LoSA: A Local Structural Approach to Adversarial Attack on the Knowledge Graph-based Question Answering System

Neha Pokharel[1], Arnab Sharma[1], Adel Memariani[1], Michael Röder[1] and Axel-Cyrille Ngonga Ngomo[1]

[1]*Heinz Nixdorf Institute, Paderborn University, Paderborn, Germany*

Abstract

Knowledge graph-based question answering (KGQA) systems are increasingly employed to retrieve accurate answers to natural language queries by leveraging structured knowledge graphs. Since such KGQA systems are frequently being deployed in many critical domains, the integrity of such systems under *adversarial threat* is of utmost importance. Although a number of works have studied how to make a KGQA system which can generate the most correct answers to the given query, the robustness of such systems is relatively less studied. To fill this gap, in this paper, we introduce an adversarial attack approach to systematically evaluate and exploit the vulnerability of KGQA systems to data poisoning attacks. More specifically, we poison the underlying knowledge graph so that the KGQA system returns wrong answers corresponding to a target question. This is done in a black-box setting, requiring only query access to the KGQA system and no internal knowledge of its architecture. Considering a KGQA system that utilizes DBpedia and Wikidata knowledge graphs, we find that our adversarial attack, albeit being simple, is quite effective in generating false answers. Additionally, we assess the *stealthiness* of our attack approach by considering the performance of the KGQA system on the untargeted queries and the underlying knowledge graphs. Our results highlight the need for a research direction in developing robust KGQA against data poisoning attacks on such systems.

1. Introduction

Knowledge graph-based question answering (KGQA) systems are increasingly essential in applications such as virtual assistants, biomedical search engines, and enterprise knowledge discovery tools [1, 2]. These systems aim to answer natural language questions by mapping them to structured queries over knowledge graphs, which offer rich semantic representations of real-world entities and relations. One of the earliest works to this end is by Berant et al. [3], which relies on semantic parsing or template-based matching to perform question-answering. The core idea therein is to map natural language phrases to logical predicates by leveraging a knowledge base and a large text corpus. With the advent of *attention*-based approaches, existing KGQA systems mostly incorporate transformer-based models to enhance question understanding and reasoning capabilities. For instance, Sun et al. proposed GRAFT-Net [4] that

Wikidata'25: Wikidata workshop at ISWC 2025

neha2022@mail.uni-paderborn.de (N. Pokharel); arnab.sharma@uni-paderborn.de (A. Sharma); adel.memariani@uni-paderborn.de (A. Memariani); michael.roeder@uni-paderborn.de (M. Röder); axel.ngonga@uni-paderborn.de (A. N. Ngomo)

CEUR Workshop Proceedings (CEUR-WS.org)

uses graph-based attention for open-domain QA, while actively retrieving relevant subgraphs during inference. In recent years, using sequence-to-sequence models for transforming natural language questions into SPARQL without relying on hand-crafted rules or statistics has shown effective performance [5, 6, 7]. SPARQL-based KGQA systems such as MST5 [7] work by translating natural language questions into SPARQL queries by jointly performing entity linking and query generation using a multilingual sequence-to-sequence model. The generated query is then executed over a knowledge graph to retrieve the answer, without relying on handcrafted rules or templates. Note that the existing works have mostly focused on building KGQA systems that achieve high accuracy and can work with multiple languages. To the best of our knowledge, only a few works [8, 9, 10] have studied the robustness of some specific types of KGQA systems, considering the potential security threats that can arise therein.

To this end, several researchers have extensively studied different *adversarial* attack strategies on the embedding models considering the *link prediction* tasks by poisoning the knowledge graph (KG) or by performing adversarial manipulations of the embedding model [11, 12, 13, 14, 15, 16, 17, 18]. The fundamental concept behind these attacks is to focus on a particular fact in the KG and poison the graph in such a way so that the link prediction for a specific triple decreases. Beyond the link prediction tasks, only a few works considered studying the robustness of the KGQA system. One of the prominent works to this end by Xi et al. [8] perform a surrogate model-based attack on a custom-made multi-hop reasoning KGQA system by poisoning the underlying biomedical KG. Ma et al. [10] recently proposed an adversarial attack on KG-augmented QA systems, which combine pretrained language models with external knowledge graphs. Through multi-round entity selection, prompt crafting, and pruning, it reconstructs a high-utility KG with minimal queries and performance loss. These approaches paved the way for some initial assessment of the robustness of some specific types of KGQA systems. However, to the best of our knowledge, no works have considered evaluating the robustness of the SPARQL-based KGQA systems. Building a robust KGQA system is of high importance given the real-world manipulations done on the open-source KGs. For instance, Wikidata has been targeted in the past by coordinated misinformation campaigns, where malicious edits introduced biased or false facts about political figures or controversial events [1].

In this work, we address this gap by introducing an adversarial approach called *localized structural attack* (LoSA) for the SPARQL-based KGQA system. Considering a black-box setting, our approach works by modifying the underlying knowledge graph to mislead the system into returning plausible yet incorrect answers. More specifically, we systematically update a small selected subset of Resource Description Framework (RDF) triples, associated with a question–answer pair, to disrupt the execution of the corresponding SPARQL query for the target question. The attack is black-box in nature since it does not require any access to internal model parameters or training data. It only leverages the system's query interface and the structure of the KG. Additionally, our attack approach LoSA is lightweight, model-agnostic, and suitable for evaluating a wide range of SPARQL-based QA systems. Since it focuses on altering only the localized region of the KG relevant to a given query, the attack remains stealthy to the other QA pairs. Note that the attack scenario we considered in this work is quite plausible. Since the KGQA systems often use KGs such as DBpedia [19], or Wikidata [20], which are open

[1] https://en.wikipedia.org/wiki/List_of_political_editing_incidents_on_Wikipedia

source, thereby giving the attackers the possibility to poison the graph.

We evaluate the effectiveness of our adversarial approach by considering a specific state-of-the-art multilingual KGQA system MST5 [7]. To this end, we consider two different underlying KGs, namely DBpedia and Wikidata. We find that even minimal, targeted manipulations can substantially impair QA performance. To this end, we also evaluate the *stealthiness* of the adversarial approach. Our results indicate our approach mostly remains undetected as it does not lead to wrong answers for the other queries, apart from the one that is targeted.

2. Related Work

In the context of performing malicious attacks on KGQA approaches, not many works can be found in the literature. Precisely, in this context, the existing works mostly focused on poisoning the KGs in order to maliciously manipulate the link prediction tasks [11, 12, 13, 14, 15, 16, 17, 18]. One of the earliest works to this end is from Zhang et al. [11] who introduced a KG poisoning attack strategy to perform adversarial attacks on the link prediction task. This is done by shifting the embedding vector of the target triple by adding or removing new connections in the KG. To make the attack stealthy, the authors proposed *indirect attacks* that involve adding or removing triples in the KG that are not directly connected to the target triple. Pezeshkpour et al. [12] used a gradient-based approach to find out the most influential neighboring triples of the target fact and remove them. However, their approach is limited to only a particular type of KGE models. Bhardwaj et al. [13] used inductive relationships such as symmetry, inversion, and composition within the knowledge graph to perform adversarial attacks. The idea therein is to exploit these relationships to add or remove triples, which ultimately helps the attackers to achieve their goal. In a later work, they [14] utilized a technique from the explainability domain, namely instance attribution, to perform data poisoning attacks on KGE models. Attribution methods are used therein to identify training triples that most influence a target prediction, which are then altered by removing or modifying one of their entities. You et al. [15] proposed black-box data poisoning attacks that maintain stealthiness by adding semantically preserving triples. Unlike prior work, they introduce indicative paths—multi-triple structures that boost the plausibility of target triples. Zhao et al.[17] used logical rules to identify triples whose removal most harms KGE performance. Kapoor et al. [18] performed adversarial attacks by considering three different attack surfaces: KG, the embeddings, and the labels of the training data.

Considering the adversarial attacks on the KGQA systems, only a few works can be found in the literature [8, 9, 10]. The work closest to ours is by Xie et al. [8] wherein they proposed ROAR (Reasoning Over Adversarial Representations), a suite of adversarial attack techniques against a knowledge graph reasoner used as a multi-hop QA system. The approach works by injecting a small set of carefully selected triples into the knowledge graph by first optimizing their embeddings in latent space to redirect reasoning outcomes toward attacker-specified answers. These optimized embeddings are then mapped back to symbolic triples using a heuristic search based on relation-specific projection operators and fitness scores. It works in an iterative co-optimization loop, alternately refining poisoning triples and misleading query components. Puerto et al. [9] introduced UKP-SQuARE, an enhanced online platform aimed at supporting research in multi-agent question answering. Amongst other features, UKP-SQuARE

also incorporates an adversarial attack module designed to evaluate the robustness of QA models against input perturbations. This component generates modified versions of the input questions to probe and expose vulnerabilities in model behavior. It supports adversarial testing as a first-class feature within its user-friendly interface, requiring no code from the user. However, this platform does not support any SPARQL-based KGQA system. In a recent work, Ma et al. [10] proposed KGDist, a prompt-based distillation attack targeting a KGQA system equipped with a language model augmented with a knowledge graph. The idea herein is to extract a specific task-relevant subgraph from a KG+LM model using the prompts and the outputs of the QA system. The attack proceeds by initializing a small set of core entities from a domain-specific corpus and iteratively expanding this set by querying the model and selecting highly confident entity pairs. Then it employs a multi-granularity prompt construction strategy to query for relationships between entities while minimizing query overhead. Finally, relation-type-based pruning is applied to remove redundant or cyclic edges, improving the extracted graph.

In contrast to the above approaches, we target a specific class of KGQA systems, namely SPARQL-based KGQA. While prior works (e.g., [12, 13, 14, 17]) also rely on graph modifications, they are tailored to link prediction or embedding-based reasoning tasks and do not consider the SPARQL query generation process. LoSA differs by explicitly aligning graph perturbations, where query generation occurs, ensuring that changes in the KG translate into systematically misleading SPARQL queries. Although LoSA also treats the SPARQL engine as a black box, its design uniquely exploits the structural dependencies of SPARQL query construction. This makes it, to the best of our knowledge, the first approach that directly performs adversarial attacks and further evaluates the robustness in SPARQL-based KGQA systems.

3. Threat Model

We define the threat model in performing the attack on the KGQA systems. The KGs often harvest data from open-source resources, such as DBPedia or Wikidata, which opens an attack window for adversaries to introduce perturbations. Specifically, in our adversarial attack approach, we assume the following threat model.

- **Access to the data:** We assume that the attacker only has access to the underlying knowledge graph, which is utilized to give answers to the given query.
- **Black-box system:** Since we assume a black-box nature of the system, we assume that the adversary has no knowledge of the internal architecture or training strategy of the target KGQA system. As mentioned beforehand, since the KGs harvest data from open source, we assume that the attacker has access to the graph.
- **Attack constraints:** Considering the attacker's access to the KGs, we assume that the adversary operates under the following practical limitations to ensure realism in our attack setting. 1. The adversary cannot create new entities or relations in the KG. 2. The adversary cannot insert repetitive triplets. 3. The adversary can remove triples from the KG considering a specific subgraph. 4. The adversary is allowed to remove only a limited number of triples. This is bounded by the *attack budget*, which essentially determines the percentage of the triples to be removed.

4. Overview of Approach

In this work, we present an adversarial attack approach to change the answer to a given query. This is done based on the structure of the KG to mislead the KGQA system by making minimal, targeted modifications. The main strategy involves, first of all, identifying the structure of the targeted QA pair. Afterwards, the triples related to the target QA pairs are removed by taking into consideration that it minimally changes any other parts of the KG. In other words, the attack should not change the answers of any other queries that are not being targeted. This basically ensures the stealthiness of the approach. We start by giving a generic description of the KGQA system on which the adversarial attack is performed.

4.1. KGQA System

Let $\mathcal{E}$ be the set of entities and $\mathcal{R}$ the set of relations that exist between these entities. A triple (h, r, t) comprises a head and a tail entity ($h, t \in \mathcal{E}$) and a relation $r \in \mathcal{R}$ that holds between them. We define a knowledge graph $\mathcal{G}$ as a collection of triples:

$$\mathcal{G} := \{(h, r, t) \in \mathcal{E} \times \mathcal{R} \times \mathcal{E}\}\,. \tag{1}$$

Given a natural language question q, a KGQA system is designed to retrieve accurate answers from a knowledge graph by first translating the question into a formal query language such as SPARQL [21, 22]. This translation maps the semantic intent of the question to a structured SPARQL query $S(q)$, which is then executed over the knowledge graph $\mathcal{G}$. The execution retrieves a subgraph $\mathcal{G}_q \subseteq \mathcal{G}$, containing the relevant triples that satisfy the query constraints. The answer a^* for the given question q is extracted directly from the result bindings of the SPARQL query. To give a concrete example, if the question q is "What is the capital of France?", the system first converts it into a SPARQL query such as,

```
SELECT ?capital WHERE {
  wd:Q142 wdt:P36 ?capital .
}
```

Here, `wd:Q142` corresponds to the Wikidata entity for France and `wdt:P36` denotes the `hasCapital` property. When executed over the knowledge graph, this query retrieves the triple $(\texttt{France}, \texttt{hasCapital}, \texttt{Paris})$, from which the answer `Paris` is extracted as a^*.

4.2. Local Structural-based Adversarial Attack

In our approach, we target a natural language question q to change its correct answer from a^* to some wrong answer $\tilde{a}$. As mentioned beforehand, herein we assume that we only have access to the underlying KG $\mathcal{G}$. To this end, firstly, given the query, we generate its corresponding SPARQL query $S(q)$ using a text-to-SPARQL translator. Since we assume the attacker has access to the KG, using $S(q)$ a subgraph $\mathcal{G}_q \subseteq \mathcal{G}$ can be retrieved by executing $S(q)$ over the knowledge graph $\mathcal{G}$. Then a minimal subset of triples $\mathcal{G}'_q \subseteq \mathcal{G}_q$ with $|\mathcal{G}'_q| \leq k$ is removed from the graph. Herein, k is the attack budget, indicating the maximum number of triples that can

be removed. The modified subgraph $\mathcal{G}_q \setminus \mathcal{G}'_q$ should then cause the KGQA system to return a false but plausible answer $\tilde{a} \neq a^*$. Essentially, our adversarial attack approach evaluates the robustness of a KGQA system by targeting a single question–answer pair at a time. Our approach comprises of the following stages, which we outline briefly below.

1. **SPARQL Generation:** Translate the target natural language question q into a SPARQL query $S(q)$ using a text-to-SPARQL generator.
2. **Subgraph Retrieval:** Execute the query on the KG $\mathcal{G}$ to retrieve a localized subgraph around the answer entity.
3. **Graph Alteration** Remove a set of triples ($\leq k$) from the KG $\mathcal{G}$ to cause the system to output an incorrect response.

Below, we describe each step in detail.

4.3. SPARQL Generation

We focus on adversarially modifying the knowledge graph with respect to a specific natural language question q. To enable this, the corresponding formal SPARQL query $S(q)$ is first obtained through a query generation function $S : \mathcal{Q} \to \mathcal{S}$ that maps natural language questions $q \in \mathcal{Q}$ to executable SPARQL queries $S(q) \in \mathcal{S}$ over a knowledge graph $\mathcal{G}$. Specifically, this process involves syntactic parsing, entity linking $\phi : q \to \mathcal{E}^*$, and relation linking $\psi : q \to \mathcal{R}^*$, as well as the instantiation of query templates. However, we treat this query generation step as a black-box component of the QA pipeline and do not intervene in its internal functionalities. The resulting SPARQL query $S(q)$ is then used as the basis for extracting the relevant subgraph of $\mathcal{G}_q$ on which our adversarial modifications operate. As an illustrative example, consider the question q as,

Who is the daughter of the person who discovered Radium?

Given this input, the KGQA system automatically identifies the entity mention "Radium" and links it to `dbr:Radium`, while recognizing relevant relations such as `dbo:discoverer` and `dbo:child`. Based on the underlying semantic structure, it generates the following SPARQL query:

```
PREFIX dbr: <http://dbpedia.org/resource/>
PREFIX dbo: <http://dbpedia.org/ontology/>

SELECT ?daughter WHERE {
  ?discoverer dbo:discoverer dbr:Radium .
  ?discoverer dbo:child ?daughter .
}
```

This query retrieves all entities bound to the variable `?daughter` such that there exists a subject `?discoverer` who is both linked to the discovery of `dbr:Radium` and is the parent of the entity in question. The corresponding set of RDF triples selected by this query forms the subgraph $\mathcal{G}_q$ targeted for our adversarial intervention.

4.4. Subgraph Retrieval

We construct a question-specific subgraph $\mathcal{G}_q \subseteq \mathcal{G}$ centered around the correct answer a^*. We begin by identifying the IRI corresponding to a^*, as returned by the execution of the SPARQL query $S(q)$. Using this IRI, we retrieve all RDF triples from the global knowledge graph $\mathcal{G}$ in which a^* appears either as the subject or object. To ensure the semantic relevance of the extracted subgraph with respect to the original question, we further restrict this set by retaining only those triples whose predicates are also referenced in the query $S(q)$. This filtering step yields a localized, semantically coherent subgraph that reflects the context in which the answer is derived. If the number of resulting triples exceeds a predefined threshold (e.g., 200), we uniformly sample a subset to limit the computational overhead and bound the attack scope. Considering the running example, such a subgraph is depicted in Figure 1.

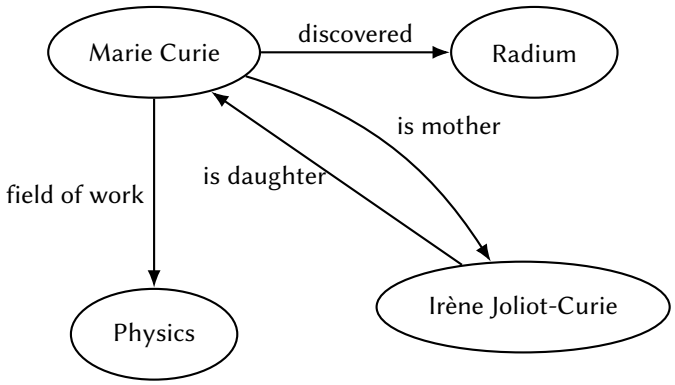

Figure 1: Subgraph retrieval for: *Who is the daughter of the person who discovered Radium?*

4.5. Graph Alteration

This stage represents the core of the adversarial attack, which aims to deliberately alter the knowledge graph $\mathcal{G}$ such that the answer returned by the KGQA system for a given query becomes incorrect. To constrain the attack's scope, we define a removal budget $k \in \mathbb{N}$ representing the maximum number of triples that can be removed. If the size of $\mathcal{G}_q$ exceeds a predefined threshold θ, we uniformly sample a subset $\mathcal{G}'_q \subseteq \mathcal{G}_q$ such that $|\hat{\mathcal{G}}_q| = k$. The triples of $\mathcal{G}'_q$ are removed from the knowledge graph $\mathcal{G}$ using a simple batch update operation used in SPARQL as SPARQL DELETE DATA. This update is executed through the SPARQL update endpoint (e.g., Apache Fuseki). After modifying the graph, the query q is given to the KGQA system which now uses a modified KG. Let $\tilde{a}$ denote the new answer returned. The attack is considered *successful* if $\tilde{a} \neq a^*$, i.e., the answer changes due to the modification of the underlying knowledge graph. This final comparison determines the effectiveness of the adversarial manipulation.

Algorithm 1 outlines a local structural adversarial attack strategy against a KGQA system. It targets a single natural language question $q \in \mathcal{Q}$ by first translating it into a SPARQL query $S(q)$ using a query generation module $\mathcal{T} : \mathcal{Q} \to \mathcal{S}$ (Step 1). The correct answer $a^* \in \mathcal{E}$ is obtained by executing $S(q)$ over the knowledge graph $\mathcal{G}$. A question-specific subgraph $\mathcal{G}_q \subseteq \mathcal{G}$

is then constructed by retrieving all RDF triples involving a^*, filtered by predicates appearing in $S(q)$ (Step 2). If the subgraph size exceeds a predefined threshold, a random sample is taken. From this subgraph, a subset $T' \subseteq \mathcal{G}_q$ of k triples is selected and deleted from $\mathcal{G}$ (Step 3). The query $S(q)$ is re-executed over the modified graph $\mathcal{G} \setminus T'$ to verify whether the returned answer $\tilde{a}$ differs from a^*. If so, the attack is deemed successful (Step 4).

Algorithm 1: Local structural adversarial attack

Input: Target question $q \in \mathcal{Q}$, SPARQL generator $\mathcal{T}$, knowledge graph $\mathcal{G}$, removal budget k

Output: Boolean success

Step 1: Answer Retrieval

$S(q) \leftarrow \mathcal{T}(q)$; `// Generate SPARQL query`

$a^* \leftarrow \texttt{Execute}(S(q), \mathcal{G})$; `// Get ground truth answer`

Step 2: Subgraph Construction

$\mathcal{G}_q \leftarrow \{(s,p,o) \in \mathcal{G} \mid s = a^* \vee o = a^*, p \in \texttt{pred}(S(q))\}$

if $|\mathcal{G}_q| > \tau$ **then**

  $\mathcal{G}_q \leftarrow \text{UniformSample}(\mathcal{G}_q, \theta)$; `// Limit to` θ `triples`

Step 3: Triple Removal

$T' \leftarrow \texttt{SelectFirstK}(\mathcal{G}_q, k)$

$\mathcal{G} \leftarrow \mathcal{G} \setminus T'$; `// Apply SPARQL DELETE DATA`

Step 4: Attack Evaluation

$\tilde{a} \leftarrow \texttt{Execute}(S(q), \mathcal{G})$

success $\leftarrow (\tilde{a} \neq a^*)$

return success

5. Evaluations & Results

In this work, we aim to find out whether the existing SPARQL-based KGQA systems are robust to the adversarial modifications done on the underlying KG. To this end, we consider a multilingual KGQA system MST5 [7] which typically utilizes either DBPedia or Wikidata knowledge graphs as the underlying knowledge sources. Note that we chose MST5 since it is an open-source SPARQL-based KGQA framework, supporting multiple languages and offering end-to-end pipelines from natural language questions to SPARQL queries. Furthermore, it is designed to work with large, real-world KGs such as DBpedia and Wikidata, making it a representative system for evaluating robustness in practical SPARQL-based QA settings.

Table 1 gives the statistical information of the DBpedia [2] and Wikidata [3] knowledge graphs. In our evaluation, we have considered the Wikidata Dump from November 2020 and DBpedia from October 2016. We consider the adversarial attack considering both the KGs. In our evaluation,

[2] https://wiki.dbpedia.org/about

[3] https://www.wikidata.org/wiki/Wikidata:Statistics

Table 1
An overview of the knowledge graphs considered as the underlying knowledge for the KGQA MST5.

Aspect	DBpedia	Wikidata
#Triples	∼1.3 billion	∼15 billion
#Number of Entities	∼6.6 million (English edition)	∼100 million
#Properties	∼3,000	>9,000
#Supported Languages	∼125	>300
#RDF Classes	∼760	∼50,000

we began with a dataset of 486 natural language questions. After mapping each question to its corresponding SPARQL query and retrieving the associated answers from the knowledge graph, 216 questions were discarded due to missing or invalid query-answer pairs, leaving 270 valid entries. From this set, we further filtered out 62 questions whose answers did not contain valid IRIs, resulting in 208 questions. Note that since the attack operates on subgraphs extracted from the KG, it requires that answers are entity IRIs to ensure that an associated subgraph can be constructed and manipulated. Literal answers (e.g., dates or numbers) do not yield meaningful graph structures for perturbation, and thus fall outside the scope of our evaluation.

Our experimental evaluation is driven mainly by the following research questions.

RQ 1. *Can our adversarial attack successfully fool the KGQA system into giving false answers?*

RQ 2. *Does our adversarial attack maintain stealthiness by preserving the correctness of QA pairs that are not directly targeted?*

Finally, to find out which QA pairs are easy to target, we consider the following RQ.

RQ 3. What specific types of questions showed vulnerability to adversarial manipulation?

Below, we discuss the evaluations and the corresponding results for each of these RQs.

RQ 1. To address this question, we evaluate the *attack success rate* of our attack approach on the multilingual KGQA system MST5 [7], using both DBpedia and Wikidata as the underlying knowledge graphs. The attack success rate quantifies the proportion of questions for which the system's predicted answer changes to another answer due to adversarial modifications made to the knowledge graph. Note that this metric is primarily used to evaluate the robustness of the KGQA system against adversarial attacks. Formally, it can be defined as follows.

$$\text{Success Rate} = \left(\frac{\text{Number of Successful Attacks}}{\text{Total Number of QA Pairs}} \right) \times 100\% \tag{2}$$

Note that, herein, we do not consider an attack as successful if, after the attack, the answer remains empty for a target question, i.e., no answers are given.

The results are illustrated in Figure 2, which shows the success rate for varying attack budget k, i.e., the number of triples allowed to be removed from the knowledge graph during the attack. Herein, we see that, on DBpedia, the attack success rate peaks at 33.3% for $k = 7$, indicating that our approach is effective in perturbing the QA system. Note that this can be interpreted as 33.7% of the QA pairs can be changed to generate false answers amongst 208 questions, i.e., the attack remains successful therein. As mentioned beforehand, the attack is considered successful only when the answer of the targeted question is changed to another answer. However, we find that if we consider the attack as successful when the answer is either changed or empty (i.e, no

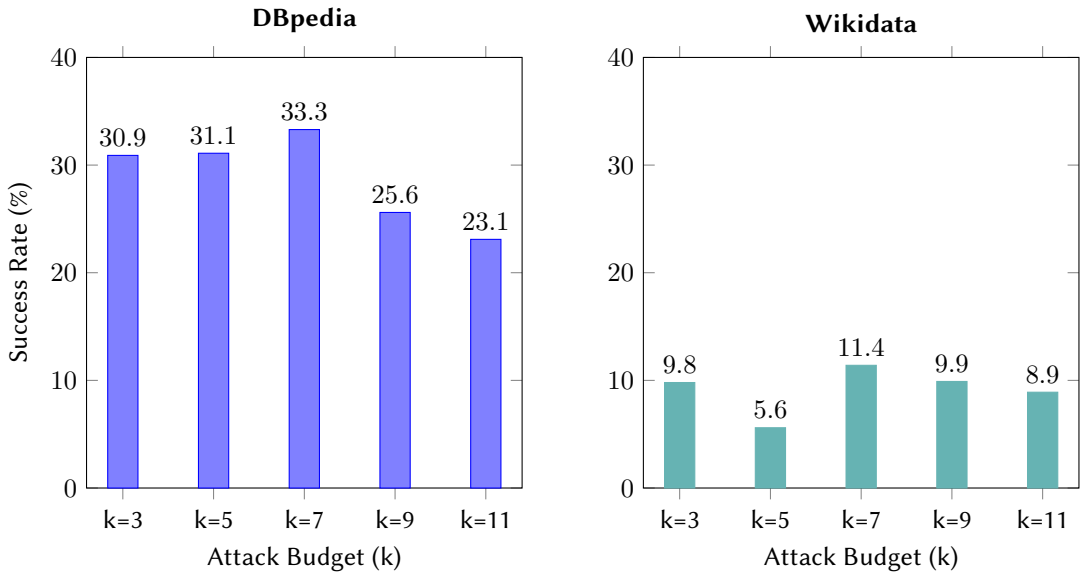

Figure 2: Attack success rate as a function of attack budget k on DBpedia and Wikidata.

answer at all), then our attack can effectively change $\sim 70\%$ of the QA pairs.

In contrast to DBpedia, the success rate on Wikidata remains lower across all budgets. Note that, to this end, we find out that, if we consider the empty answers also successful, then only around 60% of cases our attack remains successful, which is much lower than DBpedia's 70% of cases. The disparity in attack success rates between DBpedia and Wikidata stems from several structural and semantic differences in the underlying knowledge graphs. For instance, this can be attributed to structural and semantic differences in the two knowledge graphs, as well as our strict definition of success—i.e., only considering cases where the system returns a *valid but incorrect* answer (empty answers are not counted as successful).

Additionally, Wikidata has higher structural redundancy and a normalized schema (via property IDs). In contrast, DBpedia's sparser structure, flatter schema, and less standardized predicates make it easier to mislead the system into returning plausible but wrong answers. Moreover, Wikidata's rich aliasing and complex SPARQL queries often preserve answerability, whereas DBpedia's simpler structure makes the system more vulnerable to minimal, targeted deletions. These differences explain the higher success rates observed on DBpedia.

Finally, we see that with the attack budget k of 7, we get the best results for both the DBpedia and Wikidata. The performance fluctuates beyond that point and essentially degrades. This suggests possible over-pruning or irrelevant triple removal. Moreover, this can be attributed to the fact that we only consider the attack cases as successful where the system returns a valid but incorrect answer. Beyond the attack budget of 7, we mostly get empty answers, therefore lowering the attack success rate of the attack on the system.

These findings highlight the fact that KGQA robustness is not solely determined by the QA model, but is deeply influenced by the structure and design of the underlying knowledge graph.

RQ 2. To answer this research question, we look into two metrics, *functional stealthiness* and

structural stealthiness. We measure the functional stealthiness by considering the performance of the KGQA system under attack, using the GERBIL-QA framework [23]. An attack is considered functionally stealthy if it successfully degrades performance on its intended targets while causing minimal, or ideally zero, performance degradation on the QA pairs that were not targeted. To this end, we measure the F1 score of the KGQA system.

Our evaluation using the GERBIL-QA framework on the QALD dataset demonstrates that our adversarial attack exhibits strong functional stealthiness across a range of attack budgets. Specifically, the F1 scores remain consistently high: 0.9851 for a budget of $k = 3$, 0.9807 for $k = 5$, and 0.9782 for $k = 7$. These results suggest that our attack strategy causes minimal collateral damage, preserving the correctness of unrelated QA pairs while still being effective.

While the GERBIL QA analysis confirms that all attacks are functionally stealthy, a comprehensive evaluation also requires assessing their structural impact on the underlying knowledge graph. To this end, we analyze the integrity of the KG by measuring changes in three key centrality metrics after an attack: PageRank [24], betweenness-centrality [25], and eigenvector-centrality [26]. PageRank captures the global influence of a node based on the overall link structure. Betweenness Centrality reflects the extent to which a node acts as a bridge along the shortest paths between other nodes. Eigenvector Centrality identifies nodes that are not only well-connected but also linked to other highly influential nodes. Considering the most successful attack budget k of 7, we find out that pagerank, betweenness-centrality, and eigenvector-centrality measures change ~ 0.002, ~ 0.0003, and ~ 0.0530 before and after the attack. These minimal changes indicate that the attacks preserve the global structural properties of the KG to a large extent, reinforcing their stealthiness not only at the QA level but also in terms of topological footprint. This suggests that such attacks can evade standard graph integrity checks, highlighting the need for more sensitive detection methods.

RQ 3. An analysis of the successful adversarial attacks reveals that system failures occurred predominantly on complex questions that required the integration of multiple pieces of information, rather than on simple fact-based queries. In particular, vulnerabilities were most evident in questions that demanded (a) comparison and ranking, such as those involving superlatives like 'longest', 'largest', or 'oldest'; (b) conjunctive reasoning, which requires satisfying multiple constraints simultaneously, for instance, `find out the films directed by a specific individual and featuring a particular actor`; and (c) relational inference, where the system must interpret and navigate hierarchical or indirect relationships, such as familial ties or organizational affiliations.

This failure pattern underscores a critical limitation in the system's reasoning capabilities. The KGQA demonstrates strong performance on questions that require retrieving discrete facts, suggesting that such information is well-represented in the underlying knowledge graph or embedding space. However, it struggles significantly when tasked with composing, aggregating, or comparing facts across multiple entities or relations. This implies that the system's internal representation of knowledge is largely fragmented, storing facts in isolation rather than as part of a cohesive, interconnected structure.

Consequently, the KGQA system lacks robustness when reasoning chains are required. For questions that depend on multi-hop inference or the combination of multiple intermediate facts, the answer becomes susceptible to disruption at any single point in the reasoning chain. Therefore, an adversary requires only to compromise one component of this chain, such as

removing a critical triple or modifying a relation to induce a system failure. This not only highlights a potential attack surface for adversarial interventions but also points to a deeper architectural challenge in current KGQA systems, specifically the limited capacity for relational composition and structured reasoning.

6. Conclusion

In this work, we present LoSA, a localized structural attack strategy designed to systematically probe and expose vulnerabilities in SPARQL-based KGQA systems. Considering a realistic black-box threat model, LoSA perturbs only a minimal subset of RDF triples directly related to the query-specific subgraph, thereby effectively misleading the KGQA system into returning incorrect answers. Our empirical evaluation on the state-of-the-art multilingual system MST5, using both DBpedia and Wikidata KGs, demonstrates that such attacks can achieve high success rates, especially on DBpedia, while preserving functional and structural stealthiness. The attacks degrade performance on targeted queries without impacting unrelated QA pairs or disrupting global graph topology, thereby evading common detection strategies. However, we also find that using Wikidata in the KGQA system makes it inherently robust, as the graph is sparse and well-connected. Therefore, even if the attacker changes some connections, the knowledge graph remains partially intact after the poisoning.

Additionally, our analysis reveals that KGQA systems are particularly susceptible to adversarial manipulation when confronted with complex questions involving comparison, conjunction, or relational inference. These findings highlight fundamental limitations in the systems' reasoning capabilities and their reliance on loosely connected factual representations. We believe that this work can serve as a baseline, and more sophisticated attack approaches can be built in this context. More importantly, through our work, we pose an urgent need for robust defense mechanisms that can detect and mitigate such structurally minimal yet semantically impactful attacks. As part of future work, we will explore advanced attack strategies as well as adaptive defense strategies, such as graph sanitization, anomaly detection, and robust query planning, to make the KGQA pipelines robust against adversarial threats.

Acknowldgement

This work has been supported by the Ministry of Culture and Science of North Rhine-Westphalia (MKW NRW) within the project SAIL under the grant no NW21-059D, the project "WHALE" (LFN 1-04) funded under the Lamarr Fellow Network programme by the Ministry of Culture and Science of North Rhine-Westphalia (MKW NRW), and the European Union's Horizon Europe research and innovation programme under grant agreement No 101070305.

References

[1] M. Sarrouti, S. O. E. Alaoui, Sembionlqa: A semantic biomedical question answering system for retrieving exact and ideal answers to natural language questions, Artif. Intell.

Medicine 102 (2020) 101767. URL: https://doi.org/10.1016/j.artmed.2019.101767. doi:10.1016/J.ARTMED.2019.101767.

[2] A. Perevalov, A. Both, A. N. Ngomo, Multilingual question answering systems for knowledge graphs - a survey, Semantic Web 15 (2024) 2089–2124. URL: https://doi.org/10.3233/SW-243633. doi:10.3233/SW-243633.

[3] J. Berant, A. Chou, R. Frostig, P. Liang, Semantic parsing on freebase from question-answer pairs, in: Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, EMNLP 2013, 18-21 October 2013, Grand Hyatt Seattle, Seattle, Washington, USA, A meeting of SIGDAT, a Special Interest Group of the ACL, ACL, 2013, pp. 1533–1544. URL: https://doi.org/10.18653/v1/d13-1160. doi:10.18653/V1/D13-1160.

[4] H. Sun, T. Bedrax-Weiss, W. W. Cohen, Pullnet: Open domain question answering with iterative retrieval on knowledge bases and text, in: K. Inui, J. Jiang, V. Ng, X. Wan (Eds.), Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019, Association for Computational Linguistics, 2019, pp. 2380–2390. URL: https://doi.org/10.18653/v1/D19-1242. doi:10.18653/V1/D19-1242.

[5] T. Soru, E. Marx, A. Valdestilhas, D. Esteves, D. Moussallem, G. Publio, Neural machine translation for query construction and composition, arXiv preprint arXiv:1806.10478 (2018). URL: https://arxiv.org/abs/1806.10478.

[6] M. A. Borroto, F. Ricca, B. Cuteri, A system for translating natural language questions into sparql queries with neural networks: Preliminary results, in: SEBD 2021: Italian Symposium on Advanced Database Systems, RWTH Aachen, 2021, pp. 226–234.

[7] N. Srivastava, M. Ma, D. Vollmers, H. M. Zahera, D. Moussallem, A. N. Ngomo, MST5 - multilingual question answering over knowledge graphs, CoRR abs/2407.06041 (2024). URL: https://doi.org/10.48550/arXiv.2407.06041. doi:10.48550/ARXIV.2407.06041. arXiv:2407.06041.

[8] Z. Xi, T. Du, C. Li, R. Pang, S. Ji, X. Luo, X. Xiao, F. Ma, T. Wang, On the security risks of knowledge graph reasoning, in: J. A. Calandrino, C. Troncoso (Eds.), 32nd USENIX Security Symposium, USENIX Security 2023, Anaheim, CA, USA, August 9-11, 2023, USENIX Association, 2023, pp. 3259–3276. URL: https://www.usenix.org/conference/usenixsecurity23/presentation/xi.

[9] H. Puerto, T. Baumgärtner, R. Sachdeva, H. Fang, H. Zhang, S. Tariverdian, K. Wang, I. Gurevych, Ukp-square v3: A platform for multi-agent QA research, in: D. Bollegala, R. Huang, A. Ritter (Eds.), Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics: System Demonstrations, ACL 2023, Toronto, Canada, July 10-12, 2023, Association for Computational Linguistics, 2023, pp. 569–580. URL: https://doi.org/10.18653/v1/2023.acl-demo.55. doi:10.18653/V1/2023.ACL-DEMO.55.

[10] H. Ma, P. Lv, K. Chen, J. Zhou, Kgdist: A prompt-based distillation attack against lms augmented with knowledge graphs, in: E. Losiouk, A. Brighente, M. Conti, Y. Aafer, Y. Fratantonio (Eds.), The 27th International Symposium on Research in Attacks, Intrusions and Defenses, RAID 2024, Padua, Italy, 30 September 2024- 2 October 2024, ACM, 2024, pp. 480–495. URL: https://doi.org/10.1145/3678890.3678906. doi:10.1145/3678890.3678906.

[11] H. Zhang, T. Zheng, J. Gao, C. Miao, L. Su, Y. Li, K. Ren, Data poisoning attack against

knowledge graph embedding, in: Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI, 2019.
[12] P. Pezeshkpour, Y. Tian, S. Singh, Investigating robustness and interpretability of link prediction via adversarial modifications, in: 1st Conference on Automated Knowledge Base Construction, AKBC, 2019.
[13] P. Bhardwaj, J. D. Kelleher, L. Costabello, D. O'Sullivan, Poisoning knowledge graph embeddings via relation inference patterns, in: Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP, 2021.
[14] P. Bhardwaj, J. D. Kelleher, L. Costabello, D. O'Sullivan, Adversarial attacks on knowledge graph embeddings via instance attribution methods, in: Proceedings of the Conference on Empirical Methods in Natural Language Processing, EMNLP, 2021.
[15] X. You, B. Sheng, D. Ding, M. Zhang, X. Pan, M. Yang, F. Feng, Mass: Model-agnostic, semantic and stealthy data poisoning attack on knowledge graph embedding, in: Proceedings of the ACM Web Conference, WWW, 2023.
[16] Z. Zhang, F. Zhuang, H. Zhu, C. Li, H. Xiong, Q. He, Y. Xu, Towards robust knowledge graph embedding via multi-task reinforcement learning, IEEE Trans. Knowl. Data Eng. 35 (2023) 4321–4334.
[17] T. Zhao, J. Chen, Y. Ru, Q. Lin, Y. Geng, J. Liu, Untargeted adversarial attack on knowledge graph embeddings, in: Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval, 2024, ACM, 2024, pp. 1701–1711. URL: https://doi.org/10.1145/3626772.3657702.
[18] S. Kapoor, A. Sharma, M. Röder, C. Demir, A. N. Ngomo, Robustness evaluation of knowledge graph embedding models under non-targeted attacks, in: E. Curry, M. Acosta, M. Poveda-Villalón, M. van Erp, A. K. Ojo, K. Hose, C. Shimizu, P. Lisena (Eds.), The Semantic Web - 22nd European Semantic Web Conference, ESWC 2025, Portoroz, Slovenia, June 1-5, 2025, Proceedings, Part I, volume 15718 of *Lecture Notes in Computer Science*, Springer, 2025, pp. 264–281. URL: https://doi.org/10.1007/978-3-031-94575-5_15. doi:10.1007/978-3-031-94575-5_15.
[19] S. Auer, C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak, Z. G. Ives, Dbpedia: A nucleus for a web of open data, in: The Semantic Web, 6th International Semantic Web Conference ISWC, 2007.
[20] D. Vrandecic, M. Krötzsch, Wikidata: a free collaborative knowledgebase, Commun. ACM 57 (2014) 78–85. URL: https://doi.org/10.1145/2629489. doi:10.1145/2629489.
[21] O. Kolomiyets, M.-F. Moens, A survey on question answering technology from an information retrieval perspective, Information Sciences 181 (2011) 5412–5434.
[22] W. Zheng, H. Cheng, J. X. Yu, L. Zou, K. Zhao, Interactive natural language question answering over knowledge graphs, Information sciences 481 (2019) 141–159.
[23] R. Usbeck, M. Röder, M. Hoffmann, F. Conrads, J. Huthmann, A.-C. N. Ngomo, C. Demmler, C. Unger, Benchmarking question answering systems, Semantic Web 10 (2019) 293–304. URL: http://www.semantic-web-journal.net/system/files/swj1578.pdf. doi:10.3233/SW-180312.
[24] L. Page, S. Brin, R. Motwani, T. Winograd, The PageRank Citation Ranking: Bringing Order to the Web, Technical Report 1999-66, Stanford InfoLab, 1999. URL: http://ilpubs.

stanford.edu:8090/422/.
[25] L. C. Freeman, A set of measures of centrality based on betweenness, Sociometry 40 (1977) 35–41.
[26] P. Bonacich, Factoring and weighting approaches to status scores and clique identification, Journal of Mathematical Sociology 2 (1972) 113–120.