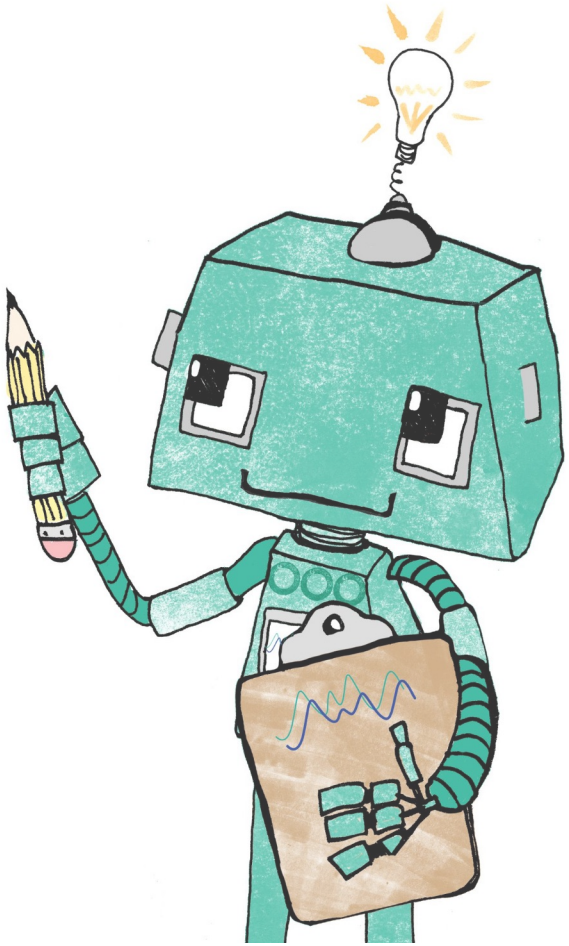
Leeds Data Science

July Meetup

1845-1850 Welcome

1850–2000 Ryan Mangan @ EfficientEther

Improving the Fidelity and Stability of Large Language Models



Housekeeping

Fire exit

- If the alarm sounds, please follow the fire exit signs
- Fire doors will be closed

Toilets

- The toilets are located on the mezzanine floor

Food / drink

- Help yourself to drinks and food throughout the session.

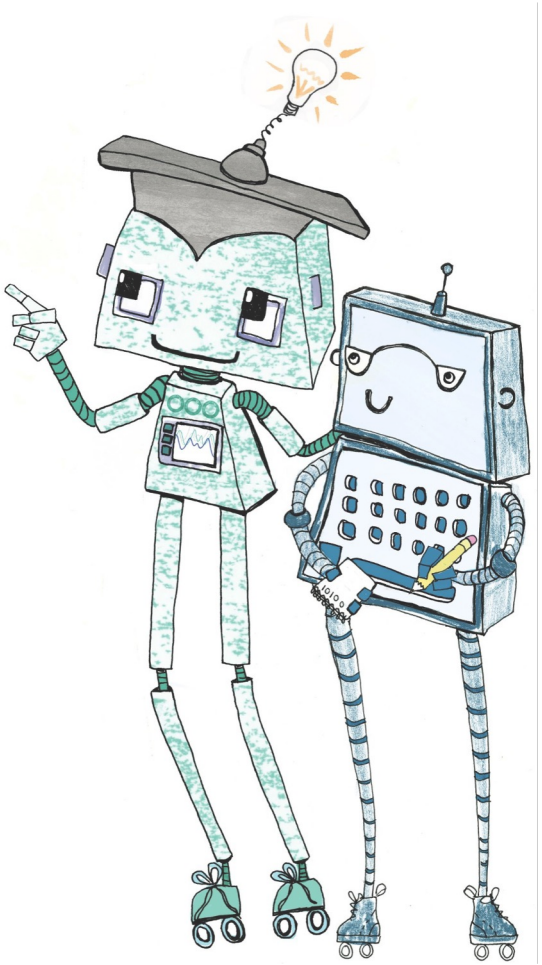
Thank you to our sponsors…

Jumping Rivers

Data Science Consultancy and Training

hello@jumpingrivers.com @jumping_uk

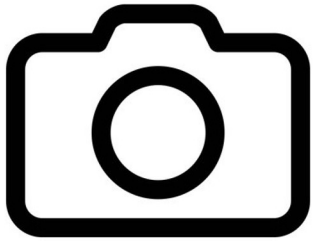
All things data science
Training
Machine learning, DevOps, infrastructure
Managed Posit services
Dashboard development and deployment



bruntwood SciTech

High quality office and hot desking space
Specialist business support which enables companies in the science and technology sector to form, collaborate, scale and grow.

July Announcements



Meta

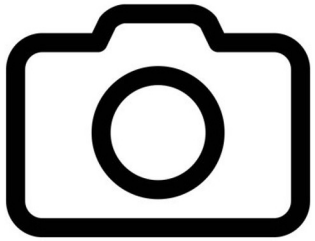
Meetups are *every two months*

Next event: **24th September**

If you would like to volunteer to give a talk: lds@jumpingrivers.com

Have an announcement you'd like included next time? Contact us via meetup.com

July Announcements



Related upcoming events

16th-27th Sept | Leeds Digital Festival | https://leedsdigitalfestival.org/

9th-10th Oct | Shiny In Production | https://shiny-in-production.jumpingrivers.com/

(25% discount code JRLEEDS)

Job opportunities near Leeds

Data visualisation developer | https://careers.libertyglobal.com/gb/en/job/REQ_00036305/

Data engineer | https://ybscareers.co.uk/job/data-engineer-in-leeds.640

Senior Data & Analytics Engineer | https://jet2careers.com/vacancy/?vId=4754

Have an announcement you'd like included next time? Contact us via meetup.com

…and now over to our speakers

Improving the Fidelity and Stability of Large Language Models

Ryan Mangan

Session Disclaimer

While the Presenter has made every effort to ensure that the information in this presentation is accurate, we do not assume and hereby disclaim any liability to any party for any loss, damage, or disruption caused by errors or omissions. This holds true whether such errors or omissions result from negligence, accident, or any other cause.

RYAN MANGAN

CEO @ EfficientEther Ltd

Ryan Mangan is a seasoned technologist with 18 years of experience across both IT professional and development environments. As the founder of EfficientEther Ltd, he focuses on leveraging AI for cloud optimisation, sustainability, Assistance and augmented actions. He is also an accomplished author and technical reviewer on a wide range of topics, including building cloud apps.

FBCS CITP
CHARTERED FELLOW
bcs The Chartered Institute for IT
Chartered IT Professional citp

MVP
Microsoft®
Most Valuable Professional

Agenda

- Intro in AI
- Intro Natural Language Processing
- Hallucinations
- Prompt Engineering
- Finetuning
- Retrieval-Augmented Generation
- Demo
- Questions

AI ?

Raise Your Hands If you are using ML ? 

Raise Your Hands If you are using NLP ? 

Raise Your Hands If you are using If you are using other AI?

High-Level AI Stages...

AI Stages	Artificial Narrow Intelligence (ANI)	Artificial General Intelligence (AGI)	Artificial Super Intelligence (ASI)
Description	Execute specific focused tasks, without ability to self-expand functionality	Perform broad tasks, reason, and improve capabilities comparable to humans	Demonstrate intelligence beyond human capabilities
Timing	Today	2050 vs 2075	TBC
Implications	Outperform humans in specific repetitive functions, such as driving, medical diagnosis, and financial advice	Compete with humans across all endeavors, such as earning university degrees and convincing humans that it is human	Outperform humans, helping to achieve societal objectives or threatening the human race

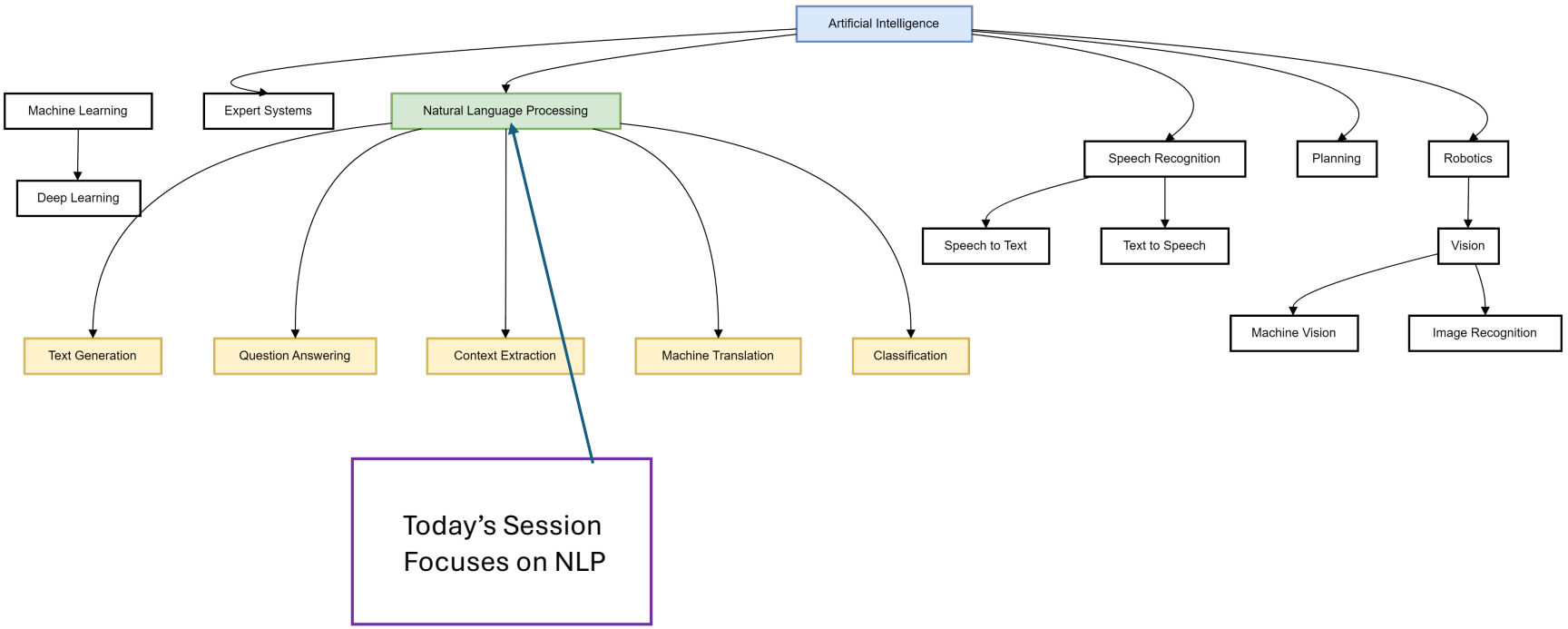
When will singularity happen? 1700 expert opinions of AGI [2024] (aimultiple.com)

OpenAI's 5 Levels of 'Super AI'...

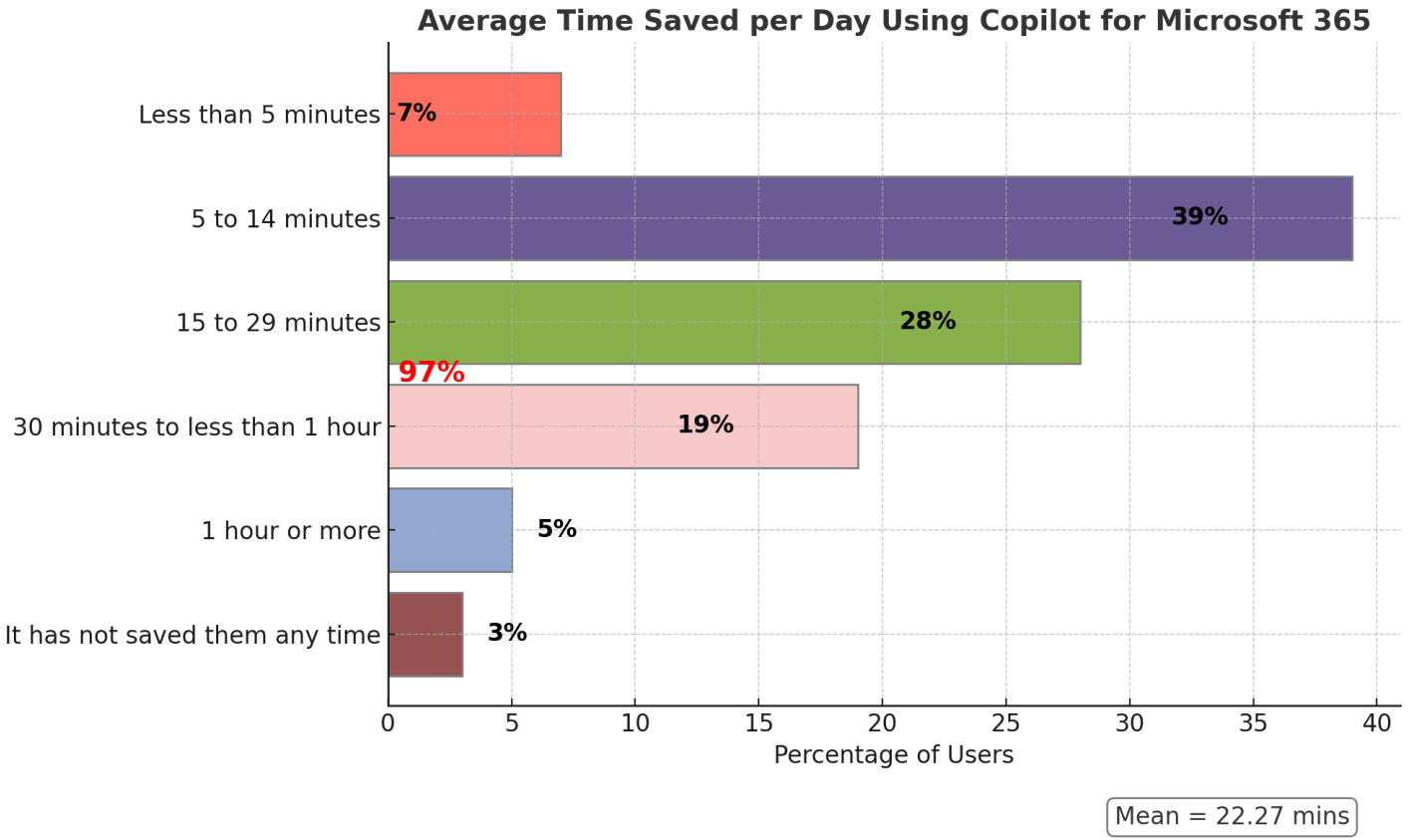
AI Levels	Description	Timing	Implications	Impact on Jobs
Level One: Conversational AI	Simple AI that can handle basic conversations and interactions	Today	Enhances customer service and user interactions	Jobs enhanced
Level Two: Reasoning AI	AI that can perform reasoning tasks, make decisions based on logic and data	Near Future	Supports complex decision-making processes	Some jobs at risk due to automation
Level Three: Autonomous AI	AI that can operate independently in complex environments	Mid-term Future	Autonomous systems in industries like transportation, logistics, and manufacturing	Significant job displacement potential
Level Four: Innovating AI	AI that can create new ideas, innovations, and improve existing processes	Long-term Future	Drives innovation and improvements across various fields	Jobs at risk; new job roles created
Level Five: Organizational AI	AI that can manage and optimize entire organizations	Distant Future	Transforms organizational structures and efficiency	Major reorganization of job roles and functions

OpenAI's 5 Levels Of 'Super AI' (AGI To Outperform Human Capability) (forbes.com)

Different Types of AI.....



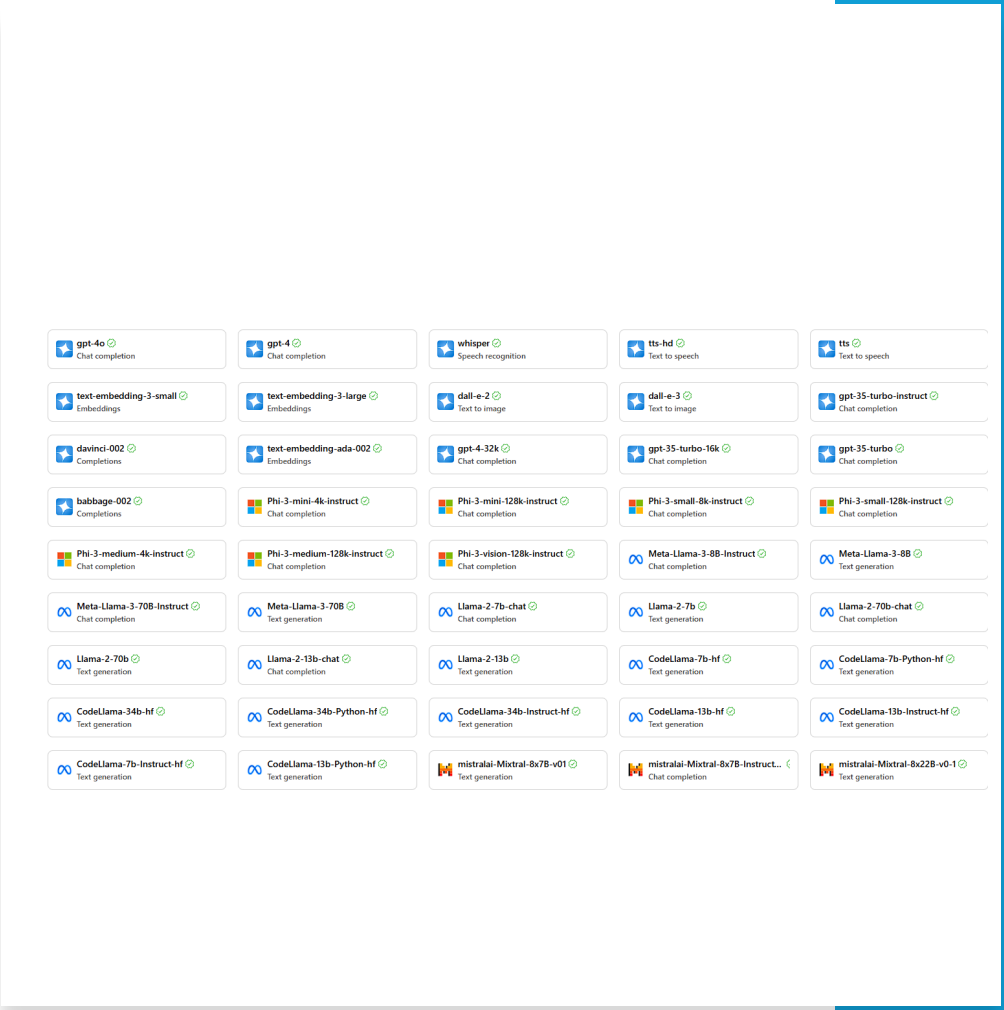
gpt-4o Chat completion	gpt-4 Chat completion	whisper Speech recognition	tts-hd Text to speech	tts Text to speech
text-embedding-3-small Embeddings	text-embedding-3-large Embeddings	dall-e-2 Text to image	dall-e-3 Text to image	gpt-35-turbo-instruct Chat completion
davinci-002 Completions	text-embedding-ada-002 Embeddings	gpt-4-32k Chat completion	gpt-35-turbo-16k Chat completion	gpt-35-turbo Chat completion
babbage-002 Completions	Phi-3-mini-4k-instruct Chat completion	Phi-3-mini-128k-instruct Chat completion	Phi-3-small-8k-instruct Chat completion	Phi-3-small-128k-instruct Chat completion
Phi-3-medium-4k-instruct Chat completion	Phi-3-medium-128k-instruct Chat completion	Phi-3-vision-128k-instruct Chat completion	Meta-Llama-3-8B-Instruct Chat completion	Meta-Llama-3-8B Text generation
Meta-Llama-3-70B-Instruct Chat completion	Meta-Llama-3-70B Text generation	Llama-2-7b-chat Chat completion	Llama-2-7b Text generation	Llama-2-70b-chat Chat completion
Llama-2-70b Text generation	Llama-2-13b-chat Chat completion	Llama-2-13b Text generation	CodeLlama-7b-hf Text generation	CodeLlama-7b-Python-hf Text generation
CodeLlama-34b-hf Text generation	CodeLlama-34b-Python-hf Text generation	CodeLlama-34b-Instruct-hf Text generation	CodeLlama-13b-hf Text generation	CodeLlama-13b-Instruct-hf Text generation
CodeLlama-7b-Instruct-hf Text generation	CodeLlama-13b-Python-hf Text generation	mistralai-Mixtral-8x7B-v01 Text generation	mistralai-Mixtral-8x7B-Instruct... Chat completion	mistralai-Mixtral-8x22B-v0-1 Text generation



n = 108; IT Leaders primarily responsible for Copilot (S02=1), excluding Don't Know

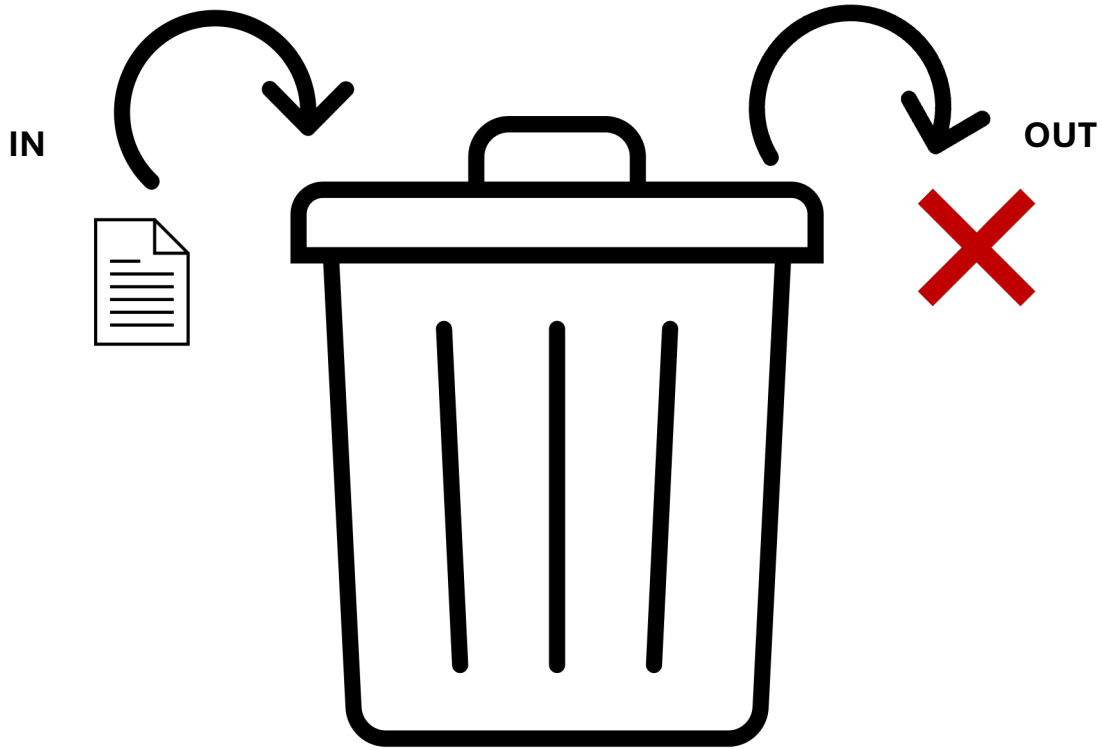
Summary thoughts before moving on

Intro Natural Language Processing



Before We Start.....

Garbage in..... Garbage out.....



Misunderstandings

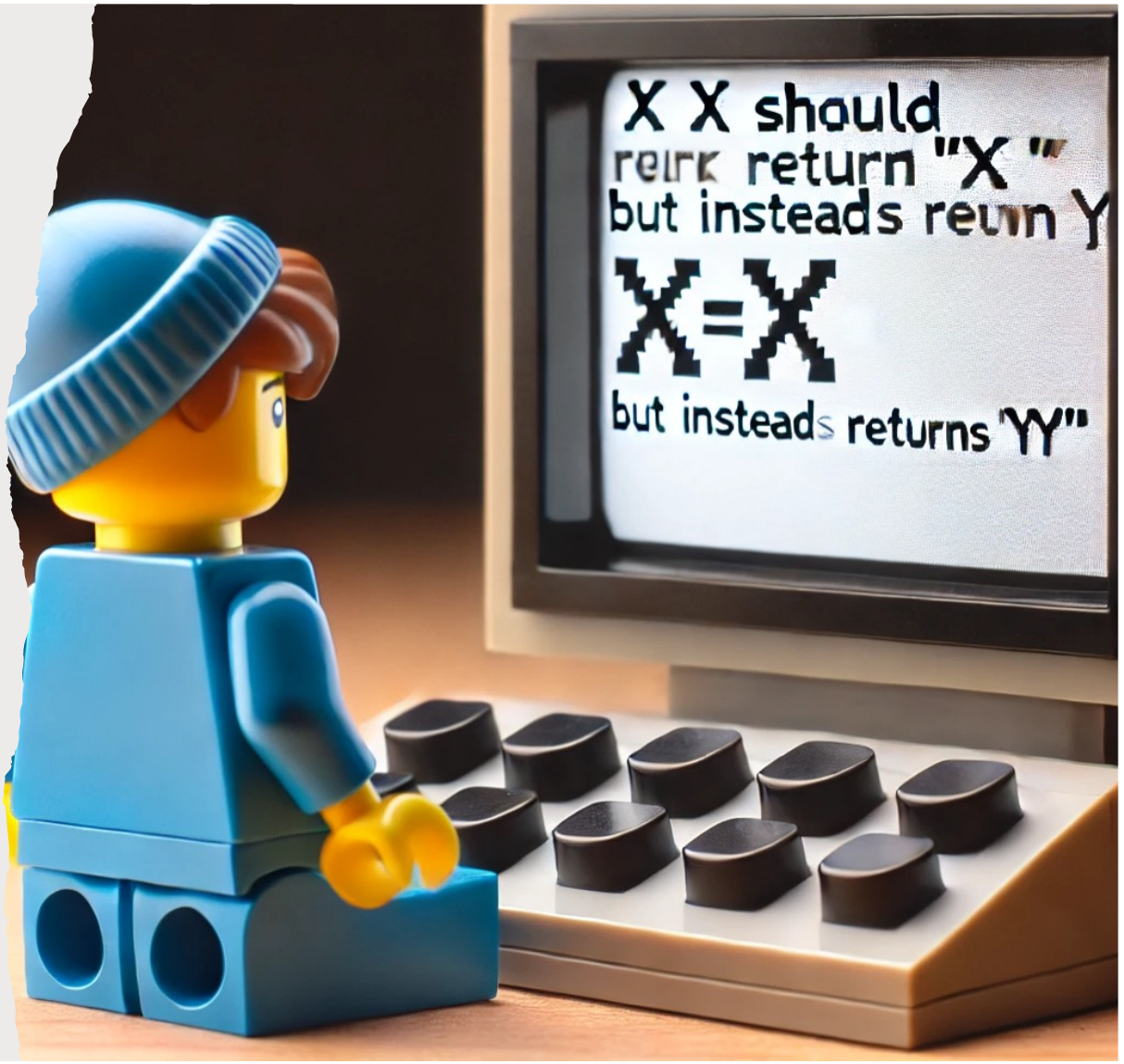
Example 1: "An English-speaking person offers a 'gift,' meaning a present, to a German speaker. However, the German speaker thinks it means 'poison,' leading to a misunderstanding'"

Misunderstandings

"**Example 2:** "An American speaker uses the word 'football,' referring to American football, but a British speaker thinks they are talking about soccer, leading to confusion."

Misunderstandings

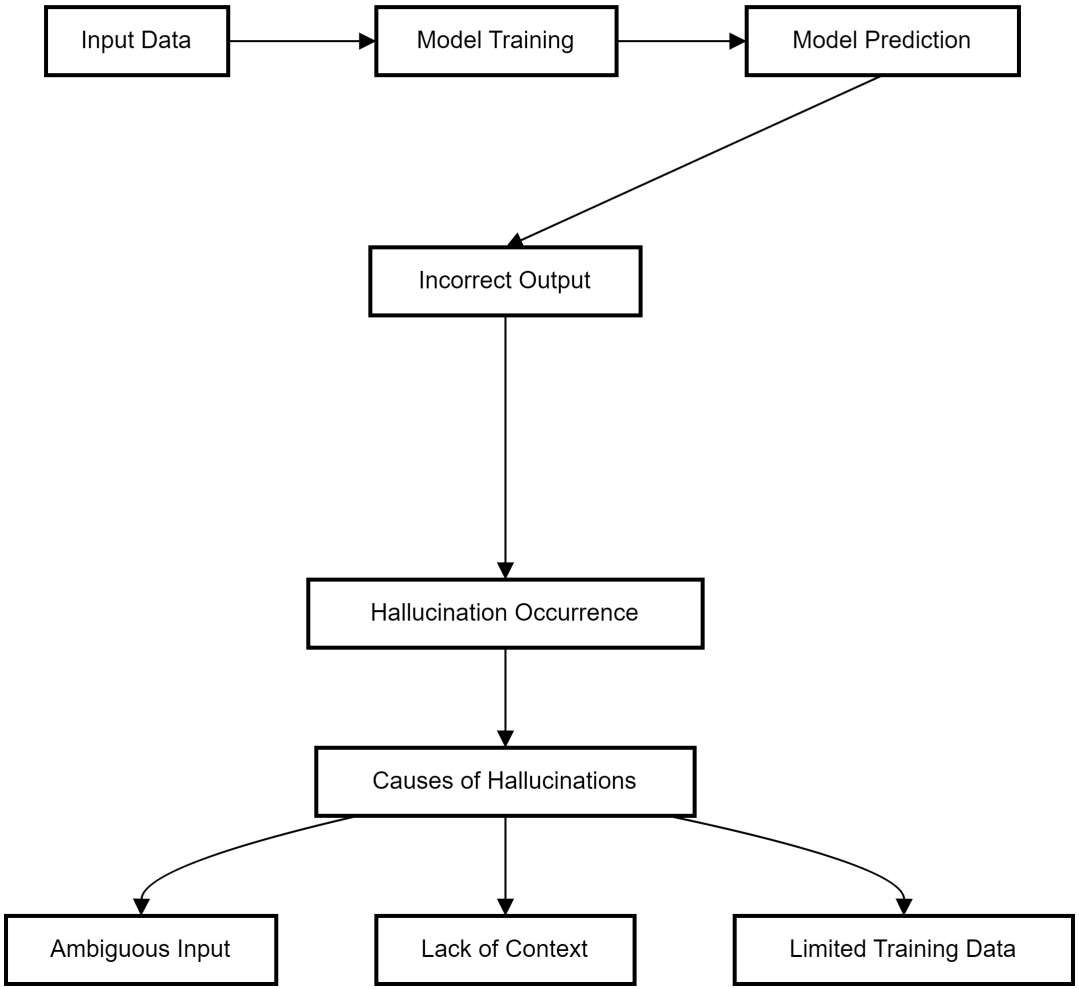
Just as language misunderstandings can occur between people from different cultures, the same issues can arise when using Natural Language Processing (NLP). If the input is not clear or uses ambiguous wording, the NLP system can produce incorrect or unexpected responses.

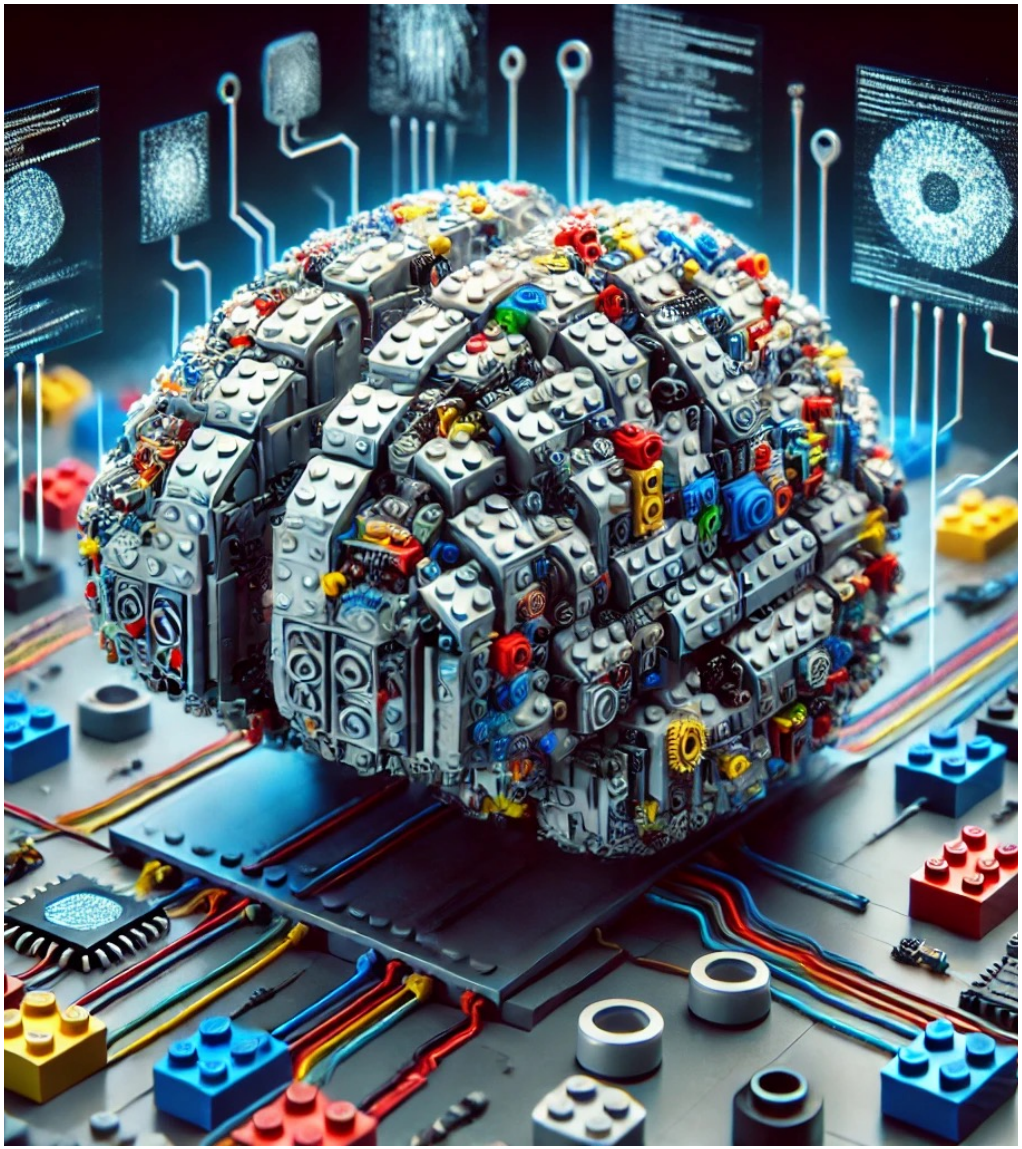


Hallucinations and why

- **Core limitation of Generative AI**
- Predictive Nature (sequence-based)
- Training Data Limitations
- Lack of Real-World Understanding
- Ambiguity and Open-Ended Queries
- Inference Under Constraints
- Presenting out-of-date or generic information when the user expects a specific, current response.
- Creating a response from non-authoritative sources.

How AI-Generated Hallucinations Occur





Model Autophagy Disorder (MAD)

MAD occurs when AI models consume their own outputs as training data, leading to a deterioration in the quality and diversity of the generated content.

As AI models consume their own outputs and amplify inaccuracies, the entropy of the generated content increases, leading to a loss of meaningful and coherent information.

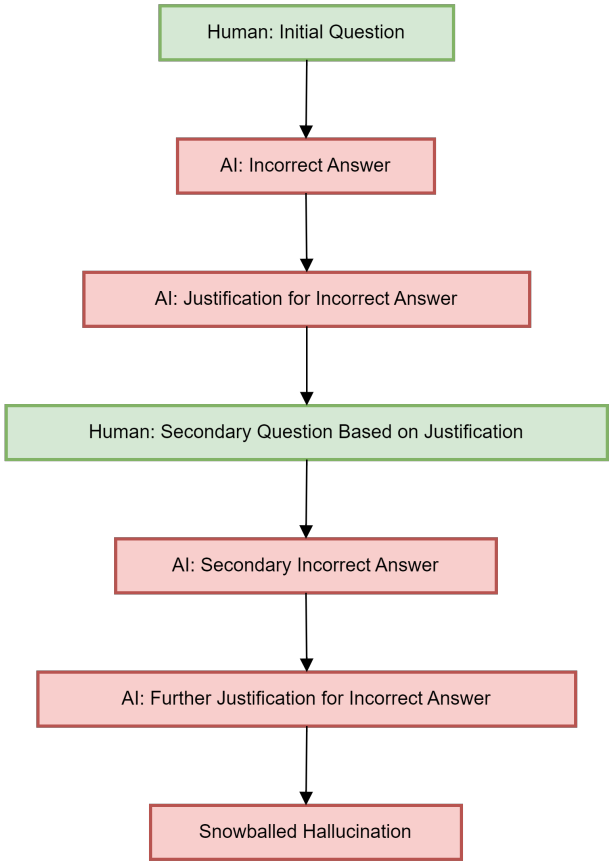
LLM's Need fresh new data. If you do not provide new, clean data, then the model will deteriorate.

[2307.01850] Self-Consuming Generative Models Go MAD (arxiv.org)

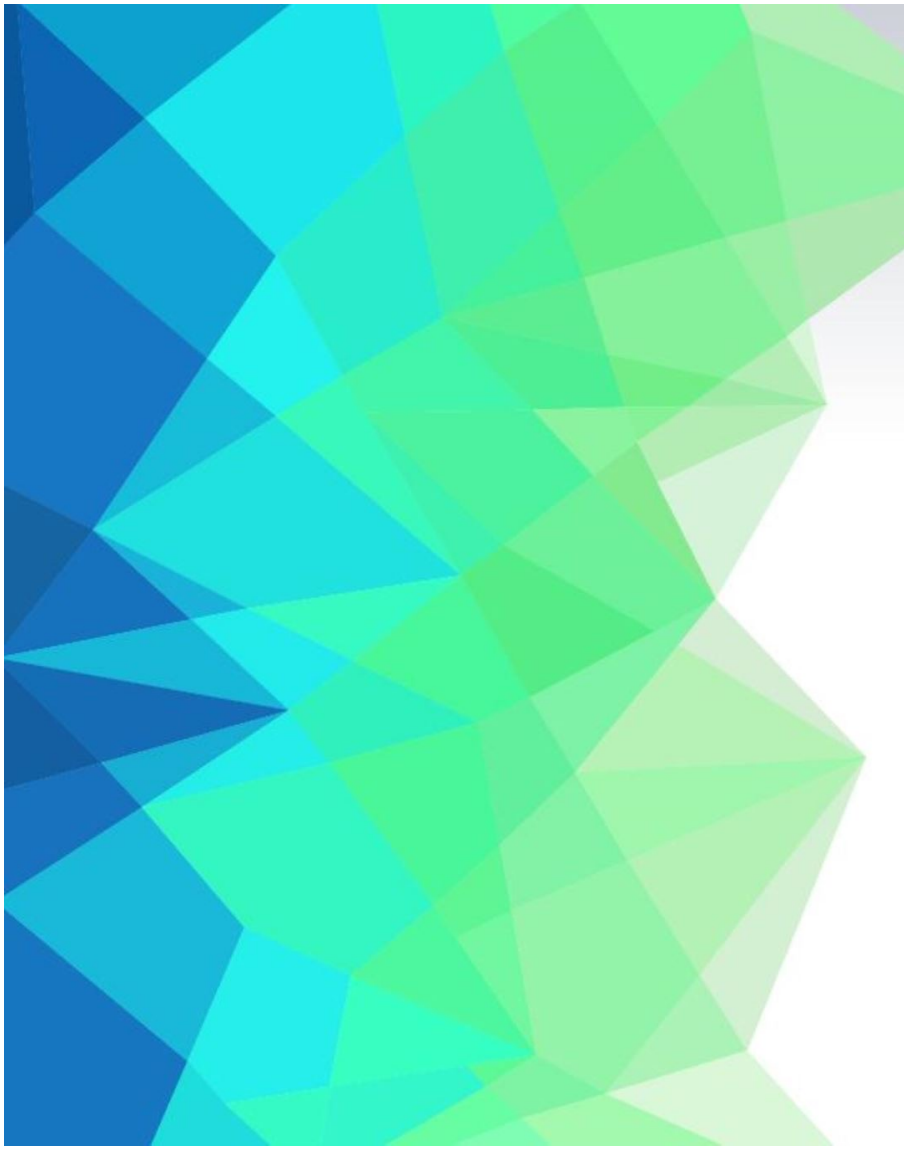
Model Hallucinations Can Snowball

[2305.13534] How Language Model Hallucinations Can Snowball (arxiv.org)

Hallucinations can snowball



Prompt Engineering

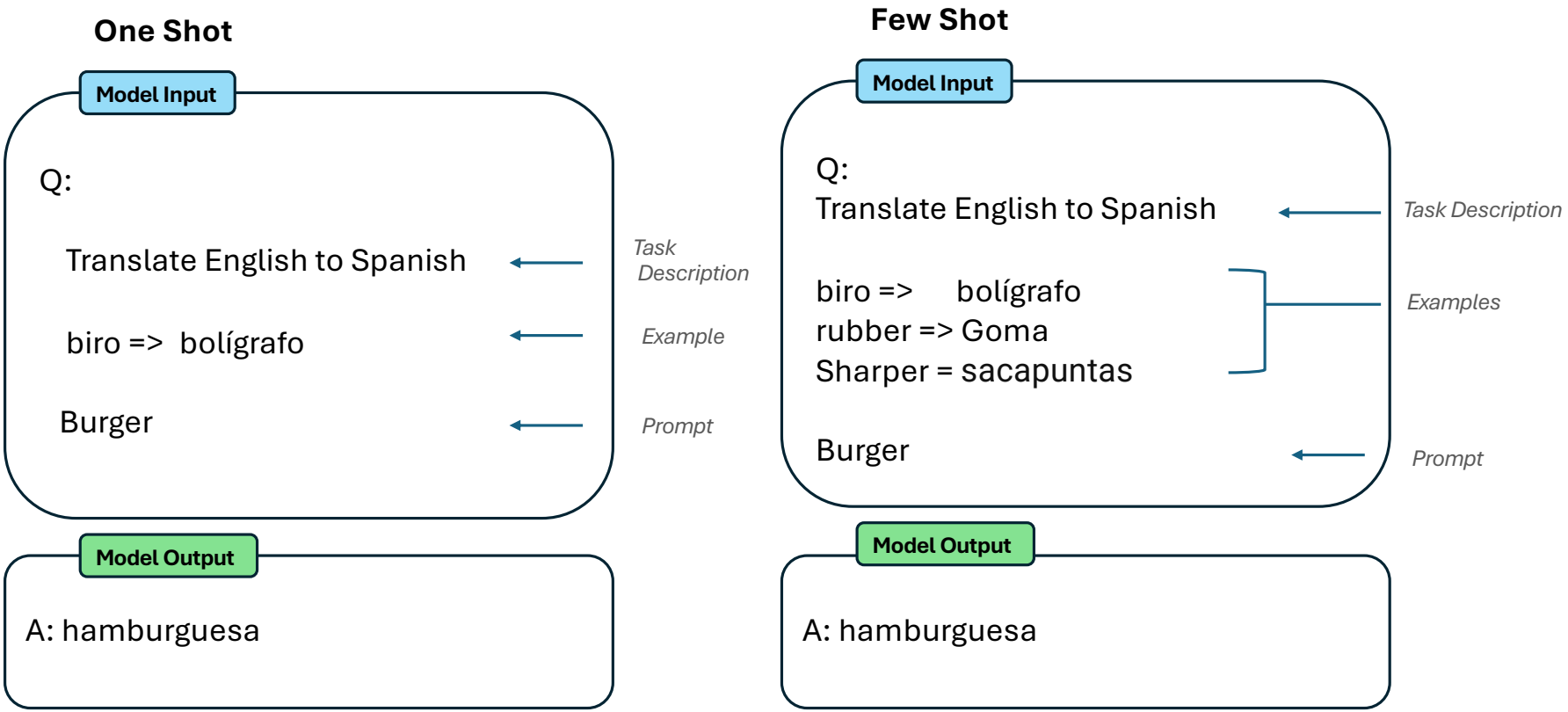


CO-STAR Prompt Framework

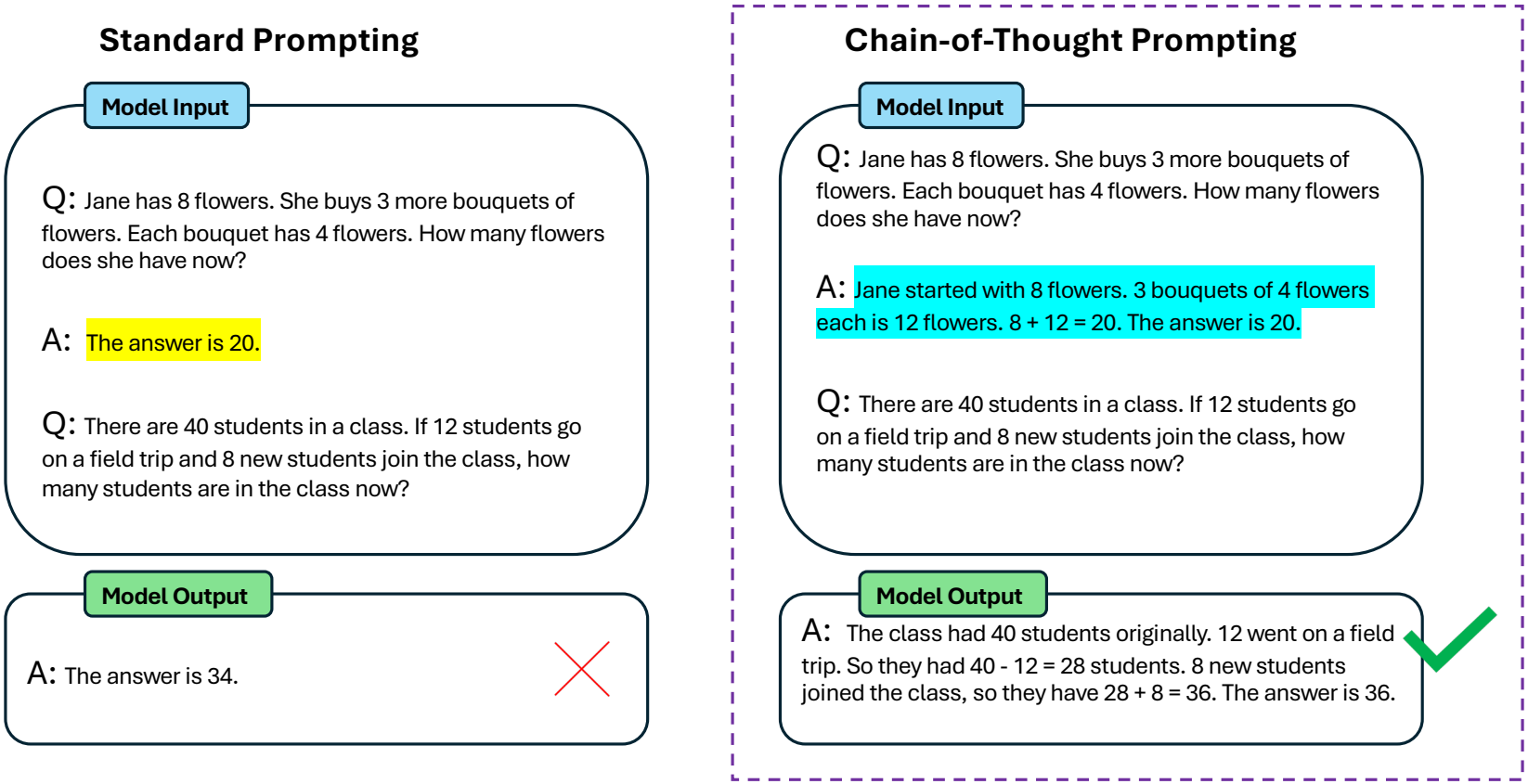
- **C: Context:** Provide background and information on the task
- **O: Objective**: Define the task that you want the LLM to perform
- **S: Style:** Specify the writing style you want the LLM to use
- **T: Tone:** Set the attitude and tone of the response
- **A: Audience:** Identify who the response is for
- **R: Response:** Provide the response format and style

2312.16171 (arxiv.org)

Few Shot Prompting

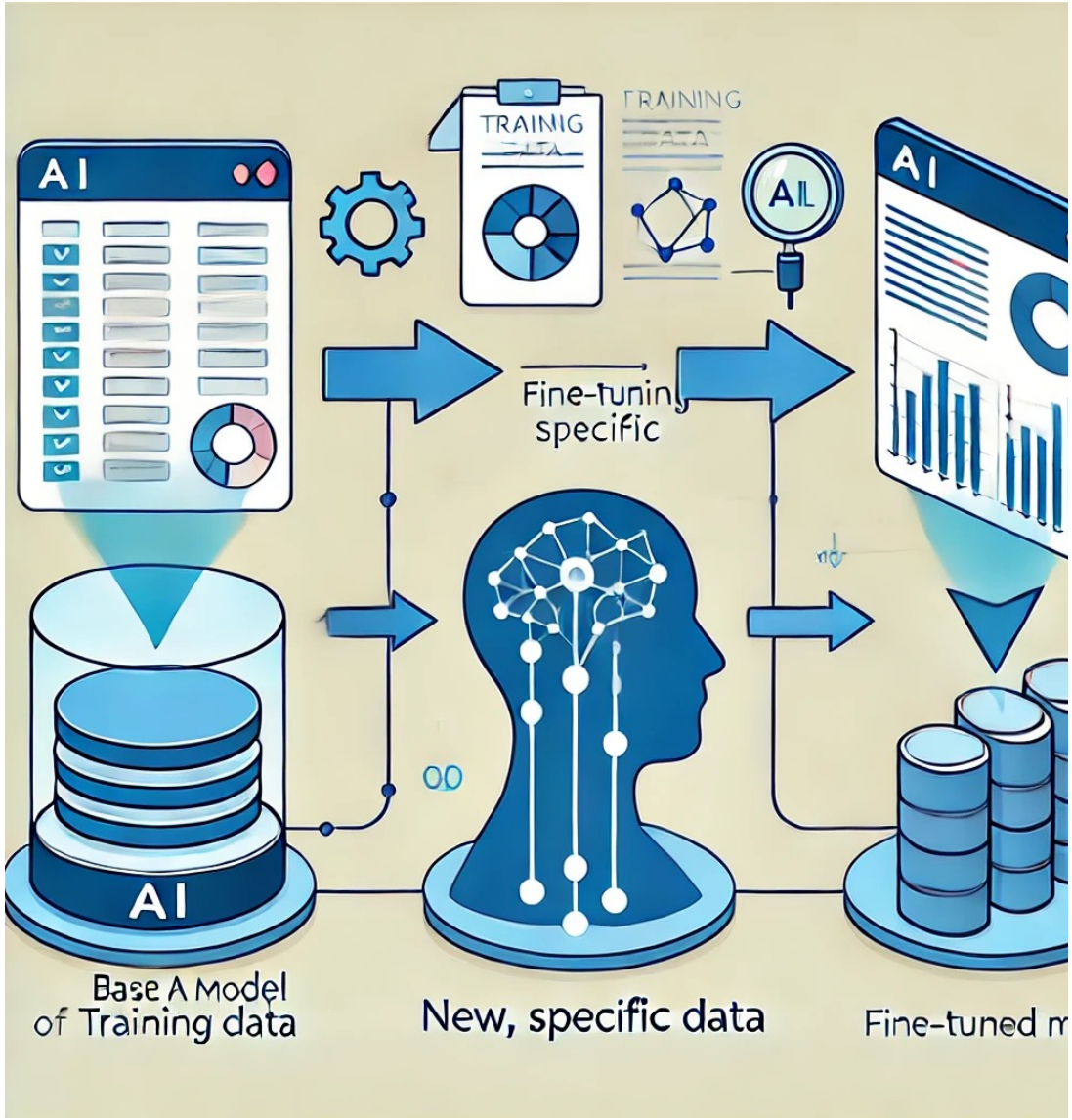


Chain-of-Thought Prompting



Training and Fine Tuning

- Training is expensive. Some organisations report they have $$$$
- Finetuning is more affordable way to improve the LLM.
- Specificity & Relevance
- Task-Specific Fine-Tuning
- Bias Amplification

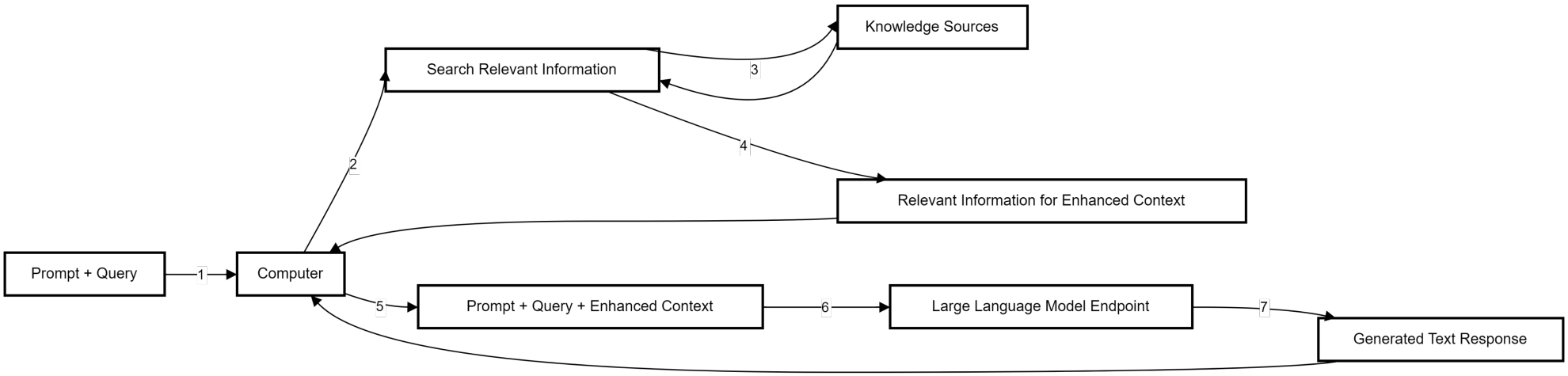


Strategies to Reduce Hallucinations and improve Responses

What are Vectors & Embeddings

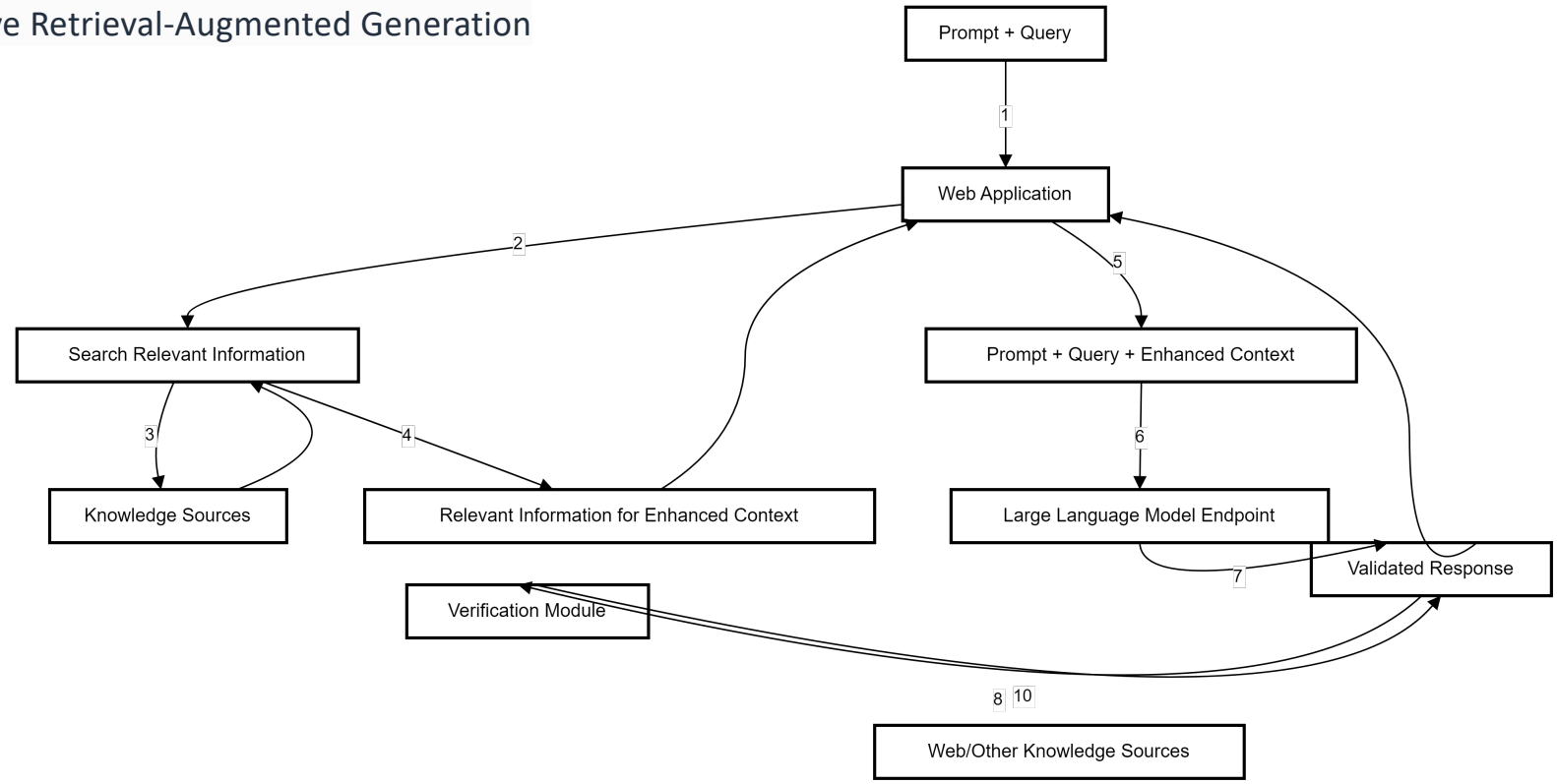
- **Vectors** in the context of machine learning and natural language processing refer to numerical representations of data. These representations, known as embeddings, map high-dimensional data (like words, sentences, or images) into a lower-dimensional space. Each point in this space corresponds to a vector, which is a list of numbers that captures the semantic meaning or features of the data.
- **Embeddings** are dense vector representations of data. For instance, in NLP, word embeddings represent words in a continuous vector space where semantically similar words are closer together. Popular embedding techniques include Word2Vec, GloVe, and BERT.

Retrieval-Augmented Generation



How RAG Works – in Concept

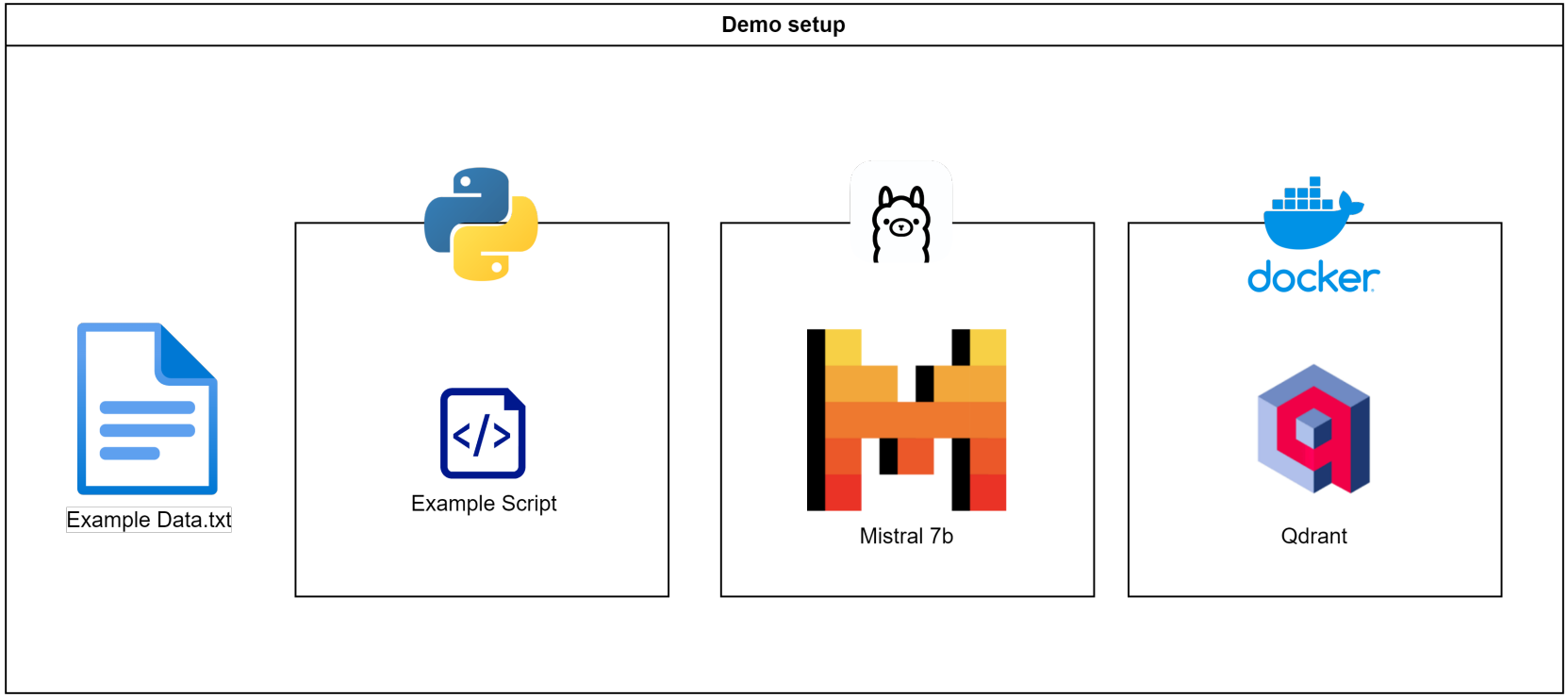
Corrective Retrieval-Augmented Generation



How CRAG Works – in Concept

Why Use RAG with AI

- Cost Affective compared to Training and fine-tuning. However, you can use it in conjunction with fine-tuning.
- Contextual Relevance
- Reduces Hallucinations
- Grounded on verifiable information improving reliability.



Lets look at an Example

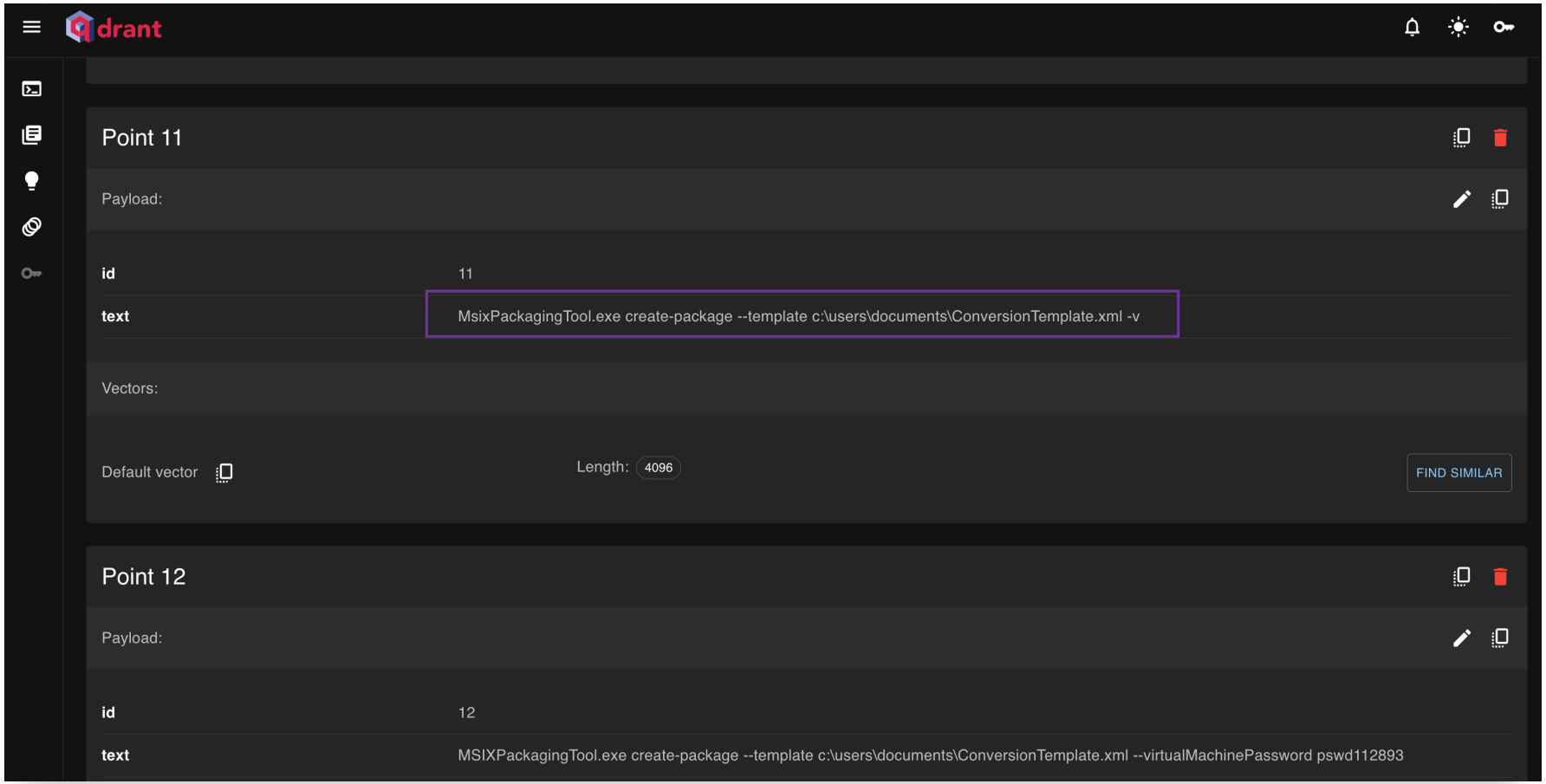
Upsert data into the Vector DB

```python
# Collection name
collection_name = "sample_data_collection"

# Delete the existing collection if it exists and create a new one
if qdrant_client.collection_exists(collection_name):
    logger.info("----------\nDELETING EXISTING COLLECTION\n----------")
    qdrant_client.delete_collection(collection_name)

logger.info("----------\nCREATING NEW COLLECTION\n----------")
# Create a new collection with the correct vector size
qdrant_client.create_collection(
    collection_name=collection_name,
    vectors_config=VectorParams(size=4096, distance=Distance.COSINE)
)

# Read sample data from file
logger.info("----------\nREADING SAMPLE DATA FROM FILE\n----------")
with open('exampledata.txt', 'r') as file:
    data = file.read()

# Split data into smaller sections for embeddings
sample_data = data.split('\n\n')  # Split by double newline for paragraphs

logger.info("----------\nGENERATING EMBEDDINGS\n----------")
# Insert sample data into Qdrant
points = []
for i, text in enumerate(sample_data):
    text = text.strip()
    if not text:
        continue
    embedding = get_embedding(text)
    point = PointStruct(
        id=i,
        vector=embedding,
        payload={"id": i, "text": text}
    )
    points.append(point)

# Upsert the points into the collection
logger.info("----------\nUPSERTING POINTS INTO QDRANT\n----------")
qdrant_client.upsert(
    collection_name=collection_name,
    points=points
)

# Log the embeddings and payloads inserted
for point in points:
    logger.info(f"Inserted point ID: {point.id}, Text: {point.payload['text']}")

# Define a query text
query_text = "What are the command line parameters for creating an MSIX package?"

# Get embedding for the query text
logger.info("----------\nGENERATING QUERY EMBEDDING\n----------")
query_embedding = get_embedding(query_text)
```

Asking a question with Vectors

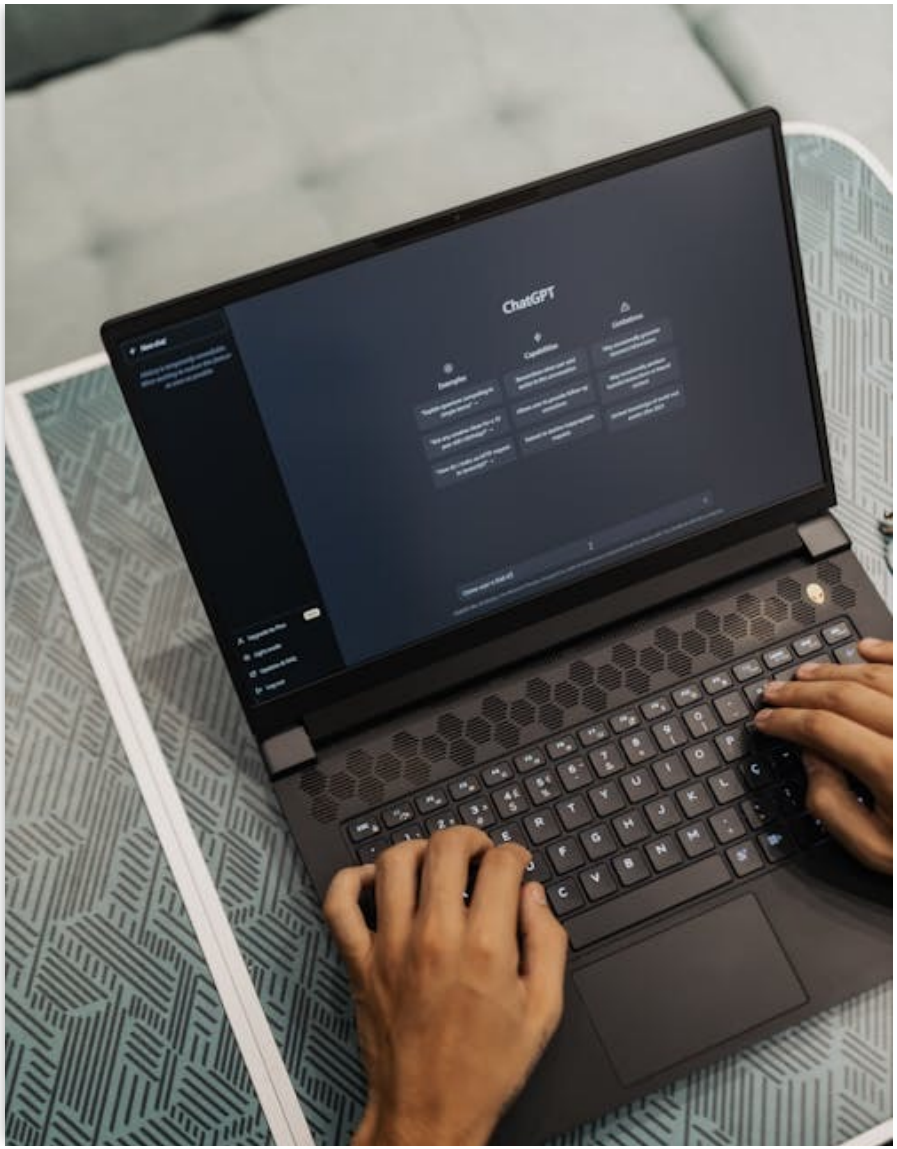
```python
# Define a query text
query_text = "What are the command line parameters for creating an MSIX package?"

# Get embedding for the query text
logger.info("----------\nGENERATING QUERY EMBEDDING\n----------")
query_embedding = get_embedding(query_text)

# Log the query embedding
logger.info(f"Query embedding: {query_embedding}")

# Search Qdrant with the query embedding
logger.info("----------\nSEARCHING QDRANT\n----------")
search_result = qdrant_client.search(
    collection_name=collection_name,
    query_vector=query_embedding,
    limit=5,  # Number of results to return
    with_payload=True
)

# Extract relevant texts from search results
relevant_texts = [result.payload['text'] for result in search_result]

# Log the search results
logger.info(f"Number of relevant vectors found: {len(search_result)}")
for result in search_result:
    logger.info(f"Search result - Score: {result.score}, Text: {result.payload['text']}")

# Construct the system prompt and completion prompt
system_prompt = "You are an expert in MSIX packaging and command line operations."
context = " ".join(relevant_texts)
completion_prompt = f"{system_prompt} Based on the following information, answer the query: {context} Query: {query_text}"

# Log the completion prompt
logger.info(f"Completion prompt: {completion_prompt}")

# Get the completion from Ollama API
logger.info("----------\nGETTING COMPLETION FROM OLLAMA\n----------")
completion = get_completion(completion_prompt)

# Print the completion
logger.info("----------\nAnswer\n----------")
logger.info(completion)
#print("Completion:", completion)
```

What are the command line parameters for creating an MSIX package?


drant
Point 11
Payload:
id
11
text
MsixPackagingTool.exe create-package --template c:\users\documents\ConversionTemplate.xml -v
Vectors:
Default vector
Length:
4096
FIND SIMILAR
Point 12
Payload:
id
12
text
MSIXPackagingTool.exe create-package --template c:\users\documents\ConversionTemplate.xml --virtualMachinePassword pswd112893

Summary

- Providing an LLM good context has been proven to improve fidelity
- Trusting the training data of a LLM is not recommended
- Controlling the context and information allows for more diverse selection of LLM’s
- RAG on its own may not be enough ; however, applying different techniques, including good prompt engineering, helps!



Questions