Link Inference Attack on Privacy-Preserving Knowledge Graphs

Emna Bouguerra[1], Ibtissam Harrouche[1], Ferran Alborch[1], and Melek Önen[1]

EURECOM, Sophia Antipolis, France

{emna.bouguerra,ibtissam.harrouche,ferran.alborch,melek.onen}@eurecom.fr

Abstract. Knowledge Graphs (KGs) are widely used to store and share structured information across sensitive domains such as healthcare, finance, and social networks. A common privacy practice is to delete sensitive relations before publishing the graph, under the assumption that removing edges is sufficient to prevent their recovery. In this paper, we challenge this assumption and show that even when a relation is fully or partially hidden, its existence leaves structural traces in the public graph that can be exploited to recover it with high accuracy. To this end, we propose a link inference attack that operates on the topology of the public graph, and evaluate it under two privacy scenarios that differ in how the adversary exploits the knowledge available to him. In the first setting where the adversary exploits all topological information, the attack achieves near-perfect discrimination (AP = 0.949, ROC-AUC = 0.999), while in the more realistic one where the adversary makes use of some semantic information, it recovers up to 74% of hidden edges. Building on these results, we further conduct a structural analysis to identify which topological properties of the graph drive the attack success, revealing that privacy risk is not uniform across entities and that certain structural patterns make specific relations significantly more vulnerable to inference than others.

Keywords: Knowledge Graphs · Link Inference Attack · Knowledge Graph Embeddings · Graph Privacy.

1 Introduction

Knowledge Graphs (KGs) are, nowadays, widely used to store and share structured information across various domains including healthcare [13], fi- nance [8], and social networks [8]. Their widespread adoption unfortunately begins to raise privacy issues: Indeed some part of the knowledge graph may include privacy-sensitive or confidential information: these could include relations such as a person's medical condition or his/her salary.

Stakeholders who own the graph, naturally wish to make use of the knowledge to train accurate machine learning models without disclosing the privacy-sensitive links/relations. A recent privacy protection practice [6] consists of pub-

lishing the knowledge graph and further making use of it, once all sensitive edges are deleted.

In this paper, we study the consequences of such a naive protection and further show that even when a sensitive relation is deleted, the remaining information and the structure of the graph can easily help an adversary to recover it. Indeed, we design a link inference attack that by processing the information on the topology of the graph and sometimes additionally some embedding information aims at inferring the hidden links of the KG. More specifically, we propose two attack scenarios that differ in how the adversary exploits the knowledge available to it: while the first one is oblivious to the semantics of the graph and makes use of all the triples in the graph, the second one is more optimized since it only processes links with entities already connected with sensitive relations.

We implement and evaluate these two scenarios and show that they result in significant leakage with non significant cost. Building on these observations, we further conduct a structural analysis to identify which topological properties of the graph drive the success of the attack. We try to extract these using Principal Component Analysis (PCA). We further observe that the leakage is not uniform. For example. highly connected entities in the graph are more exposed to these attacks.

Paper outline. Section 2 provides background information on knowledge graphs. Related work is discussed in section 3. The two proposed attacks are described and evaluated in section 4. Section 5 extends the study to identify the actual structural properties that influence privacy leakage and study this leakage across different topological patterns. Finally, conclusive remarks are provided in section 6.

2 Preliminaries

2.1 Knowledge Graphs

A **Knowledge Graph** (KG) is defined as a set of triples $\mathcal{G} = \{(h, r, t) \in \mathcal{T} \coloneqq \mathcal{E} \times \mathcal{R} \times \mathcal{E}\}$, where $\mathcal{E}$ is a set of entities and $\mathcal{R}$ is a set of relations. We denote as $h \in \mathcal{E}$ the *head* entity, as $t \in \mathcal{E}$ the *tail* entity, and as $r \in \mathcal{R}$ the *relation* type connecting them. Each triple (h, r, t) represents a directed edge from h to t labeled with relation r. KGs are a useful way to encode structured data, and as such have been used for a plethora of machine learning applications, such as recommendation systems [10], search engines [14], question answering [9], and clinical decision support [13].

2.2 Knowledge graph structural features

Knowledge graphs can be characterized by several entity-related structural metrics that capture the role and position of entities within the graph topology. In the following, we introduce the key metrics used in our analysis.

Node Degree The degree of an entity $e \in \mathcal{E}$ quantifies its connectivity within the graph, i.e. the amount of other entities involved in a triple with e. We distinguish three types of degree:

- In-degree $\deg_{in}(e) = |\{(h, r, e) \in \mathcal{T}\}|$: denotes the number of entities involved in a triple where e is a tail.
- Out-degree $\deg_{out}(e) = |\{(e, r, t) \in \mathcal{T}\}|$: denotes the number of entities involved in a triple where e is a head.
- Total degree $\deg(e) = \deg_{in}(e) + \deg_{out}(e)$: the total number of entities involved in a triple with e.

Entities with higher degree values tend to be more central and well-connected within the graph structure.

Incident Relation Counts Beyond overall connectivity, we can examine how frequently a specific entity e participates in a particular relation type r. We define:

- $\text{count}_r^{head}(e) = |\{(e, r, t) \in \mathcal{T}\}|$: the number of triples where e appears as the head with relation r.
- $\text{count}_r^{tail}(e) = |\{(h, r, e) \in \mathcal{T}\}|$: the number of triples where e appears as the tail with relation r

These counts provide a more fine-grained view of an entity's involvement in specific relationship types.

Jaccard Similarity To compare entities structurally, we use the Jaccard similarity coefficient, which measures the overlap between their neighborhoods. Given two entities e_1, e_2, we first define the neighborhood of an entity e as:

$$N(e) = \{v \in \mathcal{E} \mid (e, r, v) \in \mathcal{G} \text{ or } (v, r, e) \in \mathcal{G}\}. \tag{1}$$

$|S|$ denotes the cardinality of a given set S. The Jaccard similarity between e_1 and e_2 is then:

$$J(e_1, e_2) = \frac{|N(e_1) \cap N(e_2)|}{|N(e_1) \cup N(e_2)|}. \tag{2}$$

This coefficient ranges from 0 (indicating no shared neighbors) to 1 (indicating identical neighborhoods). A higher Jaccard similarity suggests that two entities occupy similar structural positions within the graph.

2.3 Knowledge Graph Embeddings

Knowledge Graph Embedding (KGE) models learn vector representations of entities and relations in $\mathbb{R}^d$, capturing the structural and semantic regularities of the graph, with applications in link prediction [3], question answering [9], and

recommendation systems [10]. Among these, TransE [3] represents each relation as a translation vector, optimizing:

$$f(h, r, t) = \|\mathbf{v}_h + \mathbf{v}_r - \mathbf{v}_t\|, \tag{3}$$

where $\mathbf{v}_h, \mathbf{v}_r, \mathbf{v}_t \in \mathbb{R}^d$ are the learned embeddings of the head entity, relation, and tail entity respectively. A low score indicates a likely true triple. TransE produces meaningful relation embeddings in which semantically similar relations cluster together, a property we exploit in Scenario 2 (Section 4).

3 Related Work

Knowledge graphs have emerged as a fundamental means to represent structured knowledge across diverse domains, from biomedical research to recommendation systems. As their deployment grows, so does the concern do: releasing even a partial view of a KG may expose sensitive information that was never intended to be public. This section reviews the most relevant work on inference attacks against graph-structured data and KGs specifically.

Inference Attacks on Graph-Structured Data. A foundational result by Backstrom et al. [1] demonstrates that even an anonymized social network leaks edge-level information through structural patterns, alone: given only the topology of an anonymized graph, an adversary can determine whether specific edges exist between targeted pairs of nodes. This establishes a key principle that our work extends to Knowledge Graphs: structural information alone is sufficient to mount a privacy attack, without any model access or semantic knowledge. Duddu et al. [4] are among the first to systematically quantify privacy leakage in graph embeddings, showing that GNN(Graph Neural Network) outputs expose sensitive node-level information through membership inference attacks. He et al. [5] propose the first link stealing attacks against GNN models, demonstrating that black-box access to a trained model suffices to infer whether specific edges exist in the training graph. Wu et al. [12] further show that private edges can be recovered from GNNs via influence analysis, even under differentially private training mechanisms.

Inference Attacks on Knowledge Graphs. Wang et al. [11] conduct the first empirical study of membership inference attacks against KGE models, showing that these models leak information about their training triples across multiple benchmark datasets including sensitive medical and financial KGs. Hu et al. [7] extend this line of work to the federated setting, proposing inference attacks against federated KGE models and demonstrating that collaborative training does not eliminate privacy risks. Our work is complementary to these efforts but targets a fundamentally different attack surface: rather than exploiting a trained model, we show that the raw structure of the public graph alone is sufficient to recover suppressed relations with high accuracy, without any model access or embedding.

Table 1. Comparison of the two attack scenarios.

Property	**Scenario 1**	**Scenario 2**
Candidate space	All co-occurring pairs	Semantically guided expansion
Strategy	Brute-force	Proxy-guided
Attack's cost	Expensive	Optimized
Semantic filtering	None	Via relation similarity

While our attack builds on structural features also used in link prediction, the two tasks differ fundamentally. Link prediction aims to complete an incomplete graph, whereas we target edges that were deliberately suppressed for privacy. Our attacker operates with limited supervision from $\mathcal{T}_{known}$, requires no access to a trained model unlike [12] [5] and in Scenario 2 exploits relation-type semantics via KGE embeddings, a property specific to KGs that has no equivalent in homogeneous graphs.

4 Link Inference Attack on KGs

In this section, we present a link inference attack designed to recover sensitive relations that have been deliberately hidden from a Knowledge Graph. The adversary exploits the publicly observable structure of the graph, only.

4.1 Threat Model

We model an adversary who seeks to recover a sensitive relation $r^\star$ that has been suppressed from a publicly released Knowledge Graph $\mathcal{G}_{pub}$. The attacker has two sources of information: (1) the full structure of $\mathcal{G}_{pub}$, including all non-sensitive relations, and (2) a partial set of known $r^\star$ triples representing facts that are already publicly available for a subset of entities, whether through prior releases, incomplete anonymization, or other disclosures. The attacker's goal is to exploit these two sources to infer the remaining hidden triples of $r^\star$.

Formally, let $\mathcal{T}^\star := \{(h, r^\star, t) \in \mathcal{T}\}$ denote the set of triples with sensitive relation $r^\star$, and $\mathcal{T}_{known} \subseteq \mathcal{T}^\star$ the subset known by the adversary. The attacker then observes:

$$\mathcal{G}_{pub} = \{(h, r, t) \in \mathcal{T} \mid r \neq r^\star\} \cup \mathcal{T}_{known}, \tag{4}$$

and aims to recover $\mathcal{T}^\star \setminus \mathcal{T}_{known}$. The attacker operates solely on anonymous entity identifiers, with no access to real-world entity labels or semantic knowledge.

The two scenarios we consider differ only in **how the adversary constructs the candidate space** from $\mathcal{G}_{pub}$, as the supervised training signal is the same in both cases.

4.2 Attack Description

The attack exploits the following property of KGs: even when $r^\star$ is removed, its existence leaves structural traces in $\mathcal{G}_{pub}$. Entities connected by $r^\star$ tend to share

similar structural features: they share common neighbors, co-appear in related relations, and exhibit comparable connectivity patterns. These regularities are consistent enough to be learned from the known triples $\mathcal{T}_{known}$ and generalized to unseen pairs.

Concretely, the adversary uses $\mathcal{T}_{known}$ as labeled examples to train a binary classifier that distinguishes $r^\star$ edges from non-edges, based solely on structural features extracted from $\mathcal{G}_{pub}$ (see 2.2). The trained classifier is then applied to a set of candidate pairs to rank the most likely hidden edges.

The attack follows four steps: it (1) constructs a candidate set $\mathcal{P}$ of entity pairs, (2) extracts $\mathcal{P}$'s structural features from $\mathcal{G}_{pub}$, (3) trains a binary classifier where positive examples are the known triples $\mathcal{T}_{known}$ and negative examples are all remaining pairs in $\mathcal{P}$, and (4) ranks candidates by predicted score. The two scenarios differ in Step 1, leading to distinct strategies, resulting in different costs and including semantic filtering or not as summarized in Table 1.

4.3 The two attack scenarios

Scenario 1: Naïve Attack The adversary constructs the candidate space without any semantic filtering: every entity pair co-occurring in at least one triple in $\mathcal{G}_{pub}$ is considered a potential candidate:

$$\mathcal{P} = \{(h, t) \mid \exists\, r : (h, r, t) \in \mathcal{G}_{pub}\}. \tag{5}$$

This yields an exhaustive but very large candidate space, in which true $r^\star$ edges typically represent only a small fraction of all candidates, reflecting the natural sparsity of Knowledge Graphs. Each pair is labeled against the original graph prior to masking:

$$y(h, t) = \begin{cases} 1 & \text{if } (h, r^\star, t) \in \mathcal{G}, \\ 0 & \text{otherwise.} \end{cases} \tag{6}$$

$\mathcal{P}$ is split into train, validation, and test sets via stratified sampling. Structural features are extracted from $\mathcal{G}_{pub}$ for each pair (Section 2.2). We train and compare Logistic Regression and LightGBM, with hyperparameters tuned on the validation set. Since this scenario reduces to a binary classification task over existing pairs in $\mathcal{G}_{pub}$, we report AP and ROC-AUC as evaluation metrics.

Scenario 2: Semantic-Aware Attack In this scenario, the adversary leverages $\mathcal{T}_{known}$, the set of $r^\star$ triples already known, as a seed to build a focused and semantically coherent candidate space, rather than enumerating all pairs blindly. Visible edges serve directly as training signal; the key challenge is constructing a focused candidate space for the private heads.

Relation Similarity via KGE. A TransE model [3] is trained on $\mathcal{G}_{pub}$ to rank relations by proximity to $r^\star$. For each relation r, let $\mathbf{v}_r \in \mathbb{R}^d$ denote its learned embedding. The cosine similarity between $\mathbf{v}_r$ and the target relation embedding $\mathbf{v}_{r^\star}$ is:

$$\mathrm{sim}(r, r^\star) = \frac{\mathbf{v}_r \cdot \mathbf{v}_{r^\star}}{\|\mathbf{v}_r\| \cdot \|\mathbf{v}_{r^\star}\|}. \tag{7}$$

The top-K most similar relations form the proxy set $\mathcal{R}_{sim}$.

Candidate Set Expansion. Starting from seed sets derived from visible edges,

$$\mathcal{F}_{seed} = \{h \mid (h, r^\star, t) \in \mathcal{T}_{known}\}, \quad \mathcal{G}_{seed} = \{t \mid (h, r^\star, t) \in \mathcal{T}_{known}\} \tag{8}$$

let $\mathcal{H}_r$ and $\mathcal{T}_r$ denote the head and tail entities of relation r in $\mathcal{G}_{pub}$. Each proxy relation $r \in \mathcal{R}_{sim}$ is classified as *head-like* or *tail-like* based on its overlap with the seed sets:

$$\mathcal{R}_{head\text{-}like} = \left\{ r \in \mathcal{R}_{sim} \;\middle|\; \frac{|\mathcal{H}_r \cap \mathcal{F}_{seed}|}{|\mathcal{H}_r|} \geq \alpha \;\text{ and }\; |\mathcal{H}_r \cap \mathcal{F}_{seed}| \geq n_h \right\}, \tag{9}$$

$$\mathcal{R}_{tail\text{-}like} = \left\{ r \in \mathcal{R}_{sim} \;\middle|\; \frac{|\mathcal{T}_r \cap \mathcal{G}_{seed}|}{|\mathcal{T}_r|} \geq \beta \;\text{ and }\; |\mathcal{T}_r \cap \mathcal{G}_{seed}| \geq n_t \right\}. \tag{10}$$

The fraction thresholds α, β ensure that a proxy relation involves entities of the same semantic type as $r^\star$. The minimum count thresholds n_h, n_t prevent relations with very few participants from qualifying, as a small relation can achieve a high overlap fraction purely by chance rather than by genuine similarity. Both conditions must hold simultaneously for a relation to qualify. The enriched candidate sets are then:

$$\tilde{\mathcal{F}} = \mathcal{F}_{seed} \cup \bigcup_{r \in \mathcal{R}_{head\text{-}like}} \mathcal{H}_r, \qquad \tilde{\mathcal{G}} = \mathcal{G}_{seed} \cup \bigcup_{r \in \mathcal{R}_{tail\text{-}like}} \mathcal{T}_r. \tag{11}$$

The attacker then infers that any entity in $\tilde{\mathcal{F}}$ with no visible $r^\star$ edges is a private head: $\mathcal{F}_{attack} = \tilde{\mathcal{F}} \setminus \mathcal{F}_{seed}$.

Candidate Pair Construction. Two pair sets are constructed:

- **Training/validation** ($\mathcal{P}_{vis}$): $\mathcal{F}_{seed} \times \tilde{\mathcal{G}}$, split 80/20 with stratified sampling. Labels are assigned from the full unmasked graph $\mathcal{G}$ as ground truth: $y(h, t) = 1$ if $(h, r^\star, t) \in \mathcal{G}$, and 0 otherwise.
- **Test** ($\mathcal{P}_{attack}$): $\mathcal{F}_{attack} \times \tilde{\mathcal{G}}$. Labels are unknown to the attacker and used only for offline evaluation.

Classifier Training and Ranking. The same five structural features as Scenario 1 are extracted from $\mathcal{G}_{pub}$ for each candidate pair. A LightGBM classifier is trained on $\mathcal{P}_{vis}$ with hyperparameters tuned on the validation split. For each private head $h \in \mathcal{F}_{attack}$, candidate tails in $\tilde{\mathcal{G}}$ are ranked by descending score. Since this scenario is framed as a ranking task over semantically guided candidate pairs, we primarily report Hits@1 and MRR, which directly measure ranking quality. AP is also reported for completeness, though it is less informative in ranking settings where the candidate space is focused and semantically coherent.

Table 2. Naïve Attack results on `/film/film/genre` (FB15k-237).

Model	Test AP	ROC-AUC	Gain over Random
Random Baseline	0.030	0.500	—
Logistic Regression	0.438	0.958	×14.6
LightGBM	0.949	0.999	×31.6

4.4 Experimental study

Dataset. We conduct our experiments on FB15k-237 [2], a widely used Knowledge Graph benchmark derived from Freebase. It contains 14,541 entities, 237 relation types, and 310,116 triples, covering diverse domains including film, sports, music, and people. We target three relations of varying frequency: `/film/film/genre` (medium frequency), `/award/.../award_nominee` (high frequency), and `/people/.../specialization_of` (low frequency).

Evaluation Metrics. We evaluate our attack using four metrics: Average Precision (AP), ROC-AUC, Hits@1, and Mean Reciprocal Rank (MRR), whose formal definitions are provided in Appendix A. AP and ROC-AUC are used for Scenario 1, framed as a binary classification task. Hits@1 and MRR are primarily used for Scenario 2, framed as a ranking task, with AP also reported for completeness.

Scenario 1:Naïve Attack. Table 2 reports the performance of the two classifiers against a random baseline on `/film/film/genre` (FB15k-237).

Both classifiers substantially outperform the random baseline. LightGBM achieves near-perfect discrimination (AP = 0.949, ROC-AUC = 0.999, ×31.6 over random), while Logistic Regression already yields strong performance (AP = 0.438, ROC-AUC = 0.958), confirming that even a linear model can exploit structural leakage. The gap between the two models suggests that the relationship between structural features and hidden link existence is non-linear, which the tree-based model captures more effectively. These results hold under severe class imbalance (∼3% positives), as reflected by the high ROC-AUC across both models.

The main limitation of this scenario is its cost: on FB15k-237, the candidate space reaches 283,831 pairs for a single relation, many of which are pairs that are clearly unrelated to $r^\star$ and thus trivially classified as negative. This motivates the more targeted approach of the attack in Scenario 2.

Scenario 2: Semantic-Aware Attack. Table 3 reports the coverage of $\tilde{\mathcal{F}}$ and $\tilde{\mathcal{G}}$ across three target relations. For `/film/film/genre`, expansion recovers 99.8% of all true head entities, up from 70.0% for the seed alone.

Table 4 reports performance across three target relations of varying frequency in FB15k-237.

Results reveal a clear dependence on relation frequency. For `/film/film/genre` (medium frequency), the attack achieves 74.07% Hits@1 and MRR =

Table 3. Head and tail coverage of the expanded candidate sets across three target relations. Seed→All: fraction of true entities recovered by the seed set alone. Exp.→All: fraction recovered after expansion.

Target Relation	Head Coverage		Tail Coverage	
	Seed/All	Exp./All	Seed/All	Exp./All
`.../specialization_of`	69.7%	73.1%	85.7%	97.1%
`.../award_nominee`	70.0%	74.8%	92.9%	98.9%
`/film/film/genre`	70.0%	99.8%	98.4%	99.2%

Table 4. Semantic-aware Attack results across three target relations in FB15k-237. Heads: number of private heads with at least one true hidden edge in the test set.

Relation	Freq.	AP	MRR	Hits@1	Heads	Gain
`.../award_nominee`	High	0.3241	0.5597	45.54%	112	×648.2
`/film/film/genre`	Med.	0.2935	0.8274	74.07%	563	×13.3
`.../specialization_of`	Low	0.0005	0.0659	0.00%	3	×21.7

0.8274: for nearly three quarters of hidden films, the true genre is ranked first. For `/award/.../award_nominee` (high frequency), Hits@1 is lower (45.54%) but the gain over random is the highest (×648.2), driven by the large tail space. For `/people/.../specialization_of` (low frequency), the attack largely fails (0% Hits@1, MRR = 0.066): with only 122 triples in the full KG and 3 evaluable private heads, structural features provide insufficient signal.

The lower performance of Scenario 2 relative to Scenario 1 is expected by design. Scenario 1 trains on all co-occurring pairs in $\mathcal{G}_{pub}$ with full stratified supervision, while Scenario 2 trains only on visible heads and must rank over a candidate space that it partially reconstructs from semantic proximity. The performance gap therefore reflects the cost of a more realistic threat model, where the attacker has no prior knowledge of the full candidate space.

5 Structural Leakage Analysis

In this section, we conduct two complementary studies to better understand the origin of the leakage. First, we apply PCA to the five structural features to examine their information content and inter-dependencies. Second, we show that privacy risk is not uniformly distributed across entities, and that certain structural properties make some entities significantly more vulnerable than others.

5.1 PCA on Structural Features

Our attack relies on five structural features computed for each candidate pair (h, t): `deg_head`, `deg_tail`, `inc_rel_head`, `inc_rel_tail`, and Jaccard similarity between their one-hop neighborhoods. To understand the information content of these features and their interdependencies, we apply Principal Component Analysis (PCA) to the candidate pairs of Scenario 1

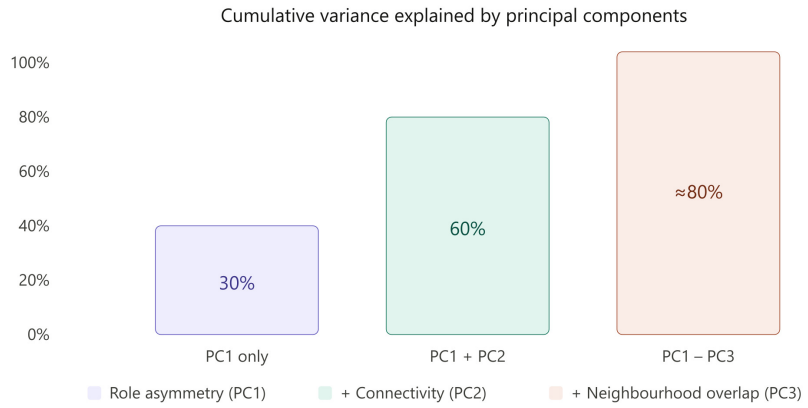

Fig. 1. Cumulative variance of the first three principal components. Three components capture approximately 80% of the total variance in the five structural features.

Table 5. Feature loadings for the first three principal components.

Feature	PC1	PC2	PC3
`deg_head`	+0.573	+0.416	−0.040
`incident_rel_head`	+0.557	+0.430	+0.129
`deg_tail`	−0.426	+0.547	+0.085
`incident_rel_tail`	−0.402	+0.514	+0.341
`jaccard`	+0.135	−0.282	+0.926

As shown in Figure 1, the first three principal components capture approximately 80% of the total variance, confirming that the five structural features are correlated and their information content compresses into three latent dimensions.

The loadings for all three components are summarized in Table 5.

PC1 — Head/Tail Asymmetry. The first component loads positively on `deg_head` (+0.573) and `inc_rel_head` (+0.557), and negatively on `deg_tail` (−0.426) and `inc_rel_tail` (−0.402), capturing the connectivity imbalance between the two endpoints: high scores indicate a hub head paired with a peripheral tail, and vice versa.

PC2 — Overall Connectivity. The second component loads positively on all degree and incident relation features, with a small negative loading on Jaccard (−0.282). Unlike PC1, it does not distinguish between head and tail it simply captures whether both entities are well-connected.

PC3 — neighborhood Overlap. The third component is dominated by Jaccard similarity (+0.926), capturing how much the local neighborhoods of the head and tail overlap, independently of their overall connectivity.

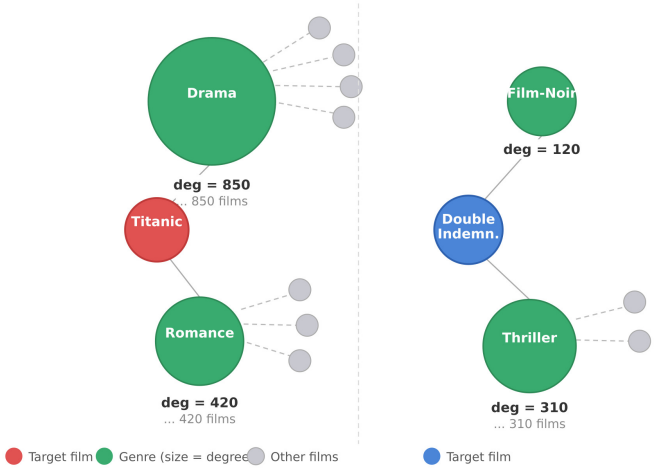

Fig. 2. Two films with contrasting genre connectivity. *Titanic* (left) is linked to high-degree genres (Drama: 850, Romance: 420), while *Double Indemnity* (right) belongs to low-degree genres (Film-Noir: 120, Thriller: 310). Node size reflects genre degree.

These three components show that structural leakage operates along three independent axes: the *imbalance* between head and tail connectivity, the *overall connectivity* of both endpoints, and the *neighborhood overlap* between them.

5.2 Privacy Risk Across Entities

The attack does not affect all entities equally. A node that is highly connected in the graph is structurally more exposed than one with few connections, as its dense neighborhood provides the attacker with stronger signal. For our target relation, this translates directly to genre popularity: a film whose genres are shared by many other films is easier to target than one whose genres are rare.

For a head h with tail set $T(h)$, we define its mean tail degree as:

$$\text{MeanDeg}(h) = \frac{1}{|T(h)|} \sum_{t \in T(h)} \deg(t), \tag{12}$$

where $\deg(t)$ is the number of heads connected to tail t. A head linked to high-degree tails sits in a dense neighborhood, providing the attacker with stronger structural signal. Figure 2 illustrates this contrast with two concrete examples. *Titanic* belongs to Drama (deg=850) and Romance (deg=420), giving it a mean genre degree of 635. *Double Indemnity* belongs to Film-Noir (deg=120) and Thriller (deg=310), giving it a mean genre degree of 215. The size difference in genre nodes directly reflects how much structural information is available to the attacker.

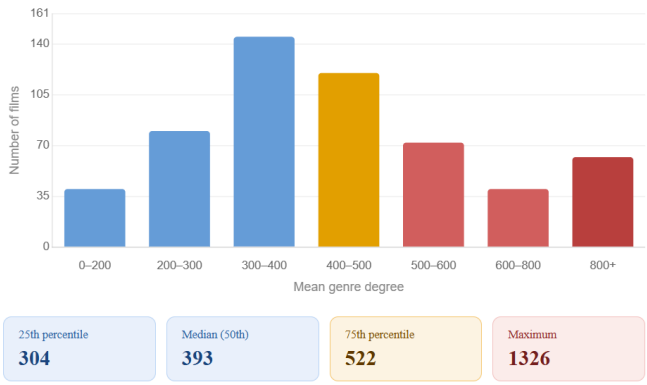

Fig. 3. Distribution of mean genre degree across films. The right-skewed distribution shows that most films belong to low-to-medium popularity genres, while a small number are linked to very popular genres.

Table 6. Attack success by mean genre degree quartile.

Genre Popularity	Mean Deg.	Hits@1	MRR	AP
Low ($\leq$ 304)	254	43%	59%	28%
Below Median (304–393)	349	74%	84%	36%
Above Median (393–522)	457	90%	93%	45%
High ($>$ 522)	701	92%	95%	49%

The distribution of MeanDeg across films is right-skewed (mean 434, median 393, 25th percentile 304, 75th percentile 522, maximum 1326), confirming that most films have low-to-medium genre connectivity, while a minority are linked to very popular genres (Figure 3).

We partition films into four quartile groups based on MeanDeg and report Hits@1, MRR, and Average Precision (AP) for each group in Table 6.

The results show a clear and consistent trend: as genre popularity increases, so does attack success across all three metrics. Films in the high popularity group reach 92% Hits@1 and 95% MRR, while those in the low popularity group drop to 43% Hits@1 and 59% MRR, showing that entities in dense regions of the graph are far more exposed than those in sparse ones. Therefore, a uniform edge removal does not provide uniform privacy guarantees. Even after the target relation is fully masked, heads associated with popular tails remain highly vulnerable due to the structural pattern carried by their neighborhood.

6 Conclusion & Future Work

In this paper, we have investigated whether hiding a sensitive relation from a Knowledge Graph is sufficient to prevent its recovery by a structural adversary. Our results demonstrate that it is not. Across the two attack scenarios, structural

features extracted from the public graph, alone, are sufficient to infer hidden triples with high accuracy. Moreover, our structural analysis reveals that privacy risk is not uniform: entities connected to popular, high-degree tail nodes are substantially more vulnerable than those linked to niche ones, meaning that uniform edge suppression does not provide uniform privacy guarantees.

These findings highlight structural leakage as a serious and underestimated threat to privacy-preserving Knowledge Graph publication. Our work also raises broader questions about what constitutes meaningful privacy in graph-structured data, and whether existing notions of privacy are adequate for the relational complexity of Knowledge Graphs.While our evaluation is limited to a single dataset, the structural properties driving the attack are generic, and validating these findings on additional KGs from sensitive domains such as biomedical or financial settings remains an important direction for future work. Several defense directions naturally follow from our findings. Graph perturbation and structural anonymization could reduce the discriminative power of the features driving the attack, while differential privacy offers formal guarantees, though its adaptation to heterogeneous KGs remains an open problem. Since privacy risk is non-uniform, any effective defense must be entity-adaptive rather than uniformly applied.

References

1. Backstrom, L., Dwork, C., Kleinberg, J.: Wherefore art thou r3579x? anonymized social networks, hidden patterns, and structural steganography. In: Proceedings of the 16th International Conference on World Wide Web. p. 181–190. WWW '07, Association for Computing Machinery, New York, NY, USA (2007). https://doi.org/10.1145/1242572.1242598
2. Bollacker, K., Evans, C., Paritosh, P., Sturge, T., Taylor, J.: Freebase: a collaboratively created graph database for structuring human knowledge. In: Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data. p. 1247–1250. SIGMOD '08, Association for Computing Machinery, New York, NY, USA (2008). https://doi.org/10.1145/1376616.1376746
3. Bordes, A., Usunier, N., Garcia-Durán, A., Weston, J., Yakhnenko, O.: Translating embeddings for modeling multi-relational data. In: Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2. p. 2787–2795. NIPS'13, Curran Associates Inc., Red Hook, NY, USA (2013), https://proceedings.neurips.cc/paper_files/paper/2013/file/1cecc7a77928ca8133fa24680a88d2f9-Paper.pdf
4. Duddu, V., Boutet, A., Shejwalkar, V.: Quantifying privacy leakage in graph embedding. In: MobiQuitous 2020 - 17th EAI International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services. p. 76–85. MobiQuitous '20, Association for Computing Machinery, New York, NY, USA (2021). https://doi.org/10.1145/3448891.3448939
5. He, X., Jia, J., Backes, M., Gong, N.Z., Zhang, Y.: Stealing links from graph neural networks. In: 30th USENIX Security Symposium (USENIX Security 21). pp. 2669–2686. USENIX Association (Aug 2021), https://www.usenix.org/conference/usenixsecurity21/presentation/he-xinlei

6. Hoang, A.T., Carminati, B., Ferrari, E.: Protecting privacy in knowledge graphs with personalized anonymization. IEEE Transactions on Dependable and Secure Computing **21**(4), 2181–2193 (2024). https://doi.org/10.1109/TDSC.2023.3300360
7. Hu, Y., Liang, W., Wu, R., Xiao, K., Wang, W., Li, X., Liu, J., Qin, Z.: Quantifying and defending against privacy threats on federated knowledge graph embedding. In: Proceedings of the ACM Web Conference 2023. p. 2306–2317. WWW '23, Association for Computing Machinery, New York, NY, USA (2023). https://doi.org/10.1145/3543507.3583450
8. Ji, S., Pan, S., Cambria, E., Marttinen, P., Yu, P.S.: A survey on knowledge graphs: Representation, acquisition, and applications. IEEE Transactions on Neural Networks and Learning Systems **33**(2), 494–514 (2022). https://doi.org/10.1109/TNNLS.2021.3070843
9. Lan, Y., He, G., Jiang, J., Jiang, J., Zhao, W.X., Wen, J.R.: Complex knowledge base question answering: A survey. IEEE Transactions on Knowledge and Data Engineering **35**(11), 11196–11215 (2023). https://doi.org/10.1109/TKDE.2022.3223858
10. Wang, X., He, X., Cao, Y., Liu, M., Chua, T.S.: Kgat: Knowledge graph attention network for recommendation. In: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. p. 950–958. KDD '19, Association for Computing Machinery, New York, NY, USA (2019). https://doi.org/10.1145/3292500.3330989
11. Wang, Y., Huang, L., Yu, P.S., Sun, L.: Membership inference attacks on knowledge graphs (2022), https://arxiv.org/abs/2104.08273
12. Wu, F., Long, Y., Zhang, C., Li, B.: Linkteller: Recovering private edges from graph neural networks via influence analysis. In: 2022 IEEE Symposium on Security and Privacy (SP). pp. 2005–2024 (2022). https://doi.org/10.1109/SP46214.2022.9833806
13. Xiang, X., Wang, Z., Jia, Y., Fang, B.: Knowledge graph-based clinical decision support system reasoning: A survey. In: 2019 IEEE Fourth International Conference on Data Science in Cyberspace (DSC). pp. 373–380 (2019). https://doi.org/10.1109/DSC.2019.00063
14. Zhou, Y., Lin, Z., Shi, J., Yang, Y., Wei, R.: Research on the construction of a knowledge graph information search engine in the web domain. In: 2023 International Conference on Evolutionary Algorithms and Soft Computing Techniques (EASCT). pp. 1–6 (2023). https://doi.org/10.1109/EASCT59475.2023.10393039

A Evaluation Metrics

Precision and Recall. Given a set of retrieved instances, precision measures the fraction of retrieved positive instances that are true positives:

$$P = \frac{|\mathrm{TP}|}{|\mathrm{TP}| + |\mathrm{FP}|}, \tag{13}$$

Recall measures the fraction of true positives among true labels:

$$R = \frac{|\mathrm{TP}|}{|\mathrm{TP}| + |\mathrm{FN}|}, \tag{14}$$

where TP, FP, and FN denote true positives, false positives, and false negatives respectively.

Average Precision (AP). The area under the precision-recall curve, summarizing model performance across all classification thresholds:

$$\mathrm{AP} = \sum_{k=1}^{K} P(k) \cdot \Delta R(k), \tag{15}$$

where $P(k)$ and $R(k)$ are the precision and recall at the k-th threshold, and $\Delta R(k) = R(k) - R(k-1)$.

ROC-AUC. The area under the receiver operating characteristic curve, measuring the probability that a randomly chosen positive instance is ranked higher than a randomly chosen negative one:

$$\text{ROC-AUC} = \frac{\sum_{x^+ \in \mathcal{P}} \sum_{x^- \in \mathcal{N}} \mathbf{1}[s(x^+) > s(x^-)]}{|\mathcal{P}| \cdot |\mathcal{N}|}, \tag{16}$$

where $\mathcal{P}$ and $\mathcal{N}$ are the sets of positive and negative instances, and $s(x)$ is the predicted score for instance x.

Rank. Given a query q and a list of candidates ranked by descending predicted score, $\mathrm{rank}(q)$ denotes the position of the first true positive in the ranked list. A rank of 1 means the true positive is the top-ranked candidate.

Hits@k. The fraction of queries for which there is a true positive candidate ranked within the first k positions

$$\text{Hits@}k = \frac{1}{|\mathcal{Q}|} \sum_{q \in \mathcal{Q}} \mathbf{1}[\mathrm{rank}(q) \leq k]\,, \tag{17}$$

where $\mathcal{Q}$ is the set of queries.

Mean Reciprocal Rank (MRR). The mean of the reciprocal rank of the first true positive across all queries:

$$\mathrm{MRR} = \frac{1}{|\mathcal{Q}|} \sum_{q \in \mathcal{Q}} \frac{1}{\mathrm{rank}(q)}. \tag{18}$$

A higher MRR indicates that true positives are consistently ranked near the top.