Complex & Intelligent Systems (2025) 11:231
https://doi.org/10.1007/s40747-025-01833-9

ORIGINAL ARTICLE

Reducing hallucinations of large language models via hierarchical semantic piece

Yanyi Liu[1] · Qingwen Yang[1] · Jiawei Tang[1] · Tiezheng Guo[1] · Chen Wang[1,2] · Pan Li[1,2] · Sai Xu[1,2] · Xianlin Gao[2] · Zhi Li[2] · Jun Liu[1] · Yingyou Wen[1,2]

Received: 8 August 2024 / Accepted: 26 February 2025 / Published online: 2 April 2025

Abstract

With the widespread application of large language models (LLMs) in natural language processing (NLP), hallucinations have become a significant impediment to their effective use of LLMs in industry applications. To address this challenge, we integrate existing hallucination detection and mitigation methods into a unified hallucination detection and mitigation framework. The framework consists of four main components: output parser, reference parser, fact verifier, and mitigator. These components collectively consolidate various hallucination detection and mitigation methods. Within this unified framework, we introduce the hierarchical semantic piece (HSP) for hallucination detection and mitigation. The HSP method extracts multi-granularity semantic pieces from both the reference material and the generated text. Sentence-level semantic pieces encapsulate global semantic information, while entity-level semantic pieces handle local semantic information. This method verifies the consistency between the generated text and the reference text at corresponding granularities, thereby enhancing the effectiveness of hallucination detection and mitigation. Experimental results show that the HSP method is very effective in detecting and mitigating hallucinations and shows lower computational resource consumption. Our method has great potential and promises for industry applications that rely on professionalism and reliability.

Keywords Large language models · Hallucination detection · Hallucination mitigation · Industry applications

Introduction

In recent years, large language models (LLMs) have demonstrated exceptional performance in natural language generation tasks, becoming a central focus in AI and NLP research [1, 2]. They are also regarded as a crucial step toward achieving artificial general intelligence (AGI) [3, 4]. With the advent of applications such as ChatGPT and GitHub Copilot, LLMs have exhibited significant potential across various scenarios [5].

In practice, LLMs often generate hallucinations in their outputs, where the generated text is fluent but disconnected from the objective facts or the context [6]. This phenomenon arises due to the probabilistic nature of the models and is

✉ Yingyou Wen
wenyingyou@mail.neu.edu.cn

Yanyi Liu
2290175@stu.neu.edu.cn

Qingwen Yang
yang_qw@neusoft.com

Jiawei Tang
2301944@stu.neu.edu.cn

Tiezheng Guo
2310725@stu.neu.edu.cn

Chen Wang
wangchen-neu@neusoft.com

Pan Li
li_pan@neusoft.com

Sai Xu
xu_s@neusoft.com

Xianlin Gao
gaoxianlin@neusoft.com

Zhi Li
zhi-li@neusoft.com

Jun Liu
liujun@cse.neu.edu.cn

1 School of Computer Science and Engineering, Northeastern University, Shenyang 110819, China

2 Neusoft AI Magic Technology Research, Shenyang 110179, China

challenging to eliminate entirely. However, it can be mitigated through various strategies, which is crucial for the application of LLMs in industry application [6, 7].

With the development of model capabilities, the frequency of hallucinations has decreased in proprietary models such as GPT-4 [8], Claude-3, and Gemini [9]. However, hallucinations still exist. To address hallucinations, researchers have proposed various strategies that encompass the entire training and inference loop [10], including pretraining [11], faithful fine-tuning [12, 13], reinforcement learning [8, 14], alignment decoding strategies [15], and post-processing [16, 17].

In practice, knowledge-enhanced methods have become the standard approach, given that the internal knowledge of the model may be outdated [18, 19]. Prominent examples of this approach include the RAG system [20]. By integrating high-quality reference data into the model input and providing supplementary information, it is possible to enhance the model's precision and reliability.

In industry applications, such as legal [21], medical [22], and financial [23] scenarios, data security and privacy are significant concerns [24, 25]. Consequently, open-source models that can be readily audited and modified are more frequently employed. These models, including Llama [26], Mistral [27], and Qwen [28], have comparatively limited capabilities and are more prone to generate hallucinations. Given the lower tolerance for hallucinations in industry applications, developing effective hallucination mitigation strategies remains crucial.

The core contributions of this paper are as follows:

- A unified hallucination detection and mitigation framework is proposed, comprising four components: output semantic decomposition, reference information retrieval and decomposition, fact verification, and hallucination mitigation. This framework unifies existing hallucination detection and mitigation methods.
- A novel approach, hierarchical semantic piece (HSP), is introduced for detecting and mitigating hallucinations. HSP extracts sentence and entity-level semantic pieces from both the reference text and the output text, verifies consistency at different levels, and enhances the effectiveness of hallucination detection and mitigation.
- The HSP method's effectiveness was validated through experiments. The results demonstrate that the HSP method is an effective approach for detecting and mitigating hallucinations. Experiments on domain-specific scenarios show that our method has great potential for industry applications.

The unified framework and the HSP method are general, modular, and extensible. By leveraging existing large language models, they can be seamlessly integrated into existing LLM applications, thereby enhancing their professionalism and reliability.

Related work

Large language models in natural language processing

With the rapid development of artificial intelligence technology, large language models (LLMs) have become increasingly popular in the field of natural language processing. LLMs have significantly advanced natural language processing technology, enabling computers to better understand and generate natural language, and providing powerful support for the development of artificial intelligence technology.

Most large language models (LLMs) are currently based on the transformer decoder architecture, such as GPT and Llama. These models are pretrained using self-supervised learning on large amounts of text data [1]. In comparison to models in encoder-only architecture like BERT [29], current LLMs mainly employ decoder architecture, which is more efficient for generative tasks. Moreover, LLMs benefit from their larger model size and more extensive training data, giving them more versatile language generation capabilities.

Prompt engineering is an essential tool in current LLM applications [30], as it involves incorporating context, instructions, and few-shot examples into the model input. This capability, known as in-context learning, allows the model to adapt to new tasks without additional fine-tuning. This has changed the way natural language processing is researched and applied, with LLMs becoming more widely used in practical scenarios.

Advanced prompt-based applications, such as RAG [31, 32] and Agent [33, 34], have emerged as a new paradigm for industry application development. By incorporating professional knowledge into the model context, RAG enhances the professionalism of large language models. Agents achieve more flexible and versatile industry application development by invoking external tools and workflow orchestration.

System engineering of large language models

Although LLMs provide powerful language generation capabilities, they also face a number of challenges, particularly in terms of the reliability and accuracy of the generated content [19]. Wang et al. proposed the concept of large language model system engineering (LLM-SE) [35] to address the problems faced by LLMs in industry applications through a systematic approach. The feature requirements of industry applications based on LLMs include professionalism, value consistency, interpretability, reliability, determinism, flexibility, stability, low latency, etc.

Two core components of LLM-SE are quality engineering and interpretation engineering. Quality engineering ensures that the content generated by LLMs is professional, consistent, and reliable. This is achieved by reviewing, monitoring, controlling, correcting, and supplementing the content to meet these quality standards. Interpretation engineering aims to enhance the understandability and acceptance of the decision-making process and the results generated by LLMs, which is essential for their successful application among professionals and end users.

Given the minimal tolerance for hallucinations in industry applications, for example in legal, medical, and financial scenarios, our research focuses on addressing this issue to enhance professionalism and reliability. From a quality engineering perspective, we aim to detect and mitigate hallucinations in large language models (LLMs) to improve their performance and reliability in industry applications.

Post-processing hallucination detection and mitigation

To mitigate hallucinations in large language models, researchers have proposed various strategies at different stages of model construction and application [7]. Post-processing methods are of particular interest, as they do not require changes to the model's parameters and can be easily applied without the need for retraining, making them more versatile.

There are two main categories of post-processing methods for reducing hallucinations.

1. Reference-free methods, which rectify hallucinations through self-reflection [36, 37], Chain of Thought (CoT) [38, 39], or by refining the decoding algorithm [40], are designed to enable models to self-evaluate and correct their outputs. However, their effectiveness is limited by the model's knowledge cutoff and the difficulty of accurate content attribution [41], making them more suitable for general text generation tasks rather than domain-specific applications.
2. Reference-based techniques, which use external knowledge or materials as reference points, detect and mitigate hallucinations by comparing the model output with credible sources. Methods such as FactScore [42] and SAFE [43] take a fact-based approach by decomposing the output text into atomic facts, which are then checked against reputable knowledge repositories such as Wikipedia. The RARR method generates verification questions based on the output text, seeks answers through web searches, and assesses the veracity of the model's output [44]. CoNLI develops a Natural Language Inference chain for the fact-checking process, which is used to validate the entities and statements within the output text [45].

Existing methods have made significant progress in hallucination detection and mitigation. In our work, we unify these methods into a modular hallucination detection and mitigation framework specifically designed for reference-based generation scenarios. Our method, hierarchical semantic piece (HSP), as a post-processing approach, enhances the effectiveness of hallucination detection and mitigation by extracting and verifying multi-level semantic pieces, thereby leveraging the reference text to ensure higher accuracy and consistency.

Methodology

Consider the knowledge-enhanced methods represented by the RAG System, the generation process can be defined as follows:

$$LLM(Ref, Query) \rightarrow Output \tag{1}$$

Here's an example of the generation process:

Generation Process Example

Reference:
Dutch programmer Guido van Rossum developed Python in 1991 after expressing frustration with the limitations of the programming language ABC. Python, which he named after the British television series Monty Pythonś Flying Circus, was publicly released in 1994.
Query:
When was Python released?
LLM Output:
Python was created by Guido van Rossum and was first released in 1994.

Our goal is to detect and mitigate hallucinations in large language models (LLMs) $Output$ in the reference-based scenario. The task is defined as follows:

Given a text generated by an LLM, $Output$, and a reference text, Ref, our task is to detect hallucinations in $Output$ and mitigate them to make the output more accurate and reliable.

$$\mathcal{F} : (Ref, Output) \rightarrow Output' \tag{2}$$

Here's a detailed example for task definition:

Hallucination Mitigation Example

Reference:
Dutch programmer Guido van Rossum developed Python in 1991 after expressing frustration with the limitations of the programming language ABC. Python, which he named after the British television series Monty Pythonś Flying Circus, was publicly released in 1994.
LLM Output:
Python was created by Guido van Rossum and was first released in 1994.
Detection:
Hallucination Detected.
Mitigation Result:
Python was created by Guido van Rossum and was first released in 1991.

Based on the task definition, we first review existing hallucination detection and mitigation methods, then propose a unified hallucination detection and mitigation framework. The framework consists of four key components: output semantic decomposition, reference information retrieval and decomposition, fact verification, and hallucination mitigation. Based on this framework, we propose a new method, hierarchical semantic piece (HSP), for hallucination detection and mitigation.

Unified hallucination detection and mitigation framework

The unified framework for hallucination detection and mitigation is inspired by various methods in the reference-based scenario. It consists of four key components: output semantic decomposition, reference information retrieval and decomposition, fact verification, and hallucination mitigation. As illustrated in Fig. 1, these four components collectively form a comprehensive hallucination detection and mitigation framework.

1. Output parser: Extracts semantic information from the text for subsequent verification.
2. Reference parser: Retrieves and extracts reference information for comparison with the output semantic information.
3. Fact verifier: Validates the output semantic information against the reference data.
4. Mitigation: Mitigates hallucinations in the output text based on the results of the fact verifier.

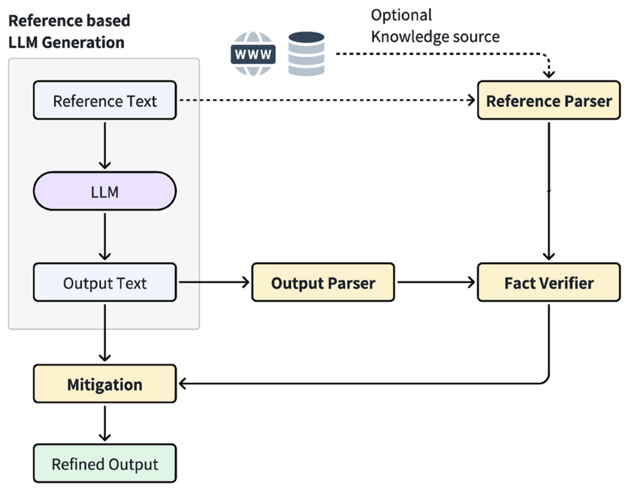

Fig. 1 Unified hallucination detection and mitigation framework

The entire framework is interdependent, with the output of each level directly influencing the final detection and mitigation results.

Existing methods of hallucination detection and mitigation can be integrated into the unified framework. Table 1 illustrates the correspondence between various hallucination detection and mitigation methods and the specific components of the framework. This allows for a clearer understanding and comparison of the differences among these methods.

In the mitigation part, nearly all methods rely on language model-based rewriting. However, the practical implementation of these methods varies differs due to variances in the outputs of preceding stages.

Based on the unified framework, we propose the hierarchical semantic piece (HSP) for hallucination detection and mitigation. Compared with existing methods, the HSP method introduces a hierarchical semantic extraction process on both the output and reference texts, enhancing the effectiveness of hallucination detection and mitigation. The following sections provide a comprehensive overview of the HSP method.

Hierarchical semantic piece (HSP)

We propose the hierarchical semantic piece (HSP) for hallucination detection and mitigation. The overall process of the HSP method is illustrated in Fig. 2.

Our method is centered on the extraction of semantic information and the consistency of semantic granularity. In HSP, both the output semantic parser and the reference semantic parser utilize the same module, the hierarchical semantic piece extractor. This extractor decomposes the semantic information in the text into two different levels: sentence-level and entity-level. In the fact verifier, we introduce an

Table 1 Existing hallucination detection and mitigation methods in the unified framework

Method	Output parser	Reference source	Reference parser	Fact verifier	Mitigation
Factscore [42]	Atomic facts extraction	Wikipedia	LM retrieval	Binary classification	None
RARR [43]	Generate questions	Google search	Text split and rearrange	Binary classification	LM Rewrite
CoNLI [45]	Sentence split entity extraction	Source texts	–	CoT based NLI	LM rewrite
SAFE [43]	Atomic facts extraction with refine	Google search	–	Iterative retrieval triple classification	LM Rewrite
HSP	Hierarchical semantic extractor	Source texts	Hierarchical semantic extractor	Triple classification	LM rewrite

evidence matching method based on embedding similarity, followed by fact verification based on LLM for the premise-hypothesis pairs. Finally, the hallucination mitigator applies the verification results to the original text, generating the refined output.

The hierarchical semantic piece (HSP) is summarized in Algorithm 1. The inputs to the algorithm are the reference text Ref and the output text $Output$ from LLM, while the outputs are the verification results V and the mitigated output $Output'$. In the algorithm, SP_r and SP_o represent the hierarchical semantic pieces extracted from the reference and output texts, respectively. For each output semantic piece sp_{oi}, the evidence E_i is obtained by matching the top-k reference semantic pieces that have the highest similarity to the output semantic piece. The fact verification results V_i are then derived based on the output semantic piece and the corresponding evidence. If any contradictions are found in the verification results, the hallucination mitigator is employed to generate the mitigated output $Output'$; otherwise, the original output is retained.

Algorithm 1 Hierarchical semantic piece (HSP) method

```
1: Input: Reference Text $Ref$, Output Text $Output$
2: Output: Verification Results $V$, Mitigated Output $Output'$
3: $SP_r \leftarrow$ HierarchicalSemanticExtractor($Ref$)
4: $SP_o \leftarrow$ HierarchicalSemanticExtractor($Output$)
5: for $sp_{oi} \in SP_o$ do
6:     $E_i \leftarrow$ EvidenceMatching($sp_{oi}, SP_r$)
7:     $V_i \leftarrow$ FactVerifier($sp_{oi}, E_i$)    ▷ $V_i \in$ {Contradiction, Neutral, Entailment}
8: end for
9: if HasContradiction($V$) then
10:    $Output' \leftarrow$ HallucinationMitigator($Output, V, E$)
11: else
12:    $Output' \leftarrow Output$
13: end if
14: return ($Output', V$)
```

The following section provides a detailed description of the components constituting the HSP method. A full example of the HSP method and detailed intermediate results are provided in the Appendix B.

Hierarchical semantic extractor

In this module, we decompose the text into sentence-level and entity-level semantic pieces to better evaluate the consistency between the model output and the reference material. The Reference text Ref and the Output text $Output$ are decomposed into sentence-level semantic pieces SP_r and SP_o:

$$Ref \rightarrow SP_r = \{sp_{r1}, sp_{r2}, \ldots, sp_{rn}\} \tag{3}$$

$$Output \rightarrow SP_o = \{sp_{o1}, sp_{o2}, \ldots, sp_{om}\} \tag{4}$$

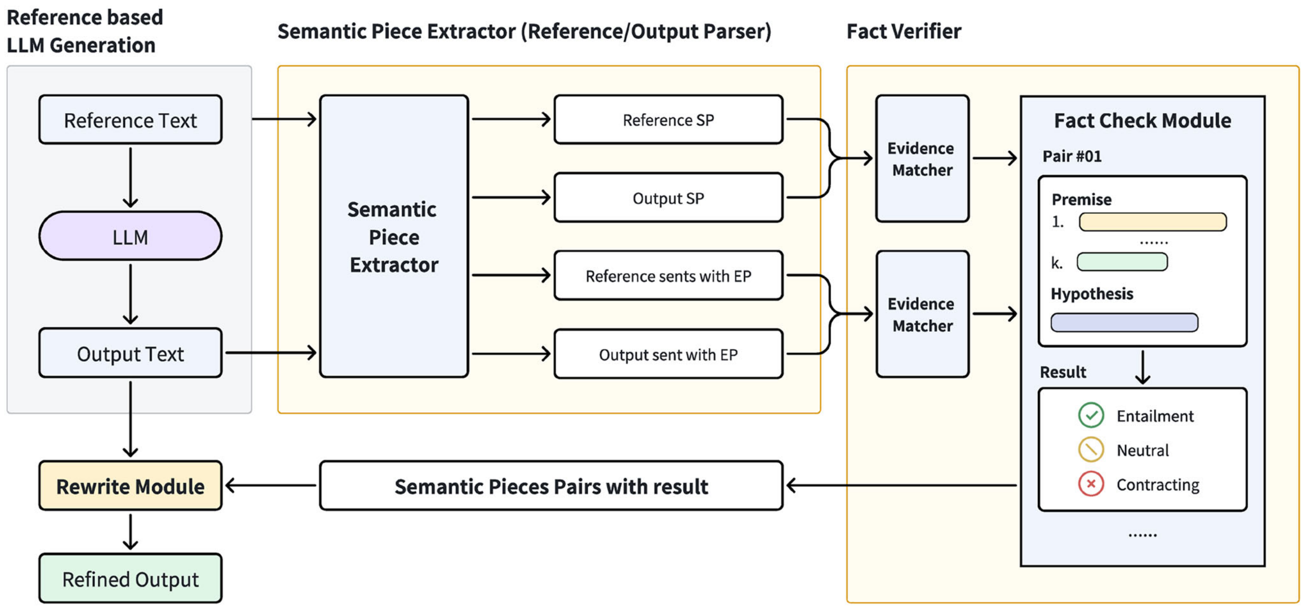

Fig. 2 Hierarchical semantic piece (HSP) overview. The yellow blocks are the key components in our method, which can be mapped to the components in the unified framework

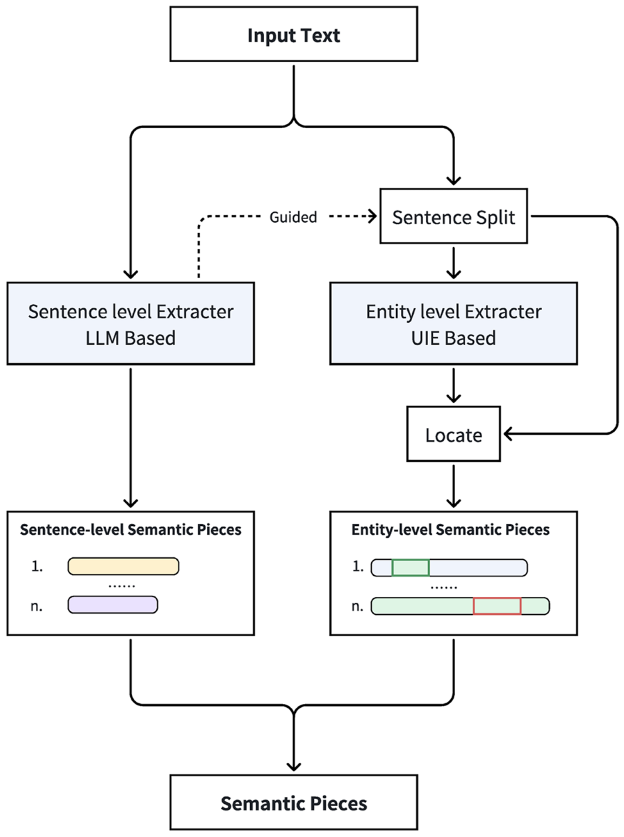
Input Text
Guided
Sentence Split
Sentence level Extracter LLM Based
Entity level Extracter UIE Based
Locate
Sentence-level Semantic Pieces
Entity-level Semantic Pieces
Semantic Pieces

Fig. 3 Hierarchical semantic extractor

The workflow is illustrated in Fig. 3. There are two levels of semantic pieces: sentence-level and entity-level.

Initially, sentence-level semantic units are extracted from the original text, aiming to decompose the content into its fundamental components. Prompt engineering is employed to ensure that each sentence-level semantic piece is an independent “claim” or "fact" that can be understood in isolation without requiring additional context.

In the second step, we extract entities from the original text based on the ERNIE-UIE [46]. The extracted entity information is combined with the corresponding original sentence in order to form an entity-level semantic piece. The entity-level semantic pieces contain structured information, represented in the form of (entity type–value), which is used for auxiliary judgment.

In the entity extraction module, predefined entity schemas are introduced, including time, number, location, and personal information. This approach narrows the scope of entities, making the extraction more targeted and simplifying the subsequent verification. The full schema is shown in Appendix D.

In practice, we filter the original sentence using sentence-level semantic pieces to remove irrelevant information before entity-level extraction, thereby reducing computational complexity.

Both sentence-level and entity-level extractions form a complementary relationship within the hierarchical extraction process. The sentence-level pieces ensure that the general conveyed ideas and claims align, which is crucial for verifying the overall consistency. Meanwhile, the entity-level pieces ensure that specific details are accurate, thus

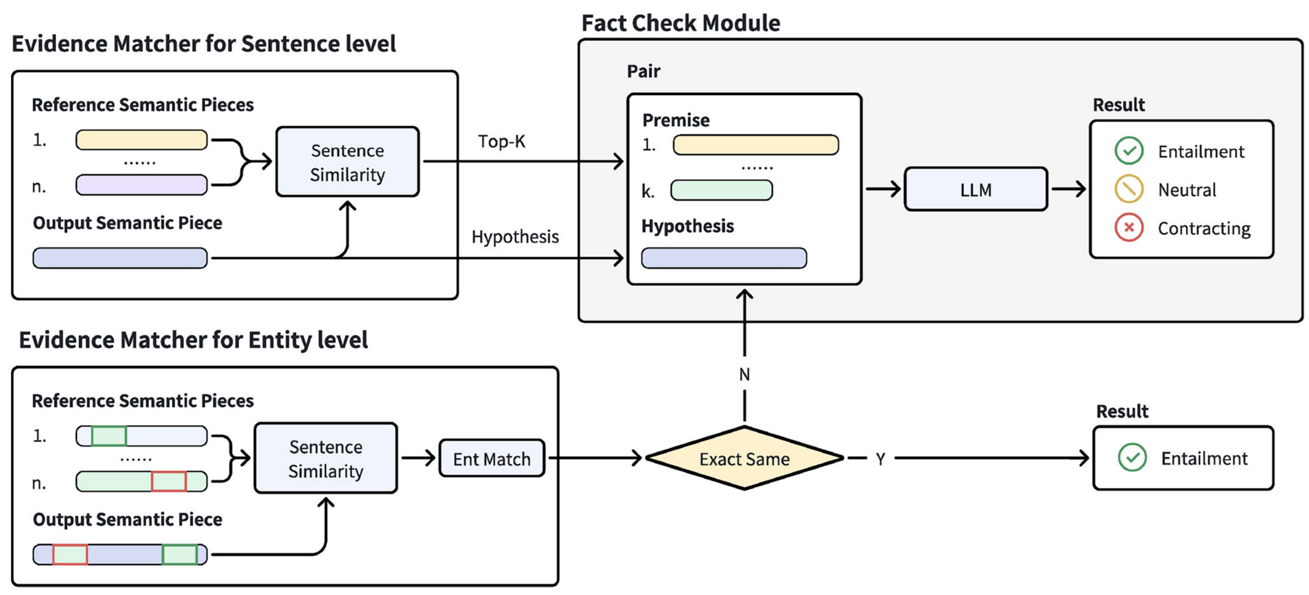

Fig. 4 Fact verifier in the HSP method

supporting the verification of precise information within the broader context.

Fact verifier for semantic piece

With the hierarchical semantic decomposition, our method applies natural language inference (NLI) to verify the consistency between the output semantic pieces and the reference semantic pieces. The workflow is illustrated in Fig. 4.

Natural language inference (NLI) task is to determine the relationship between a premise and a hypothesis, which is a common task in natural language processing. In our method, the reference sentence pieces SP_r are considered the premise set, while the output sentence pieces SP_o are considered the hypothesis set in the fact verification module. The goal is to determine the relationship between the output semantic pieces and the reference semantic pieces. Available relationships include entailment, neutral, and contradiction.

1. Entailment: The output semantic piece matches the reference semantic piece.
2. Neutral: The output semantic piece is unrelated to the reference semantic piece.
3. Contradiction: The output semantic piece contradicts the reference semantic piece.

In the fact verification module, the reference sentence pieces SP_r are considered the premise set, while the output sentence pieces SP_o are considered the hypothesis set. Considering the computational cost, it is impractical to perform fact verification on the entire combination of the whole sets. Instead, an evidence matching module is introduced to find the top-k reference semantic pieces as the evidence for each output semantic piece.

The mathematical expression for finding evidence for each output sentence piece SP_o is as follows:

$$E_i = \left\{ p_{rj} | j \in \text{Top}_k \left(\text{sim}(sp_{oi}, sp_{rj}) \right) \right\} \tag{5}$$

For each hypothesis, the evidence E_i is obtained by selecting the top k reference semantic pieces with the highest similarity to the output semantic piece sp_{oi}. The similarity function $sim(\cdot)$ calculates the similarity between two semantic pieces, specifically embedding cosine similarity is used in this case. To maintain the granularity of the semantic pieces, we keep top-3 semantic pieces as the evidence. For entity-level semantic pieces, we add an additional entity matching step; if the entity type and value are exactly the same, the evidence is considered as entailment.

After obtaining the evidence semantic pieces, they are input into the natural language inference (NLI) module for fact verification.

$$V_i = \text{NLI}(sp_{oi}, E_i) \quad \forall sp_{oi} \in SP_o \tag{6}$$

Finally, we obtain the verification results $V : \{V_0, \ldots, V_m\} \in \{\text{Entailment, Neutral, Contradiction}\}$, which are used to guide the hallucination mitigation process.

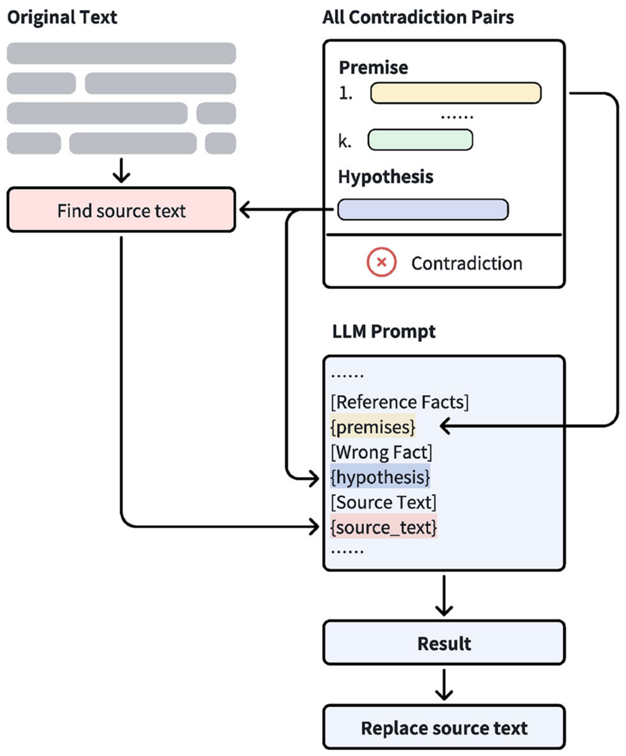

Fig. 5 Hallucination mitigator

Hallucination mitigator

The hallucination mitigator is designed to correct hallucinations detected in fact verification. The hallucination mitigator generates repair sentences for each hallucination semantic piece based on the verification results and its original text. The workflow is illustrated in Fig. 5.

For each hallucination semantic piece, the workflow is as follows:

1. Find the original sentence: To locate the relevant original sentence, the most similar sentence to the hallucination semantic piece is selected from the original text, based on the cosine similarity of the sentence embeddings.
2. Prompt LLM To Fix: We utilize the ability of LLM to fix hallucinations by providing information that includes the original sentence, the hallucination semantic piece, and the reference semantic piece. The LLM generates a repair sentence based on the input.
3. Result integration: The repair sentence is integrated into the original text to replace the hallucination semantic piece, generating the mitigated output.

If multiple hallucinations are present, the above steps are iteratively applied to each one. The final output is a revised text that has been thoroughly vetted and corrected for factual accuracy.

Here's an example of the hallucination mitigation result, the hallucinated part is marked in red, and the corrected part is marked in green.

Sentence to Fix:
A massive earthquake hit Nepal, causing widespread panic and devastation. Survivors reported seeing wild animals roaming the streets in the aftermath of the disaster.
Ground Truth:
Massive 7.8 magnitude earthquake has struck Nepal near its capital, Kathmandu. As the death toll rises, witnesses describe devastation and panic.
HSP Result:
A massive earthquake hit Nepal, causing widespread panic and devastation. Survivors reported seeing people in a state of shock and confusion in the aftermath of the disaster.
HSP Metrics
BLEU-1:(0.58 $\rightarrow$ 0.5179 $\downarrow$); Rouge-L: (0.463 $\rightarrow$ 0.4554 $\downarrow$); BertScore: (0.7486 $\rightarrow$ 0.7571 $\uparrow$); Align-score: (0.4966 $\rightarrow$ 0.9528 $\uparrow$)

Experiments

To verify the effectiveness of our proposed method, we designed two types of experiments: hallucination detection and hallucination mitigation.

Experiment setup

1. Models

To verify the generality of our proposed method, two types of models were studied. The proprietary models include the latest GPT-4o and GPT-3.5-turbo, while the open-source models include the state-of-the-art Qwen2-72B-Instruct [47] and the cost-effective Qwen1.5-32B-Chat [28].

To ensure the consistency of the results, we used the same generation parameters for all models, with temperature = 0 and top-p = 0.6.

HSP method has some hyperparameters, including the similarity function $sim(\cdot)$ and the top-k for evidence matching. To ensure fairness, we used the same hyperparameters for all models, with the SFR-Embedding-2-R model for embedding similarity and top-k = 3.

2. Evaluation Metrics

For the detection task, we define the hallucination detection task as a binary classification task, with the evaluation metric being the marco precision, recall and F1 score.

For the mitigation task, we define the hallucination mitigation task as a text rewriting task, with the evaluation metrics being Rouge1, Rouge2, RougeL, Bleu-avg, Bleu-4, BertScore [48], Alignscore [49] and FactCC [50].

The cost and latency are key metrics for LLM-based methods. However, the deployment schemes and code parallelism of the models vary, making direct evaluation based on time unfair. Therefore, we propose the Token Expansion Ratio (TER) to evaluate the efficiency of the methods.

$$\text{TER} = \frac{Token_{all}}{Token_{input}} \tag{7}$$

The token efficiency ratio (TER) measures the extra tokens needed by a method that uses a language model (LLM) compared to the number of tokens in the original input text. Specifically, $Token_{all}$ refers to the total number of tokens used by the method, while $Token_{input}$ denotes the number of tokens in the input text. A lower TER indicates higher efficiency. This metric helps to standardize the comparison among different methods by accounting for variations in tokenization rates, thus making the efficiency of various methods more comparable.

Hallucination detection

The hallucination detection task is defined as a binary classification task. We set two baselines, Alignscore [49] and LLM Prompt-based classification.

Datasets

For the hallucination detection task, QAGS [51] and SummEval [52] datasets are used to evaluate the effectiveness on general domain, HaluBench subset [53] is used to evaluate the effectiveness on domain-specific scenario. The datasets distribution is shown in Table 2.

- **QAGS**: The QAGS datasets [51] are built with CNN/Dailymaill (QAGS-CNNDM) and XSUM (QAGS-XSUM). Each sample includes three crowdsourced consistency labels, and only when all three annotators agree that the generated text is consistent with the reference text, we consider the sample to be non-hallucination.
- **SummEval**: The SummEval dataset is built upon the CNN/DailyMail dataset [52]. Each summary is labelled with three human experts on a Likert scale from 1-5 on 4 categories: consistency, coherence, fluency and relevance. We follow the TRUE benchmark [54] in taking the consistency scores and mapping a score of 5 to being non-hallucination, and anything lower to being hallucination.

Table 2 Sample distribution of datasets

Dataset	Total	Halu	Non-Halu
QAGS-CNNDM	235	122	113
QAGS-XSum	239	57	182
SummEval	1600	294	1306
HaluBench-FinanceBench	1000	500	500
HaluBench-CovidQA	1000	500	500
HaluBench-PubMedQA	1000	500	500

- **HaluBench**: The HaluBench dataset is a benchmark dataset for hallucination detection in various domains [53]. We choose FinanceBench, CovidQA and PubMedQA subsets for evaluation, which are more relevant to domain specific scenarios. All samples have been labelled with binary labels, 1 for hallucination and 0 for non-hallucination.

Experimental results

The detection results are shown in Tables 3 and 4. Due to the cost limitation, we only evaluate HaluBench dataset on Qwen1.5-32B-Chat and Qwen2-72B-Instruct models.

The HSP method achieved superior performance concerning the macro F1 score, demonstrating high precision and recall. Compared to CoNLI, our method exhibited similar performance with a slight improvement in the F1 score. Additionally, our method has an advantage in token expansion ratio (TER), showing an average reduction of approximately 30%, which indicates a lower cost in LLM tokens.

Through a series of experiments on various models, we observed a positive correlation between a model's capability and its performance in hallucination detection. The current leading model, GPT-4o, exhibited superior results across multiple methods and datasets. The top-performing open-source model, Qwen2-72B-Instruct, also demonstrated commendable performance, surpassing GPT-3.5-turbo while remaining slightly behind GPT-4o. Additionally, the cost-effective Qwen1.5-32B-Chat model achieved better results than Alignscore, making it more suitable for practical applications (Table 3).

In Table 4, the results on the HaluBench dataset show that LLM-based methods generally outperform Alignscore method on domain-specific datasets. This indicates that LLM-based methods have better generalization performance on domain-specific datasets. Additionally, the HSP method achieved better results, demonstrating its effectiveness on domain-specific datasets.

The baseline method based on LLMs performed best in terms of TER, requiring the fewest additional tokens. However, the overall performance was poor. Compared to

Table 3 Hallucination detection results with different models

Method	Model	QAGS-CNNDM				QAGS-XSUM				SummEval				Average			
		P ↑	R ↑	F1 ↑	TER ↓	P ↑	R ↑	F1 ↑	TER ↓	P ↑	R ↑	F1 ↑	TER ↓	P ↑	R ↑	F1 ↑	TER ↓
Alignscore		72.95	61.38	54.90	–	66.55	70.77	56.77	–	84.54	62.35	65.60	–	74.68	64.83	59.09	–
Prompt	Q-32B	73.98	61.00	53.97	1.42	65.16	64.01	45.15	1.41	**86.19**	60.37	62.98	1.39	75.11	61.79	54.03	1.41
CoNLI	Q-32B	**78.05**	70.39	67.45	7.25	**67.59**	**72.20**	**58.00**	6.50	85.92	66.49	70.67	7.56	**77.19**	69.69	65.37	7.10
HSP	Q-32B	76.00	**74.17**	**73.26**	4.70	65.42	69.84	56.97	3.75	76.59	**76.10**	**76.34**	5.43	72.67	**73.37**	**68.86**	4.63
Prompt	Q-72B	75.70	67.49	63.87	1.41	65.32	64.56	46.01	1.56	**88.27**	61.98	65.25	1.55	76.43	64.68	58.38	1.51
CoNLI	Q-72B	78.03	74.74	73.42	6.99	**67.17**	**72.64**	60.08	6.37	85.87	72.83	76.98	7.56	**77.02**	73.40	70.16	6.97
HSP	Q-72B	**78.22**	**77.03**	**76.42**	4.76	65.07	70.68	**61.48**	3.66	79.15	**77.90**	**78.50**	5.46	74.15	**75.20**	**72.13**	4.63
Prompt	GPT−3.5	69.22	52.84	39.15	1.18	64.47	61.54	41.19	1.40	81.32	54.97	54.58	1.23	71.67	56.45	44.97	1.27
CoNLI	GPT−3.5	**72.66**	**71.17**	70.27	6.46	**67.97**	**73.02**	59.15	6.55	**82.26**	**67.61**	71.43	7.39	**74.30**	70.60	66.95	6.80
HSP	GPT−3.5	70.45	70.51	**70.45**	4.88	65.13	71.01	**63.22**	3.71	71.26	72.71	**71.93**	5.57	68.95	**71.41**	**68.53**	4.72
Prompt	GPT-4o	79.82	73.67	71.53	1.28	67.10	71.10	56.46	1.47	84.47	67.28	71.37	1.27	77.13	70.68	66.45	1.34
CoNLI	GPT-4o	**82.40**	78.97	77.83	6.68	**71.43**	**79.12**	**66.81**	6.25	**85.96**	**72.53**	**76.73**	7.48	**79.93**	76.87	73.79	6.80
HSP	GPT-4o	81.99	**81.48**	**81.60**	5.33	68.14	75.20	66.20	3.77	77.79	74.02	75.64	5.61	75.97	**76.90**	**74.48**	4.90

P, R, F1 are short for precision, recall, F1 score, respectively. The best results for each model are in bold. Q-32B, Q-72B, GPT-3.5, and GPT-4o refer to the versions Qwen1.5-32B-Chat, Qwen2-72B-Instruct, GPT-3.5-Turbo-0125, and GPT-4o-0513, respectively

Table 4 Hallucination detection results on HaluBench

Method	Model	FinanceBench				CovidQA				PubMedQA				Average			
		P ↑	R ↑	F1 ↑	TER ↓	P ↑	R ↑	F1 ↑	TER ↓	P ↑	R ↑	F1 ↑	TER ↓	P ↑	R ↑	F1 ↑	TER ↓
Alignscore		55.63	51.90	42.35	–	77.67	74.40	73.62	–	68.60	60.20	55.13	–	67.30	32.17	57.03	–
Prompt	Q-32B	61.21	56.30	50.92	1.44	81.06	72.40	70.33	1.06	**84.13**	79.20	78.42	1.56	75.47	69.30	66.56	1.35
CoNLI	Q-32B	60.06	58.50	56.82	5.14	**84.18**	**79.70**	79.01	2.36	82.20	**82.20**	**82.20**	7.61	75.48	**73.47**	**72.68**	5.04
HSP	Q-32B	**65.77**	**60.50**	**56.90**	3.72	79.39	79.30	**79.28**	2.04	81.65	80.00	79.74	6.57	**75.60**	73.27	71.97	4.11
Prompt	Q-72B	66.54	55.90	47.45	1.64	79.47	71.00	68.76	1.08	86.62	83.10	82.68	1.53	77.54	70.00	66.30	1.51
CoNLI	Q-72B	67.61	**67.60**	**67.60**	5.60	87.37	87.00	86.97	2.22	83.63	83.50	83.48	5.11	79.54	79.37	**79.35**	6.97
HSP	Q-72B	**72.76**	**67.60**	65.65	3.63	**88.23**	**87.20**	**87.11**	3.10	**87.88**	**85.40**	**85.16**	4.40	**82.96**	**80.07**	79.31	4.63

P, R, F1 are short for precision, recall, F1 score, respectively. The best results for each model are in bold. Q-32B, Q-72B, GPT-3.5, and GPT-4o refer to the versions Qwen1.5-32B-Chat, Qwen2-72B-Instruct, GPT-3.5-Turbo-0125, and GPT-4o-0513, respectively

the baseline method based on Alignscore, the prompt-based method requires Qwen2-72B-Instruct to perform as well as well-designed small model. This indicates that research on hallucination detection methods is necessary, current large language models' capabilities are insufficient to complete hallucination detection tasks with simple prompts.

Ablation study

To verify the effectiveness of specific parameter choices and designs in our method, we conducted modular experiments on QAGS-CNNDM and QAGS-XSum datasets with Qwen2-72B-Instruct.

We introduced an evidence matching method based on embedding similarity in the evidence matching part of our method. Different similarity functions $sim(\cdot)$ were tested, including fuzzy matching based on Levenshtein distance, rerank score matching, and embedding cosine similarity matching using two different embedding models. One is the BGE-m3 [55] model based on XLM-RoBERTa, which is of moderate size, and the other is the SFR-Embedding-2-R model [56] based on Mistral-7B, which is larger and achieved 1st place in the MTEB leaderboard in June 2023. The results are shown in Table 5.

The results show that the embedding similarity method achieves the best results on both datasets, and that larger embedding models lead to better performance but also to higher computational costs. The differences in the token expansion ratio (TER) among the methods are not significant.

Regarding the selection of retrieval top-k, the impact of different k values on the results is shown in Table 6.

We can see that the top-3 perform best on the QAGS-CNNDM dataset, while the top-4 perform best on the QAGS-XSUM dataset. The results indicate that both smaller and larger values of k lead to performance degradation. It is hypothesized that when k is small, the semantic granularity is too fine, resulting in insufficient evidence. Conversely, when k is large, the introduction of excessive irrelevant information adversely affects performance. Moreover, the larger the k, the higher the TER, which means higher computational cost. To ensure consistency on different dataset, finally we choose top-3 as evidence.

In the hierarchical semantic extractor module, we introduced two levels of semantic piece, sentence-level and entity-level. The experiments tested the effectiveness of extracting semantic pieces at different levels. The results are presented in Table 7.

The results indicate that hierarchical semantic piece extraction achieves the highest performance. While sentence-level semantic piece extraction alone yields good results, entity-level semantic piece extraction alone performs poorly due to its limitation to four basic schema types, which fails to

Table 5 Effectiveness of different evidence matching methods

Method	Model	QAGS-CNNDM				QAGS-XSUM			
		P	R	F1	TER	P	R	F1	TER
Fuzzy	–	70.63	67.95	66.35	4.83	59.83	61.65	60.23	3.73
Rerank	bge-reranker-v2-m3 [55]	76.93	74.64	73.61	4.77	63.79	68.53	57.99	3.70
Embedding	bge-m3 [55]	**78.45**	76.31	75.40	4.76	63.77	68.92	60.43	3.67
Embedding	SFR-Embedding-2-R [56]	78.22	**77.03**	**76.42**	4.76	**65.07**	**70.68**	**61.48**	3.66

P, R, F1 are short for precision, recall, F1 score, respectively. The best results are in bold

Table 6 Effect of different *k* values on evidence selection

k	QAGS-CNNDM				QAGS-XSUM			
	Precision	Recall	F1	TER	Precision	Recall	F1	TER
1	70.96	68.74	67.40	4.37	62.60	66.14	**62.88**	3.39
2	75.69	73.38	72.28	4.56	64.99	70.62	62.04	3.53
3	**78.22**	**77.03**	**76.42**	4.76	65.07	70.68	61.48	3.66
4	75.79	74.51	73.81	4.94	**65.67**	**71.33**	61.06	3.79
5	73.99	72.81	72.10	5.14	64.47	69.63	59.45	3.93

The best results are in bold

Table 7 Effectiveness of different levels of semantic piece extraction

Level	QAGS-CNNDM				QAGS-XSUM			
	P	R	F1	TER	P	R	F1	TER
Sent	76.68	73.85	72.62	4.45	63.62	68.48	58.58	3.52
Ent	65.71	55.98	47.04	0.30	62.19	58.19	36.65	0.14
Sent+Ent	**78.22**	**77.03**	**76.42**	4.76	**65.07**	**70.68**	**61.48**	3.66

The best results are in bold

encompass all semantic information. However, when combined, the entity-level semantic pieces introduce additional information for the verification module to consider, thereby enhancing the overall performance of the hierarchical semantic piece extraction method.

Error analysis

To further understand the ability range of the HSP method, a manual analysis of a proportion of the error samples was conducted, and a number of primary error types were summarized. Limited by space, we provide detailed error samples in the Appendix C.

Complex text format: Some texts are in complex formats, leading to difficulty in extracting semantic pieces. Semantic piece extraction is the foundation of our method. If the extraction is incorrect, the subsequent reasoning tasks will be affected.

Wrong evidence matching: In our method, cosine similarity is used to match the evidence, for computational efficiency we use top-3 evidence. However, the embedding-based cosine similarity does not always align perfectly with the evidence. Optimal values for top-k vary across datasets, and incorrect settings can result in insufficient or excessive evidence.

NLI error: We use LLM for NLI task, sometimes the model makes mistakes in reasoning. Occasionally, the model commits reasoning errors, which affect the final outcome as this step is the concluding phase of our method.

Hallucination mitigation

For the hallucination mitigation task, we adopt an end-to-end testing method. First, we detect whether there is hallucination in the text, and then mitigate the hallucination. By comparing the original text with the refined text, we calculate the consistency with the reference text.

Datasets

We use four datasets for the hallucination mitigation task, including HaluEval Summarization, CRUD-QA-Qwen, CRUD-QA-SLM, and SummEval.

Table 8 Hallucination mitigation results on different datasets

Dataset	Method	Rouge1	Rouge2	RougeL	Bleu avg	Bleu-4	BertScore	AlignScore	FactCC
HaluEval Summ	Original	57.32	33.88	45.12	23.17	10.45	80.97	18.43	17.62
	Prompt	56.33	33.51	44.41	22.68	10.33	80.94	20.72	16.86
	HSP	**57.45**	**34.40**	**45.27**	**24.13**	**10.96**	**81.57**	**22.20**	**17.72**
CRUD QA Qwen	Original	71.51	60.86	68.33	**58.83**	**50.84**	91.75	**75.14**	–
	Prompt	70.29	59.89	67.16	58.06	50.25	91.58	72.79	–
	HSP	**72.00**	**61.00**	**68.59**	58.55	50.54	**92.05**	74.18	–
CRUD QA SLM	Original	37.32	25.16	34.37	26.00	20.70	79.86	**46.51**	–
	Prompt	33.16	22.55	30.48	23.82	19.13	78.82	43.65	–
	HSP	**40.43**	**27.90**	**37.32**	**28.18**	**22.30**	**81.38**	46.42	–
SummEval	Original	66.47	49.19	**55.98**	48.75	36.48	83.11	**35.78**	19.93
	Prompt	64.99	46.89	54.84	46.16	33.98	82.16	35.53	19.59
	HSP	**66.79**	**49.89**	55.92	**49.67**	**36.61**	**84.87**	35.63	**20.94**

The best results for each dataset are in bold. For FactCC metric, it doesn't support other language except English, so we only report the results on HaluEval Summ and SummEval dataset

- HaluEval Summ [57]: a subset of HaluEval dataset, we sample 250 samples for the hallucination mitigation task. Each sample contains hallucination text, reference text, and correct answer.
- CRUD-QA [58]: CRUD Benchmark is a benchmark for evaluating the complete RAG process. From the News QA section of the dataset, we sampled 143 instances. We use Qwen1.5-32B-Chat and an internal tiny language model to generate answers, thereby creating the CRUD-QA-Qwen and CRUD-QA-SLM datasets. The results of Qwen1.5-32B-Chat are largely acceptable, and this set will be used to ensure that our method does not over-mitigate.
- SummEval [52]: For the mitigation task, we merge all summaries in the SummEval dataset according to the reference text. For each reference text, there are multiple summaries, including hallucination content and correct content. We select all hallucination summaries to construct the mitigation task dataset, which contains 294 samples. Each sample contains hallucination text, reference text, and correct answer.

Experimental results

For comparison, we use the prompt-based method as a baseline; the prompt-based method instructs the model to repair the text directly. For all experiments, Qwen2-72B-Instruct is adopted for detection and mitigation, and the generated text is compared with the annotated correct answer. The results are shown in Table 8.

Our method effectively optimizes the output content, bringing it closer to the reference answer. On the CRUD-QA-Qwen dataset, it successfully maintains the original answer without over-mitigation. In comparison to the common prompt-based method, which often results in performance decline, our approach avoids over-mitigation. The prompt-based method tends to introduce unnecessary information and suffers from low hallucination detection accuracy, which leads to performance degradation.

Discussion and conclusion

In this paper, address the hallucination challenge in LLM-based applications such as industry scenarios, a unified hallucination detection and mitigation framework is proposed. A novel method, hierarchical semantic piece (HSP), which uses hierarchical semantic piece extraction and evidence matching grounded in embedding cosine similarity, is introduced. The experimental results show that our method has better hallucination detection performance compared to the baseline and existing methods, and reduces the token expansion ratio (TER) by 30%, providing a more efficient and cost-effective method. In terms of hallucination mitigation, the HSP method can effectively optimize the output, making it closer to the reference answer while maintaining output quality. We also conducted ablation experiments to verify the effectiveness of some designs and hyperparameters choices in our method.

Our method is built upon the capabilities of LLMs, although it has demonstrated lower computational cost compared to other LLM-Based methods, it still can't compete with optimized small models in terms of efficiency. In the future we will consider using smaller models to achieve higher efficiency and work towards real-time hallucination detection and mitigation. In addition, some hyperparameters of our processes in the current method still need to be manually tuned, and in the future we plan to dynamically optimize these hyperparameters so that the method can automatically adapt to various scenarios.

Appendix A: Prompt templates

In this section, we provide all the prompt templates we used in our experiments. All variables in the prompt templates are denoted by curly braces.

A.1 Sentence-level semantic piece extraction

Please breakdown the user input sentence into independent facts.
Output format: Use a newline character to separate each semantic piece to ensure clear distinction between each piece.
[Input sentence]
{text}

A.2 NLI task

Determine the relation between the premise and hypothesis. Available relations include entailment, contradiction, and neutral.

[Classes]
* Entailment: The hypothesis can be inferred from the premise.
* Contradiction: The hypothesis contradicts the premise.
* Neutral: The hypothesis neither entails nor contradicts the premise, or its relation cannot be determined.
Output format: Output only "Contradiction", "Entailment" to indicate the relation between the premise and hypothesis.
Now it's your turn! Please determine the relation between the premise and hypothesis, carefully consider some special information such as time, location, and numbers.
[Premise]
{premise}
[Hypothesis]
{hypothsis}

A.3 Hallucination mitigation

You are a professional editor. Here is a sentence may have issues. Please directly correct the sentence with reasoning. I will give you some reference information to help you find the error. If you can't find any error, please return Correct in reason part, remember to keep the original language.
Output Format

The output includes a reason, a correct sentence, and the final sentence. Please form them in the following format:
[Reason]
<reason>
[Correct Sentence]
<sentence>
[Final Sentence]
<final_fix_result>
Now is your time to show your editing skills.
[Reference Text]
{premises}
[Sentence to Fix]
{hypothesis}

A.4 Prompt-based hallucination detection

Give the following premise and hypothesis, determine the relationship between them, available choices are:
* Entailment: The hypothesis can be inferred from the premise.
* Contradiction: The hypothesis contradicts the premise.
* Neutral: The hypothesis neither entails nor contradicts the premise, or its relation cannot be determined.
Example 0:
Premise: The quick brown fox jumps over the lazy dog.
Hypothesis: The fox is lazy.
Relationship: Contradiction
Example 1:
Premise: Mrs. Smith is a teacher who loves to read.
Hypothesis: Mrs. Smith is a teacher.
Relationship: Entailment
Example 2:
Premise:
The sun is shining in the sky. Hypothesis: A fire is burning in the fireplace.
Relationship: Neutral Now is your turn:
[Premise]
{premise}
[Hypothesis]
{hypothesis}

A.5 Prompt-based hallucination mitigation

You are a professional editor. Here is a sentence may have issues. Please directly correct the sentence.
I will give you some reference information to help you find the error.
If you can't find any error, please return Correct in reason part, remember to keep the original language.
Output Format
The output includes the final sentence. Please form them in the following format:
[Final Sentence]

```
<final_fix_result>
[Reference]
{premise}
[Sentence]
{hypothesis}
```

Appendix B: Detailed example of HSP method

To provide a more detailed understanding of the HSP method, we present detailed examples of the hallucination detection and mitigation processes. Here is a hallucination example.

Reference Text Jane was an American magazine created to appeal to the women who grew up reading "Sassy Magazine"; Jane Pratt was the founding editor of each.First for Women is a woman's magazine published by Bauer Media Group in the USA.

Sentence to Detect While Jane and First for Women are both publications, one is aimed at women while the other is not.

B.1 Hierarchical semantic piece extraction

In this part, our method extracts hierarchical semantic pieces from the reference text and the sentence to detect. The semantic pieces are extracted at the sentence level and the entity level. The entity in the entity-level sentence piece is highlighted in red.

Semantic Pieces of Reference Text:

1. Jane was an American magazine.
2. Jane was created to appeal to the women who grew up reading "Sassy Magazine".
3. Jane Pratt was the founding editor of Jane.
4. Jane Pratt was the founding editor of Sassy Magazine.
5. First for Women is a woman's magazine.
6. First for Women is published by Bauer Media Group.
7. First for Women is published in the USA.
8. [**Jane Pratt (Type:Name)**] was the founding editor of each.
9. First for Women is a woman's magazine published by [**Bauer (Type:Name)**] Media Group in the USA.

Semantic Pieces of the Sentence:

1. Jane is a publication.
2. First for Women is a publication.
3. Jane is aimed at women.
4. First for Women is not specifically aimed at women.
5. While [**Jane (Type:Name)**] and First for Women are both publications, one is aimed at women while the other is not.

B.2 Evidence matching and fact verification

In this part, we match the semantic pieces of the sentence to detect with the reference semantic piece and verify the facts. The matched reference semantic pieces are listed, and the NLI results are provided (Table 9).
Detection Result: The sentence contains hallucination, as it contradicts the reference text.

B.3 Hallucination mitigation

In this part, we mitigate the hallucination in the sentence to detect. The refined sentence is presented, and the comparison with the reference text is provided (Table 10).

- **Original**: While Jane and First for Women are both publications, one is aimed at women while the other is not.
- **Fixed Sentence**: While Jane and First for Women are both publications aimed at women, they have different focuses and audiences.

Appendix C: Error examples

C.1 Semantic piece extraction error

In following example, the semantic piece extraction module fails to extract. The reference text has a complex format, which makes it difficult for LLM to understand. As a result, LLM directly responds original text, lead to wrong semantic piece extraction.

Table 9 Evidence matching and fact verification results

Semantic piece	Matched reference semantic piece	NLI result
Jane is a publication	- Jane was created to appeal to the women who grew up reading "Sassy Magazine" - Jane was an American magazine - Jane Pratt was the founding editor of Jane	Entailment
First for Women is a publication	- First for Women is published in the USA - First for Women is a woman's magazine - First for Women is published by Bauer Media Group	Entailment
Jane is aimed at women	- Jane Pratt was the founding editor of Jane - Jane was an American magazine - Jane was created to appeal to the women who grew up reading "Sassy Magazine"	Entailment
First for Women is not specifically aimed at women	- First for Women is a woman's magazine published by Bauer Media Group in the USA - First for Women is a woman's magazine - First for Women is published by Bauer Media Group	Contradiction
While Jane and First for Women are both publications, one is aimed at women while the other is not	- First for Women is a woman's magazine published by Bauer Media Group in the USA	Neutral

Table 10 Hallucination mitigation results

Hallucinated piece	Original sentence	Refined sentence
First for Women is not specifically aimed at women	While Jane and First for Women are both publications, one is aimed at women while the other is not	While Jane and First for Women are both publications aimed at women, they have different focuses and audiences

Complex reference text Example

Input Text:
At December 31,
Maturities
Interest
Rates %
2022
2021
Verizon Communications
< 5 Years
0.75 - 5.82
$
23,929
$
18,406
5-10 Years
…

Extract semantic pieces:

1. At December 31
2. Maturities
3. Interest
4. Rates %
5. 2022
6. 2021
7. Verizon Communications
8. < 5 Years
9. 0.75 - 5.82
10. $
11. 23,929
12. $
13. 18,406
14. 5-10 Years

…

C.2 Wrong evidence matching

In the following example, the evidence matching module fails to match the correct evidence in top-3 evidence, the correct evidence is the 4th evidence, as shown in the Table 11, leading to a wrong detection result.

Hypothesis Semantic Pieces: Shenyang boasts the highest urban population in Northeast China.

C.3 NLI error

In the following example, the NLI module makes a mistake in reasoning, leading to an incorrect result. In the hypothesis, "born with" is a key phrase that indicates a contradiction, but the model incorrectly predicts an entailment relationship.

Table 11 Reference semantic pieces and matching scores

Evidence Text	Score
Shenyang is the largest city in Northeast China by urban population	0.8949
Shenyang is the most populous city in Liaoning province	0.8508
The Shenyang metropolitan area is one of the major megalopolises in China	0.8107
Shenyang is the second-largest city in Northeast China by metropolitan population, behind Harbin.	0.8088
The Shenyang metropolitan area has a population of over 23 million	0.7967
Shenyang is a sub-provincial city in China	0.7891
Shenyang is the provincial capital of Liaoning province	0.7656
Shenyang has a population of 9,070,093 as of the 2020 census	0.7546
Shenyang's administrative region includes ten metropolitan districts	0.7036
Shenyang's administrative region includes the county-level city of Xinmin	0.6936
Shenyang's administrative region includes the counties of Kangping and Faku	0.6868

NLI Error Example

Premises:
- Maickel melamed is a venezuelan native battling muscular dystrophy.
- Despite the odds against him, maickel melamed crossed the finish line.
- Maick melamed wants to show that life is great, no matter how many problems you can have.

Hypothesis:
- Maickel Melamed was born with muscular dystrophy.

NLI Result:
- Entailment

Correct Label:
- Contradiction

Appendix D: Entity Extraction Schema

See Table 12.

Table 12 Entity extraction schema

Type	Schema
Time	Date, Calendar date, Date range, Year, Month, Time, Time range, End time, Start time, Time period, Duration
Number	Price, Cost, Expenditure, Spending, Expense, Quantity, Amount, Number, Total, Ratio, Percentage, Rate, Proportion, Maximum value, Minimum value, Percentile, Length, Area, Volume, Weight, Temperature
Location	City, Country, Province, State, Region, Area, Street, Address, Place, Building, Scenic spot, Landmark
Personal Info	Name, First name, Last name, Position, Degree, Education, Graduated from, Employer

Acknowledgements This study is supported by Liaoning Provincial Science and Technology Innovation Project in the Field of Artificial Intelligence (Project name: Research on key technologies for systems engineering of large language model) (Grant no. 2023JH26/10100005); Shenyang Young and Middle-aged Science and Technology Innovation Talent Support Program (Grant no. RC220414).

Data availability The data used in this study are available from the corresponding author upon reasonable request.

Declarations

Conflict of interest On behalf of all authors, the corresponding author states that there is no conflict of interest.

References

1. Zhao WX, Zhou K, Li J, Tang T, Wang X, Hou Y, Min Y, Zhang B, Zhang J, Dong Z, Du Y Yang C, Chen Y, Chen Z, Jiang J, Ren R, Li Y, Tang X, Liu Z, Liu P, Nie J-Y, Wen J-R (2023) A survey of large language models. arXiv:2303.18223
2. Min B, Ross H, Sulem E, Veyseh A, Nguyen TH, Sainz O, Agirre E, Heintz I, Roth D (2023) Recent advances in natural language processing via large pre-trained language models: a survey. ACM Comput Surv 56(2):1–40

3. Morris MR, Sohl-Dickstein J, Fiedel N, Warkentin T, Dafoe A, Faust A, Farabet C, Legg S (2023) Levels of agi: operationalizing progress on the path to agi. arXiv:2311.02462
4. Zhang C, Zhang C, Li C, Qiao Y, Zheng S, Dam SK, Zhang M, Kim JU, Kim ST, Choi J et al (2023) One small step for generative ai, one giant leap for agi: a complete survey on chatgpt in aigc era. arXiv:2304.06488
5. Kaddour J, Harris J, Mozes M, Bradley H, Raileanu R, McHardy R (2023) Challenges and applications of large language models. arXiv:2307.10169
6. Ji Z, Lee N, Frieske R, Yu T, Su D, Xu Y, Ishii E, Bang YJ, Madotto A, Fung P (2023) Survey of hallucination in natural language generation. ACM Comput Surv 55(12):1–38
7. Zhang Y, Li Y, Cui L, Cai D, Liu L, Fu T, Huang X, Zhao E, Zhang Y, Chen Y et al (2023) Siren's song in the ai ocean: a survey on hallucination in large language models. arXiv:2309.01219
8. Achiam J, Adler S, Agarwal S, Ahmad L, Akkaya I, Aleman FL, Almeida D, Altenschmidt J, Altman S, Anadkat S et al (2023) Gpt-4 technical report. arXiv:2303.08774
9. Team G, Anil R, Borgeaud S, Wu Y, Alayrac J-B, Yu J, Soricut R, Schalkwyk J, Dai AM, Hauth A et al (2023) Gemini: a family of highly capable multimodal models. arXiv:2312.11805
10. Tonmoy S, Zaman S, Jain V, Rani A, Rawte V, Chadha A, Das A (2024) A comprehensive survey of hallucination mitigation techniques in large language models. arXiv:2401.01313
11. Penedo G, Malartic Q, Hesslow D, Cojocaru R, Alobeidli H, Cappelli A, Pannier B, Almazrouei E, Launay J (2023) The refinedweb dataset for falcon llm: outperforming curated corpora with web data only. Adv Neural Inf Process Syst 36:79155–79172
12. Tian K, Mitchell E, Yao H, Manning CD, Finn C (2023) Fine-tuning language models for factuality. arXiv:2311.08401
13. Wan F, Huang X, Cui L, Quan X, Bi W, Shi S (2024) Mitigating hallucinations of large language models via knowledge consistent alignment. arXiv:2401.10768
14. Sun Z, Shen S, Cao S, Liu H, Li C, Shen Y, Gan C, Gui L-Y, Wang Y-X, Yang Y et al (2023) Aligning large multimodal models with factually augmented rlhf. arXiv:2309.14525
15. Li K, Patel O, Viégas F, Pfister H, Wattenberg M (2024) Inference-time intervention: eliciting truthful answers from a language model. In: Advances in neural information processing systems, vol 36
16. Gou Z, Shao Z, Gong Y, Shen Yang Y, Duan N, Chen W (2024) CRITIC: large language models can self-correct with tool-interactive critiquing. In: The 12th international conference on learning representations. https://openreview.net/forum?id=Sx038qxjek
17. Manakul P, Liusie A, Gales MJ (2023) Selfcheckgpt: zero-resource black-box hallucination detection for generative large language models. arXiv:2303.08896
18. Hu L, Liu Z, Zhao Z, Hou L, Nie L, Li J (2023) A survey of knowledge enhanced pre-trained language models. IEEE Trans Knowl Data Eng
19. Wang C, Liu X, Yue Y, Tang X, Zhang T, Jiayang C, Yao Y, Gao W, Hu X, Qi Z et al (2023) Survey on factuality in large language models: knowledge, retrieval and domain-specificity. arXiv:2310.07521
20. Shuster K, Poff S, Chen M, Kiela D, Weston J (2021) Retrieval augmentation reduces hallucination in conversation. arXiv:2104.07567
21. Novelli C, Casolari F, Hacker P, Spedicato G, Floridi L (2024) Generative ai in eu law: liability, privacy, intellectual property, and cybersecurity. arXiv:2401.07348
22. Thirunavukarasu AJ, Ting D, Elangovan K, Gutierrez L, Tan TF, Ting D (2023) Large language models in medicine. Nat Med 29(8):1930–1940
23. Paul D, Namperumal G, Surampudi Y (2023) Optimizing llm training for financial services: best practices for model accuracy, risk management, and compliance in ai-powered financial applications. J Artif Intell Res Appl 3(2):550–588
24. Yao Y, Duan J, Xu K, Cai Y, Sun Z, Zhang Y (2024) A survey on large language model (llm) security and privacy: the good, the bad, and the ugly. High-Confid Comput 100211
25. Urlana A, Kumar CV, Singh AK, Garlapati BM, Chalamala SR, Mishra R (2024) Llms with industrial lens: deciphering the challenges and prospects–a survey. arXiv:2402.14558
26. Touvron H, Martin L, Stone K, Albert P, Almahairi A, Babaei Y, Bashlykov N, Batra S, Bhargava P, Bhosale S et al (2023) Llama 2: open foundation and fine-tuned chat models. arXiv:2307.09288
27. Jiang AQ, Sablayrolles A, Mensch A, Bamford C, Chaplot DS, Casas Ddl, Bressand F, Lengyel G, Lample G, Saulnier L et al (2023) Mistral 7b. arXiv:2310.06825
28. Bai J, Bai S, Chu Y, Cui Z, Dang K, Deng X, Fan Y, Ge W, Han Y, Huang F et al (2023) Qwen technical report. arXiv:2309.16609
29. Devlin J, Chang M-W, Lee K, Toutanova K (2018) Bert: pre-training of deep bidirectional transformers for language understanding. arXiv:1810.04805
30. Marvin G, Hellen N, Jjingo D, Nakatumba-Nabende J (2023) Prompt engineering in large language models. In: International conference on data intelligence and cognitive informatics. Springer, pp 387–402
31. Lewis P, Perez E, Piktus A, Petroni F, Karpukhin V, Goyal N, Küttler H, Lewis M, Yih W-T, Rocktäschel T et al (2020) Retrieval-augmented generation for knowledge-intensive nlp tasks. Adv Neural Inf Process Syst 33:9459–9474
32. Li H, Su Y, Cai D, Wang Y, Liu L (2022) A survey on retrieval-augmented text generation. arXiv:2202.01110
33. Xi Z, Chen W, Guo X, He W, Ding Y, Hong B, Zhang M, Wang J, Jin S, Zhou E et al (2023) The rise and potential of large language model based agents: a survey. arXiv:2309.07864
34. Guo T, Chen X, Wang Y, Chang R, Pei S, Chawla NV, Wiest O, Zhang X (2024) Large language model based multi-agents: a survey of progress and challenges. arXiv:2402.01680
35. Chen W, Yan-yi L, Tie-zheng G, Da-peng L, Tao H, Zhi L, Qing-wen Y, Hui-han W, Ying-you W (2024) Systems engineering issues for industry applications of large language model. Appl Soft Comput 151:111165
36. Asai A, Wu Z, Wang Y, Sil A, Hajishirzi H (2023) Self-rag: learning to retrieve, generate, and critique through self-reflection. arXiv:2310.11511
37. Ji Z, Yu T, Xu Y, Lee N, Ishii E, Fung P (2023) Towards mitigating llm hallucination via self reflection. In: Findings of the association for computational linguistics: EMNLP 2023, pp 1827–1843
38. Wei J, Wang X, Schuurmans D, Bosma M, Xia F, Chi E, Le QV, Zhou D et al (2022) Chain-of-thought prompting elicits reasoning in large language models. Adv Neural Inf Process Syst 35:24824–24837
39. Lyu Q, Havaldar S, Stein A, Zhang L, Rao D, Wong E, Apidianaki M, Callison-Burch C (2023) Faithful chain-of-thought reasoning. arXiv:2301.13379
40. Waldendorf J, Haddow B, Birch A (2024) Contrastive decoding reduces hallucinations in large multilingual machine translation models. In: Proceedings of the 18th conference of the European chapter of the association for computational linguistics (volume 1: long papers), pp 2526–2539
41. Li D, Sun Z, Hu X, Liu Z, Chen Z, Hu B, Wu A, Zhang M (2023) A survey of large language models attribution. arXiv:2311.03731
42. Min S, Krishna K, Lyu X, Lewis M, Yih W-t, Koh PW, Iyyer M, Zettlemoyer L, Hajishirzi H (2023) Factscore: fine-grained atomic evaluation of factual precision in long form text generation. arXiv:2305.14251
43. Wei J, Yang C, Song X, Lu Y, Hu N, Tran D, Peng D, Liu R, Huang D, Du C et al (2024) Long-form factuality in large language models. arXiv:2403.18802

44. Gao L, Dai Z, Pasupat P, Chen A, Chaganty AT, Fan Y, Zhao V, Lao N, Lee H, Juan D-C, Guu K (2023) RARR: researching and revising what language models say, using language models. In: Rogers A, Boyd-Graber J, Okazaki N (eds) Proceedings of the 61st annual meeting of the association for computational linguistics (volume 1: long papers). Association for Computational Linguistics, Toronto, pp 16477–16508. https://doi.org/10.18653/v1/2023.acl-long.910. https://aclanthology.org/2023.acl-long.910
45. Lei D, Li Y, Hu M, Wang M, Yun X (2023) Chain of natural language inference for reducing large language model hallucinations. In: NeurIPS 2023 workshop on instruction tuning and instruction following
46. Lu Y, Liu Q, Dai D, Xiao X, Lin H, Han X, Sun L, Wu H (2022) Unified structure generation for universal information extraction. arXiv:2203.12277
47. Yang A, Yang B, Hui B, Zheng B, Yu B, Zhou C, Li C, Li C, Liu D, Huang F et al (2024) Qwen2 technical report. arXiv:2407.10671
48. Zhang T, Kishore V, Wu F, Weinberger KQ, Artzi Y (2019) Bertscore: evaluating text generation with bert. arXiv:1904.09675
49. Zha Y, Yang Y, Li R, Hu Z (2023) AlignScore: evaluating factual consistency with a unified alignment function. In: Rogers A, Boyd-Graber J, Okazaki N (eds) Proceedings of the 61st annual meeting of the association for computational linguistics (volume 1: long papers). Association for Computational Linguistics, Toronto, pp 11328–11348. https://doi.org/10.18653/v1/2023.acl-long.634. https://aclanthology.org/2023.acl-long.634
50. Kryściński W, McCann B, Xiong C, Socher R (2019) Evaluating the factual consistency of abstractive text summarization. arXiv:1910.12840
51. Wang A, Cho K, Lewis M (2020) Asking and answering questions to evaluate the factual consistency of summaries. In: Jurafsky D, Chai J, Schluter N, Tetreault J (eds) Proceedings of the 58th annual meeting of the association for computational linguistics. Association for Computational Linguistics, pp 5008–5020. https://doi.org/10.18653/v1/2020.acl-main.450. https://aclanthology.org/2020.acl-main.450
52. Fabbri AR, Kryściński W, McCann B, Xiong C, Socher R, Radev D (2020) Summeval: re-evaluating summarization evaluation. arXiv:2007.12626
53. Ravi SS, Mielczarek B, Kannappan A, Kiela D, Qian R (2024) Lynx: an open source hallucination evaluation model. arXiv:2407.08488
54. Honovich O, Aharoni R, Herzig J, Taitelbaum H, Kukliansy D, Cohen V, Scialom T, Szpektor I, Hassidim A, Matias Y (2022) True: re-evaluating factual consistency evaluation. arXiv:2204.04991
55. Chen J, Xiao S, Zhang P, Luo K, Lian D, Liu, Z (2024) Bge m3-embedding: multi-lingual, multi-functionality, multi-granularity text embeddings through self-knowledge distillation. arXiv:2402.03216
56. Meng R, Liu Y, Joty SR, Xiong C, Zhou Y, Yavuz S (2024) Sfrembedding-mistral: enhance text retrieval with transfer learning. Salesforce AI Research Blog, vol 3
57. Li J, Cheng X, Zhao X, Nie J-Y, Wen J-R (2023) HaluEval: a large-scale hallucination evaluation benchmark for large language models. In: Bouamor H, Pino J, Bali K (eds) Proceedings of the 2023 conference on empirical methods in natural language processing. Association for Computational Linguistics, Singapore, pp 6449–6464. https://doi.org/10.18653/v1/2023.emnlp-main.397. https://aclanthology.org/2023.emnlp-main.397
58. Lyu Y, Li Z, Niu S, Xiong F, Tang B, Wang W, Wu H, Liu H, Xu T, Chen E (2024) Crud-rag: a comprehensive Chinese benchmark for retrieval-augmented generation of large language models. arXiv:2401.17043