Applying LLMs to Infer Consistency Rules in Property Graph Data

MALVINA MICHALOUDI, 2nd International Master Program in Data and Intelligence for Smart Systems, University Claude Bernard Lyon 1, France

In the era of big data, ensuring the consistency and reliability of graph-structured data has become a significant challenge. Traditional methods for data repair and consistency maintenance often struggle with scalability and adaptability, particularly in dynamic and complex data environments. This report explores the application of Large Language Models (LLMs), specifically GPT-4, to infer consistency rules within property graphs. Through the use of zero-shot and few-shot prompting methodologies, this study investigates how LLMs can autonomously generate consistency rules and apply them to detect errors within both the schema and the dataset of graphs. The research utilizes two distinct datasets: Twitter network graphs and Movie property graphs, to evaluate the effectiveness of the proposed approach. The results demonstrate promising potential for LLMs in identifying and enforcing consistency rules with varying degrees of adherence across different rules, by using three measures for evaluating the rules based on support, coverage, and confidence. This research contributes to the growing body of work exploring the integration of advanced AI models in data management and highlights the potential for automating complex data consistency tasks.

Additional Key Words and Phrases: Large Language Models, Graph Data Management, Data Consistency, Data Integrity, Consistency Rules, Zero-Shot Prompting, Few-Shot Prompting, Twitter Graph Data, Movies Graph Data, Rule Ranking Measures

ACM Reference Format:
Malvina Michaloudi. 2024. Applying LLMs to Infer Consistency Rules in Property Graph Data. *Proc. ACM Meas. Anal. Comput. Syst.* 37, 4, Article 111 (January 2024), 22 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

In recent years, the explosion of data in the era of big data has led to increasingly complex and interconnected graph structures. Graph databases, which model data as nodes and relationships, are widely used across various domains such as social networks and entertainment industries. These databases offer powerful ways to represent and query complex relationships among entities, but ensuring their consistency and integrity poses significant challenges. Traditional data repair methods often rely on predefined rules and require extensive manual intervention, making them less efficient and scalable for handling large and dynamic datasets.

Large Language Models (LLMs), such as GPT-4, have emerged as transformative tools in natural language processing. Trained on vast amounts of text data, these models can generate human-like text and understand complex language patterns. LLMs have shown remarkable potential in a variety of tasks, from text generation to question answering, and their application has expanded into more specialized domains like data management. By leveraging their ability to understand and generate language, LLMs can be adapted to interpret and manipulate graph data, providing a new approach to extracting and applying consistency rules.

Author's address: Malvina Michaloudi, malvina.michaloudi@etu.univ-lyon1.fr2nd International Master Program in Data and Intelligence for Smart Systems,, University Claude Bernard Lyon 1, 43 Bd du 11 Novembre 1918, Lyon, 69100, France.

ACM 2476-1249/2024/1-ART111
https://doi.org/10.1145/nnnnnnn.nnnnnnn

The integration of LLMs into graph data management represents a significant advancement, offering a dynamic and scalable solution to maintaining data consistency and integrity. This study investigates how LLMs can be utilized to infer consistency rules within intricate graph structures, aiming to enhance the accuracy and efficiency of data management systems.

A property graph represents data as a directed, attributed multi-graph. Vertices and edges are rich objects with a set of labels and a set of key—value pairs, so-called properties [1]. A property graph is a type of graph model used in graph databases where not only the entities, represented as nodes and relationships, represented as edges, are stored, but both nodes and edges can also have properties associated with them. Nodes represent entities or objects such as people, places, items and edges represent how these entities are connected. Properties are key-value pairs associated with nodes and edges. For nodes, properties can store attributes of the entity for example a person's name, age etc. For edges, properties can describe the nature or attributes of the relationship.

1.1 Contributions

The research contribution of my study can be categorized as **empirical** which lies in the practical application of Large Language Models in the domain of graph data management. This research demonstrates the encouraging potential of LLMs in identifying consistency rules in real-world graph databases thus enhancing data integrity and reliability. Through rigorous experimentation and the calculation of rule ranking measures, this study empirically verifies the capability of LLMs to detect and evaluate rules. This practical demonstration not only bridges the gap between theoretical AI advancement and real-world data management challenges but also sets a precedent for future empirical studies in the application of AI for data integrity tasks in diverse domains.

1.2 Significance of the Problem

Ensuring data consistency and reliability of data within complex graph structures is critical for deriving accurate insights and making informed decisions. Traditional data repair methods rely on modules designed to process each data type in isolation and struggle when the content is diverse [2]. These limitations make it difficult when dealing with the complexity of modern graph databases.

Writing effective consistency rules can be a challenging task, especially when these rules need to be adaptable to dynamic data. Also, even when rule mining methods, such as AMIE are applied, the inferred rules can be difficult to interpret and apply effectively. The research of [3] highlighted these challenges, demonstrating that while rule-based systems can infer consistency rules, the complexity of these rules can often limit their practical application. This complexity arises from the intricacy and technicality of the rules generated, making it challenging for users or data managers to fully grasp, interpret, or implement them effectively.

Large Language Models, with their ability to process and learn from vast amounts of data, offer a scalable solution to this problem. These models have demonstrated impressive capabilities in tasks such as machine translation, text summarization and question answering [4]. Leveraging the advanced natural language processing capabilities of LLMs, such as GPT-4, provides a novel approach to graph data management. These models open up the possibility of generating consistency rules and applying them across the dataset, potentially identifying errors with reduced human intervention. However, the effectiveness of these models in this context remains a key focus of this investigation. The research will evaluate the extent to which these models can automate the identification of consistency rules and their application in practice. This represents a significant advancement over traditional methods, bringing in automation and intelligence to the process. These advanced capabilities not only enhance the accuracy of data management but also introduce a new level of automation, sophistication and reduction of time and resources required for data maintenance.

1.3 Problem Statement

In the era of big data, the complexity of graph structures poses considerable challenges for ensuring data consistency and reliability. This research explores how Large Language Models (LLMs) can be utilized to infer consistency rules within these complex graph structures. The focus is on examining whether LLMs can enhance data management by effectively identifying and applying these rules, ultimately improving the accuracy, efficiency, and integrity of data systems across various applications.

2 LITERATURE REVIEW

2.1 State of the Art

Graph data management has seen significant evolution, particularly with the introduction of Large Language Models (LLMs) like GPT-4. Unlike traditional methods that rely on static, predefined rules, LLMs dynamically learn from large datasets, enabling them to infer consistency rules and improve data reliability. One promising approach is the integration of LLMs with knowledge graphs, which enhances both data processing and knowledge representation, making it easier to manage complex data environments [2].
Generative adversarial frameworks (GANs) have also emerged as an effective technique for automated data repair. These models autonomously generate and apply data repair rules, ensuring data integrity at scale [5]. However, by encoding graph-structured data as text, LLMs can further improve reasoning and consistency rule extraction. This adaptability makes LLMs a powerful tool in managing the growing complexity of graph data, reducing human intervention and providing scalable solutions for maintaining data quality [6].

2.2 Research Connections

The research connections of my study are categorized into three distinct fields: (1) **Substantial**: this research contributes to the broader understanding of data consistency and reliability in graph databases, addressing fundamental challenges in maintaining data integrity within complex network structures; (2) **Technical**: the study advances technical methodologies by applying encoding and rule extraction techniques using Large Language Models (LLMs); (3) **Datasets**: the application of the proposed methods to real-world datasets demonstrates the practical utility of LLMs in handling diverse and complex data structures, showcasing the potential for widespread adoption in various domains requiring robust data management solutions. These connections form the backbone of the research, integrating substantial insights, technical advancements, and practical applications to advance the field of data integrity and management in social media environments.

2.3 Relevant and Similar Studies

My research, focusing on implementing Large Language Models (LLMs), a branch of Generative AI technologies, for autonomously detecting consistency rules within graph data, demonstrates relevance with several research studies within the field of graph data management and the application on Large Language Models.

Several studies have focused on how LLMs can be adapted to work with graph data, particularly in encoding graphs as text and enabling graph-based analytical tasks. Research like [6] explores different methodologies for encoding graph-structured data in a format that LLM can understand and various prompting methods for detecting consistency rules by Large Language Models (LLMs). The focus is on improving LLMs' ability to process and understand graph-based information accurately. This aligns with the broader effort to leverage LLMs beyond traditional text-based tasks, allowing them to handle complex, structured data like graphs [4].

Moreover, understanding how LLMs can infer data quality rules rather than simple labeled graphs, remains largely unexplored. While existing studies such as the [5] (Garf framework) address data repair rules for relational data, and papers like [7] demonstrates potential applications in making graph analysis more accessible but they do not specifically focus on property graphs, which offer more complexity due to their node attributes and edge

properties.

This research aims to explore a novel area of study by applying LLMs to infer data quality rules in property graphs, making the rule extraction process more accessible. Traditional rule mining methods often require domain-specific knowledge and involve complex query languages, which can be a barrier for those less familiar with graph theory. By employing LLMs to extract these rules in natural language, we can potentially automate rule generation and data quality assurance within graph databases.

In this sense, my study is distinct from these existing studies as it focuses on the scalability of LLMs to infer rules in property graphs—something that hasn't been extensively studied before. This has implications not only for social networks but for any domain where property graphs are used to represent complex, multi-relational data.

3 LLM-BASED RULE DETECTION

3.1 Encoding the Graph as Text

In order to make a LLM understand the graph, it needs to be encoded into textual format. A graph encoding function maps graph data into tokens for consumption by an LLM. There are various methodologies for representing graphs as text but in this study the incident graph encoding function is being used.

The process of encoding graphs as a text can be separated into two key inquiries: First, the encoding of nodes in the graph, and second the encoding of edges between the nodes. Regarding the encoding of nodes, the incident encoding function uses the integer encoding (e.g. Node 0). Regarding the encoding of the edges, it uses the incident encoding where the source node is connected to target nodes. Combining the node and edge encoding, the incident graph encoding uses the integer node encoding and the incident edge encoding [6]. Both the schema and the data will be encoded by using the incident graph encoding.

The full details for the incident graph encoding function are presented in the Table 1 below:

G describes a graph among 0, 1, 2, 3, 4, 5, 6, 7, and 8.
Node 0 is connected to nodes 1, 2.
Node 1 is connected to nodes 0, 2.
Node 2 is connected to nodes 0, 1, 3, 4, 5, 7.
Node 3 is connected to nodes 2, 8.
Node 4 is connected to node 2.
Node 5 is connected to nodes 2, 6.
Node 6 is connected to nodes 7, 5.
Node 7 is connected to nodes 2, 8, 6.
Node 8 is connected to nodes 3, 7.

Table 1. Incident Graph Encoding

3.1.1 Encoding Graph Schema as Text. .

Before encoding the graph schema as text, it is defined by specifying what kinds of entities exist and their types, as well as the types of relationships that exist between nodes, including the source and the target ones. Then, the defined schema is converted into a text-based serialization format that LLMs can process. JSON is often used for this purpose due to its readability and ease of use.

3.1.2 Encoding Graph Data as Text. .

Apart from the schema encoding, the graph data is converted also into a text-based format that the LLM can

effectively comprehend. The initial step involves querying the graph data to retrieve a sample representation of all the nodes and edges. This extraction includes the labels and properties for each node and the source, target and relationship types for each edge. This extraction is then exported in a JSON format. This format is chosen for its simplicity and ease of parsing, both for humans and machines. In the JSON serialization, each node is represented with its labels and properties, and each edge includes the source and target nodes as well as the type of relationship connecting them.

Concerning the encoding of the graph data, I employ again the incident graph encoding function, which ensures that nodes are described with their relationships to other nodes. Additionally, contextual information from the node properties is included to provide a richer description which enhances the LLM's ability to understand the data. For instance, for a tweet node, properties such as ID, text and creation date can be included.

3.2 Prompting Heuristics

3.2.1 Zero-Shot Prompting Methodology. .

Zero-shot prompting is an approach which simply provides the model with a task description and asks it to generate the desired output, without any prior training on the task [6]. The model predicts the answer given only a natural description of the task. A simple example is: Given as task description the sentence: Translate English to French and as prompt the word: cheese [8].

In the context of graph data management, zero-shot prompting can be employed to extract consistency rules directly from the encoded graph schema and from the encoded graph data. The first step in this process is to encode the graph schema and the data into a text-based format that the LLM can comprehend. This involves serializing the graph data into JSON format, where each node and edge is represented with labels, properties, and relationships. By providing a clear and structured representation of the graph, the LLM can parse and understand the intricate relationships and attributes within the data.

Once the graph data is encoded, the next step is to formulate the prompt, which is designed to be clear and specific, that can guide the LLM to generate consistency rules. For instance, the prompt question in my study is "Can you generate some consistency rules that could be detected in the relationships and interactions within this graph?".

Using zero-shot prompting, the LLM generates potential consistency rules based on the provided prompt and the encoded graph data. These rules are derived from the model's extensive training on diverse datasets and its ability to infer patterns and relationships. For example, the LLM might produce rules concerning a Twitter graph data, such as "Each User must have a unique username" or "Each Tweet must have a unique ID" etc.

This methodology offers several advantages. It provides flexibility, allowing the LLM to handle a wide range of tasks without the need for task-specific training data. It is efficient, leveraging the pre-trained knowledge of the LLM to quickly generate useful rules and insights, saving time and resources. Additionally, the methodology is adaptable and can be applied to various domains and datasets, making it a versatile tool for data management and analysis.

3.2.2 Few-Shot Prompting Methodology. .

Few-shot prompting is another approach which provides the model with a small number of examples of the task, along with the desired outputs [6]. A simple example is: Given again as task description the sentence: Translate English to French then three examples from three words that have been translated from English to French and as prompt again the word: cheese [8].

In the context of graph data, the few-shot prompting methodology begins with the preparation of representative examples that illustrate the types of consistency rules desired. These examples serve as a guide for the LLM, showing it how to interpret the graph data and generate appropriate rules. Once the examples are prepared, they are included in the prompt given to the LLM. The LLM uses these examples to infer patterns and generate new

rules that are consistent with the provided examples. This process helps the model to adapt its responses more accurately to the specific requirements of the graph data.

Few-shot prompting offers significant advantages over zero-shot prompting by providing the model with contextual examples, leading to more accurate and relevant outputs. It allows for a more nuanced understanding of the task and helps in generating rules that are better aligned with the specific needs of the dataset.

3.3 Analytical Procedure for Extracting Consistency Rules using LLMs

The pipeline illustrated in the Figure 1 describes the process of applying Large Language Models (LLMs), specifically GPT-4, to infer consistency rules in a graph database system. It begins with the schema and the database, which are encoded as a text using an Incident Graph Encoding Function.

First of all, the zero-shot prompting methodology is being used where the encoded graph schema and data are separately fed into the LLM. The model is prompted with a question such as, "Can you generate some consistency rules that could be detected in the relationships and interactions within this graph?" Here, the LLM is expected to infer possible consistency rules based solely on the schema and data description, without any prior examples. This initial phase allows the model to leverage its pre-trained knowledge to suggest consistency rules that may apply to the given graph structure.

Then the few-shot prompting is applied where the graph description is supplemented with a few specific examples of consistency rules. The LLM is then prompted with a follow-up question like, "Based on these examples, can you generate additional consistency rules that could be detected in the relationships and interactions within this graph dataset?" By providing a few examples, this phase allows the model to refine its understanding and generate more tailored rules, leveraging the context provided by the initial examples.

Once the consistency rules are generated from both the above methodologies, they are fed back into the LLM, now with the additional task of generating Cypher queries. These queries will be used to detect the inferred consistency rules within the graph database. By asking the LLM to produce Cypher queries, the process transitions from theoretical rule generation to practical application, enabling the rules to be tested directly within the graph database.

The generated queries are then executed within the graph database, and the results are used to evaluate the effectiveness of the inferred rules. Key metrics such as support, coverage, and confidence are calculated for each rule. These metrics provide insight into the validity and reliability of the rules, helping to assess their impact on maintaining data consistency.

In the final step, the pipeline outputs a set of consistency rules along with their corresponding support, coverage, and confidence measures. This process showcases how LLMs can be used not only to infer complex rules but also to automate their application within a graph database, significantly reducing the need for manual intervention and enhancing the overall integrity of the data.

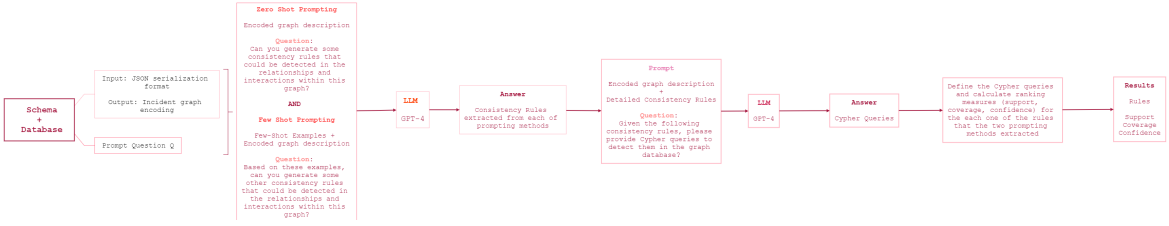

Fig. 1. Schema and Data Workflow of Consistency Rule Extraction and Evaluation using LLMs

4 EXPERIMENT DESIGN

4.1 Use Cases

This research utilizes two datasets as use cases: the Twitter network property graph database and the Movie property graph database. Each dataset presents unique characteristics making them suitable for evaluating the effectiveness of Large Language Models (LLMs) in detecting consistency rules within complex graph structures. You can find the code that has been followed for the extraction and the evaluation of consistency rules for my research on my GitHub: `GitHub Repository`

4.1.1 Twitter Network Graph. .

As illustrated in the Figure 2, the schema of the twitter graph includes diverse types of nodes and relationships, such as "User" nodes connected to "Tweet" nodes via "POSTS" relationships, "Tweet" nodes connected to "Hashtag" nodes via "TAGS" relationships, and "Tweet" nodes connected to "User" nodes via "MENTIONS" relationships etc.

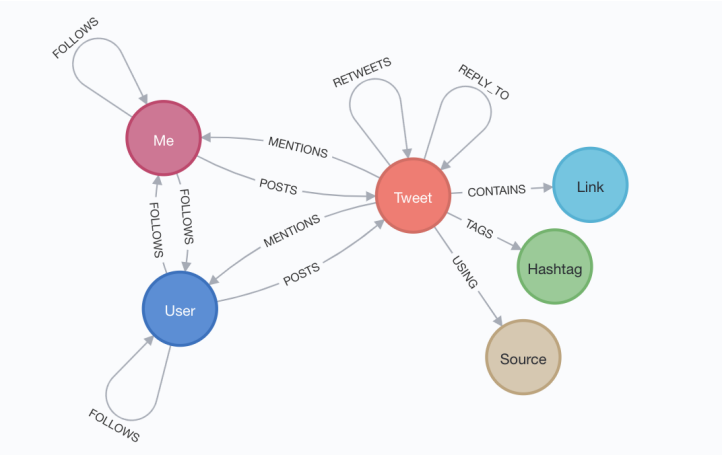

Fig. 2. Social Media Engagement: A Twitter Network Graph Schema

The Twitter data consists of 43,325 nodes and 57,896 relationships. Nodes represent entities such as users, tweets, hashtags, links, and sources, while edges represent various interactions and relationships between these entities. A representation of the Twitter property graph data is presented in the Figure 3.

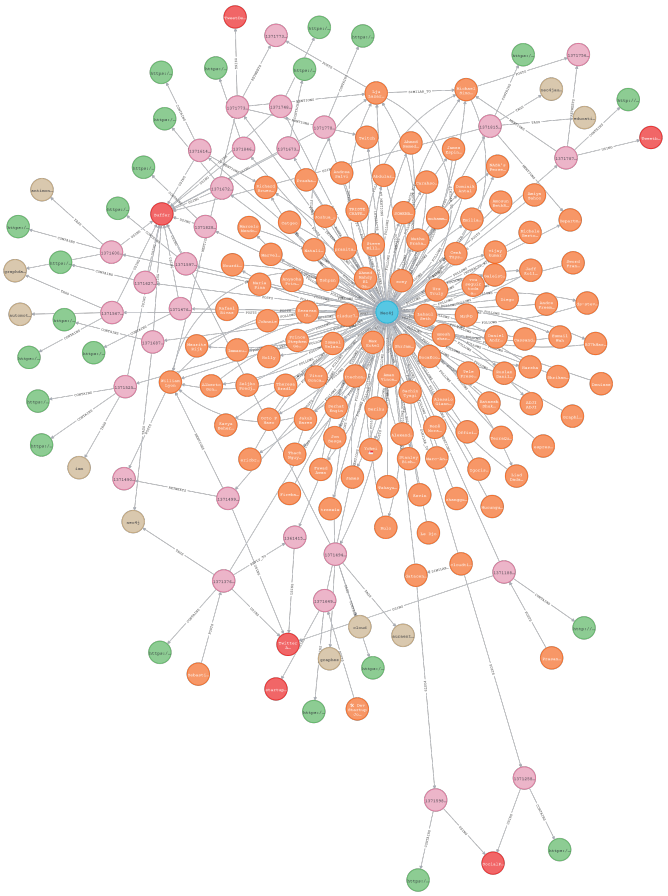

Fig. 3. A Twitter Network Property Graph

4.1.2 Movies Graph. .
The Movie data consists of 171 nodes and 253 edges as illustrated in the Figure 5. The schema of this graph is presented in the Figure 4 and reflects the intricate network of relationships within the movie industry, such as "Person" nodes connected to "Movie" nodes via roles like "ACTED IN", "DIRECTED", "PRODUCED", and "WROTE". This detailed structure is essential for ensuring the logical consistency of connections between movies and their associated personnel.

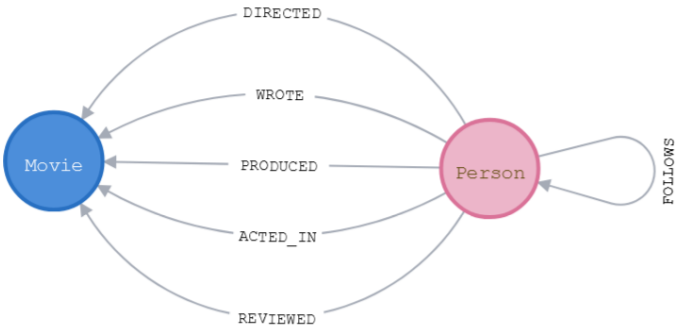

Fig. 4. The Movies Graph Schema

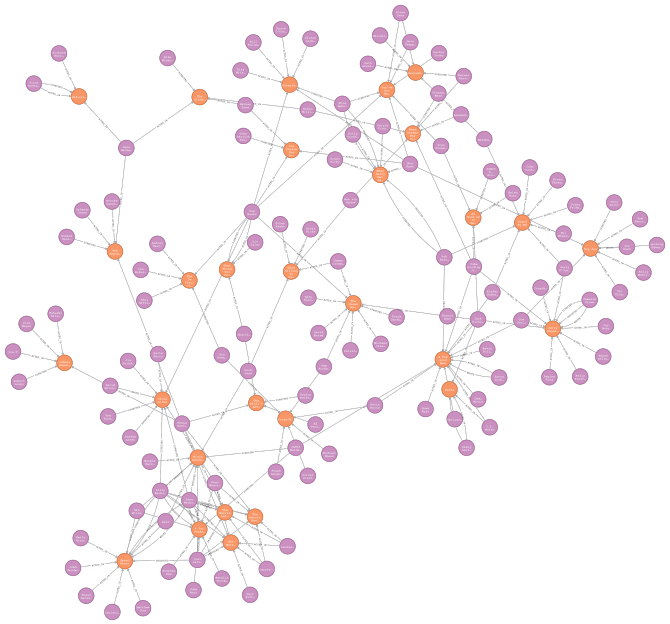

Fig. 5. The Movies Property Graph

4.2 Extracting Twitter Schema Consistency Rules

The Twitter schema is encoded using the incident graph encoding which is presented in the section 3.1.1. This function captures the graph's structure by converting it into a text-based format that includes all nodes, edges, and their properties. The incident graph encoding ensures that the graph data is represented in a way that the LLM can easily process and understand. The full details of the twitter schema incident graph encoding is presented in the Appendix in the 8.1 section.

The encoded schema, along with a prompt question, is fed into the LLM, in this case the GPT-4, to generate consistency rules. The question is framed as: "Can you generate some consistency rules that could be detected in the relationships and interactions within this graph?". Using both zero-shot and few-shot prompting methodologies,

the LLM is directed to generate and refine these rules. By applying these techniques, the LLM can generate a comprehensive set of consistency rules tailored to the specific characteristics of the Twitter graph.

4.3 Extracting Twitter Data Consistency Rules

The Twitter network property graph data comprises nodes and edges representing real-world data such as users, tweets, hashtags, links, and sources. Unlike the schema, which defines the structure, the data contains the actual instances of data that need to be checked for consistency.

The twitter graph data is encoded using again the incident graph encoding function but the difference is that it includes contextual information about the node properties. This enhances the ability of the LLM to understand the data by providing a richer description. The full details of the twitter data incident graph encoding is presented in the Appendix in the 8.2 section.

Because the encoded graph data description is large, it was necessary to break it into smaller chunks that can be processed by the LLM in manageable parts. This ensures that the LLM can effectively parse and analyze the entire dataset without being overwhelmed by its size. So, these chunks along with the prompt question “Can you generate some consistency rules that could be detected in the relationships and interactions within this graph?" are fed into the model to generate relevant consistency rules, using zero shot and few shot prompting methodology.

4.4 Extracting Movies Schema Consistency Rules

The Movie schema is encoded using the incident graph encoding function. This function captures the graph’s structure by converting it into a text-based format that includes the nodes and the relationships between those entities. The full details of the movies schema incident graph encoding is presented in the Appendix in the 8.3 section.

The same procedure as applied before is followed concerning the detection of consistency rules for the movies schema using zero-shot and few-shot prompting methods.

4.5 Extracting Movies Data Consistency Rules

The movies graph data is encoded using the incident graph encoding function with contextual information about the nodes included. The full details of the movies database incident graph encoding is presented in the Appendix in the 8.4 section.

In this case, the encoded graph description was not broken into chunks because it was not as large as the Twitter ones, so the prompt of the incident graph encoding of the movies database and the question are processed from the model to infer consistency rules, using zero shot and few shot prompting methodology.

4.6 Rule Ranking Measures

The rules that the model generated using the zero-shot and few-shot prompting methods are then evaluated for their relevance and accuracy. This involves testing the rules to identify if they are applied in the graph database. This evaluation process ensures that the rules are effective in maintaining data integrity.

To achieve this evaluation, I utilized the LLM to generate cypher queries for each rule and then estimate the quality of rules by using three ranking measures as follows:

- **Support:** Denotes the number of facts in the graph that satisfy the rule, i.e., the number of instances where the rule is true.
- **Coverage:** Normalizes the support by the number of facts for each relation in the graph, showing the proportion of the data that is correctly following the rule compared to all the data that should follow it.

- **Confidence:** Defined as the ratio of the number of facts that satisfy the rule (support) and the number of times the rule body is satisfied in the graph, showing the likelihood that if the conditions of a rule are satisfied, then the rule itself will also be satisfied [9].

5 RESULTS

5.1 Twitter Schema Consistency Rules and Ranking Measures

The Table 2 presents the consistency rules that the model of GPT-4 generated when it used both the zero-shot prompting method and the few-shot prompting method:

1. Users cannot follow themselves.
2. Tweets should not retweet/reply to themselves.
3. If a Tweet mentions a User, it must have been posted by a User.
4. If a Tweet tags a Hashtag, the Hashtag must be valid.
5. Each tweet should be posted by an existing user.
6. Tweets are unique by each User.
7. If a Tweet retweets another Tweet, the origin of the retweeted Tweet must be different from the origin of the retweeting Tweet.
8. Tweets use sources that are valid.
9. If a Tweet is a reply to another Tweet, the original Tweet's poster must be a User.
10. A retweet should only be possible if the user who is retweeting follows the original poster of the tweet.
11. Tweets contain valid URLs in Links.
12. If a Tweet mentions a User, the User must follow the person who posted the Tweet.
13. Tweets should retweet and reply to existing tweets.
14. If a User follows another User then the other User should also follow it.
15. A user cannot direct message another user unless they follow each other.
16. If a user blocks another, their tweets and any interaction should no longer appear in the blocker's timeline
17. The count of likes, retweets and replies for each tweet must be accurately reflected
18. A deleted user's tweets should be removed from the database or marked as deleted

Table 2. Zero-Shot and Few-Shot Prompting Methods Rules from the Twitter Schema

The Figure 6 below illustrates the ranking measures derived from the rules that were generated from the twitter schema, providing a comprehensive view of their adherence and relevance within the dataset.
From the first plot, which demonstrates the support for each rule indicating the number of instances that satisfy the given rule, rules such as "Users Cannot Follow Themselves", "If a Tweet tags a Hashtag, the Hashtag must be valid", "Tweets should not retweet or reply to themselves", "Tweets use sources that are valid", "Each tweet should be posted by an existing user" exhibit perfect and high support values respectively, signifying their widespread adherence in the dataset. This high level of support highlights the consistency of these rules within the Twitter graph schema.
From the second plot, which represents the proportion of existing facts that are implied by the rule, high coverage values for rules like "Tweets use sources that are valid" "Users Cannot Follow Themselves," and "Tweets are unique by each user" suggest that these rules are well-integrated within the graph, impacting a significant portion of the data. This indicates that these rules are effective in capturing the essential structure and relationships

within the Twitter dataset.

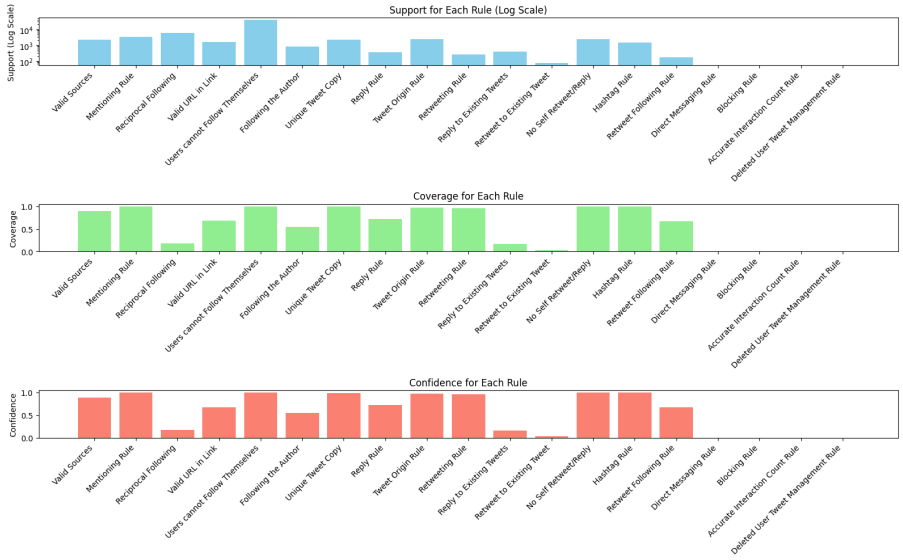

Fig. 6. Rule Ranking Measures of the Twitter Graph Schema Consistency Rules

5.2 Twitter Data Consistency Rules and Ranking Measures

The Table 3 presents the consistency rules that the model of GPT-4 generated from the encoded twitter database when both the zero-shot prompting method and the few-shot prompting method are used:

19. Every Tweet has a unique ID.
20. Every user has a unique username.
21. If a User follows another user, a directed edge should exist from one user to another.
22. Every user who follows another user follows them only once (unique follow relationship).
23. Every Tweet has one user who posted it.
24. If a Tweet mentions a user, there should be a directed edge from the tweet to the user.
25. If a Tweet contains a hashtag, there should be an edge from the tweet to the hashtag.
26. A retweeted tweet must reference the original post including the author's username in its text.
27. Every Tweet that contains a URL should have a directed edge towards that Link.
28. Each tweet should have text and creation date.
29. Every Tweet should have a valid source.
30. The tweet text that tags a hashtag should contain that hashtag within it.
31. If the identical text appears across multiple tweets, it should be a retweet.
32. A reply to a Tweet must have a relationship with the original Tweet.
33. If a user is mentioned in a tweet, the tweet should contain the tagged user's username.
34. Each user should have a unique username.

35. Every user can have followers and the following relationship must be present.
36. A reply to a Tweet must have a relationship with the original Tweet.

Table 3. Zero-Shot and Few-Shot Prompting Methods Rules from the Twitter Data

The rules “Every user has a unique username” and the rule “A reply to a Tweet must have a relationship with the original Tweet” are common in these two prompting methods. The evaluation of the Twitter data involved evaluating several key consistency rules that govern the structure and relationships within the graph. The goal was to ensure data integrity and coherence in the context of a social network graph.

The Figure 7 presents the ranking measures for each rule. Concerning the support, rules such as "Every Tweet has a unique ID" "Every user has a unique username" and " If a User follows another user, a directed edge should exist from one user to another" exhibit perfect values, reflecting their strong presence and adherence within the dataset. This very high level of support underscores the robustness of these rules in maintaining data integrity within the Twitter data.

Whereas, high coverage values for rules like "Every Tweet has a unique ID" "If a User follows another user, a directed edge should exist from one user to another" and "Every user has a unique username" suggest these rules effectively capture the essential structure and relationships within the Twitter data. The coverage metric highlights the broad applicability and relevance of these rules across the dataset.

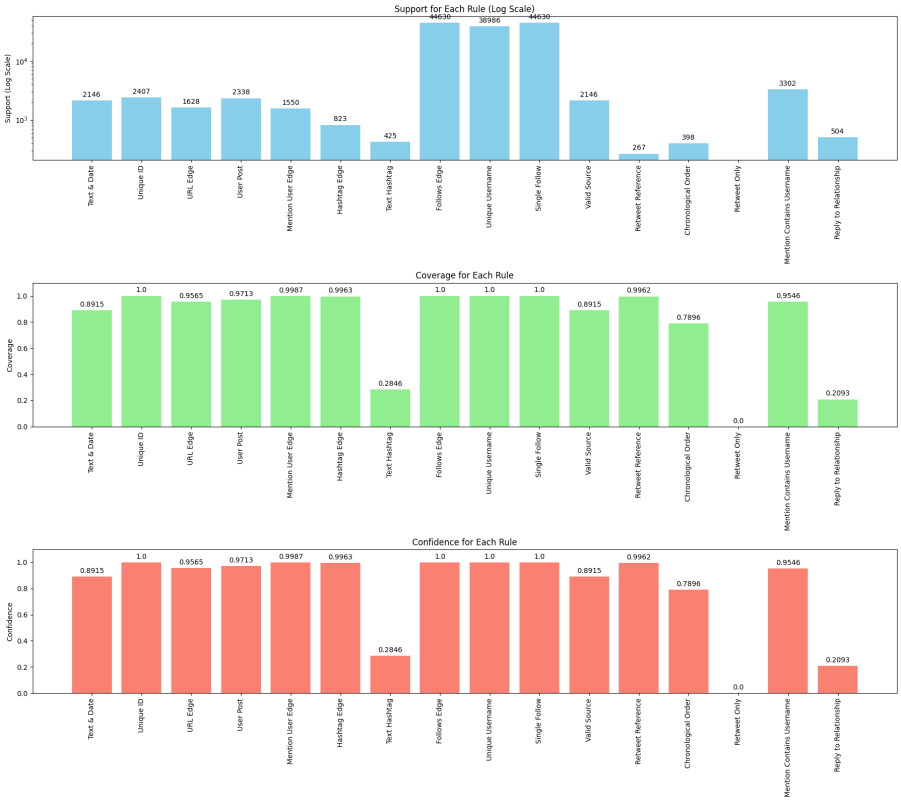

Fig. 7. Rule Ranking Measures of the Twitter Graph Database Consistency Rules

In general, the rules provide consistency and reliability within the Twitter data. By defining and enforcing these rules, the structure and relationships within the graph are maintained, ensuring that the data remains accurate and coherent. For instance, the rule “Every Tweet has one user who posted it” which guarantees that tweets are only posted by recognized users within the network, preventing orphaned tweets ensures structural consistency. Also, the rule of “Users cannot follow themselves” prevents illogical self-referential following relationships and the rule “Tweets should not retweet/reply to themselves" ensures that users do not engage in nonsensical interactions with their own tweets, maintaining logical interaction patterns.

The rule “Tweets are unique by each User” ensures that each tweet is unique, preventing duplication and spam within the dataset and the rule “Tweets should retweet and reply to existing tweets” validates that retweets reference existing tweets, preserving the authenticity of retweet actions.

5.3 Movies Schema Consistency Rules and Ranking Measures

The Table 4 below presents the consistency rules that the model of GPT-4 generated from the encoded movie graph schema when both the zero-shot prompting method and the few-shot prompting method are used:

1. A Movie must have at least one Person associated with it in the roles of director, actor, producer, or writer.
2. A Person can't review a Movie in which they have acted, directed, produced, or wrote.
3. A Person can't follow themselves.
4. A Person can't have multiple roles in the same Movie with the same role type.
5. A Movie must have a unique title and release year combination.
6. A Person can't act, direct, produce, or write a Movie before their date of birth.
7. A Movie can't be released before it was written, produced, directed or acted.
8. A Movie cannot have an actor born after the release date of the movie.
9. A Person can not have multiple roles in the same Movie of an identical nature.
10. Two Movies cannot have identical titles.
11. A Review of a Movie must be written after the movie's release date.

Table 4. Zero-Shot and Few-Shot Prompting Methods Rules from the Movies Schema

The Figure 8 illustrates the ranking measures for various rules applied to the movie schema, focusing on support, coverage, and confidence metrics. The first plot shows the support for each rule on a logarithmic scale, highlighting the number of instances where each rule is satisfied. For instance, the rule "A Movie must have at least one Person associated with it in the roles of director, actor, producer, or writer" has a support value of 38, while "A Movie cannot have an actor born after the release date of the movie" has a support value of 171, indicating that these rules are met frequently within the dataset. The second plot displays the coverage for each rule, representing the proportion of total instances where the rule could potentially apply and is indeed applicable. Notably, several rules, such as "A Person can't follow themselves" and "A Person can not have multiple roles in the same Movie of an identical nature" show full coverage (1.0), demonstrating consistent applicability across relevant instances.

The third plot presents the confidence for each rule, reflecting the likelihood that the rule holds true whenever its conditions are met. Similar to coverage, rules like "A Person can't follow themselves" achieve perfect confidence (1.0), signifying high reliability. However, certain rules, such as "A Review of a Movie must be written after the movie's release date" show no support, coverage, or confidence, indicating potential issues with these rules or limitations within the dataset. This comprehensive analysis aids in understanding the robustness and consistency of the movie schema, providing insights into data quality and potential areas for improvement.

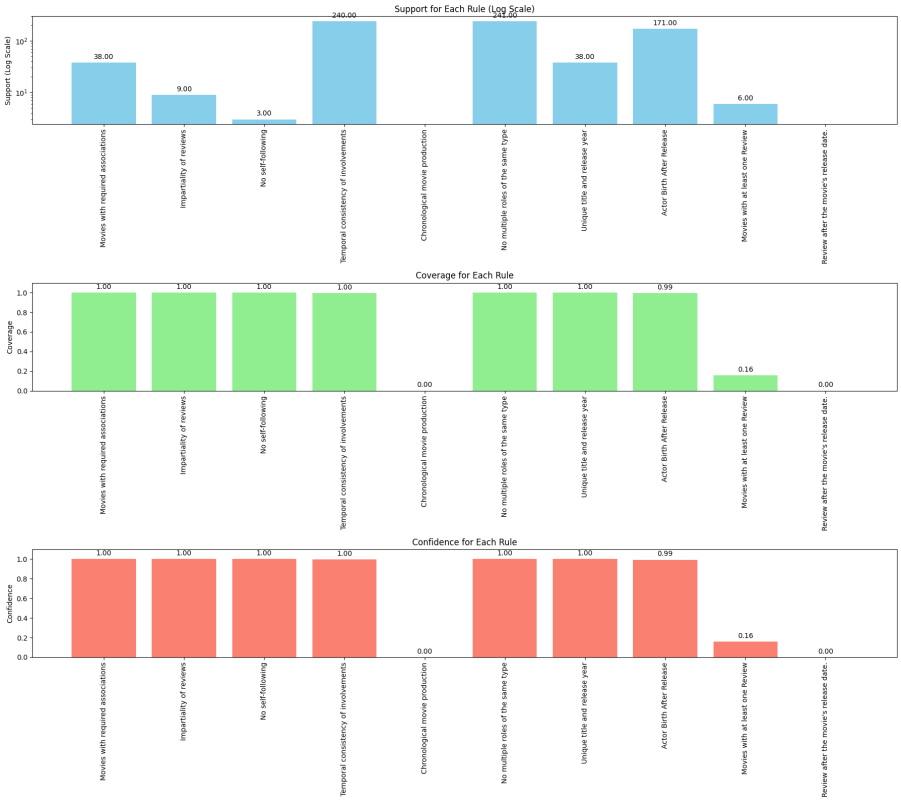

Fig. 8. Rule Ranking Measures of the Movies Graph Schema Consistency Rules

5.4 Movies Data Consistency Rules and Ranking Measures

In the Table 5 are presented the consistency rules that the model of GPT-4 generated from the encoded movie graph data when both the zero-shot prompting method and the few-shot prompting method are used:

12. If a person is listed as the director, producer, or writer of a movie, they must be associated with the film's release year and should be born before the movie release year.
13. A movie can have multiple actors, but a release date should be consistent for a specific movie throughout the graph.
14. If a person acts in a movie, the movie's release year should be later than the birth year of the person.
15. Persons can only have one birth year.
16. If a person directed a movie, they cannot act in the same movie.
17. If a person directed a movie, they cannot be listed as an actor in the same movie unless there's a rule that directors can have cameos.
18. A person can't have multiple birth years.

Table 5. Zero-Shot and Few-Shot Prompting Methods Rules from the Movies Data

The Figure 9 displays the ranking measures (support, coverage, and confidence) for various rules applied to the movie database. Concerning the support plot, the rule "If a person directed a movie, they cannot act in the same movie" has a high support value of 210, while "A movie can have multiple actors, but a release date should be consistent for a specific movie throughout the graph" has a lower support value of 38, suggesting the frequency of these rules being met in the dataset.

The coverage plot illustrates the proportion of instances where each rule is applicable and valid. Rules such as " If a person acts in a movie, the movie's release year should be later than the birth year of the person" and "If a person is listed as the director, producer, or writer of a movie, they must be associated with the film's release year and should be born before the movie release year" show full coverage (1.0), indicating that these rules consistently apply across all relevant instances. In contrast, the rule "A movie can have multiple actors, but a release date should be consistent for a specific movie throughout the graph" also shows full coverage, but the support indicates it is less frequently met.

The confidence plot reveals the likelihood that a rule holds true whenever its conditions are met. High confidence values, such as those for "If a person acts in a movie, the movie's release year should be later than the birth year of the person" and "If a person is listed as the director, producer, or writer of a movie, they must be associated with the film's release year and should be born before the movie release year" (both at 1.0), highlight the reliability of these rules. The rule "If a person directed a movie, they cannot act in the same movie" also shows high confidence at 0.9722, suggesting strong consistency. Overall, the graph demonstrates the robustness and applicability of various rules within the movie database, highlighting areas of high consistency and potential improvements in data management

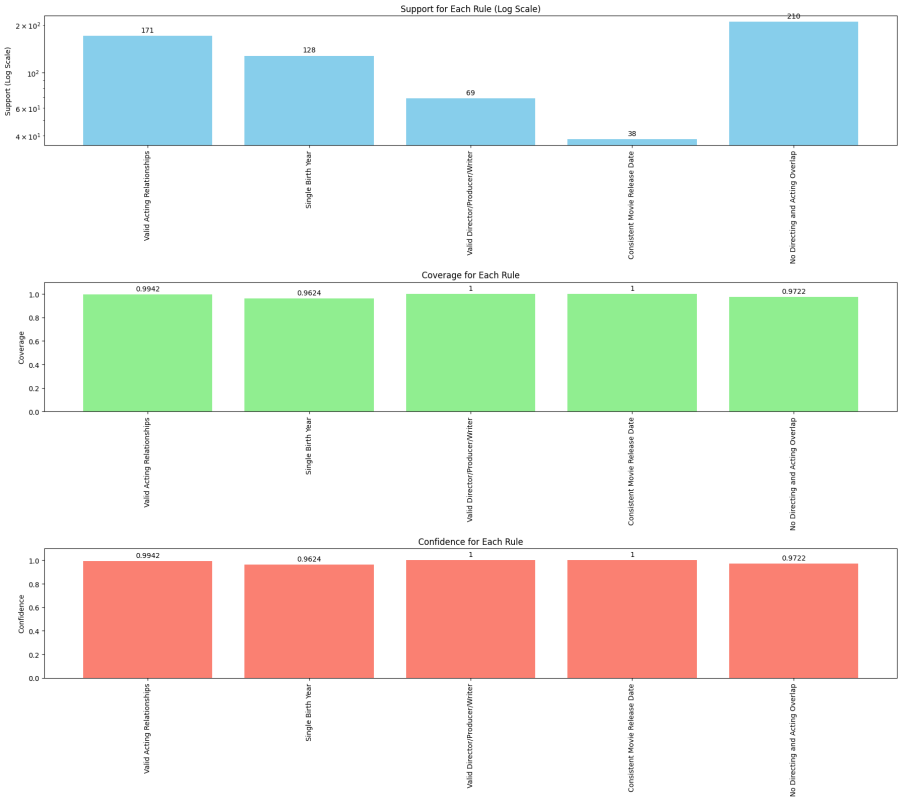

Fig. 9. Rule Ranking Measures of the Movies Graph Database Consistency Rules

6 DISCUSSION

Large Language Models (LLMs) have demonstrated remarkable capabilities in generating and extracting consistency rules from structured graph data. These rules help in identifying and correcting inconsistencies within datasets, thereby enhancing data integrity and reliability. However, there are notable challenges and limitations associated with some of the rules extracted by LLMs. These challenges underscore the need for careful evaluation and validation of the rules before their application.

Concerning the use case of the Twitter graph schema, an issue arises with the rule with the rule "A User Cannot Direct Message Another User Unless They Follow Each Other." This rule enforces a mutual consent requirement for private communication. However, the lack of a direct messaging feature or relationship in the database means this rule cannot be validated. The absence of such relationships points to a gap in the data model that needs addressing for comprehensive rule enforcement.

Ensuring privacy and preventing unwanted interactions is crucial, as highlighted by the rule "If a User Blocks Another, Their Tweets and Any Interaction Should No Longer Appear." The inability to validate this rule stems from the lack of a 'blocks' relationship in the database. Effective implementation of such rules requires a more detailed and nuanced data schema that includes blocking interactions and their consequences.

Similarly, the rule "The Count of Likes, Retweets, and Replies Must Be Accurately Reflected" emphasizes the importance of accurate interaction counts for data integrity. The absence of properties and relationships related

to likes, retweets, and replies in the database prevents the application of this rule. This highlights a critical area for data enhancement to support accurate interaction metrics. Also, the rule "A Deleted User's Tweets Should Be Removed or Marked as Deleted" is essential for maintaining database relevance by appropriately handling deleted users and their content. The lack of a 'deleted' property for users and tweets in the database underscores a significant gap. Implementing this rule requires robust mechanisms for tracking deletions and ensuring data consistency.

Concerning the rules extracted from the second use case of the movies graph, one significant challenge arises with the rule "A Movie Can't Be Released Before It Was Written, Produced, Directed, or Acted." This rule logically ensures chronological consistency in movie production. However, the support, coverage, and confidence for this rule are zero due to the absence of a 'year' property in the relationships within the database. Without this critical timestamp data, it is impossible to validate the rule, highlighting a limitation in the existing data schema rather than the rule itself.

Similarly, the rule "A Review of a Movie Must Be Written After the Movie's Release Date" aims to ensure that reviews are temporally consistent with the release dates of movies. The failure to calculate meaningful ranking measures arises from the absence of a 'review date' property in the Person node. This again underscores the importance of comprehensive and correctly attributed data properties in validating consistency rules. Another rule, "A Movie Can Have Multiple Actors, But a Release Date Should Be Consistent for a Specific Movie," ensures that a movie's release date remains consistent across its various associations. However, practical inconsistencies and real-life exceptions, such as movies with multiple release dates in different regions, make this rule controversial. While understood by humans, the rule may not always be applicable, illustrating the nuanced nature of data consistency in real-world scenarios.

The rule "If a Person Directed a Movie, They Cannot Act in the Same Movie" is straightforward and intended to maintain distinct roles for individuals within a movie. However, real-life examples of individuals who both direct and act in the same movie render this rule impractical. Such exceptions highlight the limitations of rigid rule application in complex data environments.

7 CONCLUSION

This study has demonstrated the effectiveness of Large Language Models (LLMs) in generating and validating consistency rules within complex graph datasets, such as those representing the Twitter social network and movie industry relationships. The results indicate that LLMs can significantly enhance data integrity and reliability by automating the process of rule extraction and validation, offering a scalable and efficient alternative to traditional methods.

While the study highlighted the robust capabilities of LLMs, it also revealed certain challenges, particularly when applying rigid rules to real-world scenarios with inherent complexities and exceptions. The absence of certain properties or relationships in the datasets further underscored the need for a more comprehensive data model to fully validate all inferred rules.

Looking ahead, future research could focus on refining LLMs to better handle such nuances and expanding the scope of datasets to test the models' applicability across various domains. Ultimately, this research contributes to the growing body of knowledge on the intersection of AI and data management, paving the way for more sophisticated and automated approaches to ensuring data consistency and integrity in an increasingly data-driven world.

REFERENCES

[1] Angela Bonifati, GHL Fletcher, Hannes Voigt, and Nikolay Yakovets. Querying graphs. synthesis lectures on data management. *San Rafael, CA, USA: Morgan & Claypool*, 2018.

[2] Jeff Z Pan, Simon Razniewski, Jan-Christoph Kalo, Sneha Singhania, Jiaoyan Chen, Stefan Dietze, Hajira Jabeen, Janna Omeliyanenko, Wen Zhang, Matteo Lissandrini, et al. Large language models and knowledge graphs: Opportunities and challenges. *arXiv preprint arXiv:2308.06374*, 2023.

[3] Jonathan Lajus, Luis Galárraga, and Fabian Suchanek. Fast and exact rule mining with amie 3. In *The Semantic Web: 17th International Conference, ESWC 2020, Heraklion, Crete, Greece, May 31–June 4, 2020, Proceedings 17*, pages 36–52. Springer, 2020.

[4] Bowen Jin, Gang Liu, Chi Han, Meng Jiang, Heng Ji, and Jiawei Han. Large language models on graphs: A comprehensive survey. *arXiv preprint arXiv:2312.02783*, 2023.

[5] Jinfeng Peng, Derong Shen, Nan Tang, Tieying Liu, Yue Kou, Tiezheng Nie, Hang Cui, and Ge Yu. Self-supervised and interpretable data cleaning with sequence generative adversarial networks. *Proceedings of the VLDB Endowment*, 16(3):433–446, 2022.

[6] Bahare Fatemi, Jonathan Halcrow, and Bryan Perozzi. Talk like a graph: Encoding graphs for large language models. *arXiv preprint arXiv:2310.04560*, 2023.

[7] Yun Peng, Sen Lin, Qian Chen, Lyu Xu, Xiaojun Ren, Yafei Li, and Jianliang Xu. Chatgraph: Chat with your graphs. *arXiv preprint arXiv:2401.12672*, 2024.

[8] Tom B Brown. Language models are few-shot learners. *arXiv preprint ArXiv:2005.14165*, 2020.

[9] Linhao Luo, Jiaxin Ju, Bo Xiong, Yuan-Fang Li, Gholamreza Haffari, and Shirui Pan. Chatrule: Mining logical rules with large language models for knowledge graph reasoning. *arXiv preprint arXiv:2309.01538*, 2023.

[10] Jinhao Jiang, Kun Zhou, Zican Dong, Keming Ye, Wayne Xin Zhao, and Ji-Rong Wen. Structgpt: A general framework for large language model to reason over structured data. *arXiv preprint arXiv:2305.09645*, 2023.

[11] Andrea Mauri and Alessandro Bozzon. Towards a human in the loop approach to preserve privacy in images. In *IIR*, 2021.

[12] Alessandro Bozzon, Marco Brambilla, Stefano Ceri, Andrea Mauri, and Riccardo Volonterio. Pattern-based specification of crowdsourcing applications. In *Web Engineering: 14th International Conference, ICWE 2014, Toulouse, France, July 1-4, 2014. Proceedings 14*, pages 218–235. Springer, 2014.

[13] Larissa C Shimomura, George Fletcher, and Nikolay Yakovets. Ggds: Graph generating dependencies. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, pages 2217–2220, 2020.

[14] Wenfei Fan and Ping Lu. Dependencies for graphs. *ACM Transactions on Database Systems (TODS)*, 44(2):1–40, 2019.

[15] Francesco Parisi, John Grant, et al. On database inconsistency measures. In *29th Italian Symposium on Advanced Database Systems, SEBD 2021*, volume 2994, pages 200–208. CEUR-WS, 2021.

[16] Zhikai Chen, Haitao Mao, Hang Li, Wei Jin, Hongzhi Wen, Xiaochi Wei, Shuaiqiang Wang, Dawei Yin, Wenqi Fan, Hui Liu, et al. Exploring the potential of large language models (llms) in learning on graphs. *ACM SIGKDD Explorations Newsletter*, 25(2):42–61, 2024.

[17] Jiawei Zhang. Graph-toolformer: To empower llms with graph reasoning ability via prompt augmented by chatgpt. *arXiv preprint arXiv:2304.11116*, 2023.

[18] Harshit Joshi, José Cambronero Sanchez, Sumit Gulwani, Vu Le, Gust Verbruggen, and Ivan Radiček. Repair is nearly generation: Multilingual program repair with llms. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 5131–5140, 2023.

[19] Guilong Lu, Xiaolin Ju, Xiang Chen, Wenlong Pei, and Zhilong Cai. Grace: Empowering llm-based software vulnerability detection with graph structure and in-context learning. *Journal of Systems and Software*, page 112031, 2024.

[20] Junjie Wang, Yuchao Huang, Chunyang Chen, Zhe Liu, Song Wang, and Qing Wang. Software testing with large language models: Survey, landscape, and vision. *IEEE Transactions on Software Engineering*, 2024.

[21] Bryan Perozzi, Bahare Fatemi, Dustin Zelle, Anton Tsitsulin, Mehran Kazemi, Rami Al-Rfou, and Jonathan Halcrow. Let your graph do the talking: Encoding structured data for llms. *arXiv preprint arXiv:2402.05862*, 2024.

[22] Jin Huang, Xingjian Zhang, Qiaozhu Mei, and Jiaqi Ma. Can llms effectively leverage graph structural information: when and why. *arXiv preprint arXiv:2309.16595*, 2023.

[23] Harry Li, Gabriel Appleby, and Ashley Suh. A preliminary roadmap for llms as assistants in exploring, analyzing, and visualizing knowledge graphs. *arXiv preprint arXiv:2404.01425*, 2024.

[24] Yuntong Hu, Zheng Zhang, and Liang Zhao. Beyond text: A deep dive into large language models' ability on understanding graph data. *arXiv preprint arXiv:2310.04944*, 2023.

[25] Wenfei Fan. Dependencies for graphs: Challenges and opportunities. *J. Data and Information Quality*, 11(2), feb 2019.

[26] Humza Naveed, Asad Ullah Khan, Shi Qiu, Muhammad Saqib, Saeed Anwar, Muhammad Usman, Nick Barnes, and Ajmal Mian. A comprehensive overview of large language models. *arXiv preprint arXiv:2307.06435*, 2023.

8 APPENDIX

8.1 Twitter Schema Incident Graph Encoding

The encoded graph of the twitter schema is the following:

Twitter Schema Incident Graph Encoding
G describes a social network graph among the entities Me, User, Tweet, Link, Hashtag, and Source. In this graph, Me posts Tweet; Me follows Me; Me follows User; User posts Tweet; User follows User; User follows Me; Tweet mentions Me; Tweet mentions User; Tweet retweets Tweet; Tweet reply to Tweet; Tweet contains Link; Tweet tags Hashtag; Tweet uses Source.

8.2 Twitter Data Incident Graph Encoding

The encoded graph of the Twitter data is the following:

Twitter Data Incident Graph Encoding
G describes a Twitter social network graph data. User name Neo4j (neo4j) follows User name NASA's Perseverance Mars Rover (NASAPersevere). User name Neo4j (neo4j) follows User name Galeister (galeister). User name Neo4j (neo4j) follows User name Marvello Oni (AngeliusAngel). User name Neo4j (neo4j) follows User name Kesavan Nair (Kay) (kaynairv). User name Neo4j (neo4j) follows User name James Espinosa (jamesejr7). User name Neo4j (neo4j) follows User name Holly (BytesByHolly). User name Neo4j (neo4j) follows User name Emiliano Gómez (emilianogomez33). User name Neo4j (neo4j) follows User name Marcelo Mendonça (mendonca2709). User name Neo4j (neo4j) follows User name Ahmed Hemedan (ahmed7emedan). User name Neo4j (neo4j) follows User name Ismael Velasco (DevOnAJourney). Tweet text: "Proud to help @educationgovuk deliver critical services across the #education sector with #Neo4jAuraEnterprise! L… https://t.co/p0urzRAQph" using Source name Buffer. Tweet text: "Proud to help @educationgovuk deliver critical services across the #education sector with #Neo4jAuraEnterprise! L… https://t.co/p0urzRAQph" tags Hashtag name education. Tweet text: "Proud to help @educationgovuk deliver critical services across the #education sector with #Neo4jAuraEnterprise! L… https://t.co/p0urzRAQph" tags Hashtag name neo4jauraenterprise. Tweet text: "Proud to help @educationgovuk deliver critical services across the #education sector with #Neo4jAuraEnterprise! L… https://t.co/p0urzRAQph" contains Link url https://twitter.com/i/web/status/1371815021265747970. Tweet text: "Proud to help @educationgovuk deliver critical services across the #education sector with #Neo4jAuraEnterprise! L… https://t.co/p0urzRAQph" mentions User name Department for Education (educationgovuk). Tweet text: "[Webinar] Produktdatenmanagement mit der Graphdatenbank Neo4jDonnerstag, 18. März // 11:00 CET Register here:… https://t.co/OYvzyt2TBD" using Source name Buffer. Tweet text: "[Webinar] Produktdatenmanagement mit der Graphdatenbank Neo4jDonnerstag, 18. März // 11:00 CET Register here:… https://t.co/OYvzyt2TBD" contains Link url https://twitter.com/i/web/status/1371778287970705408. User name Denisse (ArtemisaVm) follows User name Neo4j (neo4j) etc.

8.3 Movies Schema Incident Graph Encoding

The encoded graph of the Movies schema is the following:

Movies Schema Incident Graph Encoding
G describes a movie graph among the entities Movie, Person. In this graph, Person reviewed Movie; Person acted in Movie; Person produced Movie; Person wrote Movie; Person directed Movie; Person follows Person.

8.4 Movies Data Incident Graph Encoding

The encoded graph of the Movies data is the following:

Movie Data Incident Graph Encoding
G describes a Movies graph data. Person Keanu Reeves, born 1964, acted in Movie The Matrix, released 1999. Person Keanu Reeves, born 1964, acted in Movie The Matrix Reloaded, released 2003. Person Keanu Reeves, born 1964, acted in Movie The Matrix Revolutions, released 2003. Person Keanu Reeves, born 1964, acted in Movie The Devil's Advocate, released 1997. Person Keanu Reeves, born 1964, acted in Movie The Replacements, released 2000. Person Keanu Reeves, born 1964, acted in Movie Johnny Mnemonic, released 1995. Person Carrie-Anne Moss, born 1967, acted in Movie The Matrix, released 1999. Person Laurence Fishburne, born 1961, acted in Movie The Matrix Reloaded, released 2003. etc.