



Large Language Models

October 16th, 2025

Austin Perez, Ashutosh Deorukhkar, Bo Peng





Trusting Your Evidence: Hallucinate Less with Context-Aware Decoding

Weijia Shi, Xiaochuang Han, Mike Lewis, Yulia Tsvetkov, Luke Zettlemoyer, Scott Yih





Table of Contents

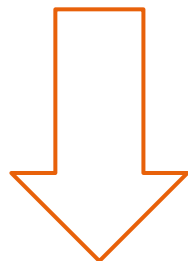
1	Overview	6	Related Work
2	Introduction	7	Conclusion
3	Methodology		
4	Experimentation		
5	Results		

Overview



Language Models Hallucinate

They can fail to pay enough attention to input context, resulting in outputs that fail to answer the query in context.



Context-Aware Decoding can help

This paper shows that CAD significantly improves the faithfulness of different LMs for summarization tasks.

Example

Input: Write a quote that ends in the word “early”.
“Better late than”

Regular: Never

Input: Write a quote that ends in the word “early”.
“Better late than”

CAD: Early



Introduction (Problem)

LMs are effective in generating coherent continuations of a prompt or document prefix.

During Generation, they rely on two sources of knowledge:

- Prior Knowledge
 - Learned in pre training, stored within model parameters;
- Context Knowledge
 - Passed as inputs in the prefix context

Discussion is still open to how a pre trained LM balances these two sources during generation



Introduction (Problem)

Research shows that LMs can fail to pay enough attention to new information introduced in the context knowledge, leading to hallucinations.

Example with LLaMA:

Model is presented with context that Argentina won the FIFA World Cups in 1978, 1986, and 2022.

When asked the question, “How many World Cups has Argentina won?”, the model still predicts “Two”, due in part to outdated training data.



Introduction (Solution)

This paper proposes using Context-Aware Decoding (CAD) to encourage the LM to pay attention to its context during generation

CAD samples from a new output distribution, which amplifies the difference between output probabilities with and without the context.

This downweights the prior knowledge when more relevant information is present in context.

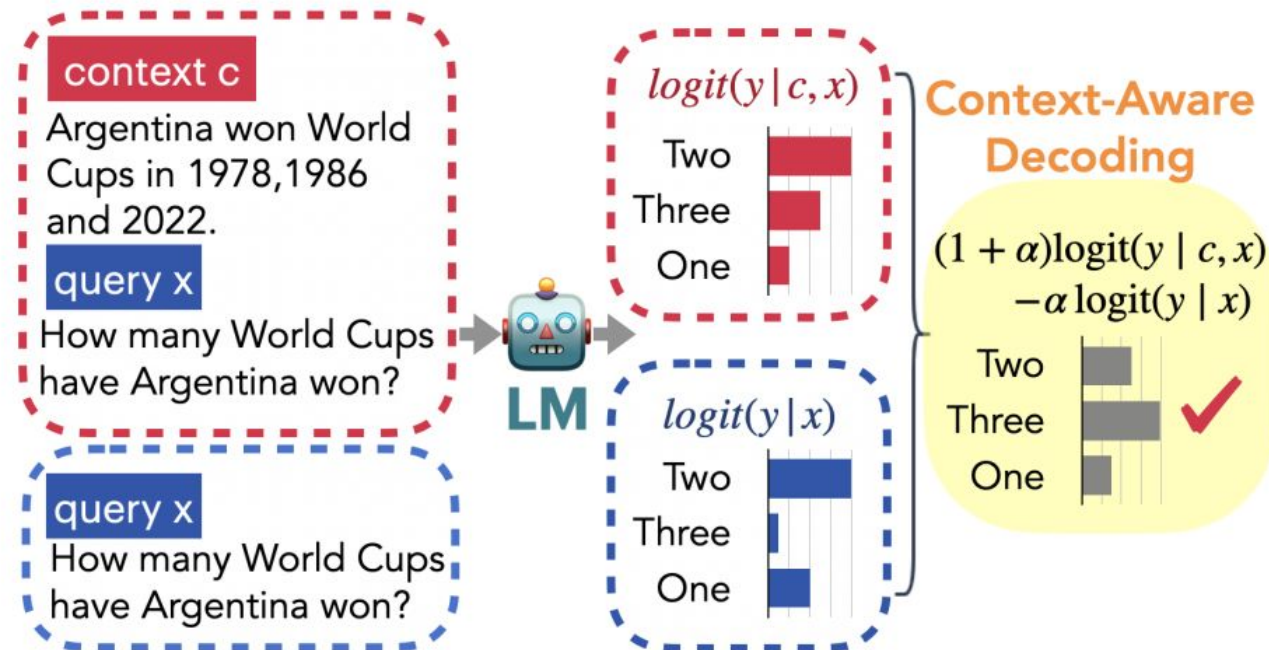


Figure 1: An illustration of context-aware decoding.



Methodology (Background)

Given LM θ , query \mathbf{x} , and context \mathbf{c} that contains external knowledge unfamiliar to LM's prior knowledge, we ask θ to generate a response \mathbf{y} .

The response can be sampled from probability distribution conditioned on \mathbf{x} and \mathbf{c} .

$$y_t \sim p_{\theta}(y_t \mid \mathbf{c}, \mathbf{x}, \mathbf{y}_{<t}) \propto \exp \text{logit}_{\theta}(y_t \mid \mathbf{c}, \mathbf{x}, \mathbf{y}_{<t})$$

As stated before, in cases where the context contains knowledge that is out-of-distribution with respect to the model, it is hypothesized that the model can struggle to effectively attend to the context and overly rely on the prior knowledge encoded in the model.



Methodology (CAD)

To mitigate these issues, we factor out the prior knowledge from the model's original output distribution contrastively, with the prior knowledge modeled as $p_{\theta}(y_t \mid \mathbf{x}, \mathbf{y}_{<t})$. We then adjust the model's original output probability distribution using the pointwise mutual information between the context and the generation conditioned on $\mathbf{x}, \mathbf{y}_{<t}$.

$$y_t \sim \tilde{p}_{\theta}(y_t \mid c, x, y_{<t}) \propto p_{\theta}(y_t \mid c, x, y_{<t}) \left(\frac{p_{\theta}(y_t \mid c, x, y_{<t})}{p_{\theta}(y_t \mid x, y_{<t})} \right)^{\alpha}$$

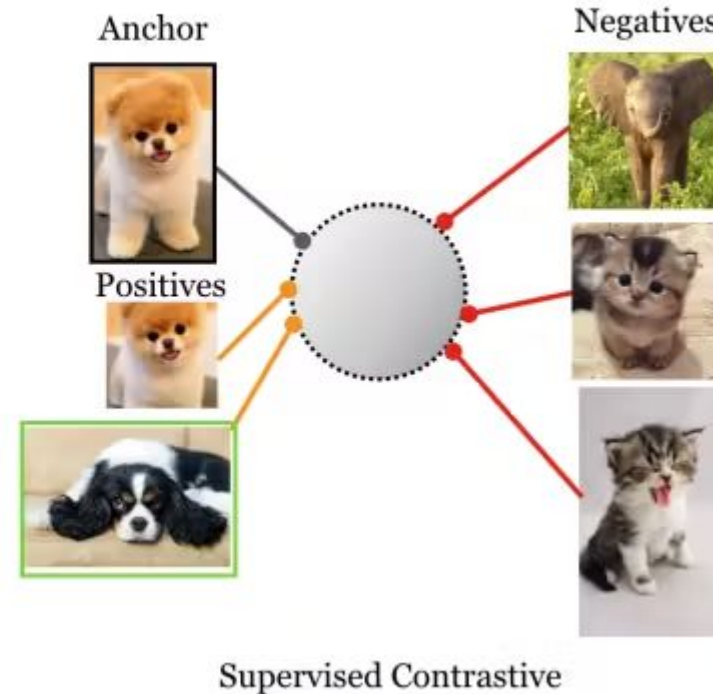
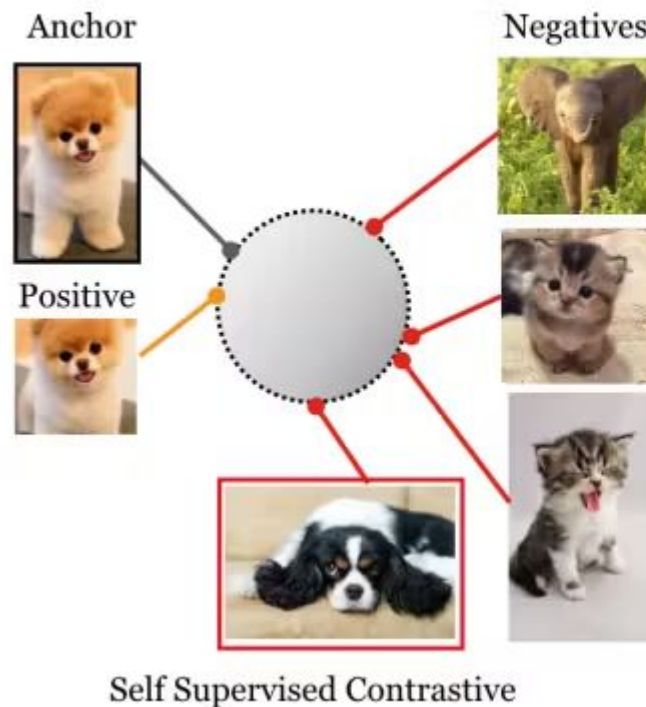
Essentially, outputs that become much more likely when the context is included are preferred. This needs to be normalized across all possible values of y_t .

$$y_t \sim \text{softmax}[(1 + \alpha) \text{logit}_{\theta}(y_t \mid c, x, y_{<t}) - \alpha \text{logit}_{\theta}(y_t \mid x, y_{<t})]$$



Methodology (CAD cont.)

Since CAD is just a contrastive ensemble between the logits of $p_{\theta}(y_t \mid \mathbf{c}, \mathbf{x}, \mathbf{y}_{<t})$ and $p_{\theta}(y_t \mid \mathbf{x}, \mathbf{y}_{<t})$, it has comparisons in other applications of AI. For example, a similar contrastive objective is universal in image generation, with \mathbf{c} being a control (anchor) to the image.





Experimentation

Several tasks that required LMs to read and reason over contexts and produce faithful outputs were evaluated.

The models were evaluated using prompting from prior works^{1,2}.

XSUM	
<i>c</i>	Article: Prison Link Cymru had 1,099 referrals in 2015-16 and said some ex-offenders were living rough for up to a year before finding suitable accommodation ...
<i>x</i>	Summarize the article in one sentence. Summary:
NQ-SWAP	
<i>c</i>	Tesla CEO Elon Musk is now in charge of Twitter , CNBC has learned ...
<i>x</i>	Who is Twitter CEO now?
MemoTrap	
<i>c</i>	Write a quote that ends in the word "early":
<i>x</i>	Better late than

Table 1: An illustration of the inputs to CAD applied to each dataset. CAD upweights the context *c* (in red) by sampling each token from $\text{softmax}[(1 + \alpha) \text{logit}_{\theta}(y_t \mid c, x, y_{<t}) - \alpha \text{logit}_{\theta}(y_t \mid x, y_{<t})]$.



Experimentation (Datasets + Metrics)

Summarization

Conducted on CNN-DM¹ and XSUM² datasets. The experiments used ROGUE-L³ to evaluate summarization quality.

BERT-Precision⁴ and FactKB⁵ were used to measure factual consistency.

Knowledge Conflicts

Conducted on MemoTrap⁶ and NQ-Swap⁷, two knowledge conflict datasets.

XSUM	
<i>c</i>	Article: Prison Link Cymru had 1,099 referrals in 2015-16 and said some ex-offenders were living rough for up to a year before finding suitable accommodation ...
<i>x</i>	Summarize the article in one sentence. Summary:
NQ-SWAP	
<i>c</i>	Tesla CEO Elon Musk is now in charge of Twitter , CNBC has learned ...
<i>x</i>	Who is Twitter CEO now?
MemoTrap	
<i>c</i>	Write a quote that ends in the word "early":
<i>x</i>	Better late than

Table 1: An illustration of the inputs to CAD applied to each dataset. CAD upweights the context *c* (in red) by sampling each token from $\text{softmax}[(1 + \alpha) \text{logit}_{\theta}(y_t | c, x, y_{<t}) - \alpha \text{logit}_{\theta}(y_t | x, y_{<t})]$.

¹(See et al., 2017) ²(Narayan et al., 2018) ³(Lin, 2004) ⁴(Pagnoni et al., 2021) ⁵(Feng et al., 2023)

⁶(Liu and Liu, 2023) ⁷(Longpre et al., 2021)



Experimentation (Models + Baselines)

CAD was applied to OPT (13B and 30B)¹, GPT-Neo (2.7B and 20B)², LLaMA (13B and 30B)³ and FLAN-T5 (XL 3B and XXL 11B)⁴.

CAD introduces a hyperparameter α to control adjustment level. For all summarization tasks $\alpha = 0.5$ and for all knowledge conflict tasks $\alpha = 1$. These were found to be effective values for each task, as a higher α was more effective in knowledge conflict where the prior knowledge needed to be factored out more.

The baselines were regular decoding following prior work^{5,6} to use greedy decoding for knowledge conflict tasks and top-p sampling with $p=0.9$ for summarization⁷. For CAD the same sampling strategies were used on top of the adjusted probability distribution



Results (Summarization)

CAD outperforms by a large margin in all 8 models across both datasets.

For LLaMA 30B in CNN-DM, CAD leads to 21% increase in ROUGE-L, 14.3% increase in factKB and 7.8% increase in BERT-P.

CAD improves the quality and factuality of generated summaries.

			CNN-DM			XSUM		
			ROUGE-L	factKB	BERT-P	ROUGE-L	factKB	BERT-P
OPT	13B	Regular	22.0	77.8	86.5	16.4	47.2	85.2
		CAD	27.4	84.1	90.8	18.2	64.9	87.5
	30B	Regular	22.2	81.7	87.0	17.4	38.2	86.1
		CAD	28.4	87.0	90.2	19.5	45.6	89.3
GPT-Neo	3B	Regular	24.3	80.5	87.5	17.6	54.0	86.6
		CAD	27.7	87.5	90.6	18.1	65.1	89.1
	20B	Regular	18.7	68.3	85.2	14.9	42.2	85.7
		CAD	24.5	77.5	89.4	19.0	63.3	90.6
LLaMA	13B	Regular	27.1	80.2	89.5	19.0	53.5	87.8
		CAD	32.6	90.8	93.0	21.1	73.4	91.7
	30B	Regular	25.8	76.8	88.5	18.7	47.7	87.1
		CAD	31.8	87.8	92.2	22.0	66.4	90.3
FLAN	3B	Regular	25.5	90.2	91.6	18.8	31.9	88.2
		CAD	26.1	93.9	92.1	19.5	35.9	88.8
	11B	Regular	25.4	90.4	91.4	19.4	29.8	88.3
		CAD	27.1	93.1	92.2	20.0	35.0	88.8

Table 2: CAD consistently outperform the regular decoding method in terms of both summary quality metric (ROUGE-L) and summary factuality (factKB and BERT-P). The best scores for each setting are boldfaced. FLAN 3B and 11B refer to FLAN-T5 XL and FLAN-T5 XXL respectively.



Results (Knowledge Conflict)

CAD shows significant improvements to regular decoding in almost all settings.

GPT-Neo 20B improved by 54.4% on Memotrap and by 128% on NQ-SWAP.

CAD proved particularly beneficial for LMs to adhere to a given context in scenarios where prior knowledge contradicts context.

Model		Decoding	Memo.	NQ	NQ-SWAP
OPT	13B	Reg.	32.5	29.2	18.8
		CAD	44.5	32.2	36.9
	30B	Reg.	28.4	29.4	14.7
		CAD	41.0	35.5	29.0
GPT.	3B	Reg.	22.5	31.9	19.1
		CAD	47.3	39.9	41.2
	20B	Reg.	37.1	22.8	16.1
		CAD	57.3	32.1	36.8
LLAMA	13B	Reg.	23.8	22.3	11.7
		CAD	57.1	33.6	36.7
	30B	Reg.	25.8	23.8	9.6
		CAD	50.6	34.0	37.7
FLAN	3B	Reg.	69.2	81.8	71.4
		CAD	72.2	80.3	73.3
	11B	Reg.	82.0	85.5	73.0
		CAD	88.7	82.5	77.1

Table 3: CAD outperforms the regular decoding method (Reg.) in all settings except for FLAN-T5 on NQ. Note that FLAN-T5 is trained on NQ dataset during instruction-finetuning.



Results (Analysis)

CAD brings consistent improvements to LMs with different sizes. Figure 2 shows further investigation into the effect of CAD on different model sizes, with clear improvement in all cases.

Figure 3 shows the effect of adjustment level α , which shows how the best value was derived for testing.

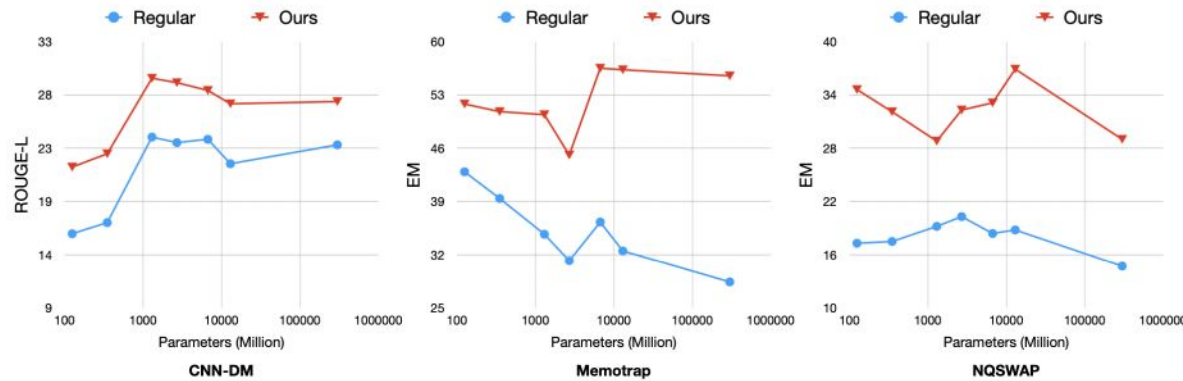


Figure 2: OPT models of varying sizes consistently benefit from CAD. The x-axis indicates the size of language models and the y-axis is the performance.

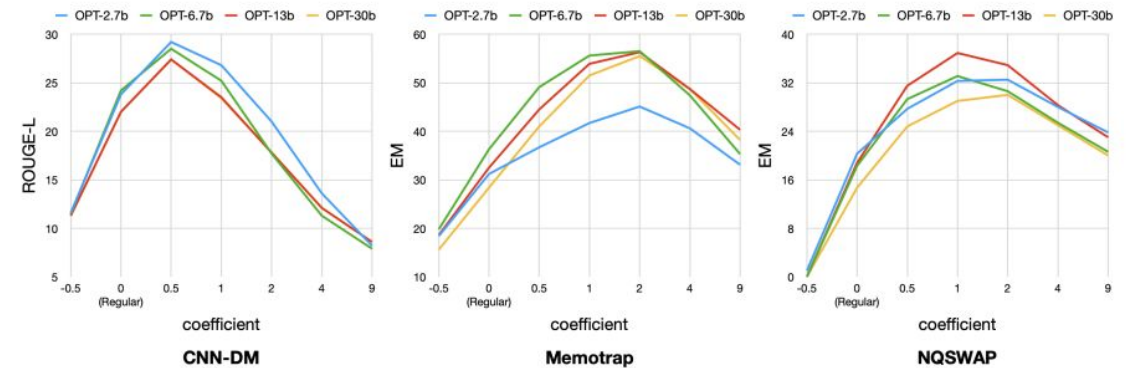


Figure 3: Effect of the adjustment level α . The y-axis is the performance and the x-axis is α .



Related Work

Summarization Factuality:

- Summarization models have a tendency to generate hallucinations (Maynez et al., 2020; Pagnoni et al., 2021)

Knowledge Conflicts:

- When presented an updated document with conflicting knowledge, we expect models to generate responses based on the provided contexts. However, simply adding documents does not always change the model's predictions. (Khandelwal et al., 2020; Shi et al., 2023; Min et al., 2022; Yasunaga et al., 2023)
- Current LMs often overlook the contexts and rely heavily on prior knowledge. (Longpre et al., 2021; Chen et al., 2022)

Contrastive Decoding Methods:

- These methods have been extensively explored for text generation, such as Coherence boosting (Malkin et al., 2021), MMI-based decoding (Li et al., 2015), and DExperts (Liu et al., 2021).

Conclusion



Problem

Off-the-shelf LMs may suffer from an insufficient attention to the supplied context compared to its learned prior knowledge leading to an unfaithful generation to the input context.

Solution

By penalizing outputs consistent with prior knowledge but inconsistent with the input context, Context-Aware Decoding provides more reliable and factual outputs across different language models of various sizes. This has been shown to be true for both summarization and knowledge conflict tasks.



How Language Model Hallucinations Can Snowball

**Muru Zhang, Ofir Press, William Merrill, Alisa Liu, Noah,
A. Smith**





Table of Contents

1	Overview
2	Introduction
3	Methodology
4	Experimentation
5	Results



When One Lie Leads to a Hundred More

- In human conversations, when someone tells a lie, they often need to tell many more lies to keep the first one hidden.
- This paper draws a similar analogy for Language Models (LMs):
 - When an LM makes **one false claim**, it may produce **more false claims** while trying to justify the very first one.
- The authors define this cascading effect as **"Snowball Hallucination"**:
 - A phenomenon where an LM's initial false statement leads to additional false claims.



What Does This Paper Examine?

- The researcher's tested **three distinct question-answering datasets** to study how hallucinations develop and compound:
 - **Primality Testing:** 500 randomly chosen primes between 1,000 and 20,000.
 - **Senator Search:** 500 questions of the form: "Was there ever a US Senator that represented the state of x and whose alma mater was y?"
 - **Graph Connectivity:** 500 questions representing 12 flight routes among 14 cities
- These datasets were evaluated using **ChatGPT 3.5 Turbo** and **GPT-4 Models**.
- Results showed that ChatGPT could recognize **67%** of its own hallucinations, while GPT-4 identified **87%**.



Why This Study Matters?

- Traditionally, these Hallucinations have often been blamed on "**knowledge gaps**".
- The Importance of the Dataset:

Dataset	Purpose
Primality Testing	Mathematical Verification
Senator Search	Factual Reasoning
Graph Connectivity	Logical Reasoning based on given rules

- Each Dataset has **two Parts**:
 - a) Original Question: Where model may hallucinate.
 - b) Verification Question: Checks if model recognizes its earlier mistake.

Snowball Hallucination in Action



Dataset	Original Question	Verification Question
Primality Testing	<p>\faUser User: Is 10733 a prime number?</p> <p>\faRobot GPT-4: No... It can be factored into 3×3577.</p>	<p>\faUser User: Is 10733 divisible by 3? Answer with either Yes or No.</p> <p>\faRobot GPT-4: No</p>
Senator Search	<p>\faUser User: Was there ever a US senator that represented the state of New Hampshire and whose alma mater was the University of Pennsylvania?</p> <p>\faRobot GPT-4: Yes... His name was John P. Hale</p>	<p>\faUser User: Was John P. Hale's alma mater University of Pennsylvania?</p> <p>\faRobot GPT-4: No... [it] was Bowdoin</p>
Graph Connectivity	<p>\faUser User: Current flight information There is a flight from city F to city K <i>[... 10 other rules cut for space ...]</i> Question: Is there a series of flights that goes from city B to city E?</p> <p>\faRobot GPT-4: Yes... the route is as follows: ... City K to City G...</p>	<p>\faUser User: <i>[...flight information given in the context...]</i> Based on the above flight information, is City K to City G a valid flight?</p> <p>\faRobot GPT-4: No, based on the above flight information, there is no direct flight from City K to City G.</p>



Summary of the Three Examples

- Primality Testing -> Mathematical Reasoning -> False Factorization -> Recognizes Math Error
- Senator Search -> Factual Reasoning -> Mixed Up Biographical Info -> Admits Factual Mistake
- Graph Connectivity -> Logical Reasoning -> Invented Non-Existent Connection -> Contradicts Earlier Claim

Snowball Hallucinations are **not limited** to a **Specific Domain**
They emerge when a model **commits too early** and prioritizes
self-consistency over correctness



Why do we expect such behavior?

Language Models commit too early and must stay consistent thereafter!

Initial Committal	Inherently Sequential Reasoning
LM's often say " Yes " or " No ", even before reasoning.	Problems like primality and graph connectivity need multi-step logic .
Once that first token is generated, the model feels " Locked In ".	Transformers cannot solve these in a single time step .
It then build explanations to justify the incorrect response.	They guess and hallucinate reasoning when forced to answer immediately.



How do we test Snowball Hallucination?

Short Recap

- **Purpose:** To evaluate whether ChatGPT and GPT-4 hallucinate further after making an initial wrong answer.
- **Method:** Create three yes/no datasets to trigger snowballing.
- **Key Idea:** If model gives a wrong answer -> it is asked to justify -> we check if justification contains any false claims.



How were ChatGPT and GPT-4 Tested?

The Experimental setup is fairly simple as:

- **Models Used:** ChatGPT (3.5 Turbo) and GPT-4 by OpenAI.
- **Mode:** Zero-Shot
- **Decoding:** Greedy Decoding
- **Two Stage Evaluation based off:**
 - Accuracy of yes/no
 - Model's ability to detect its own hallucination



Results: Both Models Snowball Easily

- ChatGPT's accuracy was approximately 39.9% and GPT-4 only 16.6%.
- GPT-4 performs surprisingly worse than ChatGPT 3.5 turbo.
- When verifying their own false claims:
 - ChatGPT detects 67% of hallucinations; GPT-4 detects 87%

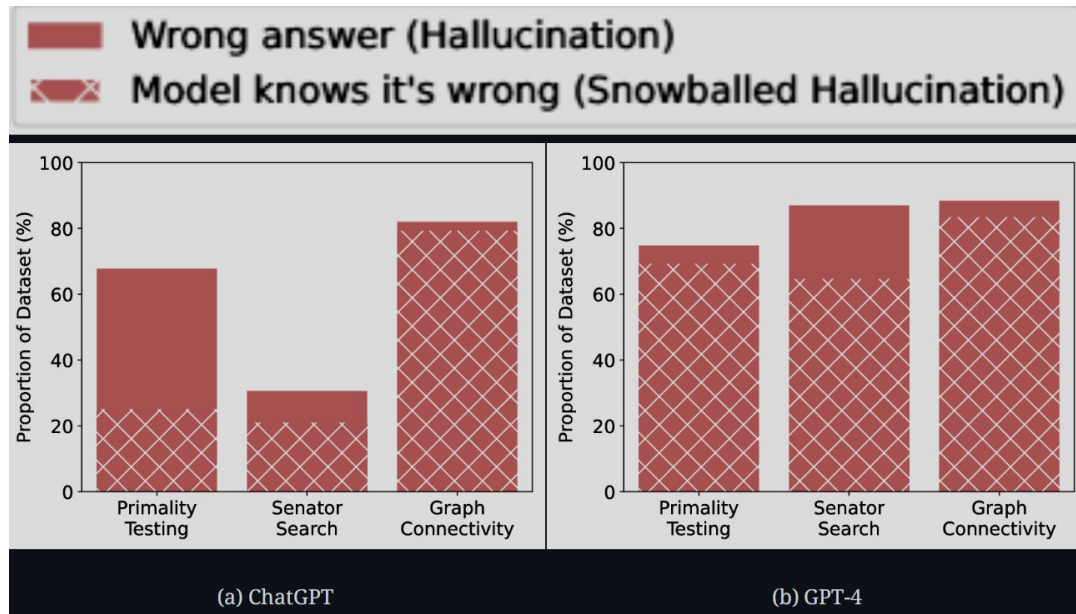


Figure 1

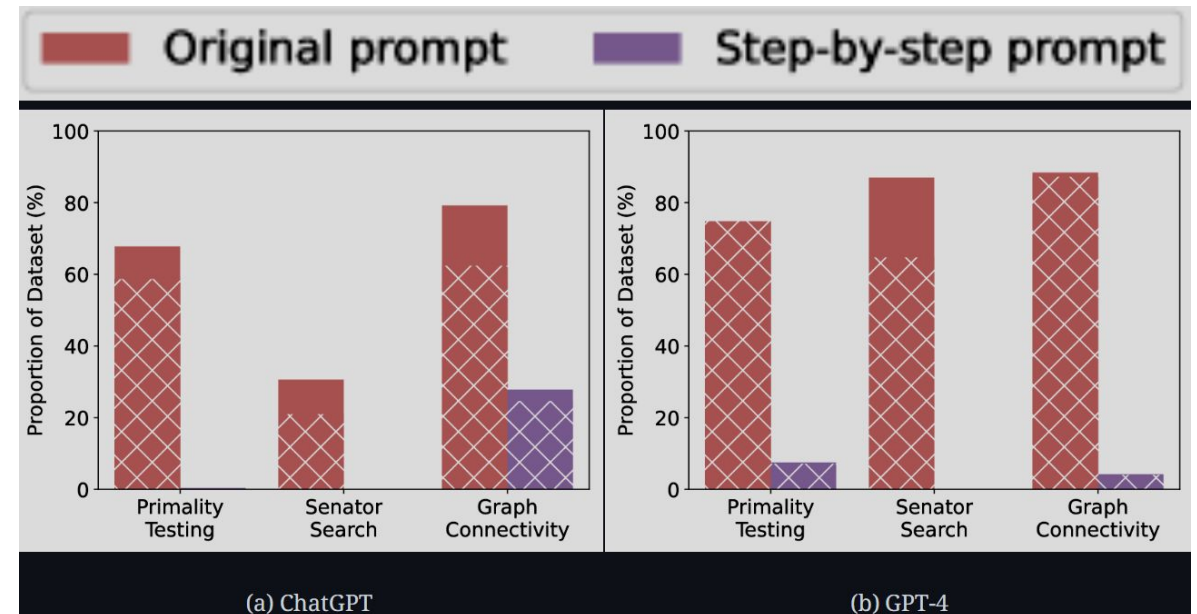


Figure 2



What do these results tell us?

- Both models fail simple yes/no tasks -> **commit early** and **justify wrong answers**.
- They can **detect hallucinations in isolation** but not in context.
- GPT-4's stronger **coherence amplifies** snowballing when it commits early.

Snowballing = When reasoning continues to **defend** the wrong initial belief.



Can we stop Hallucinations from Snowballing?

- **Root Cause:** LMs are trained to generate text coherent with prior tokens, hence once they commit to a wrong answer, they justify it further.
- **Goal:** Test if prompting or decoding/training tweaks can reduce this behavior.

Prompt
Engineering

Algorithmic Fixes



Does reasoning before answer help?

- Adding **"Let's think step by step"** -> Forces longer reasoning chain before answer.
- Accuracy Improvements:
 - Senator Search: 100%
 - Primality $\leq 10\%$ Error
 - Graph $\leq 30\%$ Error
- However, models still hallucinate within reasoning chains indicating that snowball effect still persists.

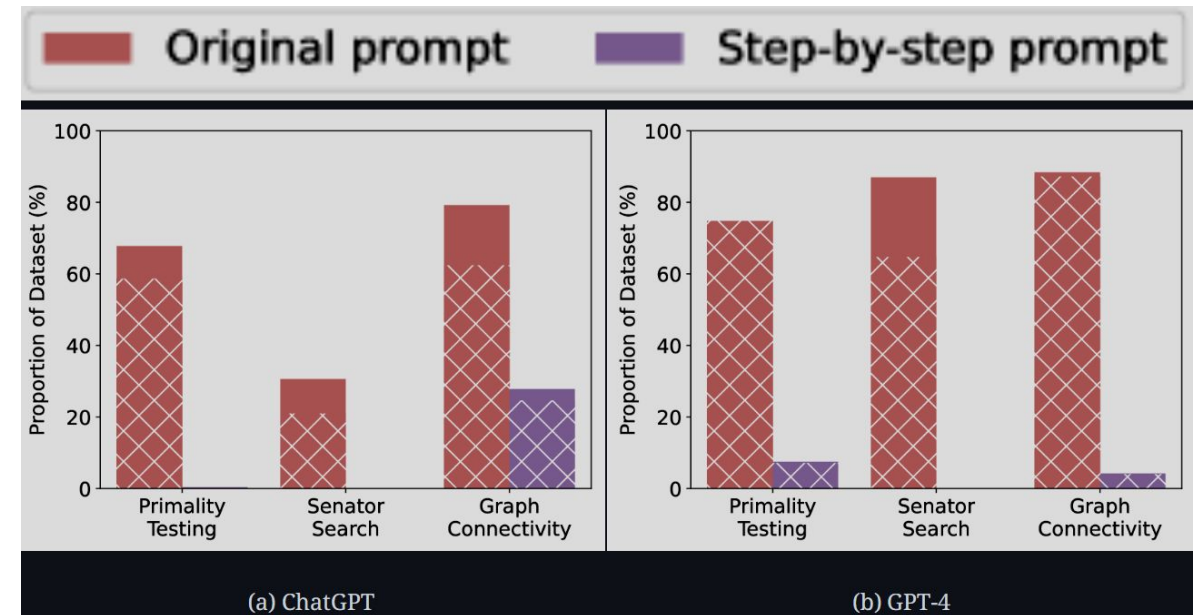


Figure 3



Do Decoding or training tweaks help?

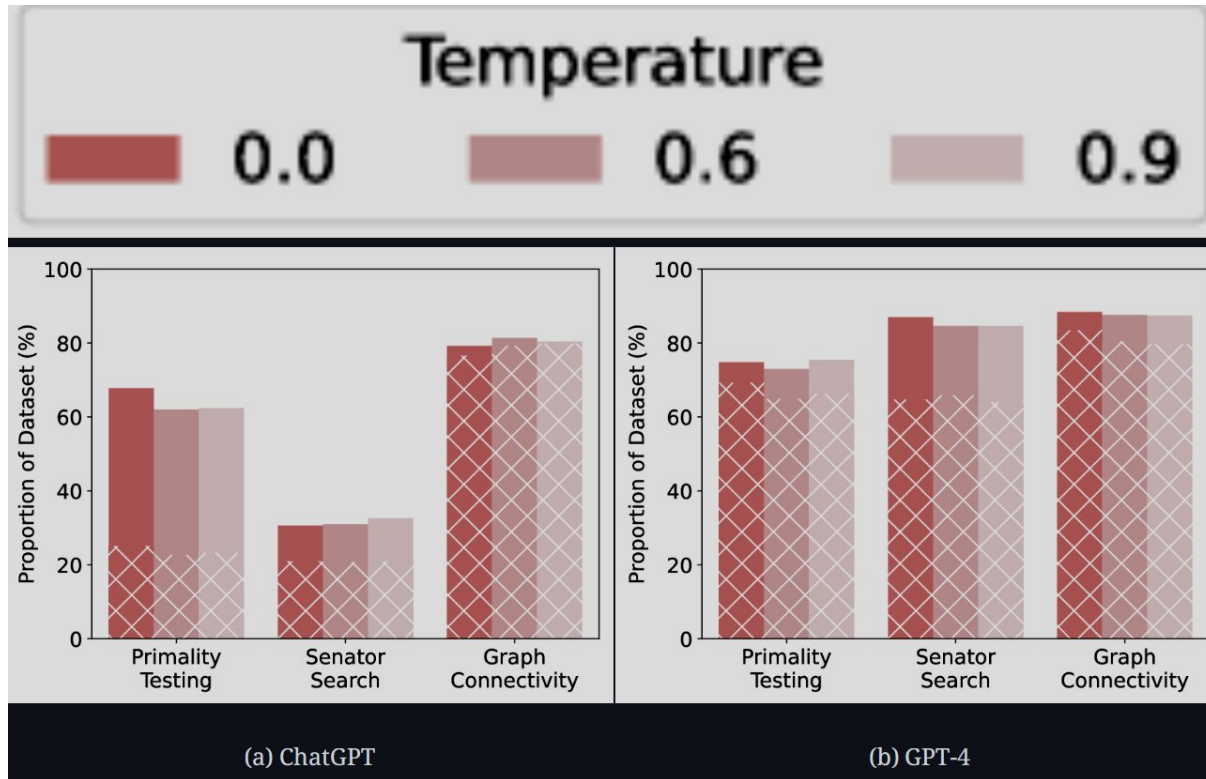


Figure 4

- **Increasing Temperature:** Has no real change, and snowball rates stay high.
- **Top-k/Nucleus Sampling:** Portrays similar issues as the model still commit too early.
- **Beam Search:** Potential aid as it keeps multiple answer paths but is not available in API.
- **Training Time:**
 - Emphasize on "Reason then Answer".
 - Fine tune with backtracking (model corrects itself).



What Have Others Found About Hallucinations?

- **Knowledge Gap as the Core Cause:**

- Rohrbach et al. (2018) showed image captioning models fabricate visual facts not grounded in the input.
- Raunak et al. (2021) highlighted that factuality issues arise when models overgeneralize beyond their training distribution.

Takeaway:
Hallucinations were
blamed on what
model does not know

- **Retrieval Augmented Models as a Fix:**

- Lewis et al. (2020, RAG) proposed Retrieval-Augmented Generation to inject relevant external documents into the prompt.
- Shuster et al. (2021) and Peng et al. (2023) showed grounding responses in retrieved facts reduces factual drift.

Takeaway:
Dominant
Assumption - More
context -> Fewer
Hallucinations



Conclusion

- Definition of Snowballing Hallucination - where early model mistakes trigger further falsehoods.
- Found that this occurs on simple factual queries.
- Percentage metric detection of ChatGPT 3.5 Turbo and GPT-4 Models.
- Fluency and coherence training may unintentionally amplify factual drift.

Small Lie turns into a snowball of coherent but false reasoning



Limitations

- Focused only on QA tasks in English.
- Tested only on two models by OpenAI (with no fine tuning or beam search).
- Future explorations:
 - Explore summarization, code generation tasks.
 - Test open weight models (with access to token probabilities)
 - Investigate training-time fixes (reason then answer, backtracking).



Hallucination Detection for Generative Large Language Models by Bayesian Sequential Estimation

Xiaohua Wang, Yuliang Yan, Longtao Huang, Xiaoqing Zheng,
Xuanjing Huang



Table of Contents



1	Introduction and Related Work
2	Method
3	Experiments
4	Conclusion and Limitations
5	Q&A



Introduction and related work

Problem: hallucination

- **Intrinsic hallucination:** the model generates content that is factually incorrect or self-contradictory.

Q: What is the capital of France?

A: The capital of France is Berlin.

- **Extrinsic hallucination:** the model fabricates information that does not exist.

Q: Describe the plot of Harry Potter 8.

A: (Invented story of a nonexistent Harry Potter 8).



Introduction and related work

Current Solutions for Hallucination Detection

- **Sampling Consistency** (Manakul et al., 2023)
Ask the model the same question multiple times.
Consistent answers → likely factual.
Advantages: Intuitive, no external knowledge.
Disadvantages: Requires many samples, computationally expensive.
- **Evidence Retrieval & Fact-Checking** (Thorne et al., 2018; Augenstein et al., 2019; Kim et al., 2023; Kamoi et al., 2023; Chen et al., 2023)
Retrieve external evidence (Wikipedia, web, knowledge graphs).
Verify claim against retrieved docs.
Advantages: High accuracy, direct verification.
Disadvantages: Fixed retrieval (e.g., 5–10 docs) → inflexible, resource-wasting, or insufficient.
- **Model Self-Evaluation** (Kadavath et al., 2022; Mündler et al., 2023; Azaria and Mitchell, 2023)
Model judges its own reliability (hidden states, “Are you sure?”).
Disadvantages: Unreliable, frequent misjudgment.



Introduction and related work

Bayesian Sequential Analysis

Origin: Wald (1947), Decision Theory + Bayesian Inference.

Goal: minimize expected risk while considering different costs:

False alarm (classifying factual as hallucination).

Miss (failing to detect hallucination).

Retrieval cost (time, computation).

Traditional Backward Induction

→ obtain the optimal solution, but high the computational complexity



Introduction and related work

Bayesian Sequential Analysis

To solve the high complexity, approximate methods:

- Cardillo & Fu (1968): assume “next step is the last step.”
→ suboptimal solution but greatly reduces the computational cost.
- Brockwell & Kadane (2003): grid approximation
→ reduce the complexity from exponential to linear.

Method in this paper

Treat each retrieved document as an evidence step.

After each document: recalculate risk.

If evidence is enough → stop retrieval.

Otherwise → continue searching.

**Ensures effective detection while reducing unnecessary retrieval
→ more accurate and efficient.**

Method



Claim Decomposition

Split a complex claim into subclaims (**1 entity + 1 attribute**).

e.g. “Giuseppe Mariani was an Italian footballer, born in Milan, died in Rome.”

→ 3 subclaims:

He was a footballer.

He was born in Milan.

He died in Rome.

Retrieve web documents for each subclaim *step-by-step*

→ Not fixed number (5 or 10) but dynamically decided

Method



**Each retrieved doc \rightarrow compute entailment score $s(C, E) \in [0, 1]$
 \rightarrow max score among segments used:**

$$s(C, E) = \max_i s(C, E_i)$$

- C : claim or subclaim generated by LLM
- E_i : the i -th retrieved document (or its text segment)

Convert score \rightarrow discrete feature (0–9):

$$f(C, E) = \lfloor 10 \cdot s(C, E) \rfloor$$

Use Naive Bayes Classifier (NBC) to update probability of being factual:

$$\pi_1(n+1) = \frac{\pi_1(n)P(f_{n+1}|\theta_1)}{(1 - \pi_1(n))P(f_{n+1}|\theta_0) + \pi_1(n)P(f_{n+1}|\theta_1)}$$

- $\pi_1(n)$: current probability that the claim is **factual** after n retrieved docs
- f_{n+1} : new entailment feature from the $(n+1)$ -th document
- θ_1 : hypothesis that the claim is factual
- θ_0 : hypothesis that the claim is hallucinated



Method

Bayesian Stop-or-Continue Decision

$$R_{stop}(n) = \min((1 - \pi_1(n))C_M, (1 - \pi_0(n))C_{FA})$$

$$R_{continue}(n) = C_{retrieve} + \sum P(f_{n+1})R_{stop}(n + 1)$$

- C_M : **miss cost** — misclassifying hallucination as factual
- C_{FA} : **false alarm cost** — misclassifying factual as hallucination
- $C_{retrieve}$: **retrieval cost** — time or computation cost

Decision rule:

- If $R_{stop}(n) < R_{continue}(n)$, stop retrieving;
- Otherwise, continue to the next document.

Final Aggregation

$$P_{factual}(C) = \min_i P_{factual}(C_i)$$

If any subclaim is judged hallucinated → mark the whole sentence as hallucination.

Method

1. Claim Decomposition
2. Document Retrieval
3. Entailment Scoring
4. Bayesian Sequential Decision
5. Aggregation of Subclaims

Algorithm 1: Hallucination detection

Input: C : A claim generated by LLMs for hallucination detection;
 $\pi_1(0)$: Initial probability of C being factual;
 K : Maximum retrieval times;
 C_{FA} : Cost of false alarm;
 C_M : Cost of miss;
 $C_{retrieve}$: Cost of retrieving an document;
Output: $P_{factual}(C)$: Probability of C being factual;

```

1  $\{C^1, C^2, \dots, C^L\} \leftarrow \text{ClaimDecompose}(C)$ ;
2 for  $i \leftarrow 1$  to  $L$  do
3    $n \leftarrow 1$ ;
4   while  $n \leq k$  do
5      $E^n \leftarrow \text{RetrieveDocument}(C^i)$ ;
6      $f_n \leftarrow \text{CalEntailmentFeature}(E^n, C^i)$ ;
7      $\pi_1(n) \leftarrow \text{NBC}(\pi_1(n-1), f_n)$ ;
8      $R_{stop}(n) \leftarrow \min((1 - \pi_1(n))C_M,$ 
9        $(1 - \pi_0(n))C_{FA})$ ;
10     $R_{continue}(n) \leftarrow$ 
11       $C_{retrieve} + \mathbb{E}_{f_{n+1}}(R_{stop}(n+1))$ ;
12    if  $R_{stop}(n) < R_{continue}(n)$  then
13      break;
14    else
15       $n \leftarrow n + 1$ ;
16    end
17  end
18   $P_{factual}^i = \pi_1(n)$ ;
19 end
20  $P_{factual}(C) = \min_i P_{factual}(C^i)$ ;
21 Return:  $P_{factual}(C)$ 

```



Experiments



Dataset

SelfCheckGPT dataset: 1,908 sentences, 298 articles (GPT-3 generated)

Labels: Accurate, Minor Inaccurate, Major Inaccurate

Grouped into: Factual vs Nonfactual

Baselines

SelfCheckGPT: multiple samples → consistency check

Self-detection (ChatGPT): model judges its own reliability

Metrics

Sentence-level: AUC-PR, Accuracy

Passage-level: Pearson / Spearman correlation

Efficiency: Avg. number of retrieved documents

Experiments



Method		Sentence-level (AUC-PR)				Passage-level (Corr.)	
		Evidence Num	Nonfact	Factual	Acc	Pearson	Spearman
Self-Detection		-	-	-	31.01	-	-
SelfCheckGPT	w/ BERTScore	20	81.96	44.23	-	58.18	55.90
	w/ QA	20	84.26	48.14	-	61.07	59.29
	w/Unigram (max)	20	85.63	58.47	-	64.71	64.91
	Combination	60	87.33	61.83	-	69.05	67.77
Our Framework	$C_M = 14, C_{FA} = 24$	3.05	82.42	57.01	80.24	71.37	64.55
	$C_M = 28, C_{FA} = 96$	6.22	86.45	61.96	82.39	81.18	74.20

Sentence-level

Comparable to SelfCheckGPT on AUC-PR

Accuracy > 80%

Passage-level

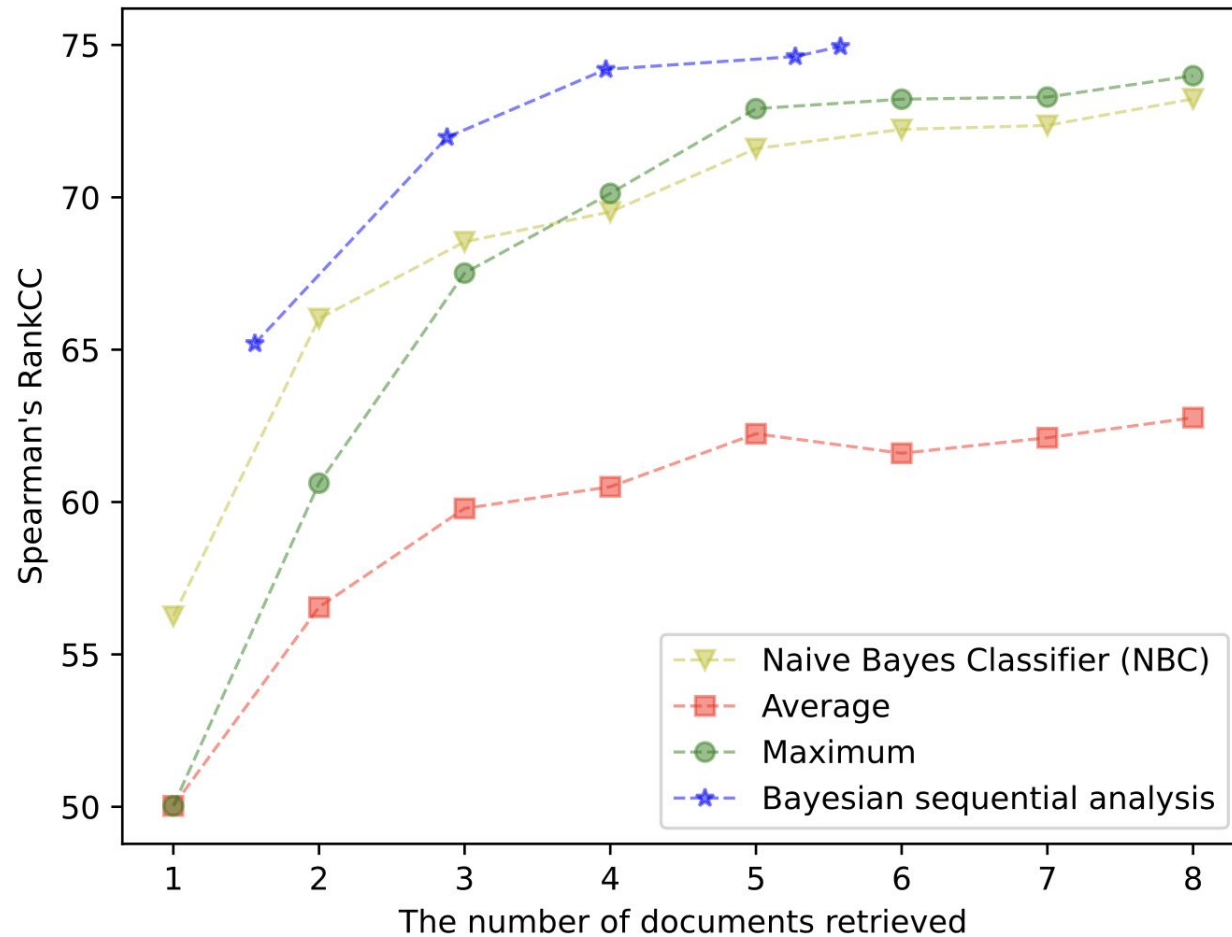
+6.43% Spearman improvement over baseline

Efficiency

SelfCheckGPT requires 20 samples,

while our framework only needs 3–6 documents to achieve better performance.

Experiments



Fixed strategies (Naive Bayes, Average, Maximum)

Require 5–8 docs for good performance

Bayesian Sequential Analysis

Outperforms others with ~3 docs

Keeps leading performance as docs increase



Experiments

Method	Sentence-level (AUC-PR)		
	Nonfact	Factual	Acc
w/o Decomposition	80.04	53.71	79.19
w Decomposition	82.42	57.01	80.24

Complex sentences = multiple facts

Subclaims → easier verification

Results

w/o decomposition → Factual AUC-PR = 53.71

w/ decomposition → Factual AUC-PR = 57.01

Overall accuracy improved

Claim decomposition is a critical step for higher accuracy



Conclusion and Limitations

Conclusion

Proposed a Bayesian sequential framework for hallucination detection.
Dynamically decides when to stop retrieval → fewer docs, higher efficiency.
Outperforms SelfCheckGPT at both sentence and passage levels.

Limitations

Relies on retrieval quality and manual cost settings (C_M , C_{FA} , $C_{retrieve}$).
Entailment scoring can be computationally expensive.



Improving Factuality and Reasoning in Language Models through Multiagent Debate

<https://arxiv.org/pdf/2305.14325>





Table of Contents

1	Overview	6	Limitations and Discussion
2	Introduction	7	Q&A
3	LangGen through Debate		
4	Experimentation		
5	Related Work		



Overview

LLMs still have room to improve through better prompting techniques, ranging from verification, self-consistency, or intermediate scratchpads.

In this paper, we are presented with another approach to improve responses where multiple LLM instances propose and debate their individual responses and reasoning processes over multiple rounds to arrive at a common final answer.

It will be shown through this paper that this approach significantly enhances mathematical and strategic reasoning across a number of tasks.



Introduction

Current methods to improve factual accuracy and reasoning in LMs are applied over a single model instance.

Multi-Agent Debate takes a different approach

Given a query, multiple LMs first generate individual candidate answers to a query. Then each model reads and critiques the responses of all other models and updates its own answer. This step is repeated over several rounds until a consensus is reached.

Introduction



In Figure 1, it can be seen that this method improved accuracy across six reasoning, factuality, and question-answering tasks.

After debating and examining the responses of other models, it was found that the population almost always converges on a single and more accurate common answer.

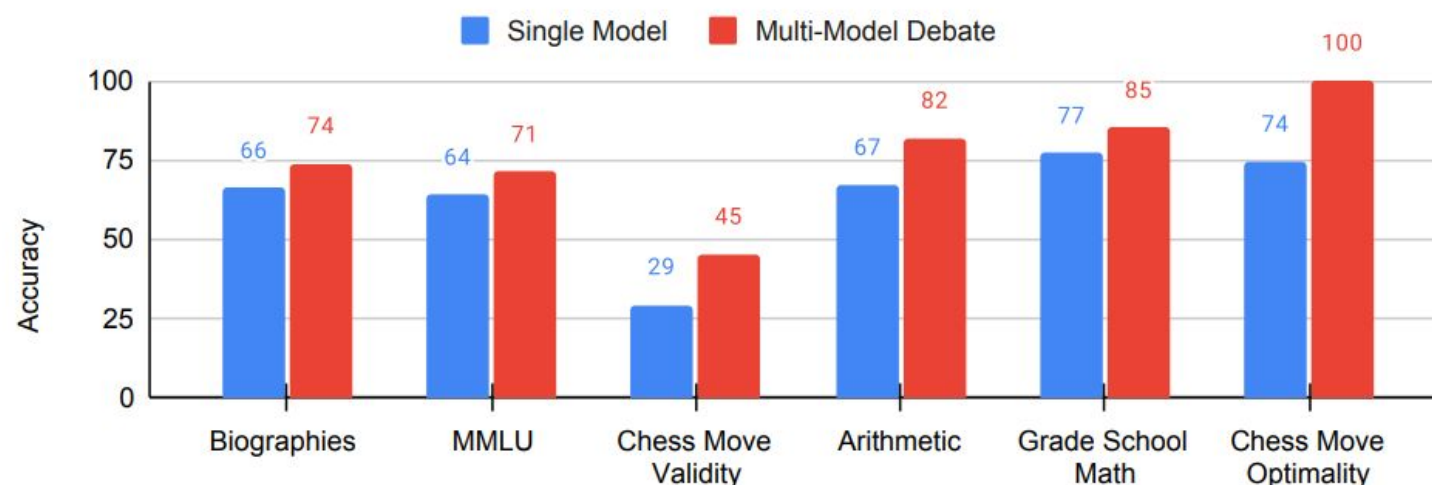


Figure 1: **Multiagent Debate Improves Reasoning and Factual Accuracy.** Accuracy of traditional inference and our multi-agent debate over six benchmarks (chess move optimality reported as a normalized score)



Language Generation through Multiagent Debate

Problem

Single LLMs often hallucinate or make reasoning mistakes.

They cannot verify or revise their own answers.

Three Steps

1. Independent Generation: Each agent writes its own answer.
2. Consensus Prompt: Give all answers back to each agent → “Read others and revise yours.”
3. Iterate 2–4 Rounds: Until answers converge or time limit.

Prompt Control

Short prompt → easily persuaded, quick agreement.

Long prompt → stubborn, longer debate but better accuracy.

Empirical Finding

No theoretical guarantee, but debate usually converges.

RLHF models are naturally “agreeable.”

	Question: What is the result of $10+20*23+3-11*18$?		Question: What is the result of $3+7*9+19-21*18$?		
Round 1	Agent 1: 269 ❌	Agent 2: 369 ❌	Agent 1: 378 ❌	Agent 2: -351 ❌	Agent 3: -357 ❌
Round 2	Agent 1: 275 ✅	Agent 2: 275 ✅	Agent 1: -293 ✅	Agent 2: -293 ✅	Agent 3: 19 ❌
	Question: What is the result of $4+23*6+24-24*12$?		Question: What is the result of $8+14*15+20-3*26$?		
Round 1	Agent 1: -244 ❌	Agent 2: -146 ❌	Agent 1: 236 ❌	Agent 2: -214 ❌	Agent 3: 210 ❌
Round 2	Agent 1: -146 ❌	Agent 2: -122 ✅	Agent 1: 160 ✅	Agent 2: 160 ✅	Agent 3: 160 ✅
Round 3	Agent 1: -122 ✅	Agent 2: -122 ✅	Agent 1: 160 ✅	Agent 2: 160 ✅	Agent 3: 160 ✅

Experiments 3.1



Model	Arithmetic (%) \uparrow	Grade School Math (%) \uparrow	Chess (Δ PS) \uparrow
Single Agent	67.0 ± 4.7	77.0 ± 4.2	91.4 ± 10.6
Single Agent (Reflection)	72.1 ± 4.5	75.0 ± 4.3	102.1 ± 11.9
Multi-Agent (Majority)	69.0 ± 4.6	81.0 ± 3.9	102.2 ± 6.2
Multi-Agent (Debate)	81.8 ± 2.3	85.0 ± 3.5	122.9 ± 7.6



Experimentation 3.2

Next we evaluate the extent to which this methodology improves factuality in LMs.

Tasks: Evaluate factuality in three different settings

- Biographies: Generate accurate historical biographies of people.
- MMLU: Factual Knowledge Questions
- Chess Move Validity: Given set of next moves, make a valid next move of a piece.

Baselines

Uses same baselines as before

Results

Debate gives the best performance in all three settings.

Model	Biographies	MMLU	Chess Move Validity
Single Agent	66.0 \pm 2.2	63.9 \pm 4.8	29.3 \pm 2.6
Single Agent (Reflection)	68.3 \pm 2.9	57.7 \pm 5.0	38.8 \pm 2.9
Multi-Agent (Debate)	73.8 \pm 2.3	71.1 \pm 4.6	45.2 \pm 2.9

Table 2: **Multiagent Debate Improves Factual Accuracy** Multi-agent debate improves the factual accuracy.



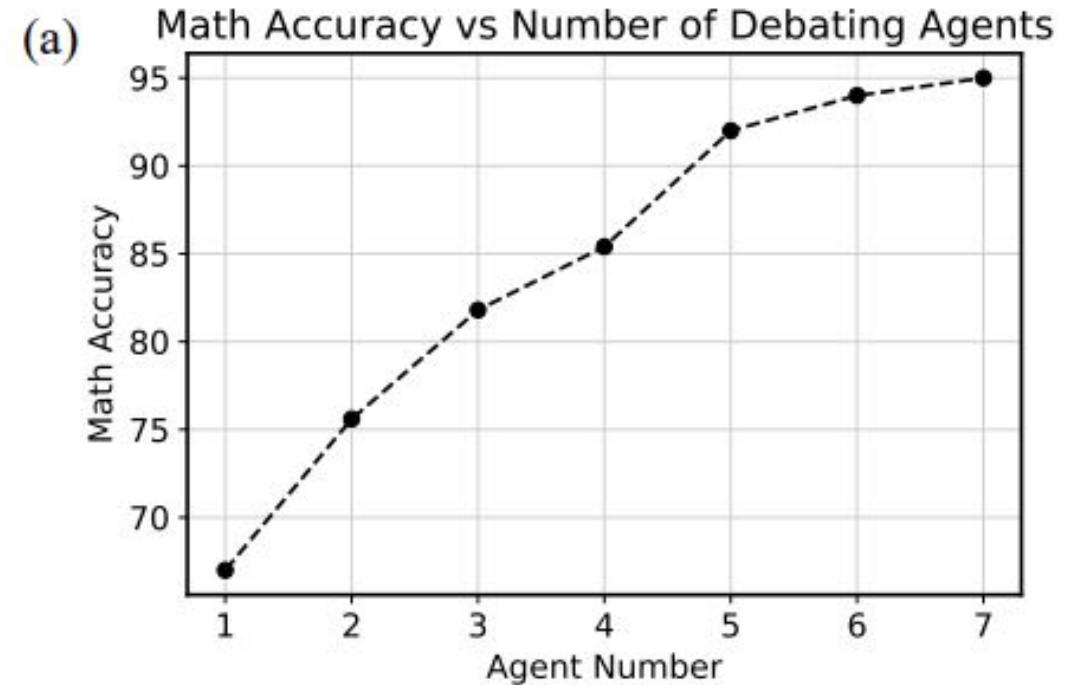
3.3: Analysis: Understanding Multiagent Debate

- We will examine the **impact** of the following **attributes** on **reasoning accuracy**:
 - Number of Agents
 - Rounds of Debate
 - Summarization
 - Prompt Diversity
- Multi-Agent discussion acts as a **self-correction mechanism**; accuracy improves as diversity and depth increases.



More Agents -> Higher Accuracy

- Performance on arithmetic tasks **increases monotonically** as number of agents increase.
- Diverse viewpoints allow agents to challenge and refine each other's reasoning.
- Outputs are summarized to prevent context-length overflow when **agents > 4**.

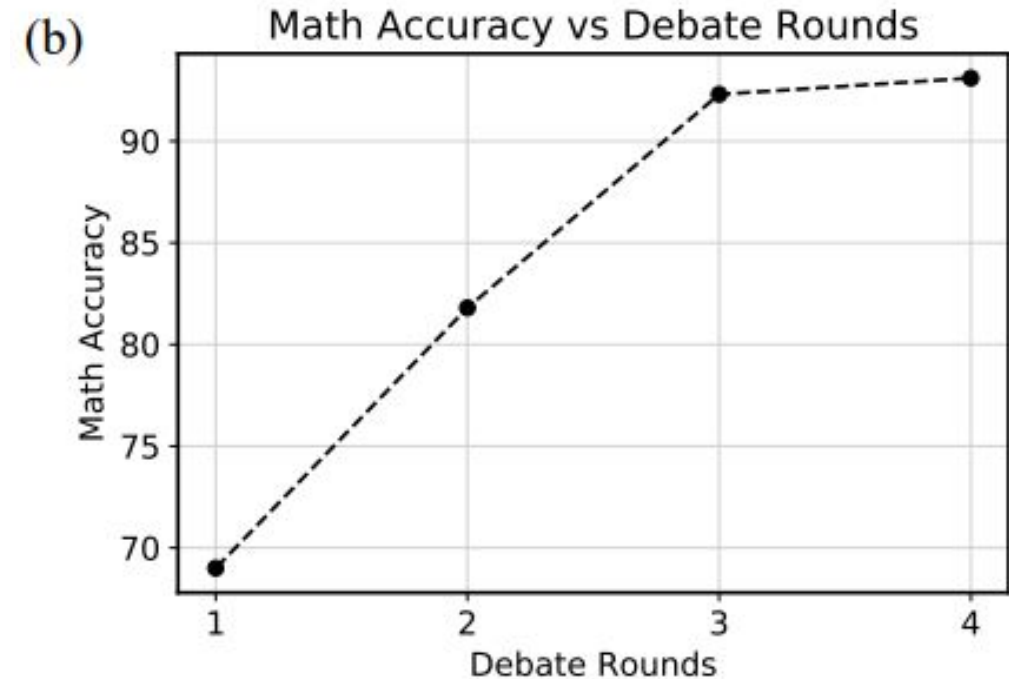


Accuracy improves with additional agents



Longer Debates Enhance Reasoning

- With **3 agents fixed**, increasing debate rounds -> steady performance gain.
- **Plateau after 4 rounds**; diminishing returns beyond this point.
- Implies trade-off between computational cost and accuracy at 4 rounds.



Improvement stabilizes
around round 4



Different LMs Used in a Debate

- Tested ChatGPT vs Bard on 20 GSM8K Math Problems.
- Individual accuracy:
 - Bard: 11/20
 - ChatGPT: 14/20
- Joint Debate: 17/20 -> collective improvement.

Question: Carla is downloading a 200 GB file. Normally she can download 2 GB/minute, but 40% of the way through the download, Windows forces a restart to install updates, which takes 20 minutes. Then Carla has to restart the download from the beginning. How long does it take to download the file?

Round 1

ChatGPT: Before the restart, Carla was able to download 40% of the file, which is $0.4 * 200 \text{ GB} = 80 \text{ GB}$. At a rate of 2 GB/minute, it would take her: $\text{time} = 80 \text{ GB} \div 2 \text{ GB/minute} = 40 \text{ minutes}$.

Because of the restart, Carla had to start over. This means she had to download the entire 200 GB file. The time it takes to download the remaining 120 GB is: $\text{time} = 120 \text{ GB} \div 2 \text{ GB/minute} = 60 \text{ minutes}$.

But Carla also had to wait 20 minutes for the Windows update to install, so the total time it took to download the file is: $\text{total time} = 40 \text{ minutes} + 60 \text{ minutes} + 20 \text{ minutes} = 120 \text{ minutes}$. ❌

Bard: Carla is downloading a 200 GB file at a speed of 2 GB/minute. This means that it will take her $200 / 2 = 100 \text{ minutes}$ to download the file.

However, Windows forces a restart to install updates, which takes 20 minutes. This means that Carla will have to restart the download from the beginning. Since Carla has already downloaded 40% of the file, she only needs to download $200 - 80 = 120 \text{ GB}$. At a speed of 2 GB/minute, it will take Carla $120 / 2 = 60 \text{ minutes}$ to download the file.

Adding the time for the Windows restart, it will take Carla $80 + 20 = 100 \text{ minutes}$ to download the file. ❌

Round 2

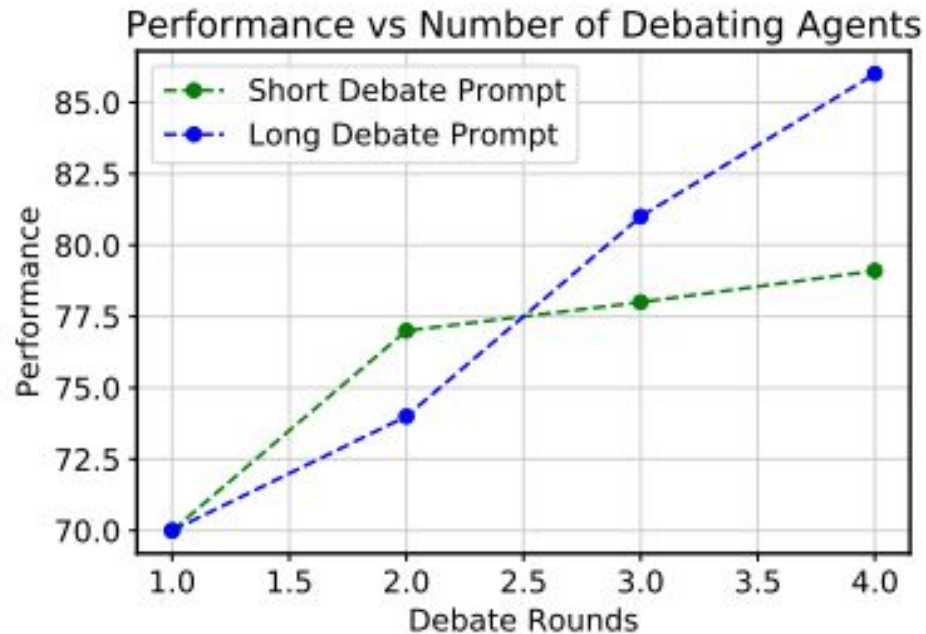
ChatGPT : Based on the calculations provided by the other agent, the correct answer is: Before the restart, Carla was able to download 40% of the file, which is $0.4 * 200 \text{ GB} = 80 \text{ GB}$. At a rate of 2 GB/minute, it would take her: $\text{time} = 80 \text{ GB} \div 2 \text{ GB/minute} = 40 \text{ minutes}$

However, because of the restart, Carla had to start over. This means she had to download the entire 200 GB file. The time it takes to download the entire 200 GB file is: $\text{time} = 200 \text{ GB} \div 2 \text{ GB/minute} = 100 \text{ minutes}$

And since Carla had to wait 20 minutes for the Windows update to install, the total time it took to download the file is: $\text{total time} = 40 \text{ minutes} + 100 \text{ minutes} + 20 \text{ minutes} = 160 \text{ minutes}$. ✅



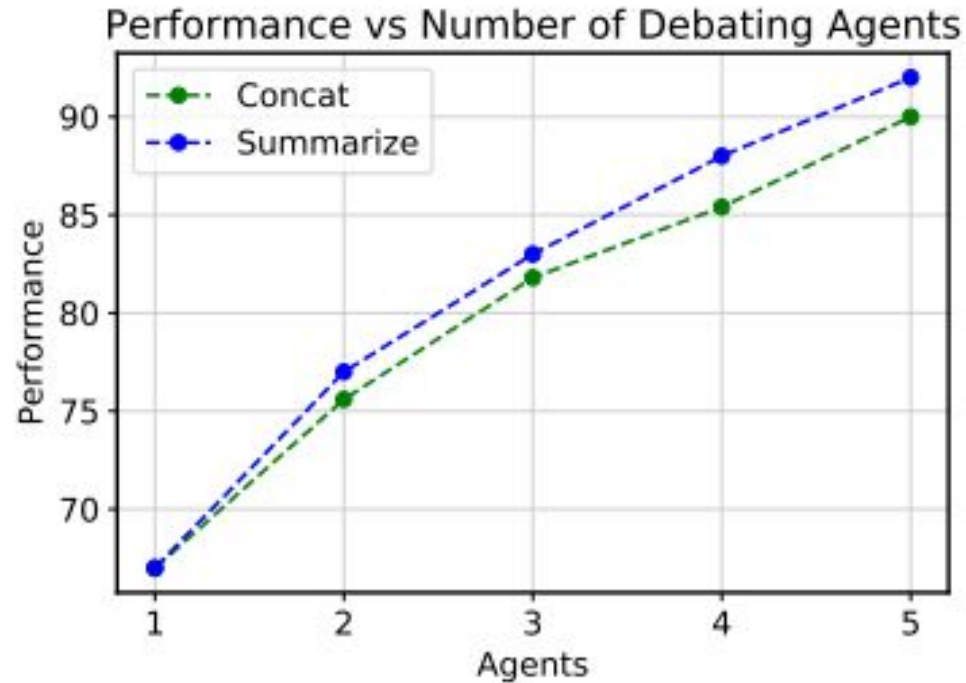
Longer Prompts -> Slower but More Accurate O/P



Longer Debates yield better Consensus

- Debate Convergence depends on Prompt Design:
 - **Short prompts:** Faster agreement; less accurate.
 - **Long prompts:** Slower debate; higher final accuracy.
- Reflects a tradeoff between speed and depth of reasoning

Summarizing Agent Responses Boosts Performance



Concat vs Summarize Performance

- Directly concatenating all responses causes context **overflow at large agent counts**.
- Summarizing other agents' answer into a compact context:
 - Reduces token usage.
 - Improves accuracy when 5 or more agents debate.
- Enable scalability of multi-agent frameworks.



Role Diverse Agents Enhance Debate Outcome

- Agents given **distinct roles**, e.g. Professor, Doctor, Mathematician.
- Produces complementary reasoning styles.
- On MMLU benchmark: **accuracy improved** 71.1% -> 74.2%.
- Highlights importance of diverse **cognitive perspectives** in collaborative LLMs.



Key Findings for Multi-Agent Debate

- More Agents -> Better Factual Accuracy.
- Up to 4 debate rounds -> Optimal Improvement.
- Summarization keeps context manageable and improves accuracy.
- Cross Model and Role Diverse agents yield stronger consensus.
- Longer prompts trade speed for deeper reasoning.



Related Work - Reasoning and Factuality in LMs

- Prior Approaches:
 - Reasoning:
 - Prompting (scratchpads, verification, chain of thought, self-reflection)
 - Fine tuning for step-by-step logical reasoning
 - Factuality:
 - RLHF and truthful dataset curation
 - External knowledge retrieval
- Paper's Approach:
 - Introduces multi-agent debate - multiple LMs discuss and challenge each other to reach better reasoning and factual conclusions.
 - Unlike majority voting or proof based verification (Irving et al. 2018) this approach lets models interactively refine their arguments in real time.



Limitations and Discussion

- **Strengths:**
 - Simple, black-box method applicable across reasoning tasks.
 - Encourages more accurate and verifiable answers.
- **Limitations:**
 - **Computational Cost:** Requires multiple LM generations and a debate mechanism.
 - **Context Handling:** Long debates cause attention limits; models focus on recent turns.
 - **Overconfidence:** Even incorrect final answers are often asserted confidently.



Q&A

Thank you !

