Computer Standards & Interfaces 97 (2026) 104113

Contents lists available at ScienceDirect

Computer Standards & Interfaces

journal homepage: www.elsevier.com/locate/csi

Co-distillation-based defense framework for federated knowledge graph embedding against poisoning attacks

Yiqin Lu, Jiarui Chen *, Jiancheng Qin

School of Electronic and Information Engineering, South China University of Technology, 510641, China

ARTICLE INFO

Keywords:
Federated learning
Knowledge graph
Poisoning attack
Knowledge distillation

ABSTRACT

Federated knowledge graph embedding (FKGE) enables collaborative knowledge sharing without data exchange, but it also introduces risks of poisoning attacks that degrade model accuracy or force incorrect outputs. Protecting FKGE from poisoning attacks becomes a critical research problem. This paper reveals the malicious strategy of untargeted FKGE poisoning attacks and proposes CoDFKGE, a co-distillation-based FKGE framework for defending against poisoning attacks. CoDFKGE deploys two collaborative knowledge graph embedding models on clients, decoupling prediction parameters from shared parameters as a model-agnostic solution. By designing distinct distillation loss functions, CoDFKGE transfers clean knowledge from potentially poisoned shared parameters while compressing dimensions to reduce communication overhead. Experiments show CoDFKGE preserves link prediction performance with lower communication costs, eliminates malicious manipulations under targeted poisoning attacks, and significantly mitigates accuracy degradation under untargeted poisoning attacks.

1. Introduction

Knowledge graphs (KGs) are structured representations of real-world entities and their relationships, supporting applications in search engines [1,2], recommendation systems [3,4], and security analysis [5,6]. Knowledge graph embedding (KGE) techniques project entities and relations into low-dimensional vector spaces, enabling efficient knowledge reasoning and completion [7]. Due to privacy regulations and data sensitivity requirements, KGs across organizations within the same domain remain fragmented despite growing data volumes. In this context, federated knowledge graph embedding (FKGE) emerges as a collaborative learning technique for sharing KG embeddings without data exchange. However, the introduction of federation mechanisms will bring new privacy risks. malicious participants can inject poisoned parameters during training or aggregation to launch a poisoning attack, degrading model accuracy or forcing incorrect outputs. Consequently, protecting FKGE systems against poisoning attacks has emerged as a critical research challenge.

Unlike graph neural network (GNN)-based models, KGE models usually rely on the translation-based model [8–11]. The embedding vectors of entity and relation in the KG are directly used as learnable parameters. KGE models utilize different score functions to measure the plausibility of triples (h,r,t). By contrasting the outputs of existing triples and negatively sampled triples, KGE models derive appropriate embedding for entities and relations. However, real-world KGs of different organizations are often incomplete, making it difficult to train high-quality knowledge graph reasoning models. Moreover, KG data often contains a large amount of private data, and direct data sharing will inevitably lead to privacy leakage. For this reason, federated learning [12] is introduced into knowledge graph reasoning.

FKGE assumes that there are multiple participants with complementary but incomplete KGs, aiming to derive optimal knowledge embeddings for each participant without data exchange. Most existing studies [13–15] model FKGE as multiple clients that maintain local KGE models and a central server. Clients train models locally and upload the model parameters to the central server, which aggregates the parameters and then returns them to the clients.

However, since the embedding vectors are directly the model parameters, FKGE is highly vulnerable to poisoning attacks. With the intent to reduce model performance, steal sensitive information, or disrupt system stability, poisoning attacks refer to malicious modifications of parameters during local training or parameter aggregation on the server. To protect the participants of FKGE, it is necessary to propose a protection mechanism against FKGE poisoning attacks.

Moreover, other related indicators in FKGE deserve attention. For example, the federated learning of KGE requires frequent parameter

* Corresponding author.
E-mail addresses: eeyqlu@scut.edu.cn (Y. Lu), ee_jrchen@mail.scut.edu.cn (J. Chen), jcqin@scut.edu.cn (J. Qin).

https://doi.org/10.1016/j.csi.2025.104113
Received 3 June 2025; Received in revised form 8 November 2025; Accepted 8 December 2025
Available online 9 December 2025

exchange, and the use of a translation-based model will submit the entity or relation embeddings, which makes the communication overhead greater than that of traditional federated learning.

Knowledge distillation [16] is a model compression technique that improves the performance of a simple (student) model by transferring the knowledge from a complex (teacher) model. Distillation-based methods are considered to be a feasible solution to combat poisoning attacks [17–19]. A teacher model can extract clean knowledge from the poisoned parameters and transfer it to a student model, thereby improving the robustness without changing the model structure. Co-distillation [20] is a variant of knowledge distillation that trains two or more models simultaneously, allowing mutual learning and information sharing. This paper aims to design a federated knowledge graph defense framework based on Co-distillation, which can enhance the model's resistance to poisoning attacks through collaborative learning without changing the original FKGE architecture.

The rest of this paper is organized as follows. Section 2 reviews the related work on FKGE and knowledge distillation. Section 3 introduces the preliminary concepts and methodologies essential for addressing FKGE poisoning attacks, with the main contributions of this paper summarized at the end of this section. In Section 4, we detail the threat model and malicious strategies for targeted and untargeted poisoning attacks in FKGE. Section 5 presents the CoDFKGE framework for defending against FKGE poisoning attacks, followed by experimental validation in Section 6. Finally, concluding remarks and future research directions are outlined in Section 7.

2. Related work

2.1. Basic FKGE framework

Early research on FKGE mainly focused on how to achieve cross-client knowledge sharing and model aggregation while protecting data privacy. FedE [13] is the first paper to introduce federated learning into KGE. FedE facilitates cross-client knowledge sharing by maintaining an entity table. Nevertheless, the mechanism of sharing entity embeddings in FedE has been proven to contain privacy vulnerabilities [21]. Attackers can leverage the embedding information to infer the existence of private triples within client datasets. Based on FedE, FedEC [14] applies embedding contrastive learning for tackling data heterogeneity and utilizes a global update procedure for sharing entity embeddings. In response to the privacy vulnerability of FedE, FedR [15] proposed a privacy-preserving relation embedding aggregation method. By sharing relation embeddings instead of entity embeddings, FedR can significantly reduce the communication overhead of privacy leakage risks while retaining the semantic information of the KG.

2.2. Knowledge distillation in FKGE

Knowledge Distillation techniques are widely applied in the FKGE field due to their advantages in model compression and knowledge transfer. To cope with the drift between local optimization and global convergence caused by data heterogeneity, FedLU [22] proposes mutual knowledge distillation. Moreover, it contains an unlearning method to erase specific knowledge from local clients. FedKD [23] uses knowledge distillation to reduce communication costs, and proposes to adaptively learn temperature to scale the scores of triples to mitigate teacher over-confidence issues. In addition to FKGE, the KGE model ColE [24] proposes co-distillation learning to exploit the complementarity of graph structure and text information. It employs Transformer and Bert for graph and text respectively, then distills selective knowledge from each other's prediction logits. Overall, existing research on knowledge distillation in FKGE primarily focuses on handling data heterogeneity, with insufficient exploration of its potential value in model security. This paper will explore the application of knowledge distillation in FKGE security to defend against poisoning attacks.

2.3. Poisoning attack in federated learning

Federated Learning (FL), due to its distributed training nature, creates favorable conditions for poisoning attacks while protecting data privacy. Poisoning attacks in federated learning have attracted significant attention from researchers [25]. In federated learning scenarios, poisoning attacks pose serious threats to model security by manipulating partial training data or local models to embed malicious behaviors [26]. The literature [27] generates stealthy backdoor triggers by extracting high-frequency features from images using discrete wavelet transform and introduces an asymmetric frequency confusion mechanism, achieving efficient backdoor attacks on multiple datasets. Meanwhile, many studies have proposed defense methods against poisoning attacks. The Literature [28] proposes the Krum method, which selects the most reliable gradient update by evaluating the consistency of gradients, thereby effectively defending against poisoning attacks. The Literature [29] proposes Fl-Defender, which improves robustness by introducing cosine similarity to adjust the weights of parameter aggregation. The literature [30] proposed a two-stage backdoor defense method called MCLDef based on Model Contrastive Learning (MCL), which can significantly reduce the success rate of backdoor attacks with only a small amount of clean data. In summary, existing research on poisoning attacks in federated learning mainly focuses on traditional deep learning domains. The design ideas of defense frameworks have laid the foundation for subsequent poisoning attack defense methods of FKGE.

2.4. Security issues in FKGE

With the development of FKGE, its security and privacy issues have attracted increasing attention, with existing research mainly focusing on privacy leakage defense. The literature [31] proposed a decentralized scalable learning framework where embeddings from different KGs can be learned in an asynchronous and peer-to-peer manner while being privacy-preserving. The literature [21] conducts the first holistic study of the privacy threat on FKGE from both attack and defense perspectives. It introduced three new inference attacks and proposed a differentially private FKGE model DP-Flames with private selection and an adaptive privacy budget allocation policy. Based on [21], the literature [32] introduces five new inference attacks, and proposed PDP-Flames, which leverages the sparse gradient nature of FKGE for better privacy-utility trade-off.

Compared with privacy leakage issues, research on defending against poisoning attacks in FKGE is still in its early stages. Traditional federated learning typically does not directly transmit original embeddings. However, entity and relation embeddings are core components in translation-based KGE, so direct transmission of embeddings is required during FKGE aggregation. Direct malicious modifications to embeddings are difficult to effectively defend against using traditional federated learning defense methods.

The recent literature [33] is the first work to systematize the risks of FKGE poisoning attacks. However, it primarily focuses on several forms of targeted poisoning attacks in FKGE, without mentioning untargeted poisoning attacks. Although this research provides some defense suggestions, such as zero-knowledge proof and privacy set intersection, it does not propose specific defense methods. In summary, the existing research lacks a systematic introduction to the untargeted poisoning attack of FKGE, and there is no complete defense method against FKGE poisoning attacks.

To address the above issues, this paper reveals the malicious strategy of FKGE untargeted poisoning attacks and proposes CoDFKGE, a co-distillation-based federating knowledge graph embedding framework for defending against poisoning attacks. The main contributions of this paper are summarized as follows.

1 We systematically define untargeted poisoning attacks in FKGE and reveal the poisoning attacks' malicious strategy, thereby enhancing threat identification in FKGE and providing a foundation for subsequent defense research.
2 We propose CoDFKGE, the first co-distillation defense framework against poisoning attacks in FKGE. By deploying bidirectional distillation models with distinct distillation loss at the client side, CoDFKGE as a model-agnostic solution decouples prediction parameters from shared parameters, thereby enhancing the model's resistance to poisoning attacks and improving robustness. We designed distinct distillation loss functions for the two models in CoDFKGE, enabling CoDFKGE to transfer clean knowledge from potentially poisoned shared parameters and compress shared parameter dimensions, which reduces communication overhead.
3 We validated the performance of CoDFKGE against poisoning attacks through experiments. The results show that without compromising link prediction performance CoDFKGE can completely eliminate targeted poisoning attacks and significantly mitigate the performance degradation caused by untargeted poisoning attacks, while simultaneously reducing communication overhead. Ablation experiments further confirm the effectiveness of the two distillation loss functions in CoDFKGE.

3. Preliminaries

3.1. Knowledge graph embedding

KG can be represented as $\mathcal{G}(\mathcal{E}, \mathcal{R}, \mathcal{T})$, where E and R are entity sets and relationship sets. $\mathcal{T}$ is a set of triples, where a triple $(h, r, t) \in \mathcal{T}$ indicates that a relationship $r \in \mathcal{R}$ connects the entities $h, t \in \mathcal{E}$.

Translation-based KGE models project entities and relationships in KGs into a continuous vector space. Models employ the scoring function $g(h, r, t; \theta)$ to evaluate the plausibility of triples, while θ represents the embedding parameters. During model training, negative samples (h, r, t') are constructed by randomly replacing the tail entities of positive triples. The training process aims to maximize the score discrepancy between positive and negative samples. Currently, most KGE models [9,11] employ the binary cross-entropy loss to measure the difference between positive and negative samples. Its mathematical expression is as Eq. (1).

$$L = -\sum_{(h,r,t)\in\mathcal{T}} \left(\log \sigma\left(g(h,r,t;\theta) - \gamma\right) + \sum_i p(h,r,t_i';\theta) \log \sigma(\gamma - g(h,r,t_i';\theta)) \right) \tag{1}$$

Among them, γ represents the margin, and (h, r, t_i') is ith negative triples. $p(h, r, t_i'; \theta)$ stands for the occurrence probability of this negative sample given the embedding parameters θ.

3.2. Federated knowledge graph embedding

FKGE is an application of federated learning that aims to fuse and share knowledge vectors from different KGs to enhance the effectiveness of KGE. Currently, most related studies are based on the framework proposed in FedE [13].

The basic framework of FKGE consists of a client set C and a central server S. Each client $c \in C$ holds a local KG $\mathcal{G}_c(\mathcal{E}_c, \mathcal{R}_c, \mathcal{T}_c)$. The entity sets of different KGs are partially overlapping, so the understanding of entities in a certain client can be supplemented by information from other clients. The server has the one-hot existence matrix $M \in \mathbb{R}^{C\times N}$ of all entities in the client, where N is the number of entities.

In each client, KGE model parameters consist of local parameters θ_L and shared parameters θ_S. During FKGE training, each epoch progresses through two sequential phases: client update and server aggregation. In the kth client update stage, client c first trains its local KGE model to update its local embedding $\theta_{L_c}^k$ and server-shared embedding $\theta_{S_c}^k$. Then, client c uploads its shared embedding $\theta_{S_c}^k$ to the server. In server aggregate stage, the central server S aggregates the shared embeddings from all clients to obtain the shared parameters θ_S^{k+1}. Finally, the server broadcasts the shared parameters θ_S^{k+1} to all clients. Entity embeddings in KGE are usually shared parameters, while relation embeddings are local parameters. Only rare literature [15] uses relation embeddings as shared parameters.

In FKGE, how the server effectively aggregates shared embeddings from different clients is a common problem. The most common FKGE server aggregation method is FedE [13], which is an improvement on FedAvg [12]. To handle the imbalance in the number of entities across different clients, FedE aggregate the shared entities using the number of occurrences in the local data as the weight w_c. This weight value can be obtained using the existence matrix M mentioned above. The mathematical expression for FedE's server aggregation method is shown in (2).

$$\theta_S^{k+1} = \sum_c w_c \theta_{S_c}^k \tag{2}$$

The final target of FKGE is to minimize the loss function of all client local triplets simultaneously through federated learning. Its optimization objective can be expressed as Eq. (3).

$$arg \min_{(\theta_{L_c}, \theta_{S_c})} \sum_c^C \mathcal{L}_c(\theta_{L_c}, \theta_{S_c}) \tag{3}$$

3.3. Knowledge distillation

Knowledge distillation is a model compression technique that transfers knowledge contained in a complex model (teacher) to a simple model (student) to improve the performance of the simple model. In the classic knowledge distillation framework, the student model's training loss comprises two components: the cross entropy loss L_{CE}, computed between its output and the true label, and the distillation loss L_{KD}, computed between its output and the teacher model's output (soft label). In practical applications, the distillation loss is usually quantified using the Kullback–Leibler divergence D_{KL} between the student model output and the soft label, and its mathematical expression is shown in Eq. (4).

$$\begin{aligned} D_{KL}\left(p_{tea} \parallel p_{stu}\right) &= \sum_i p_{tea}(i) \log\left(\frac{p_{tea}(i)}{p_{stu}(i)}\right) \\ L_{KD} = \tau^2 D_{KL}&\left(\sigma(z_{tea}^{(n)}) \parallel \sigma(z_{stu}^{(n)})\right), where\ \sigma(x) = \mathrm{softmax}\left(\frac{x}{\tau}\right) \end{aligned} \tag{4}$$

Among them, z_{tea} and z_{stu} are the logits of the teacher model and student model, respectively. τ is the temperature coefficient, which is used to control the smoothness of the output.

To allow the student model to effectively absorb the knowledge contained in the teacher model while fitting the real data distribution, the final loss function is usually the weighted sum of L_{CE} and L_{KD}.

4. Threat model

Poisoning attacks in federated learning can be categorized into targeted poisoning attacks, semi-targeted poisoning attacks, and untargeted poisoning attacks according to the intention of attackers [34]. In FKGE, a semi-targeted poisoning attack can be regarded as a special case of a targeted poisoning attack. Therefore, this paper focuses on the targeted and untargeted poisoning attack type.

4.1. Targeted poisoning attack

Targeted poisoning attacks are a attack strategy where the attacker crafts specific malicious triples that do not exist in the target system, and manipulate the target model to accept these fake triples by injecting poisoned parameters into the shared parameters. This type of attack poses a serious threat to the application of FKGE, as the false relationships it introduces can lead to reasoning errors and decision-making

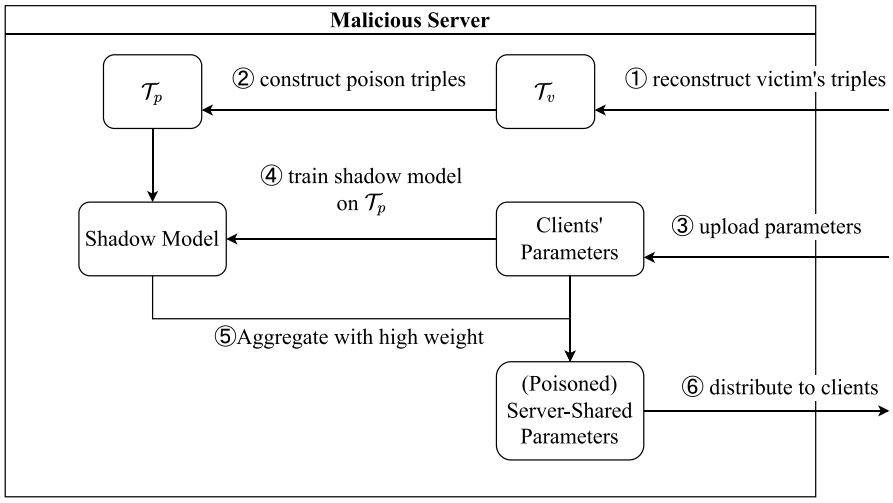


Fig. 1. Process of targeted poisoning attack.

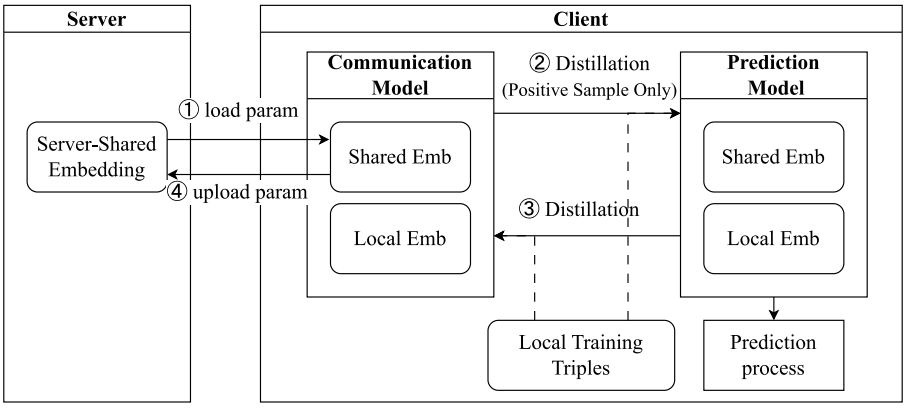


Fig. 2. Framework of CoDFKGE model.

biases in downstream tasks. For example, in financial transaction networks, a knowledge graph is constructed with transaction entities as nodes and transaction relationships as edges. Link prediction can then be applied to detect potential transaction relationships (such as money laundering or fraud). If an attacker compromises one of the participants, they can introduce false transaction relationships through targeted poisoning attacks, leading to unreasonable inferences about the victim entity.

To execute such an attack successfully, the attacker typically follows a multi-stage process that begins with victim's local information gathering. Fig. 1 shows the process of a targeted poisoning attack. In FKGE systems, while the server can observe the entities and relations each client possesses, it lacks visibility into how these elements are structured into specific triples. However, for frameworks that share entity embeddings (such as FedE [13]), recent research [21] has shown that a malicious server can use KGE scoring function to infer the victim's local relationship patterns and reconstruct the victim's triple $\mathcal{T}_v$. Armed with this inferred knowledge, the attacker strategically constructs malicious triples $\mathcal{T}_m$ that align with the victim's existing KG schema but represent false information.

The next critical attack phase involves training a shadow model, a surrogate KGE model designed to mimic the victim's learning process. The shadow model is trained on a poisoned dataset $\mathcal{T}_p$, which combines the inferred victim triples $\mathcal{T}_v$ and the malicious triples $\mathcal{T}_m$. This training strategy ensures the shadow model learns to generate embeddings that are consistent with both the victim's genuine knowledge and the attacker's deceptive information. The shadow model's parameters include θ_{S_p}, which can be initialized with the victim shared parameters θ_{S_c}, and θ_{L_p}, which approximates the victim's local model parameters θ_{L_c} from random initial values. To ensure the shadow model effectively bridges both the victim's genuine knowledge and the attacker's malicious objectives, its parameters are optimized to minimize the loss function across all triples in the poisoned dataset, as formalized in Eq. (5).

$$\arg \min_{(\theta_{S_p}, \theta_{L_p})} \sum_{(h,r,t) \in \mathcal{T}_p} L(h, r, t; \theta_{S_p}, \theta_{L_p}) \tag{5}$$

Where L is the loss function of the baseline model.

After training the shadow model, the attacker extracts the poisoned shared parameters θ_{S_p} using the same procedure that legitimate clients employ to prepare parameters for server aggregation. The attacker can aggregate the poisoned parameters θ_{S_p} with the normal clients' shared parameters. The attacker usually operates as a compromised server and assigns a disproportionately high weight to the poisoned parameters during the aggregation process to ensure that the poisoned parameter dominate the aggregated shared parameters.

The final stage of the attack exploits the implicit trust in federated systems. The victim client, unaware of the poisoning, directly incorporates the compromised aggregated parameters into its local training process without validation. As a result, the victim's model gradually learns to accept the malicious triples as valid, ultimately producing incorrect predictions on these non-existent relationships while maintaining seemingly normal performance on other parts of the KG.

4.2. Untargeted poisoning attack

The conditions for achieving a targeted poisoning attack are complex. For example, FedR [15] shares only relation embeddings (not entity embeddings), preventing attackers from inferring victim relations via entity matrices and thus avoiding targeted poisoning attacks. Even with relational data leaks, targeted poisoning attacks are difficult. Compared with sharing entity embeddings, the sparsity of relation embeddings reduces the shadow model's ability to align parameters with the victim's vector space. However, FedR has almost no defense effect against untargeted poisoning attacks.

An untargeted poisoning attack means that the attacker aims to disrupt victim model convergence or maximize the mispredictions among test cases. By maximizing the victim's loss function during training, attackers can force non-convergent predictions. The attacker can generate the poisoned shared parameter $\theta^*_{S_v}$ for the victim, which can be formalized in Eq. (6).

$$\arg\max_{\theta^*_{S_v}} \sum_{(h,r,t)\in\mathcal{T}_v} L(h,r,t;\theta^*_{S_v},\theta_{L_v}) \tag{6}$$

Among them, θ_{L_v} denotes the victim's local parameters. $\mathcal{T}_v$ is the victim's triplet set. Since it is difficult for the attacker to obtain these two parameters directory, they can use random values as guesses for θ_{L_v} and use triples of random combinations of $\mathcal{E}_v$ and $\mathcal{R}$ as guesses for $\mathcal{T}_v$.

In particular, for the TransE model [7] with the scoring function $g(h,r,t) = |h + r - t|$, the attacker can launch an untargeted poisoning attack by setting the shared parameter θ'_{S_v} sent to the victim to identical value or using negative aggregation parameters. To avoid detection, noise is often added to poisoned parameters. The prediction performance of the victim model may even be lower than that of standalone training without federated aggregation.

In general, the success of FKGE poisoning attacks relies on victims using attacker-provided aggregate parameters directly for training without validation. To prevent poisoning attacks, it is critical to isolate the parameters of the prediction model from externally provided aggregate parameters. Specifically, potentially poisoned shared parameters must be filtered before training. Meanwhile, minimizing parameter exposure to the external environment is essential. Therefore, we propose CoDFKGE, a defense FKGE framework based on co-distillation.

5. Model design

CoDFKGE is a training framework on the client side. Its training process is shown in Fig. 2. CoDFKGE initializes two baseline models with the same structure and scoring function, but for different purposes. The communication model is mainly responsible for receiving and processing shared parameters, while the prediction model is used for the final embedding and prediction. To minimize potential parameter leakage and communication overhead, the feature dimension of the communication model is intentionally designed to be smaller than that of the prediction model.

During the training process, the two models learn collaboratively through knowledge distillation. Once the communication model receives the potentially poisoned shared parameters from the server, it acts as a teacher model to transfer clean knowledge to the prediction model. Following the training of the prediction model, the roles are reversed: the prediction model becomes the teacher, and the communication model serves as the student for distillation. This stage extracts knowledge from the prediction model and compresses it into the communication model, ensuring efficient knowledge sharing while minimizing parameter exposure and communication overhead. By deploying two distinct model instances, the framework physically isolates attacker-injected parameters from the prediction model's parameters, making poisoning attacks significantly more difficult to execute. To facilitate the reproducibility of our CoDFKGE model, we provide the complete training framework pseudocode as shown in Algorithm 1.

Algorithm 1 CoDFKGE Training Framework

Require: Baseline KGE model g, Training triples $\mathcal{T}$, Learning rate η, Distillation weight β, Distillation temperature τ, Total iterations K
Initialization:
Initialize client-side prediction model with $\theta_0^P = (\theta_0^S, \theta_0^L)$ ▷ Local parameters randomly initialized
Initialize client-side communication model with reduced feature dimensions
Initialize server-side aggregated parameters $\theta_1^S = \theta_0^S$ ▷ First round initialization
Main Training Loop (Iterations $k = 1, 2, ..., K$**):**
// Client Update Phase (For each client)
for each client $c \in C$ **do**
  // Step 1: Communication to Prediction Model Distillation
  Load server-shared parameters θ_k^S ▷ Latest global shared embeddings
  Initialize communication model with $\theta^C = (\theta_k^S, \theta_{k-1}^{C_L})$
  Freeze communication model parameters ▷ Act as teacher model
  Compute distillation loss $L_k^{P_{KD}}$ using Equation (7) ▷ Only positive samples
  Compute KGE loss $L_k^{P_{KGE}}$ on training triples $\mathcal{T}$
  Update prediction model parameters $(\theta_k^{P_S}, \theta_k^{P_L})$ with:
    $\nabla\theta_k^P = \nabla(\beta L_k^{P_{KGE}} + (1-\beta) L_k^{P_{KD}})$ ▷ Gradient flows through prediction model only
    $\theta_k = \theta_k - \eta\nabla\theta_k^P, \quad where\ \theta_k = \{\theta_k^{P_L}, \theta_k^{P_S}\}$ ▷ Update prediction model parameters
  Unfreeze communication model parameters
  // Step 2: Prediction to Communication Model Distillation
  Freeze prediction model parameters θ_k^P ▷ Used as teacher model
  Compute distillation loss $L_k^{C_{KD}}$ using Equation (9) ▷ Both samples
  Update communication model parameters $(\theta_k^{C_S}, \theta_k^{C_L})$ with
    $\nabla\theta_k^C = \nabla L_k^{C_{KD}}$ ▷ Gradient flows through communication model only
    $\theta_k = \theta_k - \eta\nabla\theta_k^C, \quad where\ \theta_k = \{\theta_k^{C_S}, \theta_k^{C_L}\}$
  Upload updated shared parameters $\theta_k^{C_S}$ to server
  Unfreeze prediction model parameters
end for
// Server Aggregation Phase
Server aggregates $\theta_k^S + 1$ from all clients using baseline federated aggregate method.
Set $k = k + 1$ and repeat main loop until $k > K$ ▷ Continue Main Training Loop
return Final prediction model parameters of each client.

CoDFKGE is designed to be model-agnostic, enabling seamless integration with diverse FKGE models based on their shared parameter types. Both communication and prediction models used by CoDFKGE clients utilize the same scoring function g as the original KGE model. Clients upload and utilize shared parameters identically to the baseline model, with these parameters maintaining the same form and dimensionality as the original implementation. This parameter compatibility enables the server to aggregate updates using existing federated learning aggregation methods without modification. This design ensures that CoDFKGE preserves the original knowledge representation capabilities while maintaining consistent operational semantics with the baseline model.

5.1. Communication to prediction model distillation

In the first iteration, the model trains the prediction component following the standard procedure. Starting from the second iteration of the training process, the communication model loads the server-shared parameters θ_k^S and initializes itself jointly with the local embeddings θ_{k-1}^L from the previous iteration's local prediction model.

After the communication model receives and applies the server-shared parameters, it filters out potentially poisoned model parameters through knowledge distillation. The communication model acts as a teacher model to transfer clean knowledge to the prediction model, which serves as the student model. During this process, the prediction model parameters are frozen to ensure that the knowledge transfer direction is strictly from the communication model to the prediction model. Gradients only flow through the prediction model parameters, while the communication model parameters remain frozen, preventing gradient leakage back to potentially poisoned shared parameters.

If the communication model suffers from poisoning attacks and contains the poisoning parameter, its outputs for negative samples are not reliable. Distilling or teaching such uncertain predictions would propagate noise rather than useful knowledge. To exclude the poisoned knowledge, the prediction model should focus on positive samples during distillation, ensuring that only trustworthy knowledge is transferred. The mathematical expression for the distillation loss of the prediction model in the kth training epoch is provided in Eq. (7).

$$L_k^{P_{KD}} = \tau^2 \sum_{(h,r,t)\in\mathcal{T}} D_{KL}\left(\sigma(g(h,r,t;\theta_k^S,\theta_{k-1}^{P_L})) \parallel \sigma(g(h,r,t;\theta_k^{P_S},\theta_k^{P_L}))\right) \tag{7}$$

Among them, t is the distillation temperature coefficient, and σ is the softmax function of the ratio of the model output to t. g represents the scoring function of the prediction model, which is used to compute the KGE loss. $g(h,r,t;\theta_k^S,\theta_{k-1}^{P_L})$ represents the communication model output under server-shared parameter θ_k^S and local parameter $\theta_{k-1}^{P_L}$, and $g(h,r,t;\theta_k^{P_S},\theta_k^{P_L})$ represents the training prediction model output.

When training distillation, the model also needs to consider the KGE loss function. The overall loss function of the prediction model is the weighted sum of the KGE loss and the distillation loss, and its mathematical expression is shown in Eq. (8).

$$L_k^P = \beta L_k^{P_{KGE}} + (1-\beta) L_k^{P_{KD}} \tag{8}$$

Where, $L_{KGE}^{P_k}$ is the KGE loss of the kth epoch of the prediction model defined by Eq. (1), and β is the weight.

5.2. Prediction to communication model distillation

After training the prediction model, we train the communication model through distillation, which extracts and propagates knowledge without directly sharing prediction parameters, thereby avoiding privacy leakage. During the communication model's distillation, the output of the prediction model under positive and negative samples serves as soft labels. As Eq. (1) illustrates, the loss function must account for the probability of negative samples when balancing the impact of positive and negative predictions. Therefore, the distillation loss function of the communication model is formalized in Eq. (9).

$$\begin{aligned} L_{KD}^{C_k} = \tau^2 \sum_{(h,r,t)\in\mathcal{T}} & (D_{KL}(\sigma(g(h,r,t;\theta_k^{P_S},\theta_k^{P_L})) \parallel \sigma(g(h,r,t;\theta_k^{C_S},\theta_k^{C_L})))) \\ & + \sum_i p(h,r,t_i') D_{KL}(\sigma(g(h,r,t_i';\theta_k^{P_S},\theta_k^{P_L}) \parallel \sigma(g(h,r,t_i';\theta_k^{C_S},\theta_k^{C_L})))) \end{aligned} \tag{9}$$

Among them, $g(h,r,t;\theta_k^{C_S},\theta_k^{C_L})$ represents the communication model output. $g(h,r,t;\theta_k^{P_S},\theta_k^{P_L})$ represents the prediction model output under shared parameter $\theta_k^{P_S}$ and local parameter $\theta_k^{P_L}$. The calculation method of p follows the approach in [9], with its mathematical formulation provided in Eq. (10).

$$p(h,r,t_i') = \frac{\exp \tau_\alpha g(h,r,t_i')}{\sum_j \exp \tau_\alpha g(h,r,t_j')} \tag{10}$$

Where τ_α is the self-adversarial sampling temperature.

After the bidirectional distillation process of CoDFKGE, the communication model parameters are updated to $\theta_k^{C_S}$ and $\theta_k^{C_L}$. Client then uploads $\theta_k^{C_S}$ to the server, which aggregates these parameters from all clients using federated averaging to generate the next round's shared parameters θ_{k+1}^S.

6. Experiments

Experiments are conducted on the open available dataset FB15K-237 [35], which is a subset of Freebase, containing 14,505 entities, 544,230 triples, and 474 relations. To perform federated learning, we adopt the relational partitioning method in [22]. This method first partitions the relationships through clustering, ensuring that the triple relationships within each partition are as close as possible. Then, these partitions are divided into groups of roughly equal numbers of triples and distributed to the client. This results in tighter triple relationships within the client, better reflecting real-world scenarios.

The TransE model [7] is selected as the KGE model, serving as the foundation for all federated learning methods in the experiments—including the attacker's shadow model. To benchmark CoDFKGE, we select multiple baseline models. First, the local training model without federated learning is selected as the KGE baseline model. It does not share parameters between clients, so it has no communication overhead and is not vulnerable to poisoning attacks. Then, FedE [13] and FedR [15] are also chosen as baseline FGKE models, representing standard approaches in the field. Additionally, we implement a knowledge distillation model, which utilizes communication and prediction models similar to CoDFKGE but only processes a unidirectional knowledge distillation. Specifically, it uses the communication model as the teacher model and the prediction model as the student model to filter out poisoning knowledge, with the distillation loss function following Eq. (4).

All experiments are performed on a 72-core Ubuntu 18.04.6 LTS machine with an Intel(R) Xeon(R) Gold 5220 CPU @ 2.20 GHz and a V100S-PCIE-32GB GPU. We implemented the proposed FKGE framework and baseline model based on PyTorch Geometric [36] and distributed AI framework Ray [37]. We used KGE hyperparameter settings based on [9] and FKGE hyperparameter settings based on FedE [13]. Specifically, we used the Adam [38] optimizer with a learning rate of 1e-3. γ is 10, and self-advertise negative sampling temperature τ_α in KGE is 1. The distillation temperature τ is 2, and the coefficient β of distillation and KGE loss are both 0.5. The maximum training epoch is 400. In each epoch, the client performs 3 iterations locally before uploading the parameters to the server.

We utilize the link prediction task, a sub-task of KGE, to validate the model's accuracy. Referencing the common implementation of the link prediction, we employ the Mean Reciprocal Rank (MRR) and Hits@N as accuracy metrics. The MRR is the average of the reciprocals of the ranks of the predicted triples among all possible triples. Mathematically, if $rank_i$ is the rank of the correct triple for the ith query, and n is the total number of queries, then $MRR = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{rank_i}$. The Hits@N is the proportion of query triples for which the correct triple is present among the top N candidates generated by the model. Generally, higher values for both metrics indicate better model performance in link prediction.

Through experiments, the following research questions will be verified.

RQ1 Does CoDFKGE maintain KGE prediction performance while reducing FKGE communication overhead?

RQ2 Can CoDFKGE effectively defend against **targeted** poisoning attacks?

Table 1
Experiment result on normal link prediction.

Fed type	Model	Mem(MB)	CC(MB)	MRR	Hits@1	Hits@5	Hits@10
Local	Local(128)	57.05	–	0.4081 ± 0.0015	0.3066 ± 0.0014	0.5223 ± 0.0023	0.6077 ± 0.0015
Entity	FedE(128)	185.58	42.60	0.4082 ± 0.0004	0.3068 ± 0.0012	0.5232 ± 0.0013	0.6080 ± 0.0018
Entity	Distillation (128-128)	356.10	42.60	**0.4129** ± 0.0008	**0.3118** ± 0.0016	**0.5279** ± 0.0008	**0.6122** ± 0.0003
Entity	CoDFKGE (128-128)	356.10	42.60	0.4109 ± 0.0043	0.3097 ± 0.0041	0.5246 ± 0.0044	0.6087 ± 0.0040
Entity	Distillation (32-128)	217.39	10.65	0.3914 ± 0.0011	0.2935 ± 0.0008	0.5005 ± 0.0014	0.5838 ± 0.0032
Entity	CoDFKGE (32-128)	217.40	10.65	0.4090 ± 0.0010	0.3079 ± 0.0007	0.5233 ± 0.0019	0.6068 ± 0.0019
Relation	FedR(128)	75.49	0.69	0.4085 ± 0.0011	0.3079 ± 0.0021	0.5219 ± 0.0016	0.6066 ± 0.0017
Relation	Distillation (128-128)	151.74	0.69	**0.4106** ± 0.0013	**0.3092** ± 0.0023	**0.5242** ± 0.0008	**0.6098** ± 0.0009
Relation	CoDFKGE (128-128)	150.02	0.69	0.4065 ± 0.0007	0.3056 ± 0.0013	0.5190 ± 0.0023	0.6063 ± 0.0012
Relation	Distillation (32-128)	94.53	0.17	0.3920 ± 0.0012	0.2960 ± 0.0007	0.4996 ± 0.0019	0.5807 ± 0.0013
Relation	CoDFKGE (32-128)	93.69	0.17	0.4078 ± 0.0009	0.3060 ± 0.0007	0.5224 ± 0.0031	0.6074 ± 0.0015

RQ3 Can CoDFKGE effectively defend against **untargeted** poisoning attacks?

RQ4 Do the two proposed distillation loss functions individually contribute to poisoning defense?

6.1. Normal link prediction (RQ1)

To explore the performance of the proposed model in normal link prediction, we first tested the model on a conventional dataset. The performance of the model is measured using MRR and Hits@1, Hits@5, and Hits@10. The model is trained by federated learning and evaluated on the local test sets of clients.

Table 1 lists the performance of the local KGE model, FedE, FedR, and CoDFKGE with different dimensions. The experimental results are grouped according to the type of shared embeddings and the dimension of the prediction model. The parameter dimensions are specified in parentheses within the "Model" column. For example, CoDFKGE(32-128) denotes the CoDKGE model with a 32-dimensional communication model and a 128-dimensional prediction model. All link prediction experiments were repeated 5 times with different random seeds, and the accuracy results of all models are reported as (mean ± standard deviation). The best performing model results in each group (excluding the local model) are **bolded**. The results of the CoDFKGE (32-128) model that are better than those of Distillation(32-128) are underlined.

The performance of locally trained models is lower than most federated learning models, highlighting the advantages of sharing model parameters. High-dimensional distillation(128-128) models achieve better link prediction performance. Compared to distillation(128-128), CoDFKGE models show slightly inferior prediction performance. However, by comparing models with the same dimensions, CoDFKGE outperform both local baselines and federated baselines (FedE, FedR). The co-distillation process in CoDFKGE may lead to a loss of generalization accuracy. We believe that the main advantage of CoDFKGE is its ability to enhance the security of FKGE. In addition to the security performance demonstrated in Sections 6.2 and 6.3, it also maintains link prediction performance comparable to its baseline FKGE models.

Beyond accuracy metrics, the "CC" (Communication Cost) column reports the communication overhead per training epoch, which is calculated based on the byte size of PyTorch Embedding used in the implementation. The "Mem" column shows the GPU memory usage of federated models in MB. Distillation-based model requires maintaining two KGE models, resulting in higher computational resource consumption. Distillation-based models need larger GPU memory to store the parameters of both models. Compared to using model parameters of the same size, distillation-based models allow to compress parameters in the communication model, achieving significantly lower communication overhead. In cases of smaller communication overhead, CoDFKGE(32-128) outperforms distillation(32-128) in link prediction performance. Therefore, we believe that the CoDFKGE model does not degrade the normal link prediction performance of baseline FKGE models and can effectively reduce the communication overhead of the model.

6.2. Targeted poisoning attack experiment (RQ2)

In the targeted poisoning attack, 32 pairs of non-existent triples are selected as attack targets from the victim's KG through negative sampling to construct a poisoned triple dataset. First, a predetermined number of normal triples are selected from the victim's training triples. Subsequently, the head or tail nodes of these triples are randomly replaced, and any triples already existing in the training set are iteratively removed until 32 pairs of non-existent triples are successfully constructed. In each epoch, the shadow model undergoes the same number of local training rounds as legitimate clients on the poisoned dataset to generate poisoned parameters. The malicious server aggregates these poisoned parameters with the parameters of the normal client into shared parameters and distributes them to all clients. Attackers can assign high weights to poisoned model parameters during aggregation. Following the setup in Ref. [33], we set the weight of the attacker's aggregated poisoned triples to be 256 times that of normal triples. Experiments focus on models with shared entity parameters (required for targeted poisoning attacks) and non-federated local baselines.

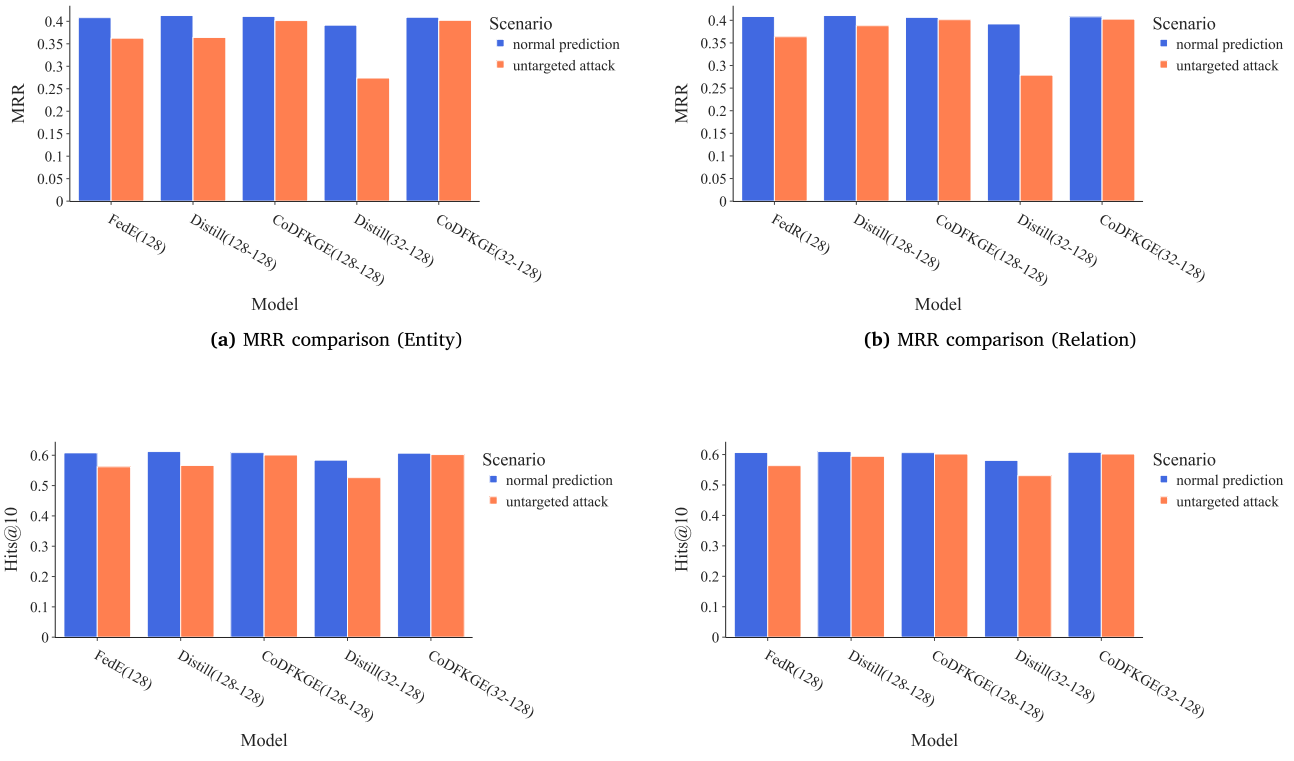
For space considerations, this section reports only MRR and Hits@10 metrics. Attack effectiveness is measured by the MRR and Hits@10 of poisoned triples on the victim. The higher metrics of the poisoned triples indicate greater vulnerability to poisoning and weaker resistance of the model to targeted poisoning attacks.

Table 2 lists the performance of baseline models and CoDFKGE under targeted poisoning attacks, grouped by the prediction model dimension. The parameter dimensions are specified in parentheses within the "Model" column. The "All Clients" column reports average performance across all clients' test sets during attacks, while "Victim Poisoned" measures the victim's performance on predicting poisoned triples. All experiments were repeated 5 times with different random seeds, and the results are reported as (mean ± standard deviation). The best performing model results are **bolded**. Moreover, the "Communication Poison" column highlights the communication model's performance on poisoned triples for CoDFKGE and the distillation model, demonstrating that both communication models are impacted by targeted poisoning attacks. Through distillation, the prediction accuracy of poisoned triples by the prediction model decreases in both cases.

For targeted poisoning attacks, the primary evaluation metrics should be the MRR and Hits@10 performance indicators of the victim model when predicting poisoned triples. The Local training model, which does not employ federated learning, remains immune to poisoning attacks, resulting in low MRR for poisoned triples, with the Hits@10 value being exactly 0. This indicates that the unpoisoned Local model does not include non-existent poisoned triples among its top 10 candidate results when making predictions. If a model incorrectly marks non-existent poisoned test triples as one of the top 10 candidates, it demonstrates that the poisoning attack has successfully manipulated the model's predictions. Therefore, we use Hits@10 as the metric to measure the Attack Success Rate (ASR).

Table 2
Experiment result under targeted poisoning attack.

Model	All clients		Victim poison		Communication poison	
	MRR	Hits@10	MRR	Hits@10(ASR)	MRR	Hits@10
Local(128, unpoisoned)	0.4081 ± 0.0015	0.6077 ± 0.0015	0.0003 ± 0.0001	0.0000 ± 0.0000	–	–
FedE(128)	0.4034 ± 0.0035	0.6004 ± 0.0029	0.4450 ± 0.0938	0.7857 ± 0.1248	–	–
Distillation(128-128)	0.4026 ± 0.0025	0.6006 ± 0.0039	0.0844 ± 0.0552	0.2000 ± 0.1311	0.4999 ± 0.1429	0.7714 ± 0.1046
CoDFKGE(128-128)	**0.4086** ± 0.0007	**0.6089** ± 0.0012	**0.0010** ± 0.0003	**0.0009** ± 0.0005	0.4694 ± 0.1511	0.6589 ± 0.1242
Distillation(32-128)	0.3821 ± 0.0022	0.5717 ± 0.0018	0.1511 ± 0.3356	0.1960 ± 0.4362	0.4919 ± 0.2364	0.6625 ± 0.1887
CoDFKGE(32-128)	0.3856 ± 0.0039	0.5740 ± 0.0054	**0.0010** ± 0.0001	0.0010 ± 0.0003	0.3794 ± 0.0032	0.5702 ± 0.005



(a) MRR comparison (Entity)

(b) MRR comparison (Relation)

(c) Hits@10 comparison (Entity)

(d) Hits@10 comparison (Relation)

Fig. 3. Performance degradation comparison.

The FedE model maintains high prediction accuracy on normal test triples when under attack, but exhibits abnormally high MRR and Hits@10 metrics for targeted poisoned triples, even exceeding those of normal triples. This indicates that targeted poisoning attacks can effectively manipulate the FedE model to generate incorrect prediction results. Similarly, in distillation-based models, their communication models are severely affected by poisoning attacks, while the impact on the prediction models is relatively minor. Although the distill(128-128) model can partially eliminate poisoning knowledge, it still remains vulnerable to the targeted poisoning attacks. Moreover, as the dimension of the communication model parameter increases, the extent of the model's vulnerability to poisoning attacks also grows.

In contrast, CoDFKGE's prediction model performs distillation learning exclusively on verified positive samples, effectively eliminating potential poisoning knowledge that might exist in negative samples. Similar to the Local training model, CoDFKGE achieves extremely low MRR and Hits@10 metrics for poisoned triples, which fully demonstrates that the CoDFKGE model can effectively defend against targeted poisoning attacks in FKGE. Furthermore, due to the compression of the communication model's dimension, the amount of information that attackers can transmit is correspondingly reduced, making the communication model in CoDFKGE(32-128) less susceptible to poisoning attacks.

6.3. Untargeted poisoning attack experiment (RQ3)

In untargeted poisoning attack experiments, the attacker returns negative aggregate parameters to the victim client, making the victim model non-converge and degrading prediction performance. The results presented in this section reflect average prediction performance on local test triples of clients.

Table 3 lists the performance of each model under untargeted poisoning attacks, grouped by the prediction model dimension and federated type. The parameter dimensions are specified in parentheses within the "Model" column. The "All Clients" column shows the average performance of all clients under untargeted poisoning attacks, and the "Victim Client" column shows the performance of the victim client. To measure the severity of the model being attacked, the MRR of the local model in Table 1 is used as a benchmark. The "Decay Ratio" column shows the ratio of performance degradation on the victim client compared to the local model shown in Table 1. All experiments were repeated 5 times with different random seeds, and the results

Table 3
Experiment result under untargeted poisoning attack.

Fed Type	Model	All clients		Victim		Decay ratio (%)	
		MRR	Hits@10	MRR	Hits@10	MRR	Hits@10
Entity	FedE(128)	0.3896 ± 0.0010	0.5939 ± 0.0009	0.3625 ± 0.0102	0.5620 ± 0.0144	11.21	7.58
Entity	Distillation(128-128)	0.3900 ± 0.0017	0.5921 ± 0.0007	0.3641 ± 0.0012	0.5664 ± 0.0018	11.82	7.54
Entity	CoDFKGE(128-128)	0.4084 ± 0.0007	0.6068 ± 0.0003	0.4017 ± 0.0010	0.6009 ± 0.0005	2.25	1.28
Entity	Distillation (32-128)	0.3024 ± 0.0208	0.5422 ± 0.0105	0.2739 ± 0.0264	0.5262 ± 0.0124	30.02	9.49
Entity	CoDFKGE (32-128)	**0.4093** ± 0.0018	**0.6081** ± 0.0014	**0.4022** ± 0.0022	**0.6023** ± 0.0011	**1.66**	**0.75**
Relation	FedR(128)	0.3915 ± 0.0010	0.5951 ± 0.0016	0.3637 ± 0.0093	0.5636 ± 0.0150	10.96	7.10
Relation	Distillation(128-128)	0.3978 ± 0.0017	0.6022 ± 0.0019	0.3881 ± 0.0023	0.5942 ± 0.0028	5.51	2.56
Relation	CoDFKGE(128-128)	0.4086 ± 0.0017	**0.6075** ± 0.0029	0.4014 ± 0.0020	**0.6018** ± 0.0037	**1.24**	**0.75**
Relation	Distillation (32-128)	0.3058 ± 0.0079	0.5463 ± 0.0029	0.2787 ± 0.0101	0.5307 ± 0.0038	27.78	8.61
Relation	CoDFKGE (32-128)	**0.4090** ± 0.0008	0.6066 ± 0.0011	**0.4026** ± 0.0008	**0.6018** ± 0.0013	1.27	0.92

Table 4
Ablation study in normal link prediction and under targeted attack.

Model	Link prediction		Targeted all clients		Targeted victim poisoning	
	MRR	Hits@10	MRR	Hits@10	MRR	Hits@10 (targeted poisoning ASR)
CoDFKGE	0.4112 ± 0.0039	0.6084 ± 0.0036	0.4086 ± 0.0007	**0.6089** ± 0.0012	**0.0010** ± 0.0003	**0.0009** ± 0.0005
Ablation(Comm)	0.4095 ± 0.0016	0.6074 ± 0.0014	0.4086 ± 0.0022	0.6076 ± 0.0021	0.0017 ± 0.0008	0.0013 ± 0.0008
Ablation(Pred)	**0.4132** ± 0.0006	**0.6116** ± 0.0012	**0.4098** ± 0.0011	0.6080 ± 0.0009	0.8086 ± 0.0064	0.9702 ± 0.0228

are reported as (mean ± standard deviation). The best and second best results in each group have been marked in **bold** and underline.

From the experimental results, it can be observed that when subjected to untargeted poisoning attacks, the CoDFKGE series models achieve optimal MRR and Hits@10 performance metrics compared to other models. In this context, all models exhibit varying degrees of decline in both their overall performance metrics and their performance metrics on victims. In Fig. 3, we present a comparison of the prediction performance of various models under normal link prediction and untargeted poisoning attack scenarios. It can be observed that the Distillation(32-128) model experiences the most significant performance degradation; for Distillation(128-128), FedE, and FedR models, their performance degradation is also substantial and cannot be ignored. These models directly incorporate poisoned global knowledge as an integral part of their own models, causing the convergence process of the models to be adversely affected. In contrast, the performance degradation of CoDFKGE models is fully within 3%. This is because even in the absence of global knowledge, the prediction model of CoDFKGE still utilizes local data knowledge for training, and its training effectiveness is comparable to that of local KGE models without knowledge sharing.

Baseline models may have their results manipulated or exhibit significant performance degradation when facing poisoning attacks. Although in link prediction experiments, distillation models exhibited advantages in performance, their defense effectiveness is extremely limited when facing poisoning attacks. In contrast, CoDFKGE remains unmanipulated when encountering targeted poisoning attacks and does not exhibit significant performance degradation when subjected to untargeted poisoning attacks, demonstrating its effective defense capability against poisoning attacks.

6.4. Ablation study (RQ4)

This section evaluates the defensive effects of applying different loss functions in CoDFKGE against poisoning attacks. Specifically, we compare the performance of models using 128-dimensional training parameters for both communication and prediction models across normal link prediction, targeted poisoning attack scenarios, and untargeted poisoning attack scenarios. Two ablation baselines were implemented: Ablation(Comm) applies the baseline loss function (Eq. (4)) solely during the communication module's distillation, while Ablation(Pred) uses it exclusively for the prediction module's distillation.

Tables 4 and 5 shows the experiment results of models with different distillation loss functions sharing entity embeddings. All experiments were repeated 5 times with different random seeds, and the results are reported as (mean ± standard deviation). The best results are **bolded**.

Experimental results demonstrate that while Ablation(Pred) performs well in conventional link prediction, its resistance to poisoning attacks lags behind the other two models due to not employing a negative sample exclusion strategy in its loss function. Among the remaining two models, while both demonstrate robust resilience against poisoning attacks, the CoDFKGE model achieves superior link prediction performance compared to Ablation(Comm). Ablation(Comm) employs a baseline loss function during the distillation training of the communication model. In contrast, the CoDFKGE model adopts the approach from [9] and utilizes self-adversarial sampling temperature τ_α to reweight negative samples, thereby enhancing the model's ability to distinguish between negative samples. Overall, the ablation experiments demonstrate that applying the proposed distillation loss functions simultaneously enhances the model's capability in defending against poisoning attacks and link prediction.

7. Conclusion

This paper proposes CoDFKGE, a co-distillation-based defense framework for FKGE poisoning attacks. As the first co-distillation defense framework against poisoning attacks in FKGE, CoDFKGE does have some limitations. First, maintaining two separate models requires higher computational resource consumption on clients. Second, the bidirectional distillation process may lead to a loss of generalization accuracy. In contrast, CoDFKGE's advantages lie in its model-agnostic applicability to existing FKGE models without compromising performance. By decoupling clients' prediction models from shared parameter models, CoDFKGE effectively filters out poisoned knowledge embedded in shared updates. CoDFKG eliminates malicious manipulations under targeted poisoning attacks, and significantly mitigates accuracy degradation under untargeted poisoning attacks. Leveraging distillation, the framework further reduces communication overhead. This work provides new ideas for enhancing the security of FKGE.

The limitations of FKGE poisoning defense research are partially rooted in the unique characteristics of KGE. When considering translation-based KGE models in FKGE, sharing entity or relation embeddings introduces risks related to both privacy preservation and poisoning attacks. Employing GNN-based KGE models in FKGE that transmit GNN parameters or gradients can alleviate these concerns. However, due to their superior robustness to sparse data and lower computational resource requirements, translation-based models still maintain unparalleled advantages in specific application scenarios.

Table 5
Ablation study under untargeted attack.

Model	Untargeted all clients		Untargeted victim		Decay ratio (%)	
	MRR	Hits@10	MRR	Hits@10	MRR	Hits@10
CoDFKGE	**0.4084** ± 0.0007	**0.6068** ± 0.0003	**0.4017** ± 0.0010	**0.6009** ± 0.0005	**2.25**	1.27
Ablation(Comm)	0.4056 ± 0.0017	0.6062 ± 0.0011	0.3996 ± 0.0018	0.6003 ± 0.0013	2.42	**1.16**
Ablation(Pred)	0.3951 ± 0.0011	0.6022 ± 0.0008	0.3852 ± 0.0009	0.5951 ± 0.0005	6.76	2.69

For future research, we recommend exploring the application of the CoDFKGE framework in more complex real-world scenarios, such as personalized FKGE problems. Additionally, in large-scale dynamic KG environments, the security landscape for FKGE may undergo significant changes, necessitating further investigation into defense methods tailored to these evolving scenarios.

CRediT authorship contribution statement

Yiqin Lu: Supervision. **Jiarui Chen:** Writing – original draft, Software, Methodology. **Jiancheng Qin:** Writing – review & editing.

Declaration of Generative AI and AI-assisted technologies in the writing process

During the preparation of this work the author(s) used deepseek in order to improve language and readability. After using this tool/service, the author(s) reviewed and edited the content as needed and take(s) full responsibility for the content of the publication.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgment

This work is supported by the Special Project for Research and Development in Key Areas of Guangdong Province, under Grant 2019B010137001.

Data availability

Data will be made available on request.

References

[1] X. Zhao, H. Chen, Z. Xing, C. Miao, Brain-inspired search engine assistant based on knowledge graph, IEEE Trans. Neural Netw. Learn. Syst. 34 (8) (2021) 4386–4400.
[2] S. Sharma, Fact-finding knowledge-aware search engine, in: Data Management, Analytics and Innovation: Proceedings of ICDMAI 2021, vol. 2, Springer, 2021, pp. 225–235.
[3] Y. Jiang, Y. Yang, L. Xia, C. Huang, DiffKG: Knowledge graph diffusion model for recommendation, in: Proceedings of the 17th ACM International Conference on Web Search and Data Mining, WSDM '24, Association for Computing Machinery, New York, NY, USA, ISBN: 9798400703713, 2024, pp. 313–321.
[4] W. Wang, X. Shen, B. Yi, H. Zhang, J. Liu, C. Dai, Knowledge-aware fine-grained attention networks with refined knowledge graph embedding for personalized recommendation, Expert Syst. Appl. 249 (2024) 123710.
[5] J. Chen, Y. Lu, Y. Zhang, F. Huang, J. Qin, A management knowledge graph approach for critical infrastructure protection: Ontology design, information extraction and relation prediction, Int. J. Crit. Infrastruct. Prot. (ISSN: 1874-5482) 43 (2023) 100634.
[6] Y. Zhang, J. Chen, Z. Cheng, X. Shen, J. Qin, Y. Han, Y. Lu, Edge propagation for link prediction in requirement-cyber threat intelligence knowledge graph, Inform. Sci. (ISSN: 0020-0255) 653 (2024) 119770.
[7] A. Bordes, N. Usunier, A. Garcia-Duran, J. Weston, O. Yakhnenko, Translating embeddings for modeling multi-relational data, Adv. Neural Inf. Process. Syst. 26 (2013).
[8] Z. Wang, J. Zhang, J. Feng, Z. Chen, Knowledge graph embedding by translating on hyperplanes, in: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 28, 2014.
[9] Z. Sun, Z.-H. Deng, J.-Y. Nie, J. Tang, Rotate: Knowledge graph embedding by relational rotation in complex space, 2019, arXiv preprint arXiv:1902.10197.
[10] Z. Zhang, J. Jia, Y. Wan, Y. Zhou, Y. Kong, Y. Qian, J. Long, Transr*: Representation learning model by flexible translation and relation matrix projection, J. Intell. Fuzzy Systems 40 (5) (2021) 10251–10259.
[11] T. Dettmers, P. Minervini, P. Stenetorp, S. Riedel, Convolutional 2d knowledge graph embeddings, in: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 32, (1) 2018.
[12] B. McMahan, E. Moore, D. Ramage, S. Hampson, B.A. y Arcas, Communication-efficient learning of deep networks from decentralized data, in: Artificial Intelligence and Statistics, PMLR, 2017, pp. 1273–1282.
[13] M. Chen, W. Zhang, Z. Yuan, Y. Jia, H. Chen, Fede: Embedding knowledge graphs in federated setting, in: Proceedings of the 10th International Joint Conference on Knowledge Graphs, 2021, pp. 80–88.
[14] M. Chen, W. Zhang, Z. Yuan, Y. Jia, H. Chen, Federated knowledge graph completion via embedding-contrastive learning, Knowl.-Based Syst. 252 (2022) 109459.
[15] K. Zhang, Y. Wang, H. Wang, L. Huang, C. Yang, X. Chen, L. Sun, Efficient federated learning on knowledge graphs via privacy-preserving relation embedding aggregation, 2022, arXiv preprint arXiv:2203.09553.
[16] G. Hinton, O. Vinyals, J. Dean, Distilling the knowledge in a neural network, 2015, arXiv preprint arXiv:1503.02531.
[17] N. Papernot, P. McDaniel, X. Wu, S. Jha, A. Swami, Distillation as a defense to adversarial perturbations against deep neural networks, in: 2016 IEEE Symposium on Security and Privacy, SP, IEEE, 2016, pp. 582–597.
[18] K. Yoshida, T. Fujino, Countermeasure against backdoor attack on neural networks utilizing knowledge distillation, J. Signal Process. 24 (4) (2020) 141–144.
[19] K. Yoshida, T. Fujino, Disabling backdoor and identifying poison data by using knowledge distillation in backdoor attacks on deep neural networks, in: Proceedings of the 13th ACM Workshop on Artificial Intelligence and Security, 2020, pp. 117–127.
[20] R. Anil, G. Pereyra, A. Passos, R. Ormandi, G.E. Dahl, G.E. Hinton, Large scale distributed neural network training through online distillation, 2018, arXiv preprint arXiv:1804.03235.
[21] Y. Hu, W. Liang, R. Wu, K. Xiao, W. Wang, X. Li, J. Liu, Z. Qin, Quantifying and defending against privacy threats on federated knowledge graph embedding, in: Proceedings of the ACM Web Conference 2023, 2023, pp. 2306–2317.
[22] X. Zhu, G. Li, W. Hu, Heterogeneous federated knowledge graph embedding learning and unlearning, in: Proceedings of the ACM Web Conference 2023, 2023, pp. 2444–2454.
[23] X. Zhang, Z. Zeng, X. Zhou, Z. Shen, Low-dimensional federated knowledge graph embedding via knowledge distillation, 2024, arXiv preprint arXiv:2408.05748.
[24] Y. Liu, Z. Sun, G. Li, W. Hu, I know what you do not know: Knowledge graph embedding via co-distillation learning, in: Proceedings of the 31st ACM International Conference on Information & Knowledge Management, 2022, pp. 1329–1338.
[25] F. Xia, W. Cheng, A survey on privacy-preserving federated learning against poisoning attacks, Clust. Comput. 27 (10) (2024) 13565–13582.
[26] J. Chen, H. Yan, Z. Liu, M. Zhang, H. Xiong, S. Yu, When federated learning meets privacy-preserving computation, ACM Comput. Surv. (ISSN: 0360-0300) 56 (12) (2024).
[27] J. Xia, Z. Yue, Y. Zhou, Z. Ling, Y. Shi, X. Wei, M. Chen, Waveattack: Asymmetric frequency obfuscation-based backdoor attacks against deep neural networks, Adv. Neural Inf. Process. Syst. 37 (2024) 43549–43570.
[28] P. Blanchard, E.M. El Mhamdi, R. Guerraoui, J. Stainer, Machine learning with adversaries: Byzantine tolerant gradient descent, Adv. Neural Inf. Process. Syst. 30 (2017).
[29] N.M. Jebreel, J. Domingo-Ferrer, Fl-defender: Combating targeted attacks in federated learning, Knowl.-Based Syst. 260 (2023) 110178.
[30] Z. Yue, J. Xia, Z. Ling, M. Hu, T. Wang, X. Wei, M. Chen, Model-contrastive learning for backdoor elimination, in: Proceedings of the 31st ACM International Conference on Multimedia, 2023, pp. 8869–8880.
[31] H. Peng, H. Li, Y. Song, V. Zheng, J. Li, Differentially private federated knowledge graphs embedding, in: Proceedings of the 30th ACM International Conference on Information & Knowledge Management, CIKM '21, Association for Computing Machinery, New York, NY, USA, ISBN: 9781450384469, 2021, pp. 1416–1425.

[32] Y. Hu, Y. Wang, J. Lou, W. Liang, R. Wu, W. Wang, X. Li, J. Liu, Z. Qin, Privacy risks of federated knowledge graph embedding: New membership inference attacks and personalized differential privacy defense, IEEE Trans. Dependable Secur. Comput. (2024).
[33] E. Zhou, S. Guo, Z. Ma, Z. Hong, T. Guo, P. Dong, Poisoning attack on federated knowledge graph embedding, in: Proceedings of the ACM Web Conference 2024, 2024, pp. 1998–2008.
[34] G. Xia, J. Chen, C. Yu, J. Ma, Poisoning attacks in federated learning: A survey, Ieee Access 11 (2023) 10708–10722.
[35] K. Toutanova, D. Chen, P. Pantel, H. Poon, P. Choudhury, M. Gamon, Representing text for joint embedding of text and knowledge bases, in: Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, 2015, pp. 1499–1509.
[36] M. Fey, J.E. Lenssen, Fast graph representation learning with PyTorch Geometric, in: ICLR Workshop on Representation Learning on Graphs and Manifolds, 2019.
[37] P. Moritz, R. Nishihara, S. Wang, A. Tumanov, R. Liaw, E. Liang, M. Elibol, Z. Yang, W. Paul, M.I. Jordan, I. Stoica, Ray: A distributed framework for emerging AI applications, in: 13th USENIX Symposium on Operating Systems Design and Implementation (OSDI 18), USENIX Association, Carlsbad, CA, ISBN: 978-1-939133-08-3, 2018, pp. 561–577.
[38] D.P. Kingma, J. Ba, Adam: A method for stochastic optimization, 2014, arXiv preprint arXiv:1412.6980.