Leveraging Knowledge Graphs to Mitigate Counting Hallucinations: A Case Study of Wikidata

Fariz Darari[1], Jaycent G. Ongris[1], Berty C. L. Tobing[2], Douglas R. Faisal[2] and On Lee[2]

[1]*Faculty of Computer Science, Universitas Indonesia, Depok 16424, Indonesia*
[2]*GDP Labs, Jakarta 12950, Indonesia*

Abstract

When answering questions, language models (LMs) often ground their responses in unstructured textual sources. However, structured sources such as knowledge graphs (KGs) often contain valuable counting or cardinality facts, e.g., the number of children a person has, the number of seasons in a TV series, or the number of branches a company operates. Leveraging KGs can help LMs reduce hallucinations in count-based queries, such as "How many children does X have?" or "How many branches does company Y have?". This work introduces the problem of counting hallucinations and proposes a novel LM-based QA approach that integrates structured counting knowledge from KGs like Wikidata to address these shortcomings. We also introduce the first benchmark dataset for counting QA, comprising over 10,000 entries with more than 30,000 counting questions. Through our experiments, we compare QA accuracy in various scenarios: using no structured counting knowledge at all and using our KGQA methods without vs. with perfect entity extraction. We also examine how performance differs between a smaller language model and a larger, more advanced model. The results on Wikidata show that incorporating structured counting knowledge leads to a substantial improvement in accuracy, with more than a 60% gain even without perfect entity extraction. This highlights the effectiveness and promise of our approach for advancing future KGQA research.

1. Introduction

The use of language models (LMs) has skyrocketed in recent years, powering applications across a wide range of domains such as travel, journalism, code writing, data analysis, and even agriculture [1]. This widespread adoption is underscored by the fact that AI agents and bots accounted for over 51% of web traffic in 2024, surpassing human activity [2]. This surge in bot traffic, largely driven by AI agents crawling websites to gather data, underscores the growing dependence on LMs for fast and accurate responses.

However, a persistent issue with LMs is hallucination, that is, the tendency to generate plausible-sounding but factually incorrect information [3]. This becomes especially problematic when users seek accurate and verifiable answers. To mitigate this, LMs should be grounded in textual sources such as internal documents and web content. Moreover, LMs can rely on structured sources, namely knowledge graphs (KGs), to ground model outputs in curated facts

Wikidata'25: Wikidata workshop at ISWC 2025
fariz@ui.ac.id (F. Darari); jaycent.gunawan@ui.ac.id (J. G. Ongris); berty.c.l.tobing@gdplabs.id (B. C. L. Tobing); douglas.r.faisal@gdplabs.id (D. R. Faisal); onlee@gdplabs.id (O. Lee)
0000-0001-6025-609X (F. Darari)

CEUR Workshop Proceedings ceur-ws.org ISSN 1613-0073
published 2025-11-25

and ontologies [4]. As reliance on LMs grows, grounding them in structured, verifiable sources such as KGs becomes increasingly important to ensure factual consistency and trustworthiness.

Among the many question types that challenge LMs, counting questions, which ask about the number of entities associated with a subject, are both frequent and error-prone. These questions usually begin with "How many..." and do not involve explicit units, relying instead on context (e.g., "How many branches does company X have?"). KGs, particularly those like Wikidata, contain structured representations of such cardinality or counting facts, often encoded through properties like "number of children" or "number of branches." In fact, Wikidata includes nearly 90 properties prefixed with "number of..." and further analysis reveals that Wikidata incorporates over 160 properties related to cardinalities[1].

This paper is motivated by the readily available counting knowledge within Wikidata[2], one of the largest and most widely-used KGs. The central question we address is how to effectively utilize this resource to answer natural language questions concerning counts, which are often susceptible to hallucination. Specifically, we aim to mitigate the problem of counting hallucinations, that is, the generation of inaccurate numerical answers to counting questions, by leveraging KGs. To the best of our knowledge, this issue has not been systematically investigated within the framework of LM-based knowledge graph question answering (KGQA).

To illustrate a case of counting hallucination, we conducted a preliminary experiment using the language model of Gemma 3 4B[3]. When asked about the number of children of Neneh Cherry (a Swedish singer-songwriter), Gemma incorrectly stated that she has four children, even listing names of (fictional) children and their partners (e.g., Lyric Vanessa Cherry with partner Eric Salmon). We then posed the same question to a more advanced model, Claude Sonnet 4, and obtained the same incorrect answer (i.e., four children). However, upon verification, we found that Neneh Cherry actually has three children. Notably, the correct count of her children is readily available on her Wikidata page (https://www.wikidata.org/wiki/Q233342). We argue that this information, along with a wealth of other counting knowledge present in Wikidata, can be further leveraged to enhance KGQA.

This paper directly addresses this issue. Our contributions are:

1. We introduce the problem of counting hallucinations in LM responses.
2. We investigate how counting facts are represented in a KG, using Wikidata as a case study.
3. We propose methods for extracting and integrating counting knowledge into KGQA systems.
4. We construct the first benchmark dataset specifically tailored for counting question answering.
5. We evaluate our approaches on a large-scale KG (that is, Wikidata) and analyze their behavior and performance.

We hope our findings provide practical guidance for designing LM-based QA systems, especially for questions involving counts. We also expect the approach to transfer to other KGs beyond Wikidata, including enterprise and domain-specific KGs.

[1] https://www.wikidata.org/wiki/Wikidata:Database_reports/List_of_properties/all
[2] https://www.wikidata.org/
[3] https://deepmind.google/models/gemma/

More broadly, we study an LM-based KGQA approach for simple, single-hop count questions on Wikidata. We focus on relations that already encode a count (e.g., number of children, number of seasons), rather than computing counts via entity enumeration or multi-hop aggregation. This focus reflects practical and ethical considerations: in many settings, entity-level records (e.g., a person's children or a company's branches) may be withheld for privacy or policy reasons, while an aggregate count is curated and published as a first-class property. Pre-encoded counts also reduce brittleness relative to COUNT-style queries, which are more sensitive to KG incompleteness and duplication, and typically require different robustness mechanisms. Accordingly, multi-hop and compositional counting are out of scope here but complementary to our setting.

The remainder of this paper is organized as follows: Section 2 reviews related work. Section 3 details our methodology for identifying and integrating counting facts. Section 4 presents our results and analysis. Section 5 discusses the implications and limitations of our findings. Finally, Section 6 concludes the paper and outlines future directions.

2. Related Work

2.1. Knowledge Graphs and RDF

A knowledge graph (KG) is a graph-based data structure designed to capture real-world knowledge. It consists of nodes representing entities and edges representing the relationships connecting them [5]. KGs can be modeled in the Resource Description Framework (RDF) [6]. An RDF graph is a set of RDF triples, consisting of three components: the subject, predicate, and object, written `<S, P, O>`. For simplicity, we do not distinguish between RDF terms (literals, IRIs, and blank nodes). Examples of triples are `<Donald Trump, child, Barron Trump>` and `<Donald Trump, number of children, 5>`. Syntactically, RDF graphs can be serialized in various formats, such as RDF/XML, Turtle, and JSON-LD. In this paper, we use the terms knowledge graphs and RDF graphs interchangeably.

2.2. Wikidata KG

Wikidata, a multilingual "Wikipedia for data", functions as a KG by centralizing and managing Wikipedia's factual information, paving the way for new applications through its structured and integrated data [7]. Following an open KG approach, Wikidata's data is openly licensed, permitting free access, use, modification, and sharing with minimal attribution and openness requirements. While providing an RDF representation, Wikidata utilizes URIs optimized for persistence and language neutrality, potentially at the cost of human readability [5].

2.3. SPARQL Queries

SPARQL [8] serves as the standard query language for retrieving information from KGs. Its core component, the basic graph pattern, consists of triple patterns that resemble RDF triples but allow variables in the subject, predicate, and object positions. These patterns identify matching subgraphs within the RDF data by substituting variables with RDF terms, resulting in a graph

equivalent to the matched portion. For example, the following SPARQL query retrieves all children of Donald Trump:

SPARQL Query

```
# we assume a default namespace for terms used here
SELECT ?child WHERE {
  :DonaldTrump :child ?child }
```

When executed against a KG, the results of the preceding query would likely include all five of Donald Trump's children: Donald Trump Jr., Ivanka Trump, Eric Trump, Tiffany Trump, and Barron Trump.

A SPARQL query can also incorporate aggregate functions, such as the `COUNT` feature. The following example demonstrates how to count the number of children of Donald Trump:

SPARQL Query

```
SELECT (COUNT(?child) AS ?cnt) WHERE {
    :DonaldTrump :child ?child }
```

A Wikidata-oriented implementation of the above SPARQL query is shown below:

SPARQL Query

```
SELECT (COUNT(?child) AS ?cnt) WHERE {
    wd:Q22686 wdt:P40 ?child }
```

Wikidata provides properties well-suited for directly representing counting information. For example, in the case of children, a SPARQL query using the "number of children" property, which readily provides the count for Donald Trump's children, is provided below, returning five. In this paper, we concentrate on such counting properties.

SPARQL Query

```
SELECT ?numOfChildren WHERE {
    wd:Q22686 wdt:P1971 ?numOfChildren }
```

2.4. KGQA

Knowledge graph question answering (KGQA) leverages the structured knowledge within a KG to answer questions posed in natural language [10]. Put simply, KGQA involves using the triples stored in a KG to address natural language queries. For example, to answer "Where is Jack Ma's birthplace?", a KG containing the triple `<Jack Ma, born in, Hangzhou>` can be queried to retrieve the entity Hangzhou. In practice, the availability of prominent open KGs such as Wikidata and DBpedia, as well as enterprise KGs maintained by organizations and companies, can be leveraged as a valuable source for KGQA. This work focuses on Wikidata, which is one of the largest and most popular KGs in the world. While various KGQA datasets exist, including

SimpleQuestionsWikidata [11], QALD-9-plus [12], QALD-10 [13], and Semantic answer type and relation prediction task (SMART) dataset [14], none of these datasets specifically focus on counting questions, as their design caters to more general QA tasks.

2.5. Hallucination

Language models (LMs) struggle with ambiguous language or concepts that fall outside their well-defined knowledge, resulting in outputs that can appear convincing but are factually wrong or off-topic [15]. This issue, commonly referred to as "hallucinations," diminishes the trustworthiness of these models. These hallucinations can be categorized as stemming from data, training, and inference [16]. This work focuses on the first cause, specifically, limitations at the knowledge boundary (e.g., long-tail knowledge, up-to-date information, and copyright-sensitive content). The problem of knowledge boundary is one area where KGs can offer mitigation by expanding the knowledge reach of LMs. Furthermore, by exposing LMs to KGs, the issue of fact-conflicting hallucination [17], where LMs generate content inconsistent with established world knowledge, can be further mitigated. This is because KGs provide a factual grounding for the processing performed by LMs.

3. Methodology

3.1. Problem Statement

In this work, we define cardinality/counting hallucination as the generation of an incorrect numerical quantity by a language model (LM) when referring to the number of instances of a person or object, as compared to the actual, verifiable number in the real world.

Key elements of this definition are as follows:

1. Incorrect numerical quantity: The core issue is a discrepancy between the model's output and the true real-world count.
2. Countable entities: This hallucination specifically applies to countable items or individuals.
3. LM generation: The incorrect quantity originates from the LM's output.
4. Verifiable reference: A true, real-world value serves as the benchmark for correctness.

3.2. Counting Knowledge in Wikidata

We conduct an analysis of Wikidata properties to identify which ones can be categorized as counting properties. Wikidata contains over 12,000 properties, among which we focus on those of type Quantity and WikibaseItem, totaling more than 2,300 properties. We exclude other property types, such as CommonsMedia, ExternalId, and EntitySchema, as they are not appropriate for representing counting information. We then perform a thorough manual review of these properties to determine which can be considered counting properties, based on the criterion that such properties must represent a countable number (i.e., a non-negative integer) of entities that can be enumerated in real-world contexts. In the end, we identify a total of 169 counting properties, ranging from number of episodes (P1113) and (number of) employees

(P1128), to number of participants (P1132) and number of wins (P1355). The full list of the counting properties is available at: https://s.id/wd-counting-properties.

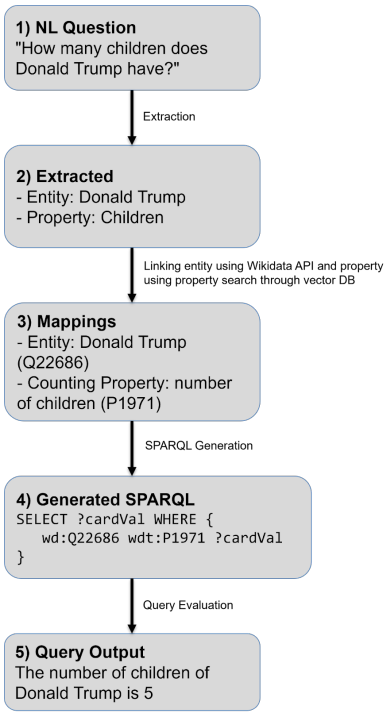

Figure 1: Text-to-SPARQL Mechanism

3.3. Approaches

To address counting hallucinations, we propose two methods for integrating knowledge from knowledge graphs (KGs) into LMs: Text-to-SPARQL and KG-to-Text. Importantly, both approaches are LM agnostic, meaning they can be applied with any LM, regardless of type or vendor. The prompts used in these two methods are accessible at: https://s.id/counting-qa-prompts. In this work, we focus on simple counting questions in which the subject entity and the target count property are already given (e.g., "How many children does X have?"), and the relation is encoded as a count in the KG. For broader question types, an initial step to validate or classify the intent would be necessary before applying either method. This intent detection could be carried out using conventional classification models or LM-based approaches. However, this aspect falls outside the scope of the present study.

3.3.1. Text-to-SPARQL

The Text-to-SPARQL approach, as displayed in Figure 1, outlines the process of transforming a natural language (NL) counting question into a structured SPARQL query to retrieve answers from a KG. It begins with question understanding by identifying the entities and properties

involved. The main entity is first extracted using an LM, which is prompted to isolate the most specific proper noun mentioned in the question. This extracted entity name is then resolved to its corresponding identifier in Wikidata using the `wbsearchentities` API[4], which returns the most relevant entity ID (QID). To facilitate property identification, we build a vector database (i.e., Chroma[5]) from Wikidata property IDs, labels, and descriptions, embedding these fields with the sentence-transformers model `all-MiniLM-L6-v2` to capture semantic similarity. We index those Wikidata properties whose type is either Quantity or WikibaseItem, as discussed in Subsection 3.2. At query time, the input NL question is matched against this indexed collection to retrieve the most relevant properties. Now that the appropriate entity ID (QID) and property ID (PID) have been obtained, an LM is prompted with pre-defined examples and instructed to generate a SPARQL query that conforms to the structure of Wikidata. This query is then executed against Wikidata, and the result is converted into a human-readable answer.

The following is the pseudocode for the Text-to-SPARQL approach.

`text_to_sparql_qa` function

```
# Assume that the global variables langmodel, vector_db, and wikidata_endpoint
    exist

def text_to_sparql_qa(question):
  entity_name = extract_main_entity(question, langmodel)
  entity_wikidata_id = link_entity(entity_name)
  property_candidates = search_properties(question, vector_db)
  sparql_query = generate_sparql(question, entity_name, entity_wikidata_id,
      property_candidates, langmodel)
  return execute(sparql_query, wikidata_endpoint)
```

3.3.2. KG-to-Text

This KG-to-Text pipeline, illustrated in Figure 2, answers NL counting questions using verbalized information from a KG (i.e., Wikidata). It begins by extracting the main entity from the input question and mapping it to its corresponding Wikidata identifier, using the same extraction and linking mechanism as described in the Text-to-SPARQL approach. Next, a SPARQL query is executed to retrieve all direct (truthy) triples associated with the identified entity. Each retrieved triple is then verbalized into human-readable form by converting the subject, predicate, and object into their English labels. These verbalized triples serve as contextual input to the LM, which subsequently generates the final answer to the counting question.

The following is the pseudocode for the KG-to-Text approach.

[4] https://www.wikidata.org/w/api.php?action=help&modules=wbsearchentities
[5] https://www.trychroma.com/

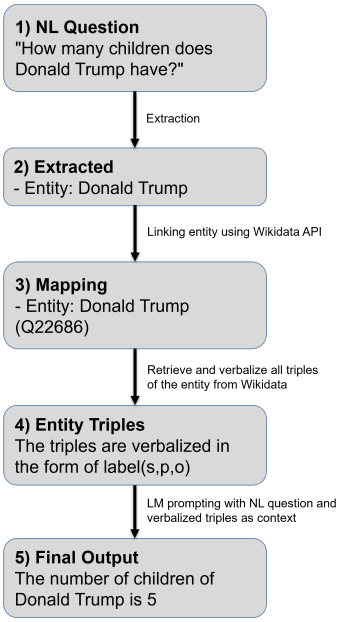

Figure 2: KG-to-Text Mechanism

`kg_to_text_qa` function

```python
# Assume that the global variables langmodel and wikidata_endpoint exist

def kg_to_text_qa(question):
  entity_name = extract_main_entity(question, langmodel)
  entity_wikidata_id = link_entity(entity_name)
  entity_context = verbalize(entity_wikidata_id)
  return ask(question, entity_context, langmodel)
```

3.4. Dataset

As mentioned in the KGQA part of Section 2, currently no dedicated dataset exists for counting question answering. We have developed such a dataset in a semi-automated manner using Wikidata and the DeepSeek LM through the following steps:

1. First, we collect all properties available in Wikidata. Next, we select only those properties that inherently provide counting information, as elaborated in Subsection 3.2.
2. From this refined set of counting properties, we sample entities that possess these properties and extract their corresponding count values.
3. Subsequently, we generate natural language (NL) questions for each entity-property pair, where the true answer to each question is from the corresponding count value in Wikidata. The generation process is supported by an LM. In our case, we rely on DeepSeek-V3-0324. The prompt for the generation is openly available at: https://s.id/counting-qa-prompts.

4. Finally, all the generated natural language questions and their corresponding numerical answers are compiled into a single dataset, which we have named the Counting QA dataset.

Currently, our Counting QA dataset contains over 10,000 entries, each accompanied by three NL question variations, totaling more than 30,000 questions focused on counting tasks. A sample of the dataset is shown in Table 1, while the full dataset is publicly available at: https://s.id/counting-qa-dataset.

Table 1
Snippet of Counting QA dataset with generated questions for selected entity-property pairs.

pId	pLabel	entityId	entityLabel	numOfX	generatedQuestions
P1971	number of children	Q51023	Vanessa Paradis	2	"How many children does Vanessa Paradis have?", "What is the number of children Vanessa Paradis has?", "How many kids has Vanessa Paradis given birth to?"
P2437	number of seasons	Q174972	Touched by an Angel	9	"How many seasons does Touched by an Angel have?", "What is the total number of seasons in Touched by an Angel?", "How many seasons were produced for Touched by an Angel?"
P8368	number of branches	Q126391861	Arab African International Bank	99	"How many branches does Arab African International Bank have?", "What is the total number of branches for Arab African International Bank?", "Can you tell me the count of Arab African International Bank's branches?"

3.5. Experiment Setup

We aim to evaluate our LM-based KGQA approaches, namely Text-to-SPARQL and KG-to-Text, using the Counting QA dataset. Additionally, we include a simple baseline (naive) method that relies solely on the pre-trained knowledge of LMs, without leveraging external structured information. To examine the impact of structured counting knowledge across different types of LMs, we select two representatives: Llama3.2 3B as a smaller LM and DeepSeek-V3-0324 as a more advanced one. To enable a modular analysis of our experimental setup, we also introduce scenarios with perfect entity extraction, in which the correct entities are provided directly, assuming flawless entity extraction and linking. This experiment design supports

clearer identification of potential sources of error. For evaluation, we use accuracy as the metric, defined as the number of correct predictions, where the predicted count equals the true count, divided by the total number of predictions. We choose accuracy due to its simplicity and ease of interpretation.

4. Results and Analysis

Table 2 presents the results of our experiments across different approaches and LMs. Due to time constraints, the experiments were performed on a randomly selected subset of 1,000 entries from our Counting QA dataset, using only the first question variation for each entry. We consider this sample size sufficiently representative of the full dataset. To ensure consistency across all experiments, we applied a fixed random seed so that the same test set was used throughout.

Table 2
Accuracy (%) comparison of different approaches across two language models.

Approach	Llama3.2 3B Acc (%)	DeepSeek-V3-0324 Acc (%)
Naive	2.60	14.10
Text-to-SPARQL	45.90	65.30
KG-to-Text	62.70	77.60
Text-to-SPARQL with Perfect Entity Extraction	65.40	82.80
KG-to-Text with Perfect Entity Extraction	98.60	99.10

The experiment results show how different approaches perform across two language models, Llama3.2 3B and DeepSeek-V3-0324. Overall, incorporating KGs significantly improves accuracy in answering counting questions, hence reducing counting hallucinations.

Starting with the simplest baseline, the Naive approach achieves the lowest performance, reaching only 2.60% accuracy with Llama3.2 and 14.10% with DeepSeek. This reveals the severe limitations of relying on an unguided LM output to provide precise numerical facts.

The Text-to-SPARQL approach, which issues explicit KG queries based on extracted entities, leads to a notable improvement. Accuracy rises to 45.90% for Llama3.2 and 65.30% for DeepSeek. This result highlights the benefits of using structured queries over KGs, though it remains somewhat limited by the quality of entity and property extraction.

The KG-to-Text approach performs even better. This method retrieves and verbalizes triples from the KG and supplies them as context for the LM to answer. It achieves 62.70% accuracy with Llama3.2 and 77.60% with DeepSeek. These outcomes indicate that providing relevant context from the KG helps keep the model anchored to the correct numerical facts.

When perfect entity extraction is introduced, meaning the entity extraction and linking are assumed to be flawless, the results improve dramatically. Under this scenario, Text-to-SPARQL with perfect entity extraction achieves 65.40% accuracy with Llama3.2 and 82.80% with DeepSeek. The KG-to-Text approach with perfect entity extraction reaches nearly perfect performance,

achieving 98.60% with Llama3.2 and 99.10% with DeepSeek. This emphasizes how essential accurate entity identification is for maximizing the benefits of integrating KGs.

Overall, these findings show that combining KGs with language models greatly enhances the reliability of counting question answers. They also demonstrate that while a more advanced model like DeepSeek consistently outperforms the smaller Llama3.2, both experience significant gains when supported by KGs and precise entity extraction & linking. This confirms the strong potential of this combined approach to reduce counting errors.

5. Discussion

5.1. KG Potential in Mitigating Counting Hallucination

Our experiments clearly highlight the potential of KGs to mitigate counting hallucinations in LMs. By grounding the LM's generation on structured facts from a KG like Wikidata, we observe a significant reduction in incorrect answers for counting questions. Methods such as Text-to-SPARQL and KG-to-Text demonstrate substantial improvements over naive approaches, achieving accuracies as high as 77.60% (without perfect entity extraction) and 99.10% (with perfect entity extraction), as compared to that of naive (with only less than 15% of accuracy). This underscores the value of integrating external structured knowledge into the QA process, effectively anchoring the model's output in verifiable data and substantially reducing the tendency to hallucinate or guess quantities.

5.2. Completeness and Timeliness of KGs

However, the effectiveness of KGs in preventing hallucinations is inherently tied to the completeness and timeliness of the underlying graph. Even large-scale KGs like Wikidata may lack up-to-date entries for rapidly evolving facts or could have incomplete coverage in certain domains, which can limit their ability to serve as an authoritative grounding source. For instance, newly formed organizations or emerging events might not yet be captured, leading to either a fallback to LM-only generation or outdated responses. Thus, while KGs offer strong potential to counter hallucinations for well-established facts, their utility can be constrained by the scope and update frequency of the graph.

5.3. Dependence on Entity Extraction and Linking

A critical dependency in KG-based approaches is the accurate extraction and linking of textual mentions to their corresponding entities (QIDs) within the KG. The effectiveness of the downstream retrieval or reasoning process heavily depends on this step. Our analysis reveals that errors in surface form recognition, ambiguities (e.g., multiple people sharing the same name), or insufficient disambiguation can lead to retrieving irrelevant or incomplete facts, directly impacting the final answer's correctness. This dependency is further amplified in structured approaches like Text-to-SPARQL, where the entity ID is used to craft explicit queries. Enhancing entity extraction and linking through context-aware disambiguation, type filtering, and robust fallback mechanisms remains essential to fully leverage the benefits of KG integration.

5.4. Scale and Latency

Next, we observe trade-offs in computational scale and latency across different approaches. Using the DeepSeek API provides access to state-of-the-art capabilities with relatively fast inference times, but incurs significant monetary costs, especially at scale. In contrast, running local models like Llama3.2 can help avoid API expenses and reduce operational costs when deployed on low-specification hardware, although this often results in slower processing speeds. However, when Llama3.2 is executed on high-specification hardware, whether locally or in the cloud, it can achieve fast inference comparable to frontier models, though at increased infrastructure cost. This suggests that while KGs help reduce hallucinations, practical deployment also requires balancing computational resources, budget constraints, and expected performance, especially when integrating with diverse LM infrastructures.

5.5. Knowledge Graph Construction

The effectiveness of our approach fundamentally depends on the availability of KGs that contain counting-related information. In this work, we leverage Wikidata, a general-purpose KG, as the primary source for counting knowledge. However, in specific domains such as personal or enterprise settings, similar knowledge might not be readily available. In these contexts, the applicability of our methods hinges on the existence or creation of relevant KGs. When such graphs are absent, users may need to construct them, either manually through domain expertise or automatically using information extraction techniques [18]. Counting knowledge itself can be incorporated in several ways. It can be added manually by domain experts, extracted directly from textual sources using dedicated methods [19], or derived through aggregation of existing KG properties to synthesize new properties that reflect countable facts.

6. Conclusions

In this paper, we have introduced the problem of counting hallucinations in language model (LM) outputs. Through a detailed study of Wikidata, we have investigated how counting facts are represented in a KG and have identified 169 properties that encapsulate countable entities, ranging from the number of episodes and number of children to the number of employees and more. We have proposed two primary LM-based KGQA methods for leveraging this structured knowledge: Text-to-SPARQL, which converts natural language counting questions into SPARQL queries, and KG-to-Text, which verbalizes knowledge triples associated with an entity and injects them into the LM's context for question answering (QA). To support systematic evaluation, we have constructed the first benchmark dataset tailored specifically for counting QA, consisting of more than 10,000 entries and 30,000 counting question variations. Experimental results have demonstrated that combining KGs with LMs substantially reduces counting hallucinations.

Future work may focus on enhancing the robustness of entity extraction and linking, which serve as core components of our framework. Another promising direction is fine-tuning LMs on the constructed counting QA dataset to improve their alignment with structured numerical knowledge. Finally, it would be valuable to extend this approach beyond Wikidata, exploring its applicability to other types of KGs, including personal and enterprise-level graphs.

Declaration on Generative AI

During the preparation of this work, the authors used ChatGPT and Gemini for grammar & spelling checking and paraphrasing. After using these tools, the authors reviewed and edited the content as needed and take full responsibility for the publication's content.

References

[1] S. Sharma, A year ago, it was just ChatGPT — now LLMs are everywhere, 2023. URL: https://venturebeat.com/ai/a-year-ago-it-was-just-chatgpt-now-llms-are-everywhere/, accessed: 2025-07-17.

[2] T. Group, Artificial Intelligence fuels rise of hard-to-detect bots that now make up more than half of global internet traffic, according to the 2025 Imperva Bad Bot Report, 2025. URL: https://cpl.thalesgroup.com/about-us/newsroom/2025-imperva-bad-bot-report-ai-internet-traffic, accessed: 2025-07-17.

[3] P. Sahoo, P. Meharia, A. Ghosh, S. Saha, V. Jain, A. Chadha, A Comprehensive Survey of Hallucination in Large Language, Image, Video and Audio Foundation Models, in: Findings of the Association for Computational Linguistics: EMNLP 2024, Association for Computational Linguistics, Miami, Florida, USA, 2024, pp. 11709–11724. doi:10.18653/v1/2024.findings-emnlp.685.

[4] M. Asjad, Integrating Graph Structures into Language Models: A Comprehensive Study of GraphRAG, 2024. URL: https://www.marktechpost.com/2024/08/24/integrating-graph-structures-into-language-models-a-comprehensive-study-of-graphrag/, last accessed: 2025-07-17.

[5] A. Hogan, E. Blomqvist, M. Cochez, C. d'Amato, G. de Melo, C. Gutiérrez, S. Kirrane, J. E. Labra Gayo, R. Navigli, S. Neumaier, A.-C. Ngonga Ngomo, A. Polleres, S. M. Rashid, A. Rula, L. Schmelzeisen, J. F. Sequeda, S. Staab, A. Zimmermann, Knowledge Graphs, Synthesis Lectures on Data, Semantics, and Knowledge, Springer, 2021. doi:10.2200/S01125ED1V01Y202109DSK022.

[6] M. L. Richard Cyganiak, David Wood, RDF 1.1 Concepts and Abstract Syntax, 2014. URL: https://www.w3.org/TR/rdf11-concepts/, accessed: 2025-07-17.

[7] D. Vrandečić, M. Krötzsch, Wikidata: a free collaborative knowledgebase, Commun. ACM 57 (2014) 78–85. doi:10.1145/2629489.

[8] S. Harris, A. Seaborne, SPARQL 1.1 Query Language, 2013. URL: https://www.w3.org/TR/sparql11-query/, accessed: 2025-07-17.

[9] T. T. Procko, O. Ochoa, Graph Retrieval-Augmented Generation for Large Language Models: A Survey, in: 2024 Conference on AI, Science, Engineering, and Technology (AIxSET), 2024, pp. 166–169. doi:10.1109/AIxSET62544.2024.00030.

[10] J. Pang, Y. Zhang, J. Deng, X. Zhu, A Survey on Information Retrieval Method for Knowledge Graph Complex Question Answering, in: 2022 China Automation Congress (CAC), 2022, pp. 1059–1064. doi:10.1109/CAC57257.2022.10055934.

[11] D. Diefenbach, T. P. Tanon, K. D. Singh, P. Maret, Question Answering Benchmarks

for Wikidata, in: International Workshop on the Semantic Web, 2017. URL: https://api.semanticscholar.org/CorpusID:24052929.

[12] A. Perevalov, D. Diefenbach, R. Usbeck, A. Both, QALD-9-plus: A Multilingual Dataset for Question Answering over DBpedia and Wikidata Translated by Native Speakers, 2022. URL: https://arxiv.org/abs/2202.00120.

[13] R. Usbeck, X. Yan, A. Perevalov, L. Jiang, J. Schulz, A. Kraft, C. Möller, J. Huang, J. Reineke, A.-C. N. Ngomo, M. Saleem, A. Both, QALD-10 – The 10th challenge on question answering over linked data: Shifting from DBpedia to Wikidata as a KG for KGQA, Semantic Web 15 (2024) 2193–2207. doi:10.3233/SW-233471.

[14] N. Mihindukulasooriya, M. Dubey, A. Gliozzo, J. Lehmann, A.-C. N. Ngomo, R. Usbeck, G. Rossiello, U. Kumar, Semantic Answer Type and Relation Prediction Task (SMART 2021), 2022. URL: https://arxiv.org/abs/2112.07606.

[15] G. Agrawal, T. Kumarage, Z. Alghamdi, H. Liu, Can Knowledge Graphs Reduce Hallucinations in LLMs? : A Survey, in: Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers), Association for Computational Linguistics, Mexico City, Mexico, 2024, pp. 3947–3960. doi:10.18653/v1/2024.naacl-long.219.

[16] L. Huang, W. Yu, W. Ma, W. Zhong, Z. Feng, H. Wang, Q. Chen, W. Peng, X. Feng, B. Qin, T. Liu, A Survey on Hallucination in Large Language Models: Principles, Taxonomy, Challenges, and Open Questions, ACM Trans. Inf. Syst. 43 (2025). doi:10.1145/3703155.

[17] Y. Zhang, Y. Li, L. Cui, D. Cai, L. Liu, T. Fu, X. Huang, E. Zhao, Y. Zhang, Y. Chen, L. Wang, A. T. Luu, W. Bi, F. Shi, S. Shi, Siren's Song in the AI Ocean: A Survey on Hallucination in Large Language Models, 2023. URL: https://arxiv.org/abs/2309.01219.

[18] L. Zhong, J. Wu, Q. Li, H. Peng, X. Wu, A Comprehensive Survey on Automatic Knowledge Graph Construction, ACM Comput. Surv. 56 (2023). doi:10.1145/3618295.

[19] P. Mirza, S. Razniewski, F. Darari, G. Weikum, Enriching Knowledge Bases with Counting Quantifiers, in: The Semantic Web – ISWC 2018, Springer International Publishing, Cham, 2018, pp. 179–197.