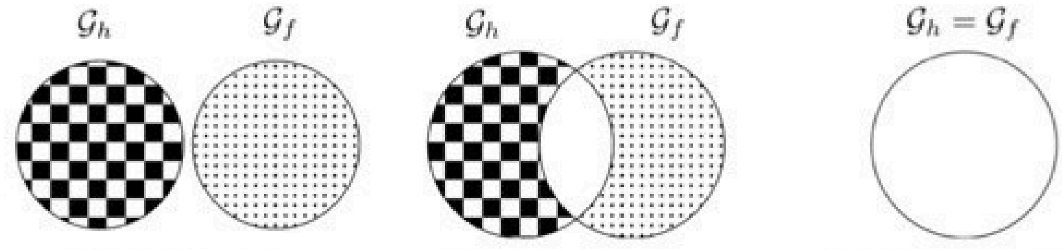

LLM Hallucination
why it happens and how to mitigate it

LLM Hallucinaten



(a) Total hallucination. (b) Some hallucination. (c) Hallucination-free.

Figure 2: Venn diagrams showing possible relations between $\mathcal{G}_h$ and $\mathcal{G}_f$.

hallucination can be classified in many ways, such as

1) intrinsic (contradicting w/ user input)
2) extrinsic

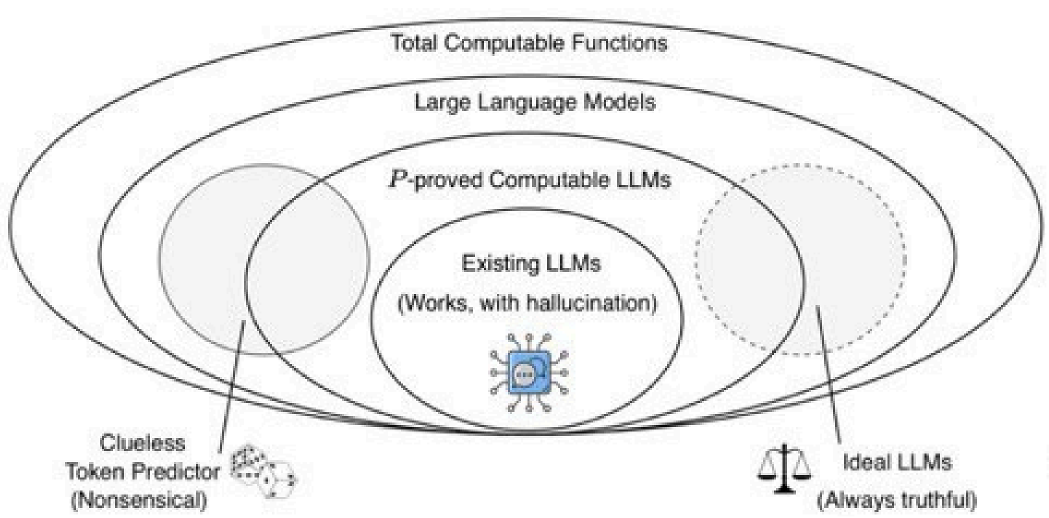
Causes

a variety of reasons,

1) data: bias, misinformation, bad quality
2) Training: architecture, losing context in long sequences, etc.
3) inference: randomness in sampling, softmax bottleneck (?)

Mitigation

for data issues, just use better Data :)
RAG, CoT and ToT also effective.
factuality-enhanced training objectives, new decoding methods to improve faithfulness



Definition 6 (Hallucination). An LLM h is hallucinating with respect to a ground truth function f, if $\exists s \in \mathcal{S}$ such that $h(s) \neq f(s)$.

hallucination can be thought of as a wrong prediction w.r.t the training data.

LLMs are essentially algorithms running on computers. They are total computable, meaning they produce outputs in finite times.

no matter how an LLM functions and is trained, as long as it's a computable function, there will always be some ground truth functions it cannot replicate.
this ground truth can be found using diagonilzation.

LLMs	s_0	s_1	s_2	s_3	s_4	$\cdots$
$\hat{h}_0$	$\hat{h}_0(s_0)$	$\hat{h}_0(s_1)$	$\hat{h}_0(s_2)$	$\hat{h}_0(s_3)$	$\hat{h}_0(s_4)$	$\cdots$
$\hat{h}_1$	$\hat{h}_1(s_0)$	$\hat{h}_1(s_1)$	$\hat{h}_1(s_2)$	$\hat{h}_1(s_3)$	$\hat{h}_1(s_4)$	$\cdots$
$\hat{h}_2$	$\hat{h}_2(s_0)$	$\hat{h}_2(s_1)$	$\hat{h}_2(s_2)$	$\hat{h}_2(s_3)$	$\hat{h}_2(s_4)$	$\cdots$
$\hat{h}_3$	$\hat{h}_3(s_0)$	$\hat{h}_3(s_1)$	$\hat{h}_3(s_2)$	$\hat{h}_3(s_3)$	$\hat{h}_3(s_4)$	$\cdots$
$\hat{h}_4$	$\hat{h}_4(s_0)$	$\hat{h}_4(s_1)$	$\hat{h}_4(s_2)$	$\hat{h}_4(s_3)$	$\hat{h}_4(s_4)$	$\cdots$
$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$
f	$\Delta(\hat{h}_0(s_0))$	$\Delta(\hat{h}_1(s_1))$	$\Delta(\hat{h}_2(s_2))$	$\Delta(\hat{h}_3(s_3))$	$\Delta(\hat{h}_4(s_4))$	$\cdots$

this can be proven in three scopes:

1) LLMs proven by a prover P to be total computable functions.
2) any computable set of LLMs
3) any individually computable LLM (most general case)

So this proves, any LLM whether running in Poly time or not, as long as computable is bound to hallucinate.

Pinch of Salt

these proofs are well-done but there are 3 things to consider:

1. LLMs are designed to perform well on specific tasks in language modeling, not to compute every possible function.
2. the definition of hallucination in this proof as $h(s) \neq f(s)$ is very strict, in real world, hallucination is context-dependent
3. the function f is deliberately constructed to differ from LLM outputs, but such a function may not be realistic and meaningful.

Hallucination is Inevitable: An Innate Limitation of Large Language Models

Ziwei Xu   Sanjay Jain   Mohan Kankanhalli
School of Computing, National University of Singapore
ziwei.xu@u.nus.edu   {sanjay,mohan}@comp.nus.edu.sg

So these proofs set forth in this paper, while are theoretically sound within the domain of computability theory, the practical impact of its limitations (the three mentioned points) are serious.

Hallucination may not be as fundamentally unavoidable as the paper suggests.

How to mitigate hallucination?

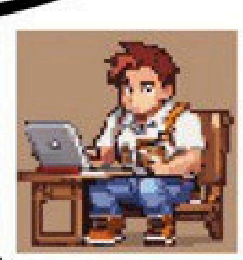
1) Increasing model size and training data
2) prompting techniques (CoT, ToT)
3) ensemble LLMs, majority voting or dialogue between models
4) external knowledge

A Survey on Hallucination in Large Language Models: Principles, Taxonomy, Challenges, and Open Questions

hallucination is rooted in psychology as "perception of an entity or event that is absent in reality"

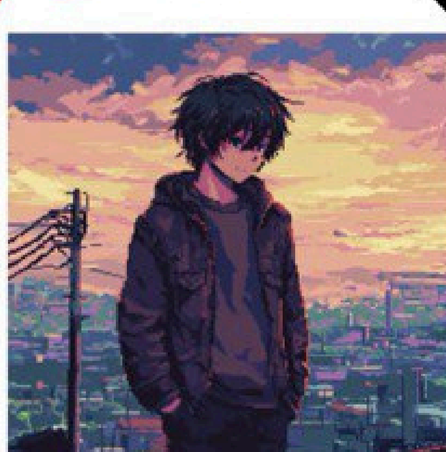
hallucination can be classified as

the range is so broad we cannot pinpoint a single origin for hallucination.



Hallucination by Data

- misinformation and bias
- knowledge boundary
  - domain knowledge
  - out-dated knowledge



- Inferior Data Utilization

means LLM is not fully leveraging Training Data. This could be due to

Knowledge shortcut: we don't fully understand how LLMs retrieve knowledge. one possibility is knowledge shortcuts. e.g. if LLM sees Canada and Toronto too much in training Data, it falsely think Toronto is the capital of Canada.

Knowledge recall failure

due to long-trail knowledge or complex scenarioss.

- **Complex Scenario** Beyond the challenges with long-tail knowledge, effective utilization of knowledge is inextricably linked with reasoning capabilities. For instance, in multi-hop question-answering scenarios, even if the LLM possesses the necessary knowledge, it may struggle to produce accurate results if multiple associations exist between questions, due to its limitations in reasoning (Zheng

Reversal Curse

A is B ✓   B is A?

trated in Table 4, although LLMs recognize Mount Everest as the world's highest peak, they fail to determine which would become the highest mountain if Everest's elevation were reduced by 500 meters, a task that requires complex reasoning ability.

Hallucination Type	Sub-Type	User Input	Model Output	Explanation
Factuality Hallucination	Factual Inconsistency	Tell me about the first person to land on the Moon.	Yuri Gagarin was the first person to land on the Moon.	The LLM's response is factual inconsistency as **Neil Armstrong was the first person to land on the Moon**, not Yuri Gagarin, who was the first human in space.
	Factual Fabrication	Tell me about the historical origins of unicorns.	Unicorns were documented to have roamed the plains of Atlantis around 10,000 BC, where they were considered sacred creatures and were often associated with royalty.	The LLM's response is a fabricated claim since **there's no verified, real-world evidence supporting the existence of unicorns in Atlantis**, especially in connection with royalty.
Faithfulness Hallucination	Instruction Inconsistency	Translate the following English question into Spanish: "What is the capital of France?"	The capital of France is Paris.	The user explicitly provided an instruction to translate the English question into Spanish. **Instead of performing the translation as instructed, the LLM chose to answer the question.**
	Context Inconsistency	Summarize the following paragraph about the Nile: The Nile is a major north-flowing river in northeastern Africa. It's traditionally considered the longest river in the world. Originating in the Great Lakes region of central Africa, it flows through several countries before emptying into the Mediterranean Sea.	The Nile originates from the mountain ranges of central Africa and flows into the Mediterranean.	The LLM's summary incorrectly states that the Nile originates from the "mountain ranges of central Africa," while **the paragraph clearly mentions that the Nile originates in the Great Lakes region of central Africa.**
	Logical Inconsistency	Solve the equation 2x + 3 = 11 step by step.	Step 1: Subtract 3 from both sides to get: 2x = 8. Step 2: Divide both sides by 2 to get: x = 3. So the result of the equation is x=3.	The first step correctly isolates the term with x to get 2x = 8. However, **the next step inaccurately divides 8 by 2 to yield a result of x = 3**, which is inconsistent with the earlier reasoning.

Hallucination From Training

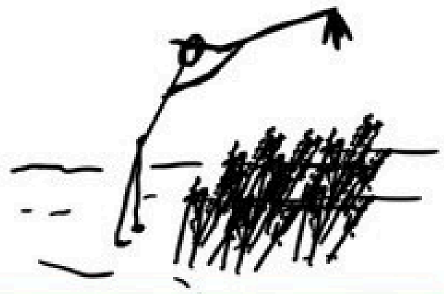
Architectural Flaws

Unidirectional Representation: LLMs train using Casual left-to-right token based training which can hinder their ability to capture some contextual dependencies. → This is actually pretty interesting and deserves some chatting with chatGPT

Attention Glitches

Exposure Bias

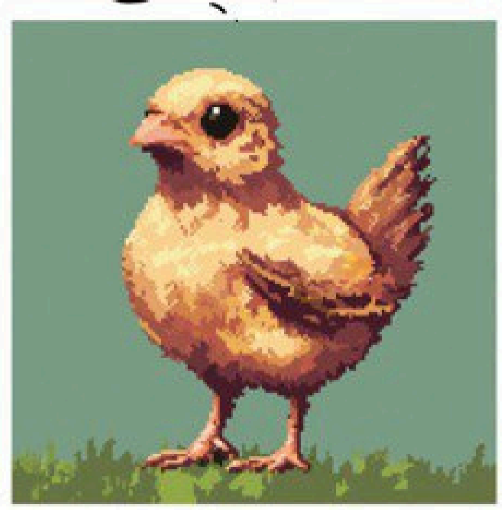
Autoregressive models learn to predict the next token based on previous correct tokens. during inference, the model only relies on the tokens generated by itself, which may be incorrect and result on a snowball effect



Alignment

SFT & RLHF essentially push the model to diverge from its internal beliefs to align with preferred outputs" even though they are factually incorrect.

Alignment does lead to weaker models eventually.



Are LLMs Random or Deterministic?

the loss function and output format of LLMS focus on the probability of next tokens.

The operation e.g matmul, sum, ... are deterministic. you get the same output given the same input.

decoder models implement randomness in their generations, by temperature, top-k, etc. This randomness causes creativity and diverse content.

why Randomness?

surprisingly, high likelihood sequence tends to be low quality, also known as likelihood trap.

Hallucination From Inference

This randomness while producing interesting outputs, could also create uncertainty and hallucination.

Insufficient Context Attention

another inference-level cause, LLMS tend to give more attention to closer tokens than the ones further in the context window, especially causing faithfulness hallucinations.

Softmax Bottleneck

a phenomena where models using softmax in their output are restricted in representing complex probabilities, especially when the layer before softmax has a lower dim than vocab size.

softmax itself makes calculations unstable so that doesn't help.

does the problem persist if we use logits w/o softmax? Yes!

it's honestly an architectural flaw rather than something related to softmax. :)

The point is to pay attention to layer ranks and avoid very low ranks that remove complex latent space relations.

Detecting Hallucination

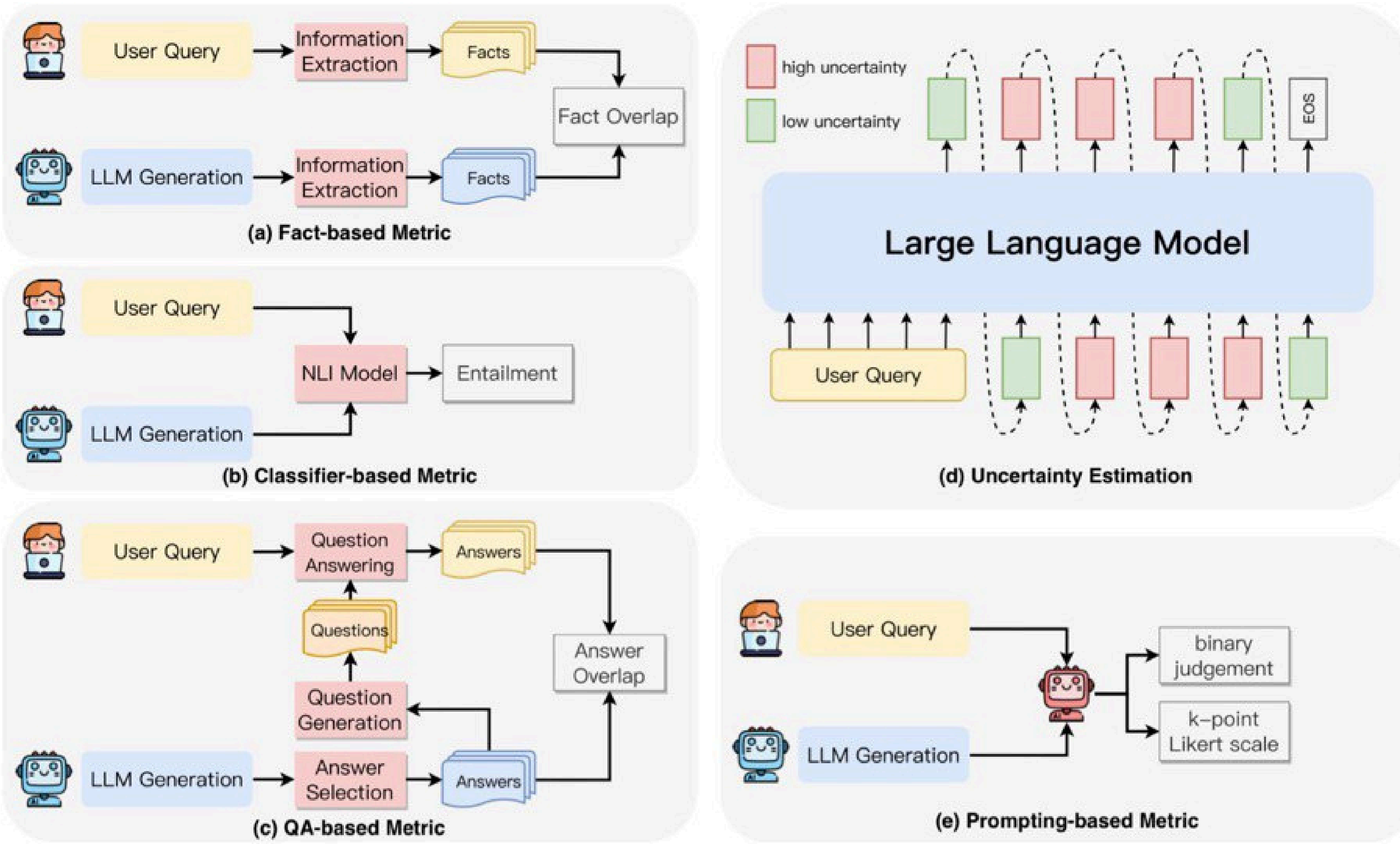
- retrieval of external knowledge
- uncertainty estimation

used via model's internal state or LLM behaviour.

Internal states can only be examined if there is access to model. An example is that low probability in tokens show the model is generally uncertain of the output.

LLM behavior analysis is best for when model is accessed through API. They typically involve measuring the factual consistency when asking the same direct/indirect prompt from the model.

another set of methods rely on measuring faithfulness which can be seen



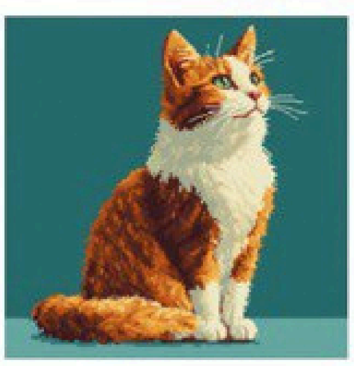


Figure 5: The illustration of detection methods for faithfulness hallucinations: **a) Fact-based Metrics**, which assesses faithfulness by measuring the overlap of facts between the generated content and the source content; **b) Classifier-based Metrics**, utilizing trained classifiers to distinguish the level of entailment between the generated content and the source content; **c) QA-based Metrics**, employing question-answering systems to validate the consistency of information between the source content and the generated content; **d) Uncertainty Estimation**, which assesses faithfulness by measuring the model's confidence in its generated outputs; **e) Prompting-based Metrics**, wherein LLMs are induced to serve as evaluators, assessing the faithfulness of generated content through specific prompting strategies.

QA based metrics: generate questions from a response, extract answers using the questions & user's resource, eval the response against the extracted answer

On the previous page we mentioned insufficient Contextual att now let's dive deeper

Self-attention tends to assign more weight to closer tokens, this is called **locality** or **positional bias**. why does this happen?

1. Natural Language Pattern shows words that are closer are more likely to be related. this creates a bias in learning Data.

2. Sinusoidal Encoding: leads to higher attention scores to nearby positions. (refer to Build LLM from scratch notes)

3. Dot-Product Similarity: Tokens with similar representations (which includes positional information) get higher attention scores.

4. Optimization Bias: it's easier for the model to learn local patterns that contribute immediately to reducing the loss function.

this is a natural aspect of human language, so it's not inherently bad, but it impacts model's capacity to capture long-term dependencies.