KSHSeek: Data-Driven Approaches to Mitigating and Detecting Knowledge-Shortcut Hallucinations in Generative Models

Anonymous ACL submission

Abstract

The emergence of large language models (LLMs) has significantly advanced the development of natural language processing (NLP), especially in text generation tasks like question answering. However, model hallucinations remain a major challenge in natural language generation (NLG) tasks due to their complex causes. We systematically expand on the causes of factual hallucinations from the perspective of knowledge shortcuts, analyzing hallucinations arising from correct and defect-free data and demonstrating that knowledge-shortcut hallucinations are prevalent in generative models. To mitigate this issue, we propose a *high similarity pruning algorithm* at the data preprocessing level to reduce spurious correlations in the data. Additionally, we design a specific detection method for knowledge-shortcut hallucinations to evaluate the effectiveness of our mitigation strategy. Experimental results show that our approach effectively reduces knowledge-shortcut hallucinations, particularly in fine-tuning tasks, without negatively impacting model performance in question answering. This work introduces a new paradigm for mitigating specific hallucination issues in generative models, enhancing their robustness and reliability in real-world applications.

1 Introduction

The emergence of large language models (LLMs) has brought a paradigm shift to natural language processing (NLP), especially in generative tasks such as question-answering (Rangapur and Rangapur, 2024; Michail et al., 2023; Qin et al., 2023) However, this revolution has also caused a growing concern, known as model hallucinations. Huang et al.(Huang et al., 2024) building on the definition of hallucinations proposed by Ji et al.(Dziri et al., 2021; Ji et al., 2023), expanded the applicability and scope of the term, classifying model hallucinations into two types: factual hallucinations and

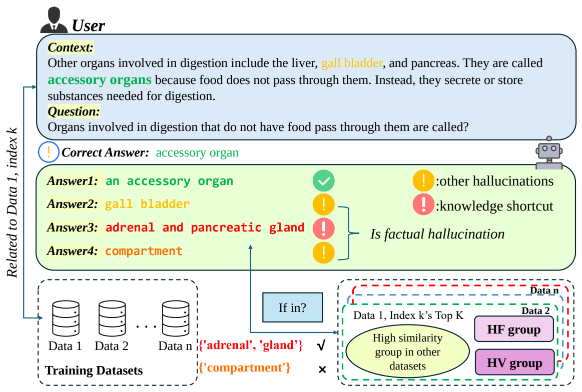

User
Context:
Other organs involved in digestion include the liver, gall bladder, and pancreas. They are called accessory organs because food does not pass through them. Instead, they secrete or store substances needed for digestion.
Question:
Organs involved in digestion that do not have food pass through them are called?
Correct Answer: accessory organ
Answer1: an accessory organ
Answer2: gall bladder
Answer3: adrenal and pancreatic gland
Answer4: compartment
:other hallucinations
:knowledge shortcut
Is factual hallucination
Related to Data 1, index k
Data 1
Data 2
Data n
Training Datasets
{'adrenal', 'gland'} √
{'compartment'} ×
If in?
Data n
Data 2
Data 1, Index k's Top K
High similarity group in other datasets
HF group
HV group

Figure 1: An example of what is the knowledge-shortcut hallucinations in CQA tasks

faithfulness hallucinations. This expanded classification provides a new paradigm for understanding model hallucinations.

We focus on factual hallucinations, and have observed a critical fact: training data has played a significant role in causing factual hallucinations. One notable example is the "floating-point comparison hallucination"[1], When the prompt *"9.11 or 9.9, which number is larger?"* is given to LLMs, many existing commercial LLMs provide incorrect answers, as illustrated in Appendix A, Table 6.

A major cause of the aforementioned hallucinations is knowledge-shortcut(Ju et al., 2024;Li et al., 2022). The training data often contains a significant amount of information such as computer system version numbers and book indexes. LLMs have learned the comparative features of this data and erroneously applied these features to the comparison of regular numbers, leading to hallucinations. In the classification proposed by Li(Li et al., 2022), this cause is referred to as a "Knowledge Shortcut". Building on this concept, we define hallucinations caused by knowledge shortcuts as knowledge-shortcut hallucinations.

[1] `https://x.com/goodside/status/1812977352085020680`

Index	C-Q	Correct-Answer	Generate-Answer	Jaccard-sim	TF_IDF-sim	AI-sim
before 1	<user|>Other...	accessory organs	liver	0.00000	0.00000	0.39682
before 2	<user|>Other...	accessory organs	assistant organ	0.00000	0.00000	**0.54355**
before 3	<user|>Other...	accessory organs	adrenal and pancreatic gland	0.00000	0.00000	0.30573
before 4	<user|>Other...	accessory organs	compartment	0.00000	0.00000	0.46903
after 1	<user|>Other...	accessory organs	an accessory organ	**1.00000**	**0.70930**	**0.90820**
after 2	<user|>Other...	accessory organs	liver	0.00000	0.00000	0.39682
after 3	<user|>Other...	accessory organs	accessory organ	**1.00000**	**1.00000**	**0.94474**
after 4	<user|>Other...	accessory organs	attached organ	**0.33333**	**0.33610**	**0.74186**

Table 1: Examples of CQA tasks before and after mitigation. Green: Correct words that appear in the C-Q text. Yellow: Incorrect words that appear in the C-Q text but are not the correct answer. Orange: Incorrect words that do not appear in the C-Q text, representing other hallucinations. Red: Words appearing in the high-frequency and high-value groups, indicating knowledge-shortcut hallucinations. See Appendix A for detailed information

Knowledge shortcut arises because language models typically do not genuinely understand the intricate and complex factual knowledge but rather rely on shortcuts. They tend to over-rely on semantically proximate positions in the pre-training data, shared high-frequency words, and the quantity of related documents (Kandpal et al., 2023).This can introduce spurious correlation biases, causing the model to produce hallucinations even when working with correct and defect-free data sources.

We focus on factual hallucinations and introduce a Context-Question-Answer (CQA) task to analyze hallucinations caused by knowledge shortcuts, termed knowledge-shortcut hallucinations. In a CQA task, the correct answer typically resides in the context, and answers from large models that deviate from the correct answer are considered factual hallucinations. However, not all factual hallucinations are knowledge-shortcut hallucinations. When the model's answer is not in the context but is found in the high-similarity group of the CQ (shown in red in Figure 1), it is considered a knowledge-shortcut hallucination. In contrast, answers found in the context (like ***Answer2(yellow)***) or outside both the context and high-similarity group (like ***Answer4(orange)***) are called other hallucinations.

A common approach for mitigating data-related hallucinations is data filtering, including strictly controlling data source(Gao et al., 2020; (Gunasekar et al., 2023)) and deduplication. Deduplication which is divided into exact and near duplicates faces challenges. Exact duplicate detection is inefficient for large datasets (Manber and Myers, 1993), while near duplicate method like hash-based algorithm MinHash (Broder, 1997) prioritize speed but miss hidden information. Semantic duplicate recognition using pre-trained models (Abbas et al., 2023) is slower and impractical for large datasets. Thus, balancing granularity in duplicate detection and processing speed, while effectively reducing knowledge-shortcut hallucinations, remains a challenge.

This paper focuses on analyzing knowledge-shortcut hallucinations triggered by high-similarity texts from correct, defect-free data. We propose a *High Similarity Pruning Algorithm* that mitigates knowledge-shortcut hallucinations from a data perspective by leveraging semantic similarity, shared high-frequency words. Furthermore, based on these characteristics and incorporating model uncertainty(Xiao and Wang, 2021; Miao et al., 2023), we design a hybrid detection method tailored for CQA tasks to identify knowledge-shortcut hallucinations effectively. Our mitigation strategy demonstrates promising results across multiple LLMs and parameter scales. Notably, in the fine-tuning of nanoGPT-large, it successfully reduces knowledge-shortcut hallucinations by 6.5%.

Table 1 shows the same color-coding as in Figure 1. We present a real CQA example and compare responses before and after mitigation. Detailed results in appendix A show significant improvement across similarity metrics, a reduction in knowledge-shortcut hallucinations, and overall higher response quality.

Overall, the contributions of our paper can be summarized as follows:

- We investigate the mechanisms and patterns underlying knowledge-shortcut hallucinations driven by accurate and defect-free data. We identify their general characteristics and reveal their widespread presence in LLMs.

- We propose a novel detection method that combines semantic similarity and the uncertainty of LLM-generated outputs in CQA tasks. This method enables the quantitative evaluation of knowledge-shortcut hallucinations across different LLMs and training strategies (e.g., fine-tuning vs. training from scratch).

- To mitigate knowledge-shortcut hallucinations, we introduce a *Data High Similarity Pruning Algorithm* based on the identified generation mechanisms of such hallucinations. Quantitative evaluations demonstrate that this algorithm significantly improves the generation quality of LLMs and excels in suppressing hallucinations. The source code for our approach is available at github.

2 Methodology

2.1 Overview

The CQA task, characterized by simple answers and clear facts, is particularly well-suited for the study of knowledge-shortcut hallucinations. Through prior analysis, we found that such hallucinations arise from the misleading effect of high-frequency co-occurring and highly similar words in the training data (e.g., “9.11” being interpreted as a computer version number or directory, which is larger than “9.9”). The essence of our mitigation strategy is to filter high-frequency co-occurring and highly similar terms in the training data based on specific metrics (reducing or balancing the occurrences of "9.11" as a version number or directory). The focus of knowledge-shortcut hallucinations detection is to determine whether a factual error is caused by the misdirection of high-frequency and highly similar co-occurring entries in the training data, based on the background and the question (e.g., whether a comparison of computer version numbers or directories, like "9.11 is larger than 9.8," exists in the training data).

Building on these insights, we have refined our knowledge-shortcut mitigation strategy (Section 2.2) and hallucination detection approach (Section 2.3), with the overall framework shown in Figure 2.

We carefully selected three metrics for measuring text similarity: Jaccard similarity, TF-IDF similarity, and pre-trained model similarity. Through extensive engineering optimization, we ensured that our text similarity metrics not only maintain fine granularity but also enhance runtime efficiency, as detailed in Appendix B.1. Experimental results (Section 3, Appendix C) confirm that our mitigation strategy is simple to implement and highly effective, demonstrating the robustness of the strategy and validating its performance in knowledge-shortcut hallucination detection.

2.2 Data High Similarity Pruning

We define high-frequency co-occurring words and highly similar words as the high similarity group, with the specific concept outlined in Appendix B.2. Based on the definition of the high-similarity group, we designed the *High Similarity Pruning Algorithm* shown in Figure 2, which helps generative models reduce the occurrence of knowledge-shortcut hallucinations.

Given a batch of fine-tuning or training data from n different categories $(data_1, data_2, ..., data_n)$, we define the following steps for *data1*: with hyperparameters $(K_1, K_2, \alpha_1, \alpha_2)$, we compute the set $R_{1,j\in n}$ for deletion:

1) For each row in $data_1$, compute the top K_1 Jaccard and TF-IDF similarity values with the remaining $(n-1)$ datasets. Record the corresponding indices and values. **2)** Calculate the top K_2 most frequent indices (High-Frequency group, G_{HF}) and the top K_2 largest values (High-Value group, G_{HV}). **3)** Combine the four groups (*HF* and *HV* for both Jaccard and TF-IDF similarities), remove duplicates, and identify rows in $data_{j\in n}$ to delete, denoted as $R_{1,j\in n}$(Equation 1):

$$R_{1,j\in n} = Set(\alpha_1 G_{HF} + \alpha_2 G_{HV}) \qquad (1)$$

4) Iterate over all n datasets to compute the final set R_{all} for deletion across all datasets with Equation 2:

$$R_{all} = Set(\sum_{i=1}^{n}\sum_{j\neq i}^{n} R_{i,j}) \qquad (2)$$

2.3 Detection of Knowledge-Shortcut Hallucination

Detecting knowledge-shortcut hallucinations requires distinguishing them from other hallucinations. We focus on fact-based question-answering tasks with a CQA structure, where the correct answer is embedded in the context. To detect knowledge-shortcut hallucinations, we propose a method combining similarity features and self-check uncertainty measurement(Miao et al., 2023). The pseudocode is in Appendix B.3.

1) For a given context-question pair (CQ), we compute the most similar entry CQA_{ij} from the

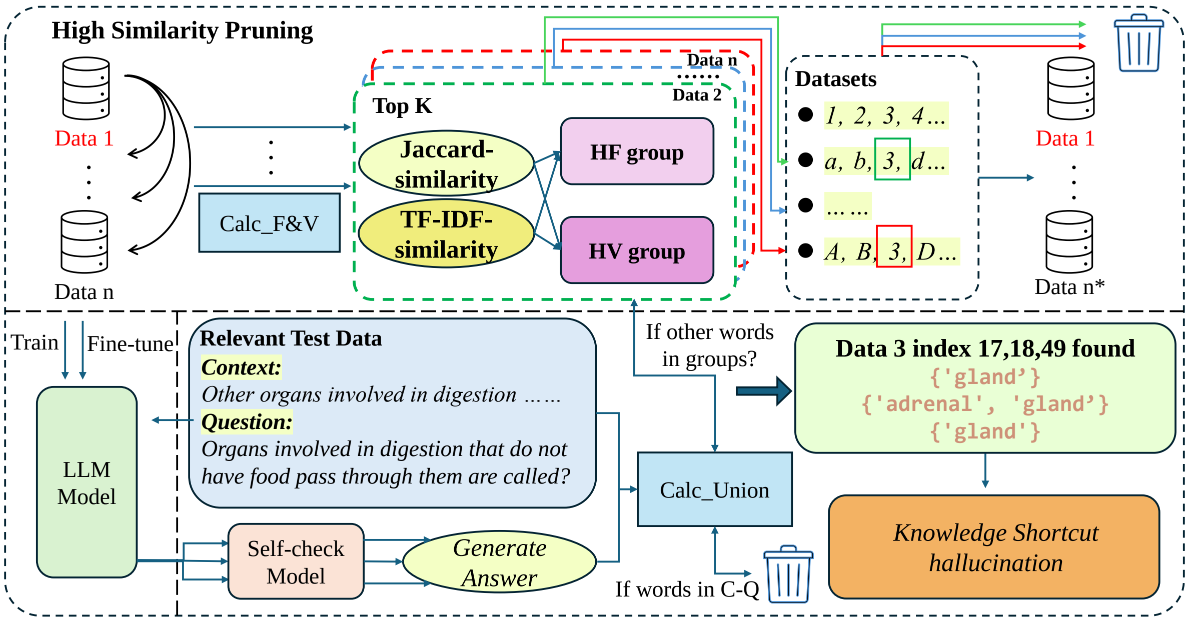


Figure 2: Overview of detection and mitigation.

datasets $(data_1, ..., data_n)$, where i denotes the dataset and j the row index. **2)** We calculate the Jaccard and TF-IDF similarity scores between this entry and others from $data_{k\neq i}$, identifying the high-frequency group (G_{HF}) and high-value group (G_{HV}). **3)** In the self-check module, the model generates an answer A_o. We then regenerate m responses $(A_1, \ldots, A_m)$ from the same input. If A_o significantly differs from $(A_1, \ldots, A_m)$, the response is flagged as a potential hallucination. The variation is quantified by Equation 3, where $m = 5$ and $\alpha_3 = 0.2$.

$$\frac{\sum_{l=1}^{m} 1 - Sim(A_o, A_l)}{m} \leq \alpha_3 \tag{3}$$

4) We compute the difference set S_o between A_o and CQ (Equation 4). If S_o is non-empty, the process continues:

$$S_o = Set(A_o) - Set(CQ) \tag{4}$$

5) Finally, we calculate the intersection between S_o and the high-frequency (G_{HF}) and high-value (G_{HV}) groups from CQA (Equation 5). If non-empty, we conclude that A_o is a knowledge-shortcut hallucination.

$$Set(A_o) \cap Set(CQA_{G_{HF},G_{HV}}) \tag{5}$$

2.4 Metrics of Effectiveness Evaluation

To evaluate the mitigation method, we design metrics that measure performance differences of the same model under identical parameter configurations, both before and after applying the method.

Coarse-grained metrics: 1) Number of Non-Zero and Non-Empty Similarity Rows: Count the rows where the similarity between generated and correct answers is non-zero and non-empty, before and after mitigation. 2) Average Similarity: Calculate the average similarity across the test set. These metrics offer a macroscopic view of the method's overall impact.

Fine-grained metrics: The fine-grained approach directly counts the number of knowledge-shortcut hallucinations in the test set. By comparing the numbers before and after applying the mitigation method, this metric offers a straightforward and clear evaluation of the method's effectiveness.

3 Experiments

3.1 Experiments Setting

3.1.1 Datasets

We selected four datasets with a CQA structure from the generative text datasets on Hugging Face as training or fine-tuning datasets $(data_1, data_2, data_3, data_4)$, along with a hallucination test dataset as the ablation test dataset. The four CQA datasets belong to different domains, forming a diverse training or fine-tuning datasets. In terms of data quantity selection, not all data from the four datasets were used. The large discrepan-

Dataset	Category	Number of rows
sciq	science	11679
financial-qa-10K	finance	7000
trivia-cqa	miscellaneous	14000
QASports	Basketball	14453

Table 2: General description of the CQA datasets

Dataset	Number of rows	Reduction magnitude
sciq	11679	0%
financial-qa-10K	6962	0.542%
trivia-cqa	13926	0.529%
QASports	14376	0.533%

Table 3: For sciq test, after mitigation

cies in the total volume of data across different datasets could make it difficult for the model to learn long-tail knowledge(Kandpal et al., 2023), thus negatively impacting the experimental results. Therefore, we selected portions of data from *trivia-cqa*[2] and *QASports*[3] that closely matches the sample sizes of *sciq*[4] and *financial-qa-10K*[5]. Details of this data selection can be found in Table 2.

For the test sets, we selected 600 samples from the *sciq* test dataset, a natural sciences dataset focused on objective facts, as the related test set. Additionally, we selected 513 samples from the *llm hallucination*[6] test dataset as an unrelated test set to evaluate the method's performance under different conditions.

3.1.2 Model Selection

We conducted our experiments on two generative models which is distributed under the MIT License. We used the model according to the terms specified in the license: nanoGPT[7] and TinyLlama[8]. For nanoGPT, we selected three parameter scales: gpt2-large (774M), gpt2-medium (350M), and gpt2 (124M), to perform fine-tuning and training experiments. For TinyLlama(Zhang et al., 2024), we conducted fine-tuning experiments using the LoRA(Hu et al., 2021) (Low-Rank Adaptation) method at the 1.1B parameter scale.

3.1.3 Implementation Details

Our experiments consist of three phases: assessment, mitigation, and detection.

Assessment: We used the *sciq* dataset ($data_1$) and progressively combined it with three additional datasets to form four training datasets. We trained three nanoGPT models (gpt2-large, gpt2-medium, gpt2) using both training and fine-tuning approaches, while the TinyLlama model was fine-tuned only. All models were evaluated on the *sciq test* using the CQA task. The results from the models trained on $sciq(data_1)$ alone served as the baseline for the related test set. We assessed the similarity changes when additional datasets were incorporated, analyzing both increases and decreases. Since there is no baseline for the unrelated test set, similarity changes are not evaluated for it.

Mitigation: We compared the performance of the models before and after applying the mitigation method, using consistent test sets. The mitigation parameters were set as: $(K_1, K_2, \alpha_1, \alpha_2) = (50, \text{lens} \times 0.006, 0.4, 0.1)$. $K_1 = 50$ corresponds to the Top-K parameter in nanoGPT, a key factor in hallucination generation. K_2 is related to dataset length, with a value of 0.006 to avoid excessive data removal. The value of 0.4 prioritizes high-frequency overlapping data in pruning. We chose $\alpha_1 + \alpha_2 = 0.5$ to balance the influence of High-Frequency (HF) and High-Value (HV) groups. The *High Similarity Pruning* increases data source independence, reducing semantic overlap between unrelated categories.

For instance, applying the *data high similarity pruning algorithm* to the *sciq test* yields the updated data quantities, as shown in Table 3.

Detection: We performed detection and metric evaluations for all models before and after mitigation on both test sets. To ensure result stability, we repeated experiments with different random seeds. The experiments were conducted on two NVIDIA 3090 GPUs (24 GB each). The source code is available on GitHub.

3.2 Experiment 1: Evaluation—-The Prevalence of Knowledge-Shortcut Hallucinations

We trained models on the four datasets described in Table 1 and conducted CQA tasks on a related test set comprising 600 samples. Using the results from training on $sciq(data_1)$ as the baseline, we evaluated the accuracy changes in model answers to

[2]https://huggingface.co/datasets/tilyupo/trivia_cqa
[3]https://huggingface.co/datasets/PedroCJardim/QASports
[4]https://huggingface.co/datasets/allenai/sciq
[5]https://huggingface.co/datasets/virattt/financial-qa-10K
[6]https://huggingface.co/datasets/C0uchP0tat0/llm_hallucinations
[7]https://github.com/karpathy/nanoGPT
[8]https://github.com/jzhang38/TinyLlama

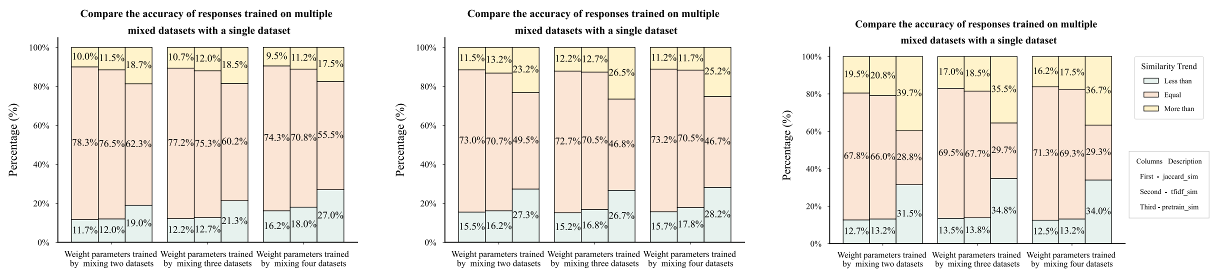


Figure 3: Evaluation of nanoGPT models under the fine-tuning method across three parameter scales: normal, medium and large(from left to right). Each x-axis represents a different similarity metric: Jaccard similarity, TF-IDF similarity, and Pre-trained model-based similarity, respectively.

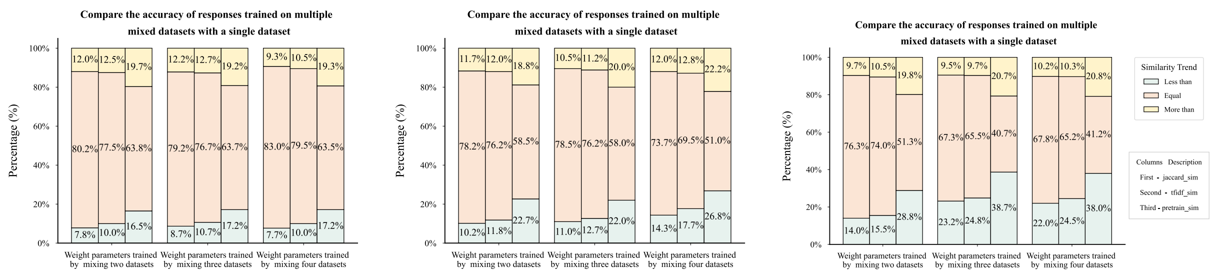


Figure 4: Evaluation of nanoGPT models under the training method across three parameter scales: normal, medium and large(from left to right). Each x-axis represents a different similarity metric: Jaccard similarity, TF-IDF similarity, and Pre-trained model-based similarity, respectively.

the same questions after mixing additional datasets into the training data.

Models trained on a single category of data and tested on the corresponding category's test set can fully demonstrate the model's performance. By progressively mixing other datasets into the training data and repeating the testing process, we revealed the widespread presence of knowledge-shortcut hallucinations from a macro perspective.

As shown in Figure 3, for models fine-tuned with different parameter scales, the similarity between generated answers and correct answers, measured by all three similarity metrics (Jaccard similarity, TF-IDF similarity, and pre-trained model similarity), decreased to varying extents as more datasets were mixed in. Compared to the baseline, models trained with multiple mixed datasets showed a higher proportion of "less" labels than "more" labels in their responses (The model trained on the *sciq* dataset serves as the baseline. If the responses generated by models trained with additional mixed datasets show an increase in any of the three similarity metrics, they are labeled as "more"; if the similarity decreases, they are labeled as "less"). Similar trends were observed in models trained using the full training method, as illustrated in Figure 4. The evaluation results and analysis after applying the mitigation strategy can be found in Appendix C.1.

This phenomenon is also observed in the fine-tuning of the *TinyLlama 1.1B* model, with the corresponding results presented in Appendix C.2.

Our evaluations demonstrate that knowledge-shortcut hallucinations are widespread across various datasets and models. This trend is consistently observed across different model architectures, parameter scales, and training methods, highlighting the significant impact of dataset composition on the accuracy and reliability of generative models.

3.3 Experiment 2: Mitigation–Effectiveness Before and After Applying the High Similarity Pruning Algorithm

Using the same training methods and parameters, we conducted CQA tasks on the models before and after applying the mitigation strategy. We first evaluate whether the 0.4% reduction in training data induced by the mitigation strategy would adversely affect model training performance. While the primary objective of this mitigation strategy is to reduce knowledge shortcut-induced model hallucinations, we aim to ensure that its implementation does not compromise model performance on CQA tasks. We employ two coarse-grained evaluation

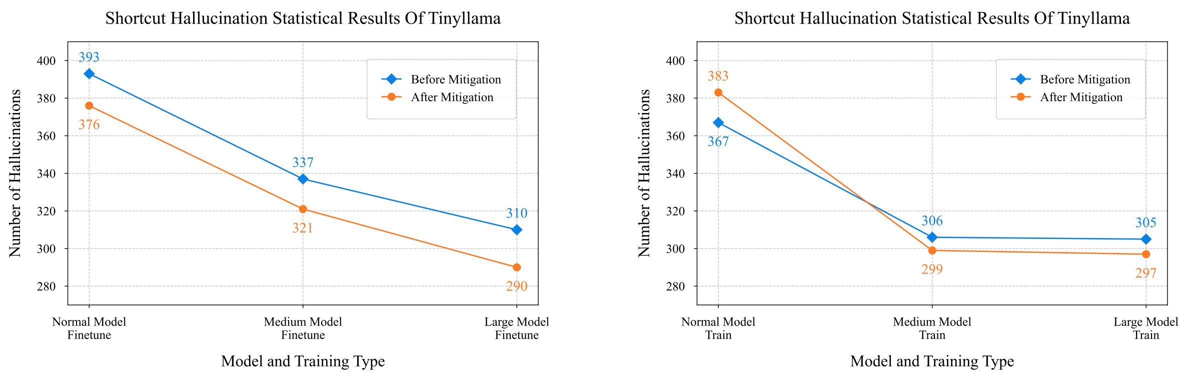


Figure 5: The number of Knowledge-Shortcut hallucination in CQA tasks before and after mitigation

Type	**Parameter**	**After mitigation**			**Before mitigation**		
		Jaccard	**TF-IDF**	**Pre-train**	**Jaccard**	**TF-IDF**	**Pre-train**
fine-tune	large	**1649**	**1656**	2400	1646	1652	2400
		0.62541	**0.61355**	**0.79622**	0.62188	0.61136	0.76381
	medium	**1388**	**1399**	2400	1368	1380	2400
		0.51048	**0.50020**	**0.69581**	0.50859	0.49707	0.69392
	normal	**956**	**979**	2400	952	975	2400
		0.32638	**0.32113**	**0.56459**	0.32576	0.31948	0.56337
train	large	**1852**	**1861**	2400	1838	1849	2400
		0.71533	**0.70426**	**0.83314**	0.71288	0.69956	0.82451
	medium	**1737**	**1745**	2400	1699	1709	2400
		0.66219	**0.65106**	**0.79507**	0.64394	0.63185	0.78482
	normal	1341	1352	2400	**1373**	**1385**	2400
		0.49678	0.48617	0.68270	**0.50845**	**0.49828**	**0.68748**

Table 4: Results with related test, mitigation and before mitigation for various nanoGPT parameters. The integer above each row represents the number of non-zero and non-empty result rows (out of a total of 2400), while the decimal below indicates the overall average similarity. Values in **bold** denote significant results.

metrics: the count of non-zero non-empty similarity rows and average similarity score. The strategy is considered effective if these metrics demonstrate comparable or slightly improved performance post-implementation.

As shown in Table 4, across nearly all training configurations and model scales of nanoGPT, the mitigation strategy generally yields marginal improvements in both metrics. An exceptional performance decline observed in the GPT-2 (124M) model under one specific training configuration will be analyzed in the section 6.

The reduction in training data directly leads to a decrease in training time, improving the efficiency of generative model training or fine-tuning. The reduced data did not impact testing on the *related* test set. In the other experiments presented in Table 5, we also explored the impact on the *unrelated* test set, where the fluctuations in both metrics remained minimal within the proportion of data reduction (For the unrelated test set, the train method is not meaningful, so we focus only on the fine-tune method). Furthermore, after applying the mitigation strategy, models trained on mixed datasets showed a decreasing trend in the proportion of "less" labels when compared to before mitigation in appendix C.1.

3.4 Experiment 3: Detection–Reduction of Knowledge-Shortcut hallucinations

To evaluate the effectiveness of the mitigation strategy from a finer-grained perspective, we employed the knowledge-shortcut hallucination fusion detection method. Specifically, we directly counted the number of knowledge-shortcut hallucinations in the test set generated by models of different param-

Type	Parameter	After mitigation			Before mitigation		
		Jaccard	TF-IDF	Pre-train	Jaccard	TF-IDF	Pre-train
fine-tune	large	1090	1126	2056	**1118**	**1160**	2056
		0.41782	0.37675	**0.64697**	**0.42160**	**0.37675**	0.64240
	medium	**976**	**1023**	2056	956	1006	2056
		0.36990	**0.33733**	0.60472	**0.37286**	0.33688	**0.60864**
	normal	**632**	**675**	2056	601	648	2056
		0.22673	**0.20910**	**0.49422**	0.21640	0.19913	0.48892

Table 5: Results with unrelated test, before and after mitigation. The integer above each row represents the number of non-zero and non-empty result rows (out of a total of 2056), while the decimal below indicates the overall average similarity. Values in **bold** denote significant results.

eter scales and training methods before and after applying the mitigation strategy. This straightforward approach provides a clear demonstration of the mitigation strategy's effectiveness.

Our detections show that in large-parameter fine-tuning models, the mitigation strategy performs exceptionally well in suppressing knowledge-shortcut hallucinations. Even for the training method, the strategy proved effective in reducing hallucinations, validating the method's efficacy in mitigating knowledge-shortcut hallucinations in generative models. The results are presented in Figure 5.

We provide a reproducible set of repeated experimental results in Appendix C.3. The training and generation code has been open-sourced on GitHub. The results from this set of experiments align well with those presented in the main text, indirectly validating the robustness of our method.

4 Related Works

Significant progress has been made in the study of hallucinations in LLMs. Xu(Xu et al., 2024) argue that eliminating hallucinations in LLMs is impossible. Numerous techniques have emerged to mitigate LLM hallucinations, with notable approaches including Retrieval-Augmented Generation (RAG)(Lewis et al., 2020), knowledge retrieval(Varshney et al., 2023), CoNLI(Lei et al., 2023), and CoVe(Dhuliawala et al., 2023). Our aim is to clarify the underlying causes of knowledge-shortcut hallucinations and minimize such hallucinations as much as possible.

Shortcut learning is a critical area of research in LLM hallucinations, with knowledge shortcuts representing the manifestation of shortcut learning at the data level. Geirhos(Geirhos et al., 2020) and Du(Du et al., 2023) suggest that dataset bias is the starting point of shortcut learning, and many excellent works have focused on alleviating shortcut learning from the data perspective, such as identifying biased sentencesLei(Lei et al., 2022), data shortcuts(Friedman et al., 2022), or replacing datasets with more balanced ones(Tang et al., 2023). Tang's research(Tang et al., 2021) indicates that fine-tuned language models can learn and even amplify biases present in the training datasets, leading to poor performance in downstream tasks, which aligns with our experimental findings. Despite these advancements, existing works have yet to fully explore all the ways in which dataset bias can manifest. Our goal is to conduct an in-depth study of one type of hallucination triggered by correct, defect-free data sources in large models and propose a feasible method for mitigating and predicting such illusions.

5 Conclusion

In conclusion, we have conducted a finer-grained study of a specific type of hallucination originating from the data perspective and proposed a novel method for mitigating this hallucination, along with a fusion detection method for such hallucinations. Our approach demonstrates through experiments that, when handling specific question-answering tasks, it can significantly reduce knowledge-shortcut hallucinations in the fine-tuning process while maintaining the performance of generative models and stabilizing answer similarity. This provides a new paradigm for addressing hallucinations in generative models.

6 Limitations

Normal parameter scale results. In the experiment shown in Figure 5, the mitigation effect on the nanoGPT (124M) model was relatively poor. This phenomenon persisted in repeated experiments (Figure 11), suggesting a potential explanation. Given the small parameter scale and limited training data, the model may struggle to learn the patterns of knowledge shortcuts effectively. As a result, applying the mitigation strategy does not yield significant improvements. This observation indicates that our mitigation approach is better suited for larger-scale models, aligning with the experimental results observed in large parameter models.

Runtime and applicable tasks. Our mitigation strategy demonstrated outstanding performance in fine-tuning tasks, with a stable and significant reduction in knowledge-shortcut hallucinations. This suggests that the strategy is more suitable for fine-tuning rather than pretraining tasks. From a data scale perspective, pretraining datasets are typically vast, whereas fine-tuning datasets are relatively smaller. As a result, the computational overhead introduced by our mitigation strategy is entirely acceptable in fine-tuning scenarios.

Detection method applicability. Unlike the general mitigation strategies at the data preprocessing level, our knowledge shortcut hallucination detection method is specifically designed for CQA tasks and is data-dependent. As such, we did not assess the superiority of this method; rather, it serves as an evaluation technique for the number of knowledge shortcut hallucinations before and after applying our mitigation strategy. Due to its data dependency, this detection method is also applicable to fine-tuning tasks.

Chain-of-Thought technology. We have also tested floating-point comparison issues on the latest commercial large models utilizing "chain-of-thought" (CoT) reasoning, such as ChatGPT o1 and DeepSeek R1. Although they ultimately provided the correct answers, doubts arose during the reasoning process. ChatGPT o1, for instance, initially gave an incorrect answer but corrected itself in the subsequent reasoning steps. Therefore, CoT technology represents a potential approach for mitigating all factual hallucinations. However, the phenomenon of knowledge shortcuts may not directly affect the final outcome, yet it still misleads the model's reasoning process.

7 Ethics Statement

This research does not involve human subjects, private data, or personally identifiable information. All datasets used in our experiments are publicly available and were collected from open-access sources such as HuggingFace. Our mitigation and detection techniques target hallucinations in large language models caused by spurious correlations in the data, aiming to improve model reliability and reduce potential harms due to misinformation. We foresee no significant ethical risks associated with this work. Nevertheless, we acknowledge that detecting hallucinations is an ongoing challenge, and further work is needed to ensure fairness and robustness in broader real-world deployments.

8 Acknowledgements

References

Amro Kamal Mohamed Abbas, Kushal Tirumala, Daniel Simig, Surya Ganguli, and Ari S Morcos. 2023. Semdedup: Data-efficient learning at web-scale through semantic deduplication. In *ICLR 2023 Workshop on Multimodal Representation Learning: Perks and Pitfalls*.

A Broder. 1997. On the resemblance and containment of documents. In *Proceedings of the Compression and Complexity of Sequences 1997*, page 21.

Shehzaad Dhuliawala, Mojtaba Komeili, Jing Xu, Roberta Raileanu, Xian Li, Asli Celikyilmaz, and Jason Weston. 2023. Chain-of-verification reduces hallucination in large language models. *arXiv preprint arXiv:2309.11495*.

Mengnan Du, Fengxiang He, Na Zou, Dacheng Tao, and Xia Hu. 2023. Shortcut learning of large language models in natural language understanding. *Communications of the ACM*, 67.

Nouha Dziri, Andrea Madotto, Osmar Zaïane, and Avishek Joey Bose. 2021. *Neural Path Hunter: Reducing Hallucination in Dialogue Systems via Path Grounding*. Association for Computational Linguistics, Online and Punta Cana, Dominican Republic.

Dan Friedman, Alexander Wettig, and Danqi Chen. 2022. Finding dataset shortcuts with grammar induction. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 4345–4363.

Leo Gao, Stella Biderman, Sid Black, Laurence Golding, Travis Hoppe, Charles Foster, Jason Phang, Horace He, Anish Thite, Noa Nabeshima, et al. 2020. The pile: An 800gb dataset of diverse text for language modeling. *arXiv preprint arXiv:2101.00027*.

Robert Geirhos, Jörn-Henrik Jacobsen, Claudio Michaelis, Richard Zemel, Wieland Brendel, Matthias Bethge, and Felix A Wichmann. 2020. Shortcut learning in deep neural networks. *Nature Machine Intelligence*, 2.

Suriya Gunasekar, Yi Zhang, Jyoti Aneja, Caio César Teodoro Mendes, Allie Del Giorno, Sivakanth Gopi, Mojan Javaheripi, Piero Kauffmann, Gustavo de Rosa, Olli Saarikivi, et al. 2023. Textbooks are all you need. *arXiv preprint arXiv:2306.11644*.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*.

Lei Huang, Weijiang Yu, Weitao Ma, Weihong Zhong, Zhangyin Feng, Haotian Wang, Qianglong Chen, Weihua Peng, Xiaocheng Feng, Bing Qin, and Ting Liu. 2024. *A Survey on Hallucination in Large Language Models: Principles, Taxonomy, Challenges, and Open Questions*. Association for Computing Machinery, New York, NY, USA.

Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Ye Jin Bang, Andrea Madotto, and Pascale Fung. 2023. Survey of hallucination in natural language generation. *ACM computing surveys*, 55.

Tianjie Ju, Yijin Chen, Xinwei Yuan, Zhuosheng Zhang, Wei Du, Yubin Zheng, and Gongshen Liu. 2024. Investigating multi-hop factual shortcuts in knowledge editing of large language models. *arXiv preprint arXiv:2402.11900*.

Nikhil Kandpal, Haikang Deng, Adam Roberts, Eric Wallace, and Colin Raffel. 2023. *Large language models struggle to learn long-tail knowledge*. ICML'23. JMLR.org.

Deren Lei, Yaxi Li, Mengya Hu, Mingyu Wang, and Xi Yun. 2023. Chain of natural language inference for reducing large language model hallucinations. In *NeurIPS 2023 Workshop on Instruction Tuning and Instruction Following*.

Yuanyuan Lei, Ruihong Huang, Lu Wang, and Nick Beauchamp. 2022. Sentence-level media bias analysis informed by discourse structures. In *Proceedings of the 2022 conference on empirical methods in natural language processing*, pages 10040–10050.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems*, 33:9459–9474.

Shaobo Li, Xiaoguang Li, Lifeng Shang, Zhenhua Dong, Cheng-Jie Sun, Bingquan Liu, Zhenzhou Ji, Xin Jiang, and Qun Liu. 2022. How pre-trained language models capture factual knowledge? a causal-inspired analysis. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 1720–1732.

Udi Manber and Gene Myers. 1993. Suffix arrays: a new method for on-line string searches. *siam Journal on Computing*, 22(5):935–948.

Ning Miao, Yee Whye Teh, and Tom Rainforth. 2023. Selfcheck: Using llms to zero-shot check their own step-by-step reasoning. *arXiv preprint arXiv:2308.00436*.

Andrianos Michail, Stefanos Konstantinou, and Simon Clematide. 2023. Uzh_clyp at semeval-2023 task 9: Head-first fine-tuning and chatgpt data generation for cross-lingual learning in tweet intimacy prediction. In *The 61st Annual Meeting Of The Association For Computational Linguistics*.

Chengwei Qin, Aston Zhang, Zhuosheng Zhang, Jiaao Chen, Michihiro Yasunaga, and Diyi Yang. 2023. Is chatgpt a general-purpose natural language processing task solver? In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 1339–1384.

Aryan Rangapur and Aman Rangapur. 2024. The battle of llms: A comparative study in conversational qa tasks. *arXiv preprint arXiv:2405.18344*.

Ruixiang Tang, Mengnan Du, Yuening Li, Zirui Liu, Na Zou, and Xia Hu. 2021. Mitigating gender bias in captioning systems. In *2021 World Wide Web Conference, WWW 2021*, pages 633–645. Association for Computing Machinery, Inc.

Ruixiang Tang, Dehan Kong, Longtao Huang, and Hui Xue. 2023. Large language models can be lazy learners: Analyze shortcuts in in-context learning. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 4645–4657.

Neeraj Varshney, Wenlin Yao, Hongming Zhang, Jianshu Chen, and Dong Yu. 2023. A stitch in time saves nine: Detecting and mitigating hallucinations of llms by validating low-confidence generation. *arXiv preprint arXiv:2307.03987*.

Yijun Xiao and William Yang Wang. 2021. On hallucination and predictive uncertainty in conditional language generation. *arXiv preprint arXiv:2103.15025*.

Ziwei Xu, Sanjay Jain, and Mohan Kankanhalli. 2024. Hallucination is inevitable: An innate limitation of large language models. *arXiv preprint arXiv:2401.11817*.

Peiyuan Zhang, Guangtao Zeng, Tianduo Wang, and Wei Lu. 2024. Tinyllama: An open-source small language model. *arXiv preprint arXiv:2401.02385*.

A Example

In Table 1, the word marked in red in *Answer3* is identified because it repeatedly appears in the high

Model Name	Answer
Chatgpt-4	×
Chatgpt-4o	×
Gemini Advanced	×
Claude	×
Kimi	×
Cici	×
ERNIE Bot	✓

Table 6: Large Model Floating Point Comparison

similarity group between the test input (CQ) and other datasets. This is further detailed in Table 9. The first three rows of the table indicate that, in the top 50 rows with the highest Jaccard similarity to this CQ from the *trivia(data3)* dataset, words ***{gland}***, ***{adrenal, gland}***, and ***{gland}*** were found in rows 17, 18, and 49 (with indices 1326, 11791, and 1303 in the data3 dataset, respectively). Therefore, we define the incorrect answer containing the red-marked word as a knowledge-shortcut hallucination.

B Methodology

B.1 Metrics of Text Similarity

This paper aims to develop a framework for detecting and mitigating knowledge-shortcut hallucinations and to validate the effectiveness of the proposed method. From the perspective of data, the root cause of knowledge-shortcut hallucinations is intuitively reflected in the presence of high textual similarity within the training data. To quantify textual similarity, we employ three mathematical approaches:

Jaccard similarity. Jaccard similarity is the most straightforward measure of the overlap between two sets. It calculates the ratio of the intersection to the union of the sets, providing a naive yet effective way to assess similarity. The numerator denotes the intersection of sets A and B, and the denominator denotes the union of sets A and B.

$$Jaccard_{sim} = \frac{|A \bigcap B|}{|A \bigcup B|} \tag{6}$$

TF-IDF similarity. TF-IDF(Term Frequency-Inverse Document Frequency) similarity leverages statistical measures of word importance across documents, enabling a more nuanced comparison that considers term frequency and discriminative power. The t in the formula denotes the word and d denotes the document. So we can get $TF-IDF_{sim}$

$$TF-IDF_{sim}(t,d) = TF(t,d) \cdot IDF(t) \tag{7}$$

$$TF(t,d) = \frac{count(t,d)}{\sum_{k \in d} count(k,d)} \tag{8}$$

$$IDF(t) = log\frac{N}{1+DF(t)} \tag{9}$$

Pre-trained model-based similarity. A Pre-trained model-based similarity measure uses pre-trained language models to compute semantic similarity between text pairs. This method captures contextual and latent relationships in text, providing a more sophisticated and accurate measure compared to traditional approaches. Finally, we choose the paraphrase-miniLM-v12-v2[9] of the sentence-transformers library, because it performs fast, accurate sentence similarity evaluation.

We utilize the above three metrics to measure the similarity between the generated responses and the correct answers, leveraging their respective advantages. Jaccard similarity provides a simple and intuitive method for quickly assessing the magnitude of similarity between two texts. TF-IDF similarity incorporates the influence of term frequency, reducing the impact of high-frequency words on sentence similarity. Pre-trained model-based similarity evaluates the similarity from a semantic perspective, offering fine-grained corrections for discrepancies, such as those between *"4"* and *"four."* By combining these three metrics, we achieve a multidimensional evaluation of sentence similarity.

We also performed several engineering optimizations in the code. Given that the generated responses in the CQA task are relatively short, variations in singular and plural forms of nouns, verb conjugations, and adjective-adverb transformations could significantly impact Jaccard similarity and TF-IDF similarity. To address this, we utilized the *nltk*[10] library to implement a lemmatization method, improving the accuracy of Jaccard and TF-IDF similarity measurements. This refinement enhances the granularity of overall text similarity evaluation metrics.

B.2 The Defination of High Similarity Group

Datasets used in pretraining or fine-tuning often consist of multiple semantically distinct sources,

[9]https://huggingface.co/sentence-transformers/paraphrase-MiniLM-L12-v2

[10]https://www.nltk.org/

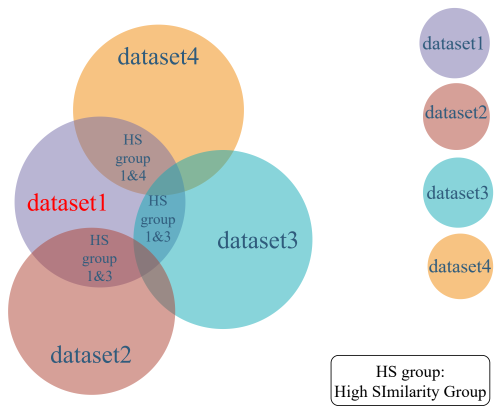


Figure 6: An example of data high similarity

such as mathematics, art, and agriculture. Our focus is on subsets of high textual similarity within these sources. As shown in Figure 6, for example, high similarity between *data1* and subsets of *data2*, *data3*, and *data4* forms what we call *High Similarity Group* (HS group), represented as *HS group 1&2*, *HS group 1&3*, and *HS group 1&4*. These high similarity texts can convey different meanings, misleading generative models during training or fine-tuning. This increases the risk of knowledge-shortcut hallucinations when queries are input.

B.3 Pseudocode of Mitigation and Detection

Algorithm 1 Mitigation: High Similarity Pruning Algorithm

```
Input: $(data_1, ..., data_n)$; $K_1, K_2, \alpha_1, \alpha_2$
Output: Mitigation strategy
1: for $k = 1$ to $n$ do
2:     for $i = 0$ to row($data_k$) do
3:         for $j \neq k$ to $n$ do
               $T_{i,j}(CQA_{G_{HF},G_{HV}}) \leftarrow K_1, data_j$;
               $G_{j,HF}, G_{j,HV} \leftarrow T_{i,j}(CQA_{G_{HF},G_{HV}})$;
4:         end for
           $R_{k,j} \leftarrow Set(\alpha_1 G_{j,HF} + \alpha_2 G_{j,HV})$;
5:     end for
       $R_{all} \leftarrow Set(R_{k,j}), K_2$;
6: end for
       $(data_1, ...data_n)' \leftarrow (data_1, data_n) - R_{all}$;
```

Algorithm 2 Knowledge Shortcut Hallucination Detection

```
Input: Context-question pair $(CQ)$; Similarity threshold $\alpha_3$; $m = 5$
Output: Detection of knowledge-shortcut hallucination
1: for $j = 0$ to $len(CQ)$ do
2:     $T(CQA_{ij}) \leftarrow (data_1, ...data_n)$;
3:     $G_{HF}, G_{HV} \leftarrow CQA_{ij}$;
4:     for $l = 1$ to $m$ do
5:         Generate response $A_l$;
6:         if $\frac{\sum_{l=1}^{m} 1 - Sim(A_o, A_l)}{m} > \alpha_3$ then
           $S_o = Set(A_o) - Set(CQ)$;
7:             if $S_o$ is not empty then
8:                 $flag \leftarrow Set(A_o)$
                   $\cap Set(CQA_{G_{HF},G_{HV}})$;
9:                 if $flag \neq \emptyset$ then
10:                    Return True;
11:                else Return False;
12:                end if
13:            else Return False;
14:            end if
15:        else Return False;
16:        end if
17:    end for
18: end for
```

C Results of the Experiment

C.1 Evaluation after applying mitigation strategies

Here, we show the evaluation results of different parameter-scale models and training methods on the same test set after applying the mitigation strategy. Figures 7 and 8 show the evaluation results after applying the mitigation strategy, corresponding to Figures 3 and 4. A comparison reveals that the proportion of *less* labels has generally decreased, while the proportion of *more* labels has increased, leading to a more balanced and coordinated distribution. These findings indicate that our strategy effectively reduces the influence of unrelated datasets on the model's generated answers, thereby improving output quality.

It is worth noting that the normal parameter scale of nanoGPT is only 124M, resulting in greater fluctuations in the experiments. Compared to larger parameter models, this introduces some instability, leading to deviations in certain results.

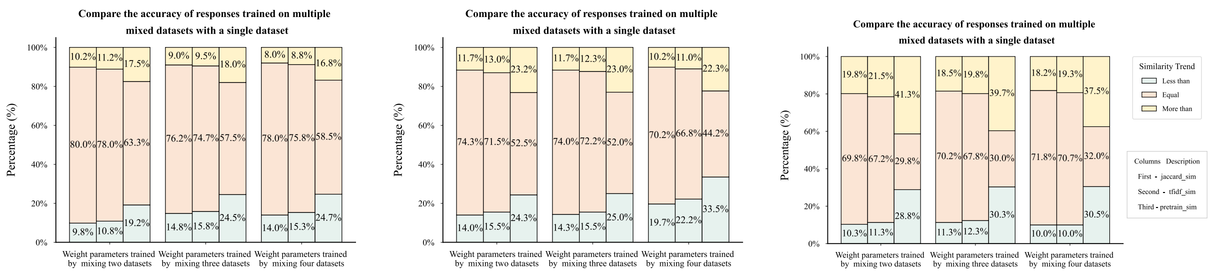


Figure 7: After mitigation, fine-tuning: nanoGPT large, medium, normal

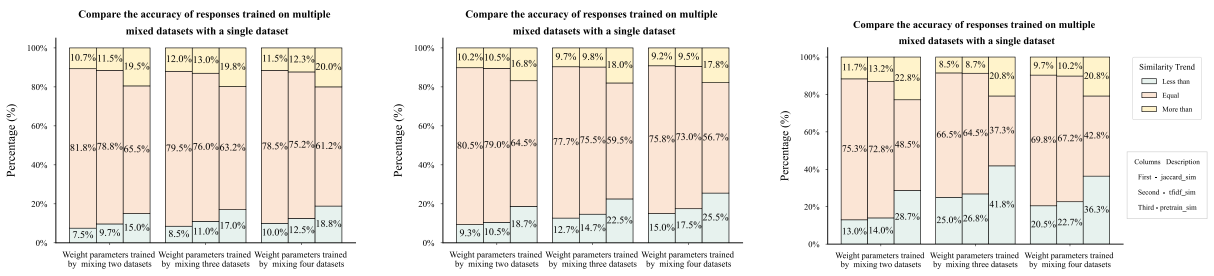


Figure 8: After mitigation, train : nanoGPT large, medium, normal

C.2 Experiment Results of Tinyllama

To further investigate the generalization effectiveness of our proposed method, we conducted the same evaluation, mitigation, and prediction experiments using the TinyLlama model, employing only the fine-tuning approach. The results and trends remained consistent with those observed in nanoGPT, demonstrating the robustness and applicability of our method across different model architectures.

Figure 10 presents the evaluation results of TinyLlama before and after applying the mitigation strategy, while Table 7 summarizes the macro-level performance metrics of TinyLlama's generated responses under both conditions. Furthermore, in TinyLlama's response generation process, we set the sampling parameter Top-k to 5. To ensure consistency, we maintained the hallucination detection parameter K_1 equal to *Top-k* and conducted an ablation study to explore the impact of different K_1 values on the detection of knowledge-shortcut hallucinations. As shown in Figure 9, after applying the mitigation strategy, the number of detected knowledge-shortcut hallucinations remained consistently lower than before across all K_1 values, with minimal variation in detection differences.

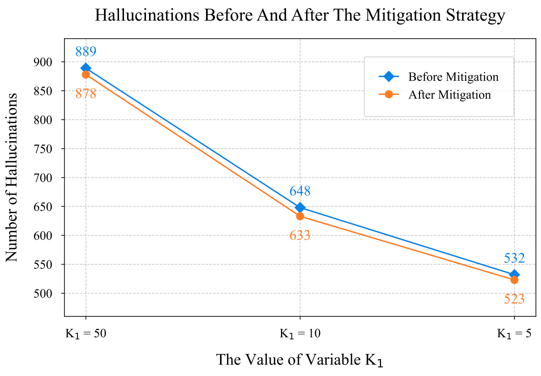
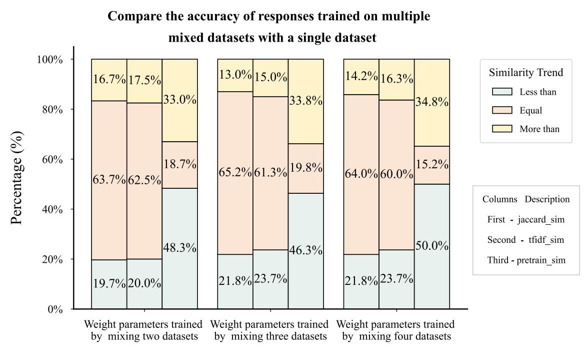


Figure 9: Effectiveness of Different K_1 Values on Tinyllama Knowledge-Shortcut Hallucination Detection

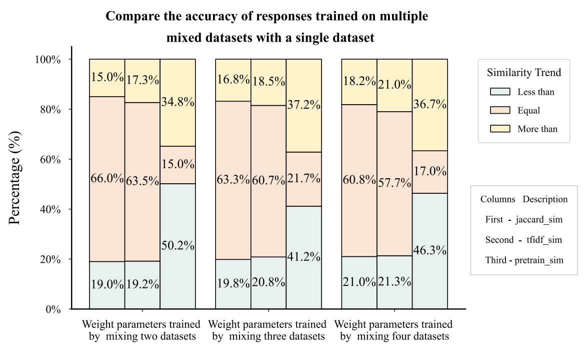
C.3 Reproducibility of Experimental Results

To minimize the impact of randomness on our experimental results, we conducted multiple repeated experiments and have open-sourced all code and results. The full reproducible experiments can be accessed and executed to obtain the exact experimental outcomes.

For these repeated experiments, we primarily focused on nanoGPT. We retrained, generated responses, and conducted testing to obtain a comprehensive set of experimental results, which are presented below.



(a) Evaluation of Tinyllama, before mitigation



(b) Evaluation of Tinyllama, after mitigation

Figure 10: (a) Before mitigation, (b) After mitigation. Illustration of the evaluation results with mitigation strategies.

Type	**Parameter**	**After mitigation**			**Before mitigation**		
		Jaccard	**TF-IDF**	**Pre-train**	**Jaccard**	**TF-IDF**	**Pre-train**
fine-tune	1.1b	**1273**	**1321**	2400	1243	1280	**2400**
		0.39285	**0.39105**	**0.68855**	0.38596	0.38347	0.68374

Table 7: Results with mitigation and before mitigation for Tinyllama. Values in **bold** denote significant results.

Type	**Parameter**	**After mitigation**			**Before mitigation**		
		Jaccard	**TF-IDF**	**Pre-train**	**Jaccard**	**TF-IDF**	**Pre-train**
fine-tune	large	**1638**	**1647**	2400	1612	1618	2400
		0.61965	**0.60919**	**0.76633**	0.61193	0.60121	0.76393
	medium	1380	1390	2400	**1388**	**1399**	2400
		0.50376	0.49043	0.69370	**0.51097**	**0.50020**	**0.69581**
	normal	949	967	2400	**995**	**1012**	2400
		0.32107	0.31457	0.55900	**0.34018**	**0.33170**	**0.57477**
train	large	**1858**	**1870**	**2400**	1844	1853	2398
		0.71944	**0.70863**	**0.83367**	0.71422	0.70078	0.82887
	medium	**1742**	**1755**	2400	1716	1725	2400
		0.66459	**0.65435**	**0.79688**	0.65474	0.64318	0.79082
	normal	**1370**	**1382**	2400	1331	1346	2400
		0.50535	**0.49617**	**0.68918**	0.49451	0.48467	0.67837

Table 8: Results with mitigation and before mitigation for various nanoGPT parameters. Values in **bold** denote significant results.

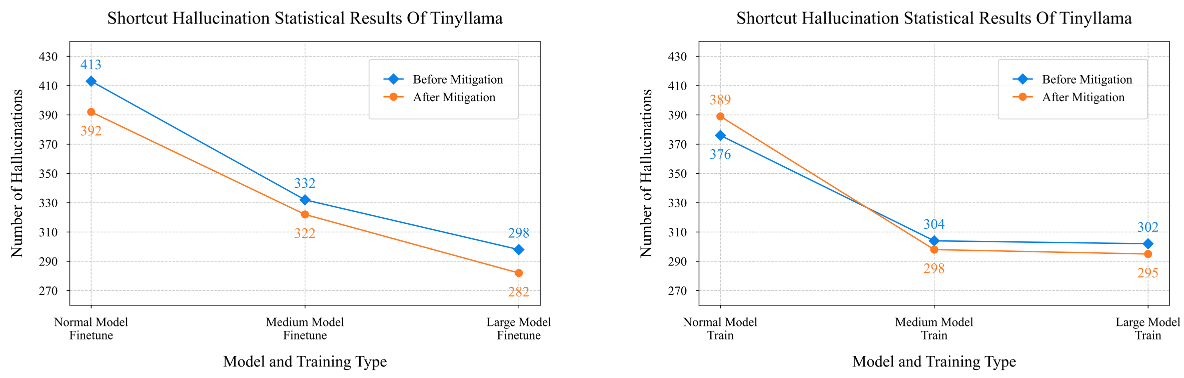


Figure 11: The number of Knowledge-Shortcut hallucination in CQA tasks before and after mitigation in new experiments

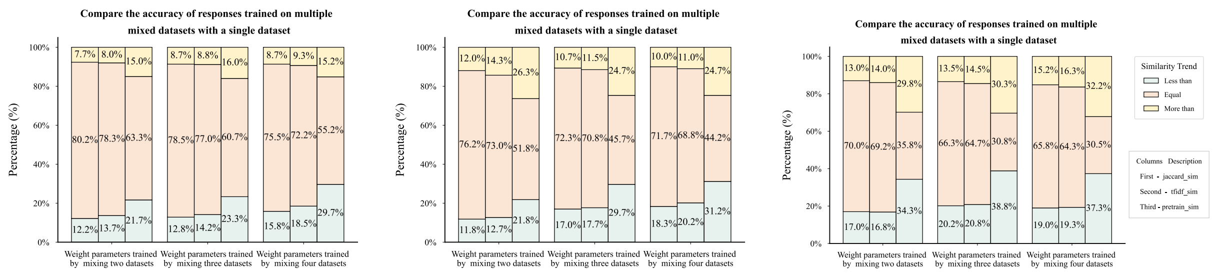


Figure 12: Reproduce experiments, before mitigation, fine-tuning: nanoGPT large, medium, normal

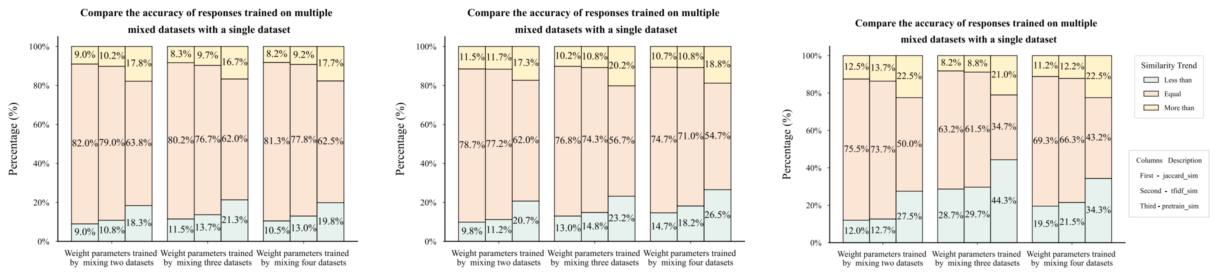


Figure 13: Reproduce experiments, before mitigation, train: nanoGPT large, medium, normal

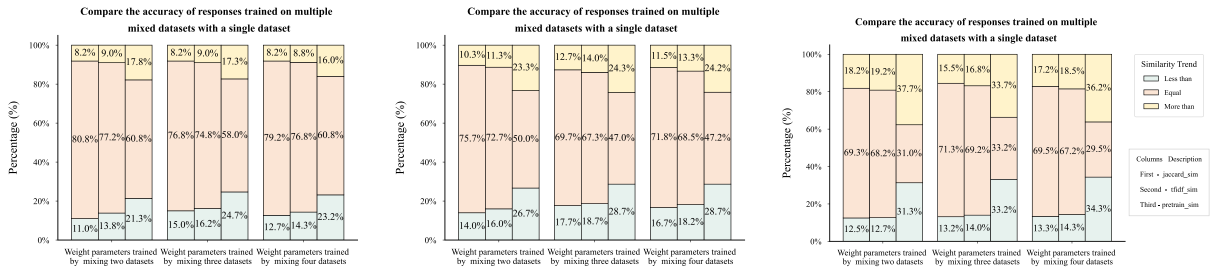


Figure 14: Reproduce experiments, after mitigation, fine-tuning: nanoGPT large, medium, normal

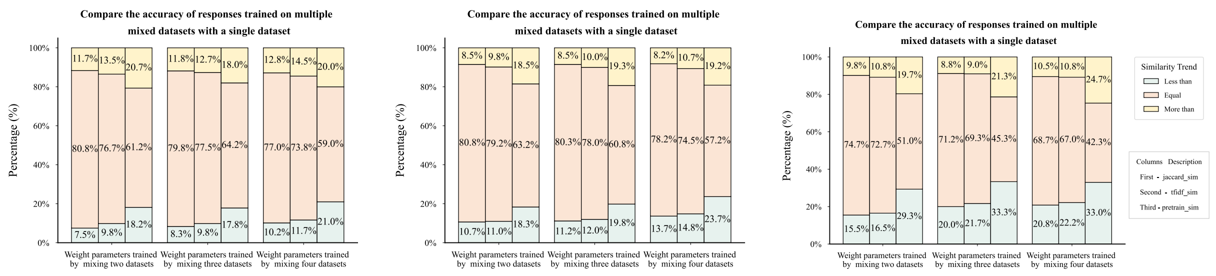


Figure 15: Reproduce experiments, after mitigation, train: nanoGPT large, medium, normal

Type	HS Index	Dataset	Row Index	Context
Jaccard	17	3	1326	Insulin is a hormone made up of a small polypeptide protein that is secreted by the pancreas, which acts as both an endocrine and exocrine gland. Endocrine glands are the system of glands that secrete hormones to regulate body functions. Exocrine glands aid in digestion.
Jaccard	18	3	11791	Epinephrine (ep-uh-nef-rin, -reen) is also known as adrenaline. It is a hormone that is secreted by the adrenal glands.
Jaccard	49	3	1303	The thyroid gland is one of the body's most important endocrine organs......
TF-IDF	45	3	11235	Pinnipeds have streamlined, spindle-shaped bodies with reduced or non-existent external ear flaps, rounded heads, flexible necks, limbs modified into flippers, and small tails...... The mammary glands and genitals of pinnipeds can retract into the body.

Table 9: Examples of high-frequency co-occurring words found in the high similarity group