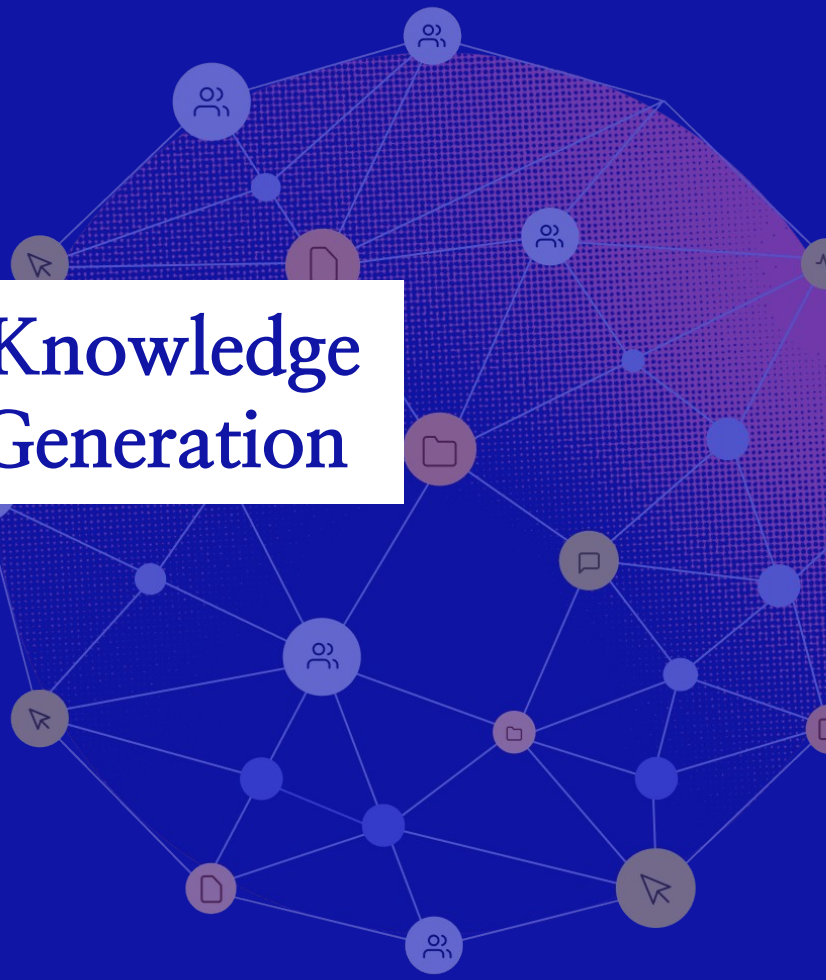
glean

Separating Coherence from Knowledge with Retrieval-Augmented Generation



WSAI

April 19, 2023

Agenda

1. Introduction
2. ChatGPT: Language vs. Implicit World Knowledge
3. RAG (Retrieval Augmented Generation)
4. Hallucinations: Entailment

Chau Tran

Tech Lead, LLMs and Vector Search

Glean

Language

Today's LLMs

- Trained to model **language**
- Emergent abilities at scale:
  - Coherence
  - Reasoning
  - Implicit knowledge

“Why is the sky…”

Knowledge

Knowledge

- Implicit world knowledge
  - Hallucinations
  - Factuality issues

Knowledge

- Missing knowledge
  - Recency (ChatGPT → 2021)
  - Non-public data

RAG

Retrieval Augmented Generation (RAG)

- Retrieval
  - Store knowledge outside of LLM model weights

- Use generative LLMs for their strengths:
  - Reasoning
  - Synthesis

Retrieval Augmented Generation (RAG)

- Results:
  - Knowledge-grounded
  - Factually generated output
  - LLMs as “agents” to connect to tools and knowledge

RAG / Agent

The New England Journal of Medicine is a registered trademark of [QA("Who is the publisher of The New England Journal of Medicine?") → Massachusetts Medical Society] the MMS.

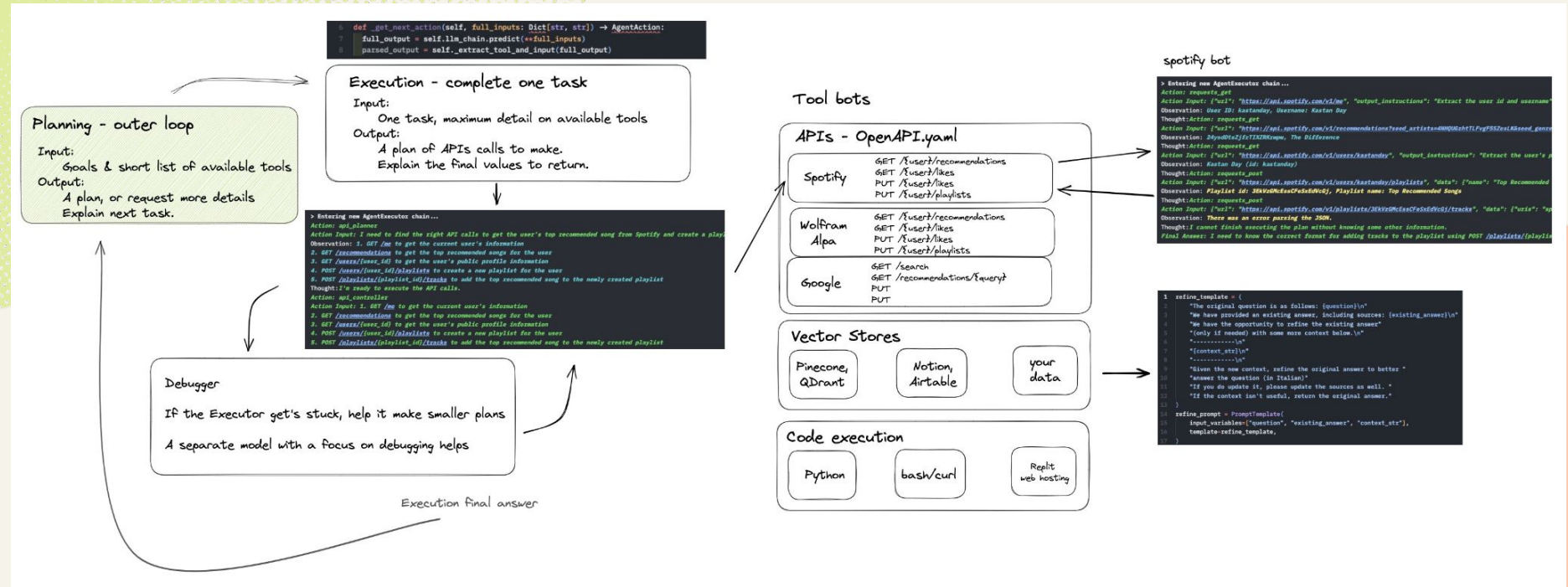
Out of 1400 participants, 400 (or [Calculator(400 / 1400) → 0.29] 29%) passed the test.

The name derives from "la tortuga", the Spanish word for [MT("tortuga") → turtle] turtle.

The Brown Act is California's law [WikiSearch("Brown Act") → The Ralph M. Brown Act is an act of the California State Legislature that guarantees the public's right to attend and participate in meetings of local legislative bodies.] that requires legislative bodies, like city councils, to hold their meetings open to the public.

Figure 1: Exemplary predictions of Toolformer. The model autonomously decides to call different APIs (from top to bottom: a question answering system, a calculator, a machine translation system, and a Wikipedia search engine) to obtain information that is useful for completing a piece of text.

RAG / Agent

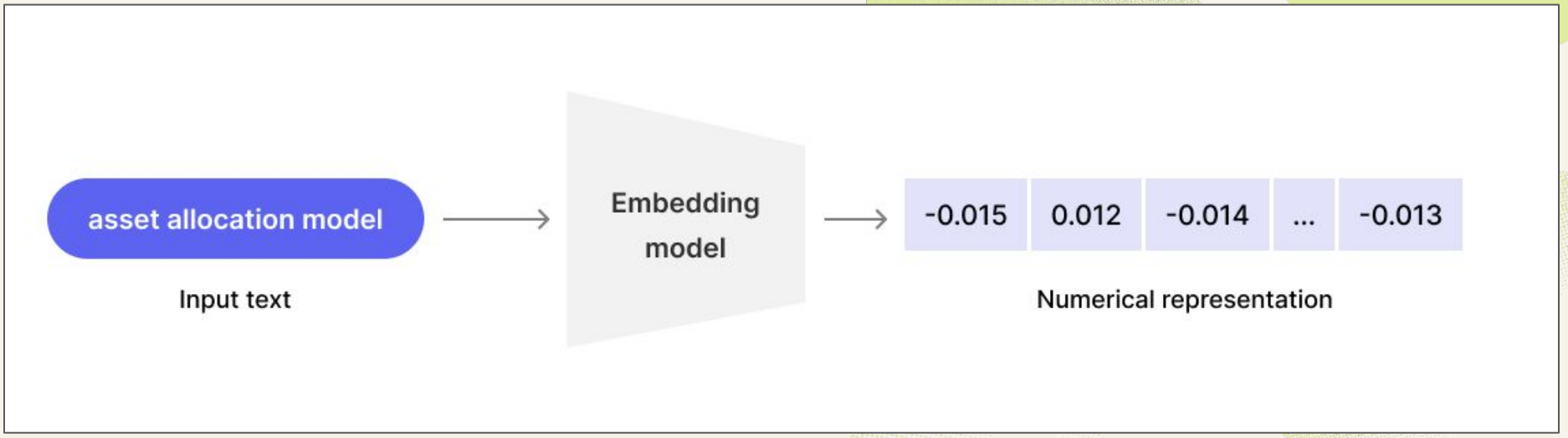

def _get_next_action(self, full_inputs: Dict[str, str]) → AgentAction:
full_output = self.llm_chain.predict(**full_inputs)
parsed_output = self._extract_tool_and_input(full_output)
Planning - outer loop
Input:
Goals & short list of available tools
Output:
A plan, or request more details
Explain next task.
Execution - complete one task
Input:
One task, maximum detail on available tools
Output:
A plan of APIs calls to make.
Explain the final values to return.
> Entering new AgentExecutor chain...
Action: api_planner
Action Input: I need to find the right API calls to get the user's top recomended song from Spotify and create a playl
Observation: 1. GET /me to get the current user's information
2. GET /recommendations to get the top recommended songs for the user
3. GET /users/{user_id} to get the user's public profile information
4. POST /users/{user_id}/playlists to create a new playlist for the user
5. POST /playlists/{playlist_id}/tracks to add the top recommended song to the newly created playlist
Thought:I'm ready to execute the API calls.
Action: api_controller
Action Input: 1. GET /me to get the current user's information
2. GET /recommendations to get the top recommended songs for the user
3. GET /users/{user_id} to get the user's public profile information
4. POST /users/{user_id}/playlists to create a new playlist for the user
5. POST /playlists/{playlist_id}/tracks to add the top recommended song to the newly created playlist
Debugger
If the Executor get's stuck, help it make smaller plans
A separate model with a focus on debugging helps
Execution final answer
Tool bots
APIs - OpenAPI.yaml
Spotify
GET /{user}/recommendations
GET /{user}/likes
PUT /{user}/likes
PUT /{user}/playlists
Wolfram Alpa
GET /{user}/recommendations
GET /{user}/likes
PUT /{user}/likes
PUT /{user}/playlists
Google
GET /search
GET /recommendations/{query}
PUT
PUT
Vector Stores
Pinecone, QDrant
Notion, Airtable
your data
Code execution
Python
bash/curl
Replit web hosting
spotify bot
> Entering new AgentExecutor chain...
Action: requests_get
Observation: User ID: kastanday, Username: Kastan Day
Thought:Action: requests_get
Observation: Kastan Day (id: kastanday)
Thought:Action: requests_post
Observation: Playlist id: 3EkVzGMcEssCFeSxEdVcGj, Playlist name: Top Recommended Songs
Thought:Action: requests_post
Observation: There was an error parsing the JSON.
Thought:I cannot finish executing the plan without knowing some other information.
refine_template = (
"The original question is as follows: {question}\n"
"We have provided an existing answer, including sources: {existing_answer}\n"
"We have the opportunity to refine the existing answer"
"(only if needed) with some more context below.\n"
"------------\n"
"{context_str}\n"
"------------\n"
"Given the new context, refine the original answer to better "
"answer the question (in Italian)"
"If you do update it, please update the sources as well. "
"If the context isn't useful, return the original answer."
)
refine_prompt = PromptTemplate(
input_variables=["question", "existing_answer", "context_str"],
template=refine_template,
)

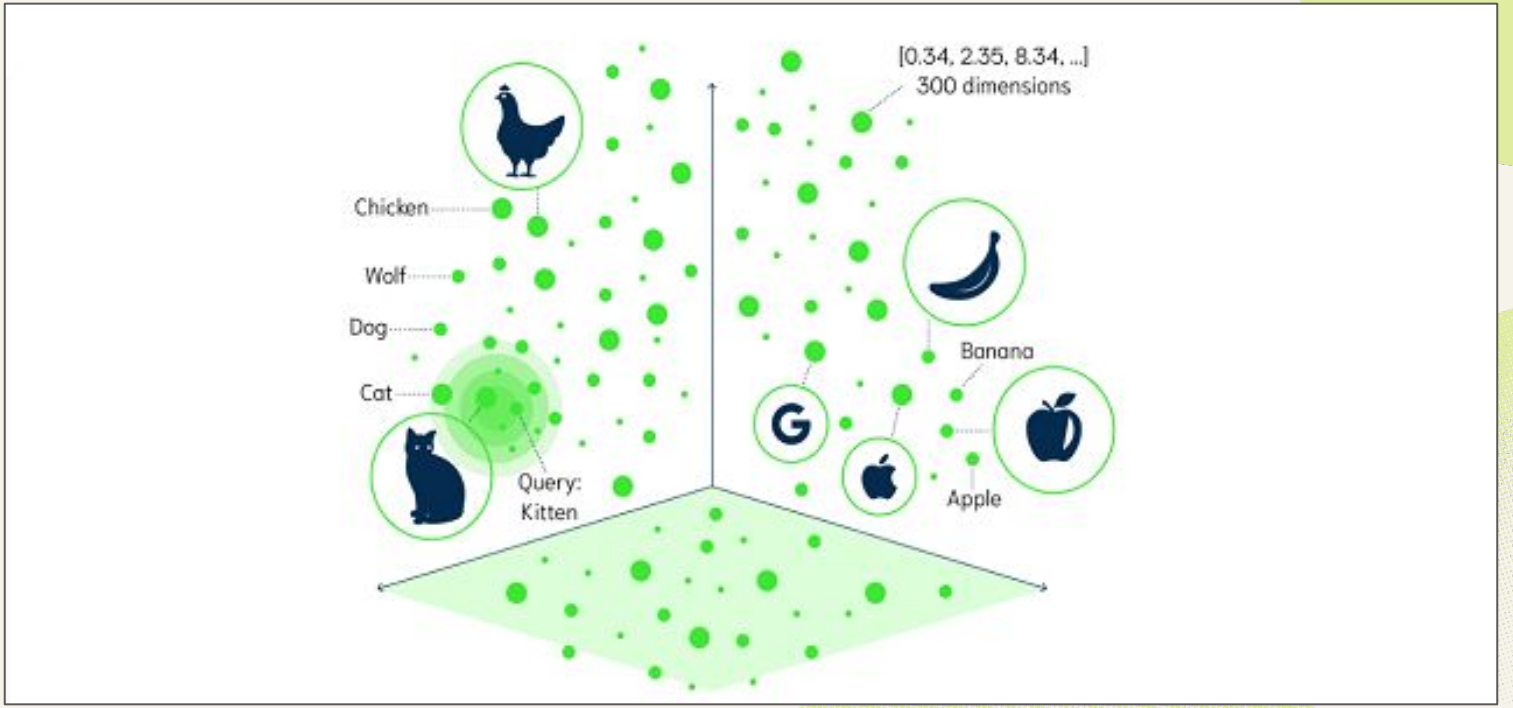
Domain-Adapted Retrieval

- Retrieving the right knowledge is a search problem

- Hybrid search (SOTA)
  - Keywords & terms
  - Dense (embeddings)

Fine-tuned for specific corpora & knowledge


asset allocation model
Input text
Embedding model
-0.015
0.012
-0.014
...
-0.013
Numerical representation


[0.34, 2.35, 8.34, ...]
300 dimensions
Chicken
Wolf
Dog
Cat
Query:
Kitten
Banana
Apple

Entailment as a Hallucination Defense

- For LLM input X (relevant knowledge), is generated output Y **entailed** by X?

- Note: requires X to be source of knowledge (RAG)

TEXT	HYPOTHESIS	ENTAILMENT
• *Eyeing the huge market potential, currently led by Google, Yahoo took over search company Overture Services Inc last year.*	• *Yahoo bought Overture.*	• **TRUE**

Q&A

If we don’t get to your question live, we’ll still make sure to get you an answer.

A member of our team will follow up with you after today’s event.

glean

Thank you!

Interested in more information?

☞ glean.com/get-a-demo