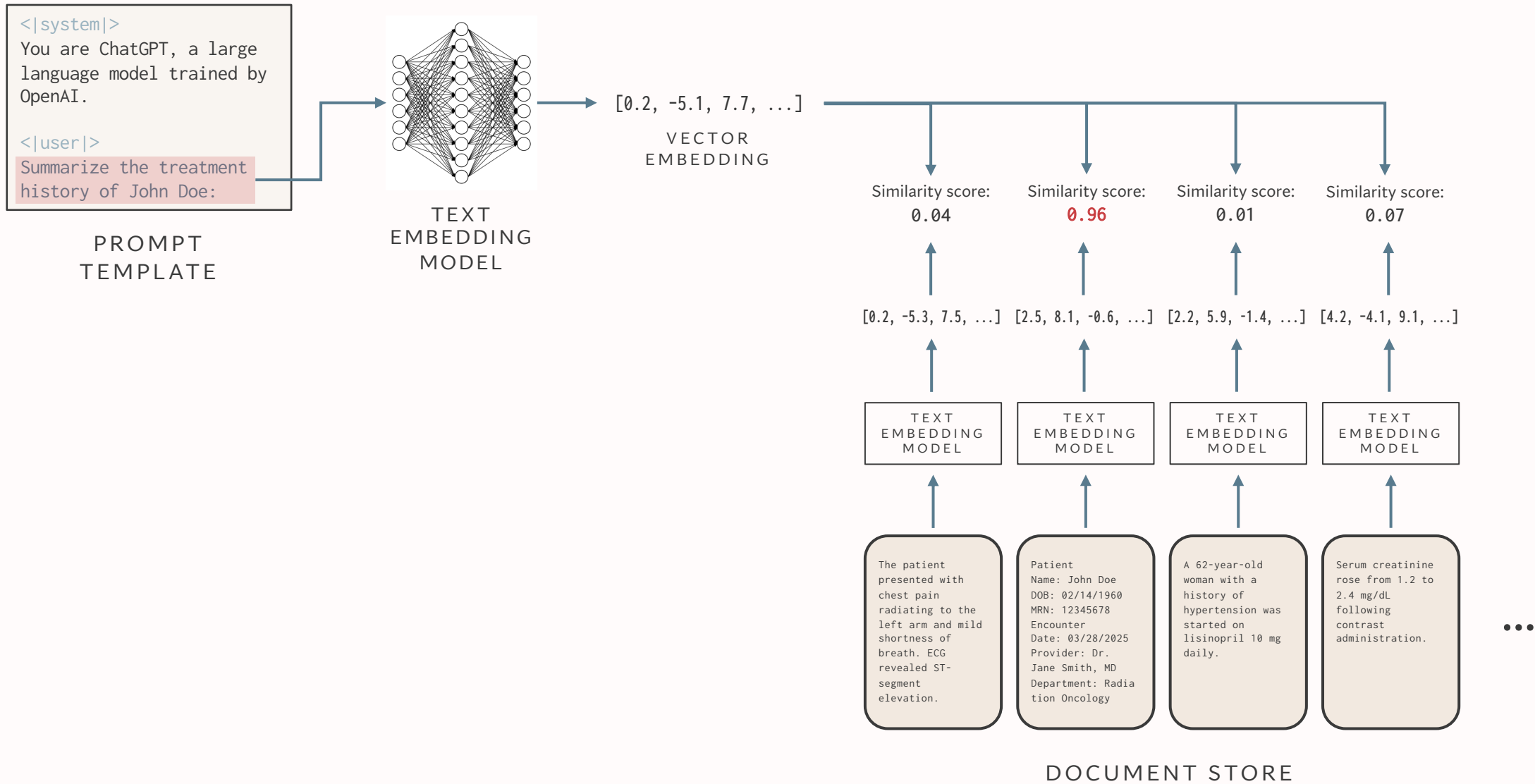
- Use a cheaper/faster LLM to find relevant chunks
- Easy to implement
- *Much* slower/more expensive than keyword-based RAG
- **Impractical if you have tens of thousands of chunks**
- *Note:* the term “two-stage RAG” is not an established term in the literature

Vector RAG

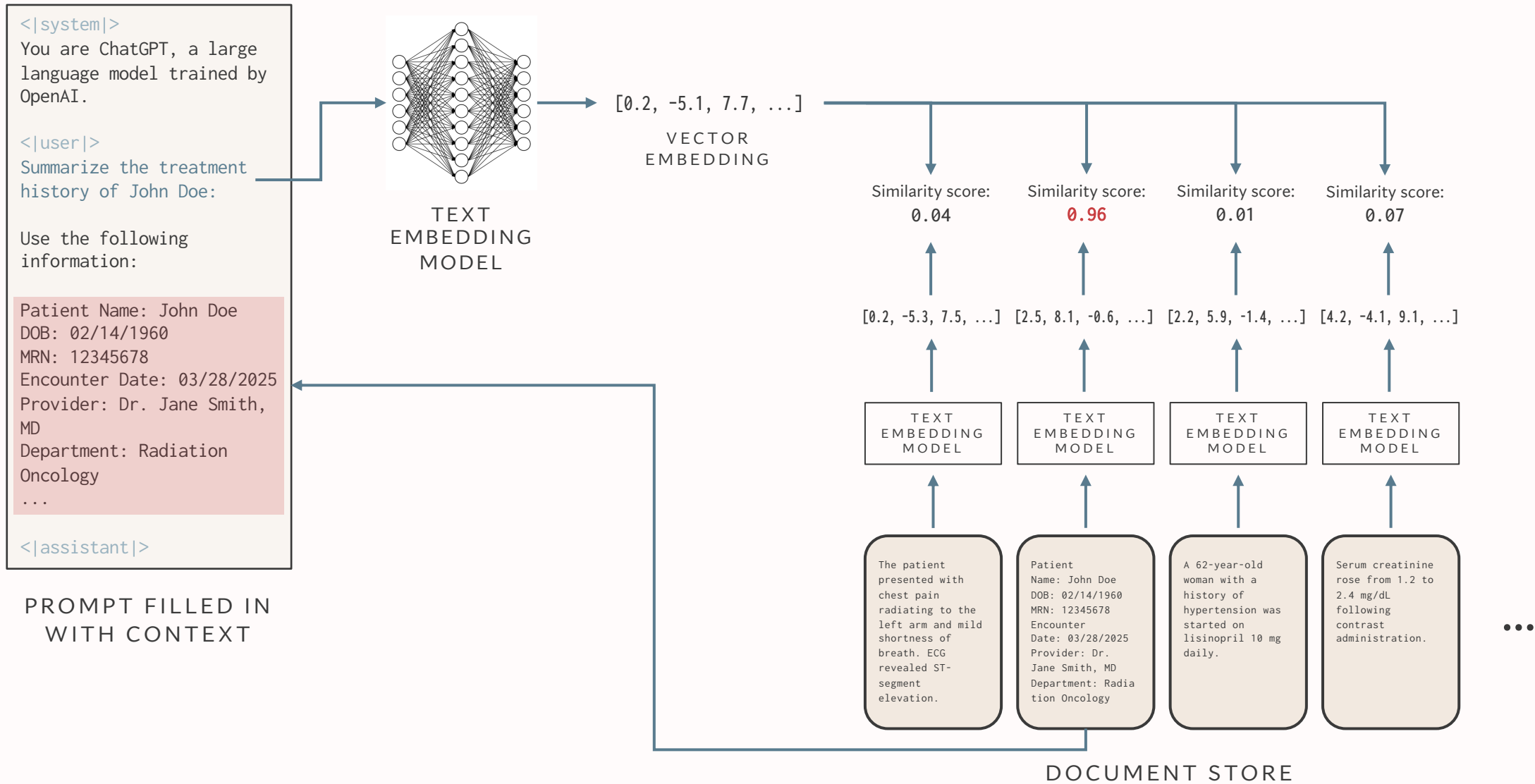
3. VECTOR-BASED RAG

- Use an *embedding model* to represent each chunk as a *vector*
- Use the same embedding model to represent the user query as a vector
- Compute *vector similarity measure* between the query vector and the chunk vectors
- Select the most similar chunks according to this measure
- Insert those chunks into the prompt

Vector RAG



Vector RAG



Vector RAG

```
<|system|>
You are ChatGPT, a large
language model trained by
OpenAI.

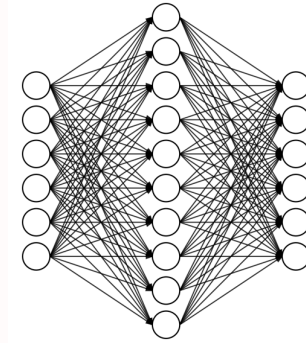
<|user|>
Summarize the treatment
history of John Doe:

Use the following
information:

Patient Name: John Doe
DOB: 02/14/1960
MRN: 12345678
Encounter Date: 03/28/2025
Provider: Dr. Jane Smith,
MD
Department: Radiation
Oncology
...

<|assistant|>
```

PROMPT FILLED IN
WITH CONTEXT



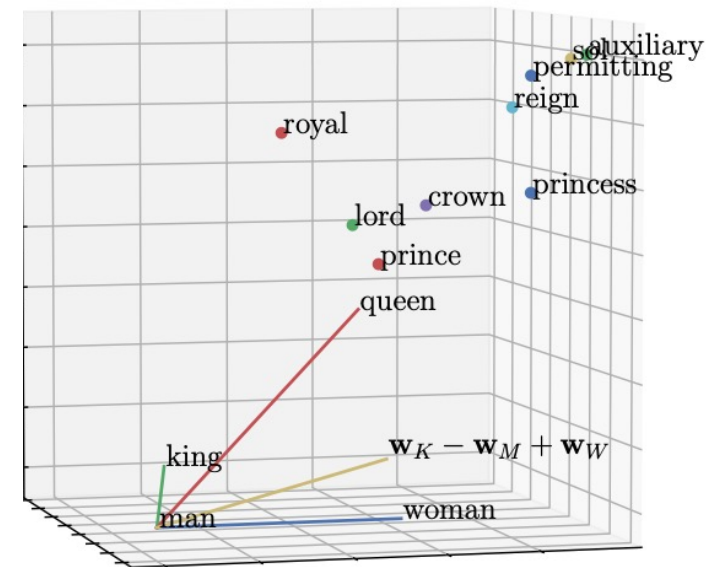
LARGE
LANGUAGE
MODEL

```
John Doe has a history of
prostate cancer managed
with external beam
radiation therapy, with
follow-up visits
demonstrating stable PSA
levels and no evidence of
recurrence.
```

LLM OUTPUT

Embeddings

- Embeddings are a vector representation of data that has some useful properties
- Typically, we desire that data that has *similar* vector representations are *semantically similar*.

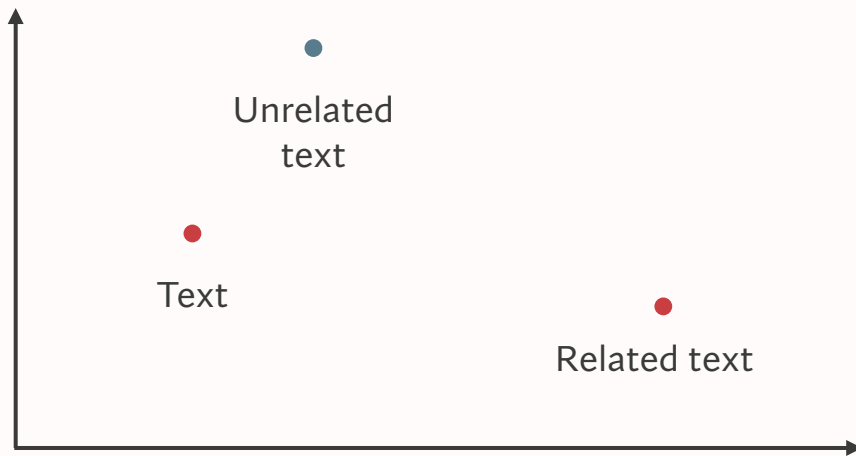


Space of word embeddings

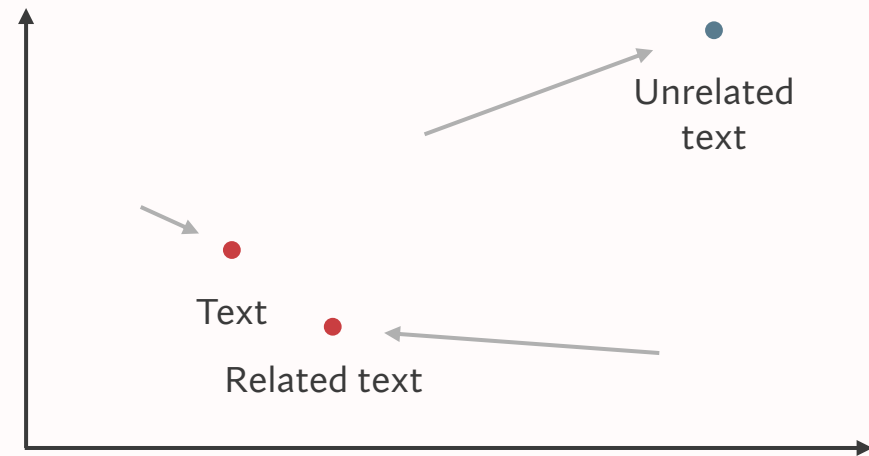
Allen, Carl, and Timothy Hospedales. "Analogies explained: Towards understanding word embeddings." In International Conference on Machine Learning, pp. 223-231. PMLR, 2019.

Embeddings

- How do we create such *semantically meaningful* vector embeddings?
- We train a separate text embedding model.
- **CONTRASTIVE LEARNING:** learn embeddings where:
 - Similar pairs are close (*in vector space*)
 - Dissimilar pairs are far apart (*in vector space*)



EMBEDDINGS BEFORE LEARNING



EMBEDDINGS AFTER LEARNING

Agentic AI

Giving LLMs tools

Agentic AI

- An AI agent is an LLM that has been given access to “tools” that it can invoke by sending messages.
- EXAMPLES:
 - Calculators
 - Internet search
 - Database search (*RAG can be thought of as a special case of agentic AI*)
 - Coding environment
 - Email client
 - Etc.
- Tools enable LLMs to:
 1. Overcome inherent weaknesses
 2. Act on the outside world

Tools to overcome weaknesses

- **WEAKNESS:** *LLMs were not trained on information regarding current affairs.*
- **TOOL:** *Web search, e.g., Google/Bing.*

EXAMPLE:

```
<|system|>
```

```
You are ChatGPT, a large language model trained by OpenAI. You have access to a web search tool. If you need to perform a web search to answer a user query, reply with `web_search("search_phrase")`.
```

Tools to overcome weaknesses

- **WEAKNESS:** *LLMs were not trained on information regarding current affairs.*
- **TOOL:** *Web search, e.g., Google/Bing.*

EXAMPLE:

```
<|system|>
```

```
You are ChatGPT, a large language model trained by OpenAI. You have access to a web search tool. If you need to perform a web search to answer a user query, reply with `web_search("search_phrase")`.
```

```
<|user|>
```

```
What is the stock price of NVIDIA today?
```

Tools to overcome weaknesses

- WEAKNESS: *LLMs were not trained on information regarding current affairs.*
- TOOL: *Web search, e.g., Google/Bing.*

EXAMPLE:

```
<|system|>
You are ChatGPT, a large language model trained by OpenAI. You have
access to a web search tool. If you need to perform a web search to
answer a user query, reply with `web_search("search_phrase")`.

<|user|>
What is the stock price of NVIDIA today?

<|assistant|>
`web_search("NVIDIA stock price today")`
```

Tools to overcome weaknesses

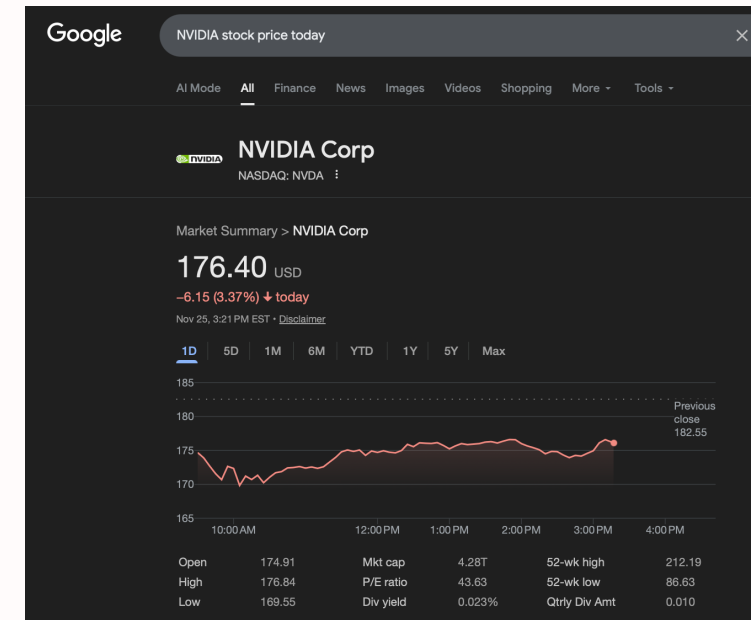
- **WEAKNESS:** *LLMs were not trained on information regarding current affairs.*
- **TOOL:** *Web search, e.g., Google/Bing.*

EXAMPLE:

```
<|system|>
You are ChatGPT, a large language model trained by OpenAI. You have
access to a web search tool. If you need to perform a web search to
answer a user query, reply with `web_search("search_phrase")`.

<|user|>
What is the stock price of NVIDIA today?

<|assistant|>
`web_search("NVIDIA stock price today")`
```



Google search

Tools to overcome weaknesses

- **WEAKNESS:** *LLMs were not trained on information regarding current affairs.*
- **TOOL:** *Web search, e.g., Google/Bing.*

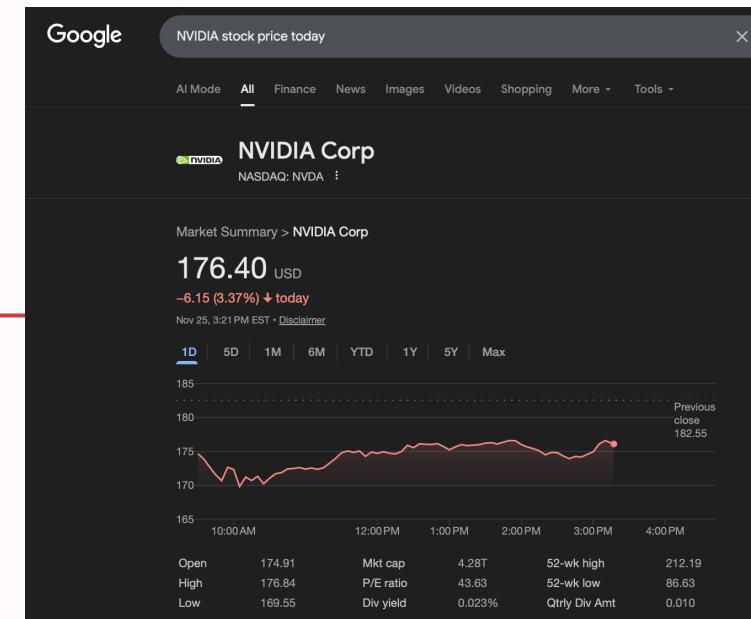
EXAMPLE:

```
<|system|>
You are ChatGPT, a large language model trained by OpenAI. You have
access to a web search tool. If you need to perform a web search to
answer a user query, reply with `web_search("search_phrase")`.

<|user|>
What is the stock price of NVIDIA today?

<|assistant|>
`web_search("NVIDIA stock price today")`

<|tool|>
NVIDIA Corp NASDAQ: NVDA
Market Summary > NVIDIA Corp
176.40 USD
-4.15 (-3.97%)
```



Google search

Tools to overcome weaknesses

- **WEAKNESS:** *LLMs were not trained on information regarding current affairs.*
- **TOOL:** *Web search, e.g., Google/Bing.*

EXAMPLE:

```
<|system|>
You are ChatGPT, a large language model trained by OpenAI. You have
access to a web search tool. If you need to perform a web search to
answer a user query, reply with `web_search("search_phrase")`.

<|user|>
What is the stock price of NVIDIA today?

<|assistant|>
`web_search("NVIDIA stock price today")`

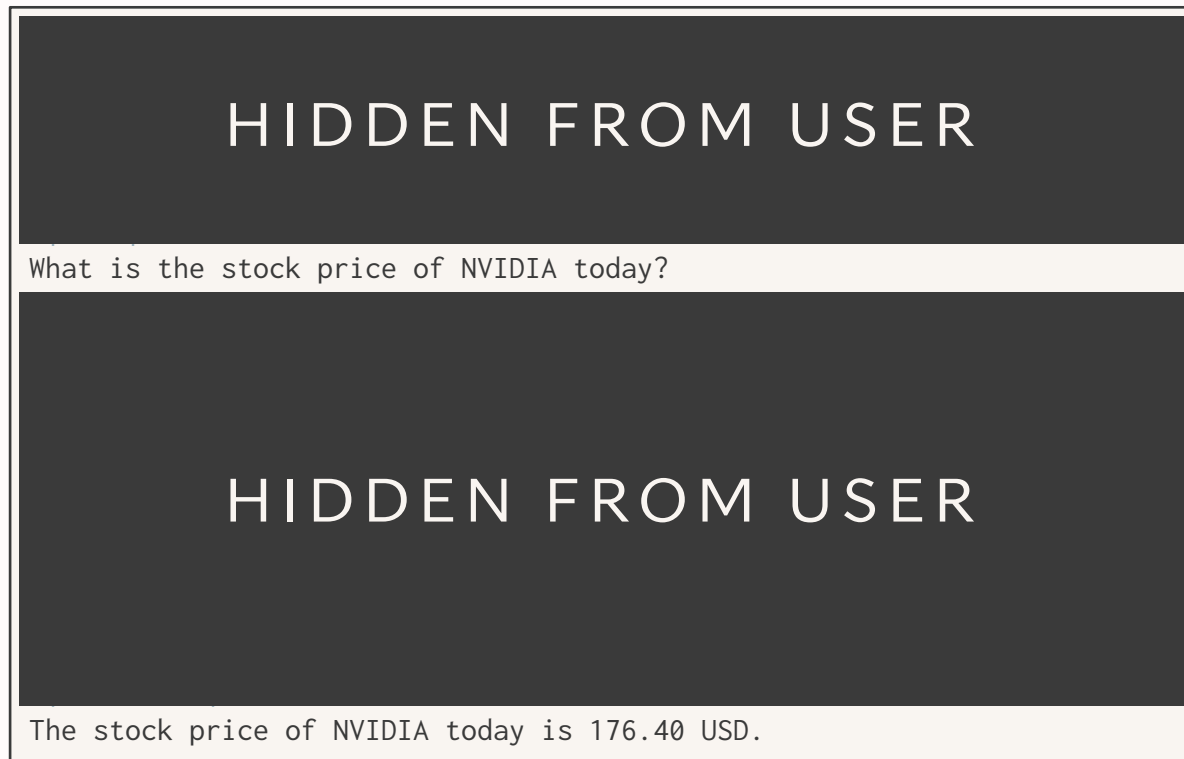
<|tool|>
NVIDIA Corp NASDAQ: NVDA
Market Summary > NVIDIA Corp
176.40 USD
-4.15 (-3.97%)

<|assistant|>
The stock price of NVIDIA today is 176.40 USD.
```

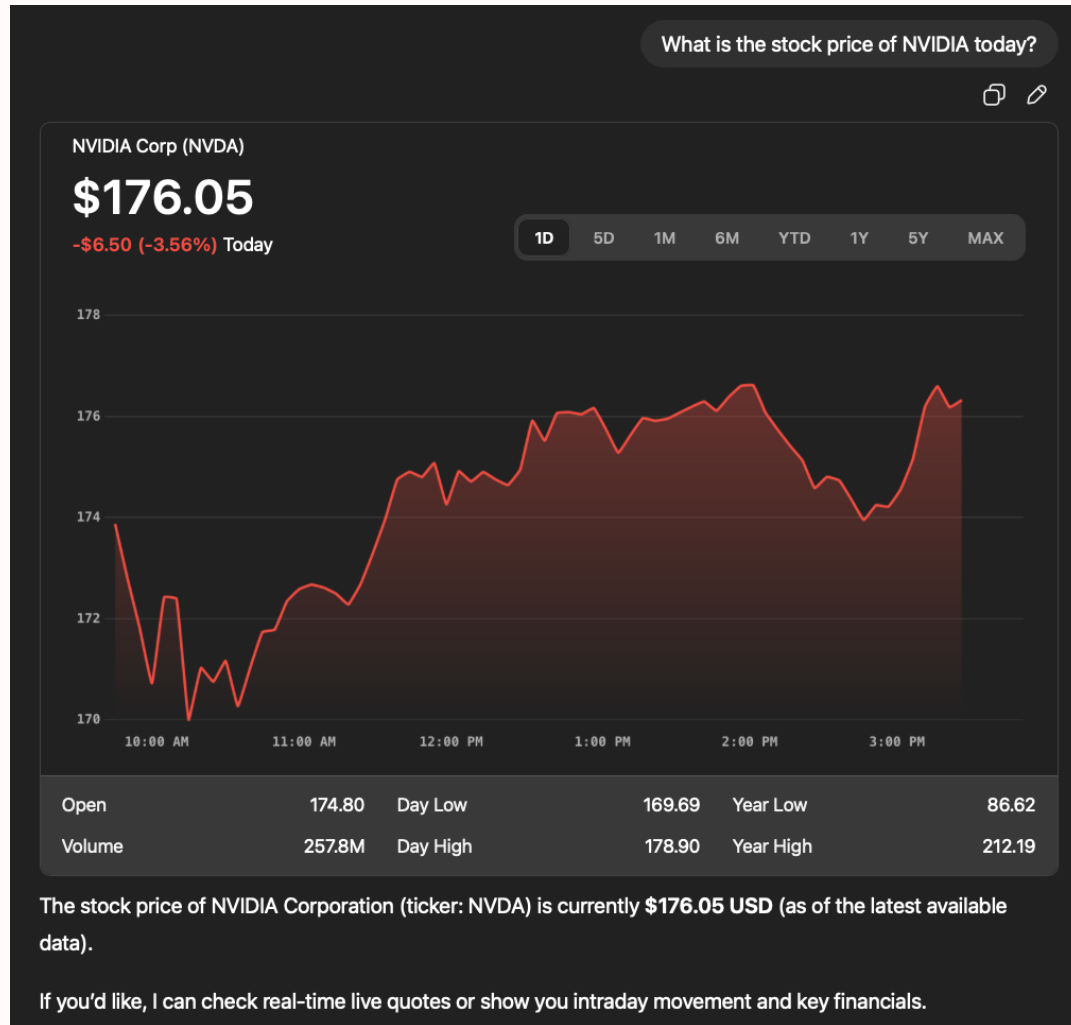
Tools to overcome weaknesses

- **WEAKNESS:** *LLMs were not trained on information regarding current affairs.*
- **TOOL:** *Web search, e.g., Google/Bing.*

EXAMPLE:



Tools to overcome weaknesses



ChatGPT, accessed
November 25th, 2025

Tools to overcome weaknesses

- LLMs learn *everything* by statistical association—including math.
- If you ask an LLM “What is $2 + 2$?”, it can only answer correctly if it saw the string “ $2 + 2 = 4$ ” sufficiently many times in its training data.
 - *If instead the string “ $2 + 2 = 5$ ” is more common, then the LLM will answer “ $2 + 2 = 5$ ”!*
- This is fine for small numbers, but what about questions like:
 - What is $25,678,905 / 12,907,786$?
 - What is the square root of $8,921,579,902$?
- These questions are almost certainly *not* in the training data.
- SOLUTION: *calculator program tool*.

Calculator tool use

EXAMPLE:

```
<|system|>
```

```
You are ChatGPT, a large language model trained by OpenAI. You have access to a calculator tool.
```

```
If you need to perform an addition, reply with `add(first_number, second_number)`.
```

```
If you need to perform a division, reply with `divide(numerator, denominator)`.
```

```
...
```

Calculator tool use

EXAMPLE:

```
<|system|>
```

```
You are ChatGPT, a large language model trained by OpenAI. You have access to a calculator tool.
```

```
If you need to perform an addition, reply with `add(first_number, second_number)`.
```

```
If you need to perform a division, reply with `divide(numerator, denominator)`.
```

```
...
```

```
<|user|>
```

```
What is 25,678,905 / 12,907,786?
```

Calculator tool use

EXAMPLE:

`<|system|>`

You are ChatGPT, a large language model trained by OpenAI. You have access to a calculator tool.

If you need to perform an addition, reply with
``add(first_number, second_number)``.

If you need to perform a division, reply with
``divide(numerator, denominator)``.

...

`<|user|>`

What is $25,678,905 / 12,907,786$?

`<|assistant|>`

``divide(25678905, 12907786)``

Calculator tool use

EXAMPLE:

`<|system|>`

You are ChatGPT, a large language model trained by OpenAI. You have access to a calculator tool.

If you need to perform an addition, reply with ``add(first_number, second_number)``.

If you need to perform a division, reply with ``divide(numerator, denominator)``.

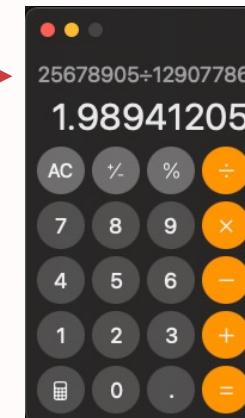
...

`<|user|>`

What is 25,678,905 / 12,907,786?

`<|assistant|>`

``divide(25678905, 12907786)``



Calculator app

Calculator tool use

EXAMPLE:

```
<|system|>
You are ChatGPT, a large language model trained by OpenAI. You have
access to a calculator tool.

If you need to perform an addition, reply with
`add(first_number, second_number)`.

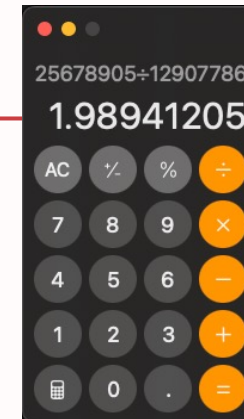
If you need to perform a division, reply with
`divide(enumerator, denominator)`.

...

<|user|>
What is 25,678,905 / 12,907,786?

<|assistant|>
`divide(25678905, 12907786)`

<|tool|>
1.98941205
```



Calculator app

Calculator tool use

EXAMPLE:

```
<|system|>
```

```
You are ChatGPT, a large language model trained by OpenAI. You have access to a calculator tool.
```

```
If you need to perform an addition, reply with `add(first_number, second_number)`.
```

```
If you need to perform a division, reply with `divide(numerator, denominator)`.
```

```
...
```

```
<|user|>
```

```
What is 25,678,905 / 12,907,786?
```

```
<|assistant|>
```

```
`divide(25678905, 12907786)`
```

```
<|tool|>
```

```
1.98941205
```

```
<|assistant|>
```

```
25,678,905 / 12,907,786 is 1.98941205.
```

Calculator tool use

EXAMPLE:

HIDDEN FROM USER

What is $25,678,905 / 12,907,786$?

HIDDEN FROM USER

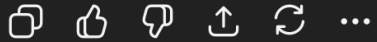
$25,678,905 / 12,907,786$ is 1.98941205.

Calculator tool use

What is 25,678,905 / 12,907,786?

$$\frac{25,678,905}{12,907,786} \approx 1.9894$$

So the ratio is about **1.99**.



ChatGPT, accessed November 25th, 2025

Multiple tool use

LLMs that can take actions

Tools to act on the outside world

AI scheduling and booking agent:

```
<|system|>
```

```
You are ChatGPT, a large language model trained by OpenAI. You can use external tools.  
To use a tool, reply with the exact syntax:
```

```
calendar.check_availability(date)  
appointments.book(type, datetime)
```

Tools to act on the outside world

AI scheduling and booking agent:

<|system|>

You are ChatGPT, a large language model trained by OpenAI. You can use external tools.
To use a tool, reply with the exact syntax:

```
calendar.check_availability(date)
appointments.book(type, datetime)
```

<|user|>

I need you to book a doctor's appointment for me for next Tuesday morning.

Tools to act on the outside world

AI scheduling and booking agent:

<|system|>

You are ChatGPT, a large language model trained by OpenAI. You can use external tools.
To use a tool, reply with the exact syntax:

```
calendar.check_availability(date)
appointments.book(type, datetime)
```

<|user|>

I need you to book a doctor's appointment for me for next Tuesday morning.

<|assistant|>

```
calendar.check_availability("2025-02-04 09:00-12:00")
```

Tools to act on the outside world

AI scheduling and booking agent:

<|system|>

You are ChatGPT, a large language model trained by OpenAI. You can use external tools.
To use a tool, reply with the exact syntax:

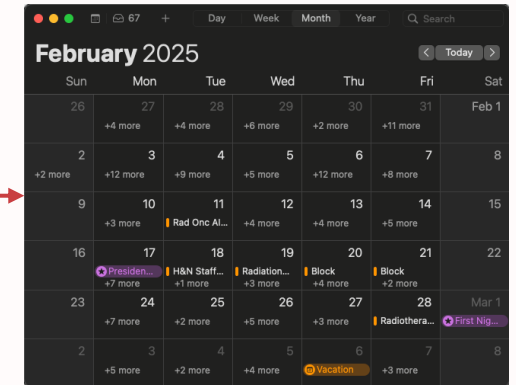
```
calendar.check_availability(date)
appointments.book(type, datetime)
```

<|user|>

I need you to book a doctor's appointment for me for next Tuesday morning.

<|assistant|>

```
calendar.check_availability("2025-02-04 09:00-12:00")
```



Calendar app

Tools to act on the outside world

AI scheduling and booking agent:

<|system|>

You are ChatGPT, a large language model trained by OpenAI. You can use external tools.
To use a tool, reply with the exact syntax:

```
calendar.check_availability(date)
appointments.book(type, datetime)
```

<|user|>

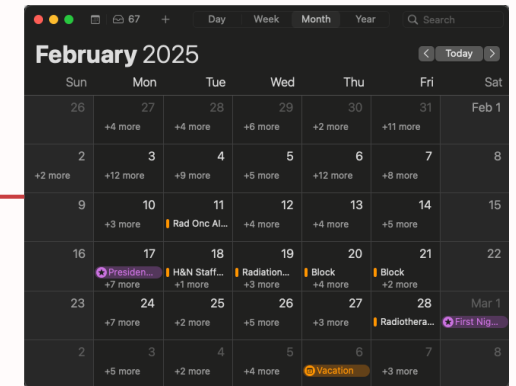
I need you to book a doctor's appointment for me for next Tuesday morning.

<|assistant|>

```
calendar.check_availability("2025-02-04 09:00-12:00")
```

<|tool|>

```
{"available_slots": ["2025-02-04T10:00", "2025-02-04T11:30"]}
```



Calendar app

Tools to act on the outside world

AI scheduling and booking agent:

```
<|system|>
You are ChatGPT, a large language model trained by OpenAI. You can use external tools.
To use a tool, reply with the exact syntax:

calendar.check_availability(date)
appointments.book(type, datetime)

<|user|>
I need you to book a doctor's appointment for me for next Tuesday morning.

<|assistant|>
calendar.check_availability("2025-02-04 09:00-12:00")

<|tool|>
{"available_slots": ["2025-02-04T10:00", "2025-02-04T11:30"]}

<|assistant|>
appointments.book("doctor", "2025-02-04T10:00")
```

Tools to act on the outside world

AI scheduling and booking agent:

```
<|system|>
You are ChatGPT, a large language model trained by OpenAI. You can use external tools.
To use a tool, reply with the exact syntax:

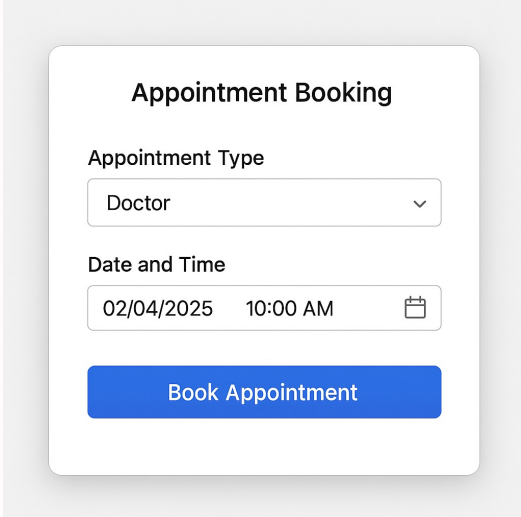
calendar.check_availability(date)
appointments.book(type, datetime)

<|user|>
I need you to book a doctor's appointment for me for next Tuesday morning.

<|assistant|>
calendar.check_availability("2025-02-04 09:00-12:00")

<|tool|>
{"available_slots": ["2025-02-04T10:00", "2025-02-04T11:30"]}

<|assistant|>
appointments.book("doctor", "2025-02-04T10:00")
```

A screenshot of an appointment booking application. It features a white card with rounded corners on a light gray background. The card is titled "Appointment Booking" in bold black text. Below the title, there are two sections: "Appointment Type" with a dropdown menu showing "Doctor" and a downward arrow, and "Date and Time" with a date field showing "02/04/2025", a time field showing "10:00 AM", and a calendar icon. At the bottom of the card is a prominent blue button with the text "Book Appointment" in white.

Appointment booking app

Tools to act on the outside world

AI scheduling and booking agent:

```
<|system|>
You are ChatGPT, a large language model trained by OpenAI. You can use external tools.
To use a tool, reply with the exact syntax:

calendar.check_availability(date)
appointments.book(type, datetime)

<|user|>
I need you to book a doctor's appointment for me for next Tuesday morning.

<|assistant|>
calendar.check_availability("2025-02-04 09:00-12:00")

<|tool|>
{"available_slots": ["2025-02-04T10:00", "2025-02-04T11:30"]}

<|assistant|>
appointments.book("doctor", "2025-02-04T10:00")

<|tool|>
{"status": "confirmed", "appointment_id": "APT-982314", "datetime": "2025-02-04T10:00"}
```

Appointment Booking

Appointment Type
Doctor

Date and Time
02/04/2025 10:00 AM

Book Appointment

Appointment booking app

Tools to act on the outside world

AI scheduling and booking agent:

<|system|>

You are ChatGPT, a large language model trained by OpenAI. You can use external tools.
To use a tool, reply with the exact syntax:

```
calendar.check_availability(date)
appointments.book(type, datetime)
```

<|user|>

I need you to book a doctor's appointment for me for next Tuesday morning.

<|assistant|>

```
calendar.check_availability("2025-02-04 09:00-12:00")
```

<|tool|>

```
{"available_slots": ["2025-02-04T10:00", "2025-02-04T11:30"]}
```

<|assistant|>

```
appointments.book("doctor", "2025-02-04T10:00")
```

<|tool|>

```
{"status": "confirmed", "appointment_id": "APT-982314", "datetime": "2025-02-04T10:00"}
```

<|assistant|>

Your doctor's appointment is confirmed for Tuesday at 10:00 AM.

Tools to act on the outside world

AI scheduling and booking agent:

HIDDEN FROM USER

I need you to book a doctor's appointment for me for next Tuesday morning.

HIDDEN FROM USER

Your doctor's appointment is confirmed for Tuesday at 10:00 AM.

Tools to act on the world

- *Many* more such use-cases
- Agentic AI is an extremely rapidly-expanding field
- Researchers and developers are still in the process of understanding the opportunities and risks involved

Question & Answer