Poisoning Knowledge Graph Embeddings via Relation Inference Patterns

Peru Bhardwaj[1]   **John Kelleher**[2*]   **Luca Costabello**[3*]   **Declan O'Sullivan**[1*]

[1] ADAPT Centre, Trinity College Dublin, Ireland
[2] ADAPT Centre, TU Dublin, Ireland
[3] Accenture Labs, Ireland
`peru.bhardwaj@adaptcentre.ie`

Abstract

We study the problem of generating data poisoning attacks against Knowledge Graph Embedding (KGE) models for the task of link prediction in knowledge graphs. To poison KGE models, we propose to exploit their inductive abilities which are captured through the relationship patterns like symmetry, inversion and composition in the knowledge graph. Specifically, to degrade the model's prediction confidence on target facts, we propose to improve the model's prediction confidence on a set of decoy facts. Thus, we craft adversarial additions that can improve the model's prediction confidence on decoy facts through different inference patterns. Our experiments demonstrate that the proposed poisoning attacks outperform state-of-art baselines on four KGE models for two publicly available datasets. We also find that the symmetry pattern based attacks generalize across all model-dataset combinations which indicates the sensitivity of KGE models to this pattern.

1 Introduction

Knowledge graph embeddings (KGE) are increasingly deployed in domains with high stake decision making like healthcare and finance (Noy et al., 2019), where it is critical to identify the potential security vulnerabilities that might cause failure. But the research on adversarial vulnerabilities of KGE models has received little attention. We study the adversarial vulnerabilities of KGE models through data poisoning attacks. These attacks craft input perturbations at training time that aim to subvert the learned model's predictions at test time.

Poisoning attacks have been proposed for models that learn from other graph modalities (Xu et al., 2020) but they cannot be applied directly to KGE models. This is because they rely on gradients of
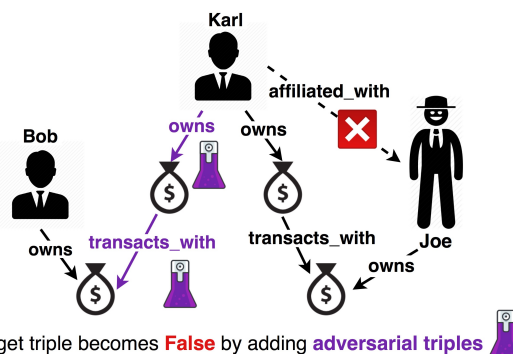

Figure 1: Composition based adversarial attack on fraud detection. The knowledge graph consists of two types of entities - Person and BankAccount. The target triple to predict is $(Karl, \texttt{affiliated\_with}, Joe\_the\_mobster)$. Original KGE model predicts this triple as True. But a malicious attacker adds adversarial triples (in purple) that connect $Karl$ with a non-suspicious person Bob through composition pattern. Now, the KGE model predicts the target triple as False.

all possible entries in a dense adjacency matrix and thus, do not scale to large knowledge graphs with multiple relations. The main challenge in designing poisoning attacks for KGE models is the large combinatorial search space of candidate perturbations which is of the order of millions for benchmark knowledge graphs with thousands of nodes. Two recent studies (Zhang et al., 2019a; Pezeshkpour et al., 2019) attempt to address this problem through random sampling of candidate perturbations (Zhang et al., 2019a) or through a vanilla auto-encoder that reconstructs discrete entities and relations from latent space (Pezeshkpour et al., 2019). However, random sampling depends on the number of candidates being sampled and the auto-encoder proposed in Pezeshkpour et al. (2019) is only applicable to multiplicative KGE models.

In this work, we propose to exploit the inductive abilities of KGE models to craft poisoned examples against the model. The inductive abilities of KGE models are expressed through different connectiv-

---

*Equal contribution by last authors.

ity patterns like symmetry, inversion and composition between relations in the knowledge graph. We refer to these as *inference patterns*. We focus on the task of link prediction using KGE models and consider the adversarial goal of *degrading* the predicted rank of *target* missing facts. To degrade the ranks of target facts, we propose to carefully select a set of decoy facts and exploit the inference patterns to improve performance on this decoy set. Figure 1 shows an example of the use of composition pattern to degrade KGE model's performance.

We explore a collection of heuristic approaches to select the decoy triples and craft adversarial perturbations that use different inference patterns to improve the model's predictive performance on these decoy triples. Our solution addresses the challenge of large candidate space by breaking down the search space into smaller steps - (i) determining adversarial relations; (ii) determining the decoy entities that most likely violate an inference pattern; and (iii) determining remaining adversarial entities in the inference pattern that are most likely to improve the rank of decoy triples.

We evaluate the proposed attacks on four state-of-art KGE models with varied inductive abilities - DistMult, ComplEx, ConvE and TransE. We use two publicly available benchmark datasets for link prediction - WN18RR and FB15k-237. Comparison against the state-of-art poisoning attacks for KGE models shows that our proposed attacks outperform them in *all* cases. We find that the attacks based on symmetry pattern perform the best and generalize across all model-dataset combinations.

Thus, the main contribution of our research is an effective method to generate data poisoning attacks, which is based on inference patterns captured by KGE models. Through a novel reformulation of the problem of poisoning KGE models, we overcome the existing challenge in the scalability of poisoning attacks for KGE models. Furthermore, the extent of effectiveness of the attack relying on an inference pattern indicates the KGE model's sensitivity to that pattern. Thus, our proposed poisoning attacks help in understanding the KGE models.

2   Problem Formulation

For a set of entities $\mathcal{E}$ and a set of relations $\mathcal{R}$, a knowledge graph is a collection of triples represented as $\mathcal{KG} = \{(s, \mathbf{r}, o) \,|\, s, o \in \mathcal{E} \, and \, \mathbf{r} \in \mathcal{R}\}$, where $s, \mathbf{r}, o$ represent the subject, relation and object in a triple. A Knowledge Graph Embedding (KGE) model encodes entities and relations to a low-dimensional continuous vector space $\mathbf{e}_s, \mathbf{e}_\mathbf{r}, \mathbf{e}_o \in \mathbb{R}^k$ where k is the embedding dimension. To do so, it uses a scoring function $f : \mathcal{E} \times \mathcal{R} \times \mathcal{E} \to \mathbb{R}$ which depends on the entity and relation embeddings to assign a score to each triple $f_{sro} = f(\mathbf{e}_s, \mathbf{e}_\mathbf{r}, \mathbf{e}_o)$. Table 1 shows the scoring functions of state-of-art KGE models studied in this research. The embeddings are learned such that the scores for true (existing) triples in the knowledge graph are higher than the scores for false (non-existing) triples in the knowledge graph.

Multiplicative vs Additive Interactions: The scoring functions of KGE models exhibit multiplicative or additive interactions (Chandrahas et al., 2018). The multiplicative models score triples through multiplicative interactions of subject, relation and object embeddings. The scoring function for these models can be expressed as $f_{sro} = \mathbf{e}_\mathbf{r}^\top \mathcal{F}(\mathbf{e}_s, \mathbf{e}_o)$ where the function $\mathcal{F}$ measures the compatibility between the subject and object embeddings and varies across different models within this family. DistMult, ComplEx and ConvE have such interactions. On the other hand, additive models score triples through additive interactions of subject, relation and object embeddings. The scoring function for such models can be expressed as $f_{sro} = -\left\| \mathbf{M}_\mathbf{r}^1(\mathbf{e}_s) + \mathbf{e}_\mathbf{r} - \mathbf{M}_\mathbf{r}^2(\mathbf{e}_o) \right\|$ where $\mathbf{e}_s, \mathbf{e}_o \in \mathbb{R}^{k_\mathcal{E}}$, $\mathbf{e}_\mathbf{r} \in \mathbb{R}^{k_\mathcal{R}}$ and $\mathbf{M}_\mathbf{r} \in \mathbb{R}^{k_\mathcal{E} \times k_\mathcal{R}}$ is the projection matrix from entity space $\mathbb{R}^{k_\mathcal{E}}$ to relation space $\mathbb{R}^{k_\mathcal{R}}$. TransE has additive interactions.

Inductive Capacity of KGE models: The general intuition behind the design of the scoring functions of KGE models is to capture logical properties between relations from the observed facts in the knowledge graph. These logical properties or *inference patterns* can then be used to make downstream inferences about entities and relations. For example, the relation is_owned_by

Model	Scoring Function
DistMult	$\langle \mathbf{e}_s, \mathbf{e}_\mathbf{r}, \mathbf{e}_o \rangle$
ComplEx	$\Re(\langle \mathbf{e}_s, \mathbf{e}_\mathbf{r}, \overline{\mathbf{e}_o} \rangle)$
ConvE	$\langle \sigma(\mathrm{vec}(\sigma([\overline{\mathbf{e}_\mathbf{r}}, \overline{\mathbf{e}_s}] * \mathbf{\Omega}))\mathbf{W}), \mathbf{e}_o \rangle$
TransE	$-\left\| \mathbf{e}_s + \mathbf{e}_\mathbf{r} - \mathbf{e}_o \right\|$

Table 1: Scoring functions f_{sro} of the KGE models used in this research. For ComplEx, $\mathbf{e}_s, \mathbf{e}_\mathbf{r}, \mathbf{e}_o \in \mathbb{C}^k$; for the remaining models $\mathbf{e}_s, \mathbf{e}_\mathbf{r}, \mathbf{e}_o \in \mathbb{R}^k$. Here, $\langle \cdot \rangle$ denotes the tri-linear dot product; σ denotes sigmoid activation function, $*$ denotes 2D convolution; $\bar{\cdot}$ denotes conjugate for complex vectors, and 2D reshaping for real vectors in ConvE model; $\|\cdot\|$ denotes l-p norm

is inverse of the relation owns, and when the fact $(Account42, \texttt{is\_owned\_by}, Karl)$ is true, then the fact $(Karl, \texttt{owns}, Account42)$ is also true and vice versa. A model that can capture inversion pattern can thus predict missing facts about owns based on observed facts about is_owned_by. The most studied inference patterns in the current literature are symmetry, inversion and composition since they occur very frequently in real-world knowledge graphs. In this work, we use these patterns to investigate the adversarial vulnerability of KGE models.

Link Prediction: Since most of the existing knowledge graphs are incomplete, a standard use case of KGE models is to predict missing triples in the $\mathcal{KG}$. This task is evaluated by an entity ranking procedure. Given a test triple $(s, \texttt{r}, o)$, the subject entity is replaced by each entity from $\mathcal{E}$ in turn. These replacements are referred to as *synthetic negatives*. The KGE model's scoring function is used to predict scores of these negative triples. The scores are then sorted in descending order and the rank of the correct entity is determined. These steps are repeated for the object entity of the triple.

The state-of-art evaluation metrics for this task are (i) *MR* which is the mean of the predicted ranks, (ii) *MRR* which is the mean of the reciprocals of predicted ranks and (iii) *Hits@n* which count the proportion of correct entities ranked in top-n. In the filtered setting (Bordes et al., 2013), negative triples that already exist in the training, validation or test set are filtered out. That is, their scores are ignored while computing the ranks. Depending on the domain of use, either subject or object or both ranks of the test triple are used to determine the model's confidence[1] in predicting a missing link.

Poisoning Attacks on KGE models: We study poisoning attacks for the task of link prediction using KGE models. We focus on targeted attacks where the attacker targets a specific set of missing triples instead of the overall model performance. We use the notation $(s, \texttt{r}, o)$ for the *target triple*; in this case, s, o are the *target entities* and $\texttt{r}$ is the *target relation*. The goal of an adversarial attacker is to degrade the ranks of missing triples which are predicted highly plausible by the model. The rank of a highly plausible target triple can be degraded by improving the rank of less plausible *decoy triples*. For a target triple $(s, \texttt{r}, o)$, the decoy triple for degrading the rank on object side would be $(s, \texttt{r}, o')$ and the decoy triple for degrading the

rank on subject side would be $(s', \texttt{r}, o)$. Thus, the aim of the adversarial attacker is to select decoy triples from the set of valid synthetic negatives and craft *adversarial edits* to improve their ranks. The attacker does not add the decoy triple itself as an adversarial edit, rather chooses the adversarial edits that would improve the rank of a missing decoy triple through an inference pattern.

Threat Model: To ensure reliable vulnerability analysis, we use a white-box attack setting where the attacker has full knowledge of the target KGE model (Joseph et al., 2019). They cannot manipulate the model architecture or learned embeddings directly; but only through *addition* of triples to the training data. We focus on adversarial *additions* which are more challenging to design than adversarial deletions for sparse knowledge graphs[2].

As in prior studies (Pezeshkpour et al., 2019; Zhang et al., 2019a), the attacker is restricted to making edits only in the neighbourhood of target entities. They are also restricted to 1 decoy triple for each entity of the target triple. Furthermore, because of the use of filtered settings for KGE evaluation, the attacker cannot add the decoy triple itself to the training data (which intuitively would be a way to improve the decoy triple's rank).

3 Poisoning Knowledge Graph Embeddings through Relation Inference Patterns

Since the inference patterns on the knowledge graph specify a logic property between the relations, they can be expressed as Horn Clauses which is a subset of FOL formulae. For example, a property represented in the form $\forall x, y : (x, \texttt{owns}, y) \Rightarrow (y, \texttt{is\_owned\_by}, x)$ means that two entities linked by relation owns are also likely to be linked by the inverse relation is_owned_by. In this expression, the right hand side of the implication $\Rightarrow$ is referred to as the *head* and the left hand side as the *body* of the clause. Using such expressions, we define the three inference patterns used in our research.

Definition 3.1. The **symmetry** pattern $\mathcal{P}_s$ is expressed as $\forall x, y : (x, \texttt{r}, y) \Rightarrow (y, \texttt{r}, x)$. Here, the relation $\texttt{r}$ is symmetric relation.

---

[1] KGE models do not provide model uncertainty estimates.

[2] For every target triple, the possible number of adversarial additions in the neighbourhood of each entity are $\mathcal{E} \times \mathcal{R}$. For the benchmark dataset FB15k-237, this is of the order of millions; whereas the *maximum* number of candidates for adversarial deletion are of the order of thousands.

Definition 3.2. The **inversion** pattern $\mathcal{P}_i$ is expressed as $\forall x, y : (x, \mathtt{r_i}, y) \Rightarrow (y, \mathtt{r}, x)$. Here, the relations $\mathtt{r_i}$ and $\mathtt{r}$ are inverse of each other.

Definition 3.3. The **composition** pattern $\mathcal{P}_c$ is expressed as $\forall x, y, z : (x, \mathtt{r_1}, z) \wedge (z, \mathtt{r_2}, y) \Rightarrow (x, \mathtt{r}, y)$. Here, the relation $\mathtt{r}$ is a composition of $\mathtt{r_1}$ and $\mathtt{r_2}$; and the $\wedge$ is the conjunction operator from relational logic.

The mapping $\mathcal{G} : \mathcal{V} \rightarrow \mathcal{E}$ of variables $\mathcal{V}$ in the above expressions to entities $\mathcal{E}$ is called a grounding. For example, we can map the logic expression $\forall x, y : (x, \mathtt{owns}, y) \Rightarrow (y, \mathtt{is\_owned\_by}, x)$ to the grounding $(Karl, \mathtt{owns}, Account42) \Rightarrow (Account42, \mathtt{is\_owned\_by}, Karl)$. Thus, a KGE model that captures the inversion pattern will assign a high prediction confidence to the head atom when the body of the clause exists in the graph.

In the above expressions, the decoy triple becomes the head atom and adversarial edits are the triples in the body of the expression. Since the decoy triple is an object or subject side negative of the target triple, the attacker already knows the relation in the head atom. They now want to determine (i) the adversarial relations in the body of the expression; (ii) the decoy entities which will most likely violate the inference pattern for the chosen relations and; (iii) the remaining entities in the body of the expression which will improve the prediction on the chosen decoy triple. Notice that the attacker needs all three steps for composition pattern only; for inversion pattern, only the first two steps are needed; and for symmetry pattern, only the second step is needed. Below we describe each step in detail. A computational complexity analysis of all the steps is available in Appendix A.

3.1 Step1: Determine Adversarial Relations

Expressing the relation patterns as logic expressions is based on relational logic and assumes that the relations are constants. Thus, we use an algebraic approach to determine the relations in the head and body of a clause. Given the target relation $\mathtt{r}$, we determine the adversarial relations using an algebraic model of inference (Yang et al., 2015).

Inversion: If an atom $(x, \mathtt{r}, y)$ holds true, then for the learned embeddings in multiplicative models, we can assume $\mathbf{e}_x \circ \mathbf{e_r} \approx \mathbf{e}_y$; where $\circ$ denotes the Hadamard (element-wise) product. If the atom $(y, \mathtt{r_i}, x)$ holds true as well, then we can also assume $\mathbf{e}_y \circ \mathbf{e_{r_i}} \approx \mathbf{e}_x$. Thus, $\mathbf{e_r} \circ \mathbf{e_{r_i}} \approx \mathbf{1}$ for inverse relations $\mathtt{r}$ and $\mathtt{r_i}$ when embeddings are learned

from multiplicative models. We obtain a similar expression $\mathbf{e_r} + \mathbf{e_{r_i}} \approx 0$ when embeddings are learned from additive models.

Thus, to determine adversarial relations for *inversion* pattern, we use the pre-trained embeddings to select $\mathtt{r_i}$ that minimizes $| \mathbf{e_{r_i}} \mathbf{e_r}^T - 1 |$ for multiplicative models; and $\mathtt{r_i}$ that minimizes $| \mathbf{e_{r_i}} + \mathbf{e_r} |$ for additive models.

Composition: If two atoms $(x, \mathtt{r_1}, y)$ and $(y, \mathtt{r_2}, z)$ hold true, then for multiplicative models, $\mathbf{e}_x \circ \mathbf{e_{r_1}} \approx \mathbf{e}_y$ and $\mathbf{e}_y \circ \mathbf{e_{r_2}} \approx \mathbf{e}_z$. Therefore, $\mathbf{e}_x \circ (\mathbf{e_{r_1}} \circ \mathbf{e_{r_2}}) \approx \mathbf{e}_z$. Hence, relation $\mathtt{r}$ is a composition of $\mathtt{r_1}$ and $\mathtt{r_2}$ if $\mathbf{e_{r_1}} \circ \mathbf{e_{r_2}} \approx \mathbf{e_r}$. Similarly, for embeddings from additive models, we can model composition as $\mathbf{e_{r_1}} + \mathbf{e_{r_2}} \approx \mathbf{e_r}$.

Thus, to determine adversarial relations for *composition* pattern, we use pre-trained embeddings to obtain all possible compositions of $(\mathtt{r_1}, \mathtt{r_2})$. For multiplicative models, we use $\mathbf{e_{r_1}} \circ \mathbf{e_{r_2}}$ and for additive models we use $\mathbf{e_{r_1}} + \mathbf{e_{r_2}}$. From these, we choose the relation pair for which the Euclidean distance between the composed relation embeddings and the target relation embedding $\mathbf{e_r}$ is minimum.

3.2 Step2: Determine Decoy Entities

We consider three different heuristic approaches to select the decoy entity - soft truth score, ranks predicted by the KGE model and cosine distance.

Soft Logical Modelling of Inference Patterns
Once the adversarial relations are determined, we can express the grounding for symmetry, inversion and composition patterns for the decoy triples. We discuss only object side decoy triple for brevity -

$$\mathcal{G}_s : (o', \mathtt{r}, s) \Rightarrow (s, \mathtt{r}, o')$$
$$\mathcal{G}_i : (o', \mathtt{r_i}, s) \Rightarrow (s, \mathtt{r}, o')$$
$$\mathcal{G}_c : (s, \mathtt{r_1}, o'') \wedge (o'', \mathtt{r_2}, o') \Rightarrow (s, \mathtt{r}, o')$$

If the model captures $\mathcal{P}_s$, $\mathcal{P}_i$ or $\mathcal{P}_c$ to assign high rank to the target triple, then the head atom $(s, \mathtt{r}, o')$ of a grounding that violates this pattern is a suitable decoy triple. Adding the body of this grounding to the knowledge graph would improve the model performance on decoy triple through $\mathcal{P}_s$, $\mathcal{P}_i$ or $\mathcal{P}_c$.

To determine the decoy triple this way, we need a measure of the degree to which a grounding satisfies an inference pattern. We call this measure the *soft truth score* $\phi : \mathcal{G} \rightarrow [0, 1]$ - it provides the truth value of a logic expression indicating the degree to which the expression is true. We model the soft truth score of grounded patterns using t-norm based fuzzy logics (Hájek, 1998).

The score f_{sro} of an individual atom (i.e. triple) is computed using the KGE model's scoring function. We use the sigmoid function $\sigma(x) = 1/(1 + \exp(-x))$ to map this score to a continuous truth value in the range $(0, 1)$. Hence, the soft truth score for an individual atom is $\phi(s, \mathbf{r}, o) = \sigma(f_{sro})$. The soft truth score for the grounding of a pattern can then be expressed through logical composition (e.g. $\wedge$ and $\Rightarrow$) of the scores of individual atoms in the grounding. We follow (Guo et al., 2016, 2018) and define the following compositions for logical conjunction ($\wedge$), disjunction ($\vee$), and negation ($\neg$):

$$\phi(a \wedge b) = \phi(a) \cdot \phi(b),$$
$$\phi(a \vee b) = \phi(a) + \phi(b) - \phi(a) \cdot \phi(b),$$
$$\phi(\neg a) = 1 - \phi(a).$$

Here, a and b are two logical expressions, which can either be single triples or be constructed by combining triples with logical connectives. If a is a single triple $(s, \mathbf{r}, o)$, we have $\phi(a) = \phi(s, \mathbf{r}, o)$. Given these compositions, the truth value of any logical expression can be calculated recursively (Guo et al., 2016, 2018).

Thus, we obtain the following soft truth scores for the groundings of symmetry, inversion and composition patterns $\mathcal{G}_s$, $\mathcal{G}_i$ and $\mathcal{G}_c$ -

$$\phi(\mathcal{G}_s) = \phi(o', \mathbf{r}, s) \cdot \phi(s, \mathbf{r}, o') - \phi(o', \mathbf{r}, s) + 1$$
$$\phi(\mathcal{G}_i) = \phi(o', \mathbf{r_i}, s) \cdot \phi(s, \mathbf{r}, o') - \phi(o', \mathbf{r_i}, s) + 1.$$
$$\phi(\mathcal{G}_c) = \phi(s, \mathbf{r_1}, o'') \cdot \phi(o'', \mathbf{r_2}, o') \cdot \phi(s, \mathbf{r}, o')$$
$$\qquad - \phi(s, \mathbf{r_1}, o'') \cdot \phi(o'', \mathbf{r_2}, o') + 1$$

To select the decoy triple $(s, \mathbf{r}, o')$ for symmetry and inversion, we score all possible groundings using $\phi(\mathcal{G}_s)$ and $\phi(\mathcal{G}_i)$. The head atom of grounding with minimum score is chosen as decoy triple.

For composition pattern, the soft truth score $\phi(\mathcal{G}_c)$ for candidate decoy triples $(s, \mathbf{r}, o')$ contains two entities (o', o'') to be identified. Thus, we use a greedy approach to select the decoy entity o'. We use the pre-trained embeddings to group the entities o'' into k clusters using K-means clustering and determine a decoy entity with minimum soft truth score for each cluster. We then select the decoy entity o' with minimum score across the k clusters.

KGE Ranks: We use the ranking protocol from KGE evaluation to rank the target triple against valid subject and object side negatives $(s', \mathbf{r}, o)$ and $(s, \mathbf{r}, o')$. For each side, we select the negative triple that is ranked just below the target triple (that

Adversarial Attack Step	Sym	Inv	Com
Determine Adversarial Relations	n/a	Alg	Alg
Determine Decoy Entities	Sft Rnk Cos	Sft Rnk Cos	Sft Rnk Cos
Determine Adversarial Entities	n/a	n/a	Sft

Table 2: A summary of heuristic approaches used for different steps of the adversarial attack with symmetry (Sym), inversion (Inv) and composition (Com) pattern. Alg denotes the algebraic model for inference patterns; Sft denotes the soft truth score; Rnk denotes the KGE ranks; and Cos denotes the cosine distance.

is, $negative\_rank = target\_rank + 1$). These are suitable as decoy because their predicted scores are likely not very different from the target triple's score. Thus, the model's prediction confidence for these triples might be effectively manipulated through adversarial additions. This is in contrast to very low ranked triples as decoy; where the model has likely learnt a low score with high confidence.

Cosine Distance: A high rank for the target triple $(s, \mathbf{r}, o)$ against queries $(s, \mathbf{r}, ?)$ and $(?, \mathbf{r}, o)$ indicates that $\mathbf{e}_s, \mathbf{e}_o$ are similar to the embeddings of other subjects and objects related by $\mathbf{r}$ in the training data. Thus, a suitable heuristic for selecting decoy entities s' and o' is to choose ones whose embeddings are dissimilar to $\mathbf{e}_s, \mathbf{e}_o$. Since these entities are not likely to occur in the neighbourhood of o and s, they will act adversarially to reduce the rank of target triple. Thus, we select decoy entities s' and o' that have maximum cosine distance from target entities s and o respectively.

3.3 Step3: Determine Adversarial Entities

This step is only needed for the composition pattern because the body for this pattern has two adversarial triples. Given the decoy triple in the head of the composition expression, we select the body of the expression that would maximize the rank of the decoy triple. We use the soft-logical model defined in Step 2 for selecting decoy triples. The soft truth score for composition grounding of decoy triple is given by $\phi(\mathcal{G}_t) = \phi(s, \mathbf{r_1}, o'') \cdot \phi(o'', \mathbf{r_2}, o') \cdot \phi(s, \mathbf{r}, o') - \phi(s, \mathbf{r_1}, o'') \cdot \phi(o'', \mathbf{r_2}, o') + 1$. We select the entity o'' with maximum score because this entity satisfies the composition pattern for the decoy triple and is thus likely to improve the decoy triple's ranks on addition to the knowledge graph.

	WN18RR	FB15k-237
Entities	40,559	14,505
Relations	11	237
Training	86,835	272,115
Validation	2,824	17,526
Test	2,924	20,438

Target		WN18RR	FB15k-237
	DistMult	1,315	3,342
	ComplEx	1,369	3,930
	ConvE	1,247	4,711
	TransE	1,195	5,359

Table 3: Statistics for the datasets WN18RR and FB15k-237. We removed triples from the validation and test set that contained unseen entities to ensure that we do not add new entities as adversarial edits. The numbers above (including the number of entities) reflect this filtering.

4 Evaluation

The aim of our evaluation is to assess the effectiveness of proposed attacks in *degrading* the predictive performance of KGE models on missing triples that are predicted true. We use the state-of-art evaluation protocol for data poisoning attacks (Xu et al., 2020). We train a clean model on the original data; then generate the adversarial edits and add them to the dataset; and finally retrain a new model on this poisoned data. All hyperparameters for training on original and poisoned data remain the same.

We evaluate four models with varying inductive abilities - DistMult, ComplEx, ConvE and TransE; on two publicly available benchmark datasets for link prediction[3]- WN18RR and FB15k-237. We filter out triples from the validation and test set that contain unseen entities. To assess the attack effectiveness in *degrading* performance on triples predicted as true, we need a set of triples that are predicted as true by the model. Thus, we select as *target triples*, a subset of the original test set where each triple is ranked ≤ 10 by the original model. Table 3 provides an overview of dataset statistics and the number of target triples selected.

Baselines: We compare the proposed methods against the following baselines -

Random_n: Random edits in the neighbourhood of each entity of the target triple.

Random_g1: Global random edits in the knowledge graph which are not restricted to the neighbourhood of entities in the target triple and have 1 edit per decoy triple (like symmetry and inversion).

Random_g2: Global random edits in the knowledge graph which are not restricted to the neigh-

bourhood of entities in the target triple and have 2 edits per decoy triple (like composition).

Zhang et al.: Poisoning attack from (Zhang et al., 2019a) for edits in the neighbourhood of subject of the target triple. We extend it for both subject and object to match our evaluation protocol. Further implementation details available in Appendix B.2.

CRIAGE: Poisoning attack from (Pezeshkpour et al., 2019). We use the publicly available implementation and the default attack settings[4]. The method was proposed for edits in the neighbourhood of object of the target triple. We extend it for both entities to match our evaluation protocol and to ensure fair evaluation.

Implementation: For every attack, we filter out adversarial edit candidates that already exist in the graph. We also remove duplicate adversarial edits for different targets before adding them to the original dataset. For Step 2 of the composition attack with ground truth, we use the elbow method to determine the number of clusters for each model-data combination. Further details on KGE model training, computing resources and number of clusters are available in Appendix B. The source code to reproduce our experiments is available on GitHub[5].

4.1 Results

Table 4 and 5 show the reduction in MRR and Hits@1 due to different attacks on the WN18RR and FB15k-237 datasets. We observe that the proposed adversarial attacks outperform the random baselines and the state-of-art poisoning attacks for all KGE models on both datasets.

We see that the attacks based on symmetry inference pattern perform the best across all model-dataset combinations. This indicates the sensitivity of KGE models to symmetry pattern. For DistMult, ComplEx and ConvE, this sensitivity can be explained by the symmetric nature of the scoring functions of these models. That is, the models assign either equal or similar scores to triples that are symmetric opposite of each other. In the case of TransE, the model's sensitivity to symmetry pattern is explained by the translation operation in scoring function. The score of target $(s, \mathtt{r}, o)$ is a translation from subject to object embedding through the relation embedding. Symmetry attack adds the adversarial triple $(o', \mathtt{r}, s)$ where the relation is same

---

[3]https://github.com/TimDettmers/ConvE

[4]https://github.com/pouyapez/criage

[5]https://github.com/PeruBhardwaj/InferenceAttack

1880

		DistMult		ComplEx		ConvE		TransE	
		MRR	Hits@1	MRR	Hits@1	MRR	Hits@1	MRR	Hits@1
Original		0.90	0.85	0.89	0.84	0.92	0.89	0.36	0.03
Baseline Attacks	Random_n	0.86 (-4%)	0.83	0.84 (-6%)	0.80	0.90 (-2%)	0.88	0.28 (-20%)	0.01
	Random_g1	0.88	0.83	0.88	0.83	0.92	0.89	0.35	0.02
	Random_g2	0.88	0.83	0.88	0.83	0.91	0.89	0.34	0.02
	Zhang et al.	0.82 (-8%)	0.81	0.76 (-14%)	0.74	0.90 (-2%)	0.87	**0.24 (-33%)**	**0.01**
	CRIAGE	0.87	0.84	-	-	0.90	0.88	-	-
Proposed Attacks	Sym_truth	0.66	0.40	**0.56 (-33%)**	**0.24**	**0.61 (-34%)**	**0.28**	0.57	0.36
	Sym_rank	0.61	0.32	**0.56 (-33%)**	**0.24**	0.62	0.31	0.25	0.02
	Sym_cos	**0.57 (-36%)**	**0.32**	0.62	0.43	0.67	0.44	**0.24 (-33%)**	**0.01**
	Inv_truth	0.87	0.83	0.86	0.80	0.90	0.87	0.34	0.03
	Inv_rank	0.86	0.83	0.85	0.80	0.89 (-4%)	0.85	0.25	0.02
	Inv_cos	0.83 (-8%)	0.82	0.80 (-10%)	0.79	0.90	0.88	0.25 (-30%)	0.01
	Com_truth	0.86	0.83	0.86	0.81	0.89	0.86	0.53 (+49%)	0.27
	Com_rank	0.85 (-5%)	0.80	0.83	0.77	0.89	0.84	0.57	0.32
	Com_cos	0.86	0.77	0.82 (-8%)	0.70	0.88(-4%)	0.83	0.53 (+49%)	0.27

Table 4: Reduction in MRR and Hits@1 due to different attacks on the **target split** of WN18RR. First block of rows are the baseline attacks with random edits; second block is state-of-art attacks; remaining are the proposed attacks. For each block, we report the *best* relative percentage difference from original MRR; computed as $(original - poisoned)/original * 100$. Lower values indicate better results; best results for each model are in bold. Statistics on the target split are in Table 3.

as the target relation, but target subject is the object of adversarial triple. Now, the model learns the embedding of s as a translation from o' through relation $\mathbf{r}$. This adversarially modifies the embedding of s and in turn, the score of $(s, \mathbf{r}, o)$.

We see that inversion and composition attacks also perform better than baselines in most cases, but not as good as symmetry. This is particularly true for FB15k-237 where the performance for these patterns is similar to random baselines. For the composition pattern, it is likely that the model has stronger bias for shorter and simpler patterns like symmetry and inversion than for composition. This makes it harder to deceive the model through composition than through symmetry or inverse. Furthermore, FB15k-237 has high connectivity (Dettmers et al., 2018) which means that a KGE model relies on a high number of triples to learn target triples' ranks. Thus, poisoning KGE models for FB15k-237 will likely require more adversarial triples per target triple than that considered in this research.

The inversion pattern is likely ineffective on the benchmark datasets because these datasets do not have any inverse relations (Dettmers et al., 2018; Toutanova and Chen, 2015). This implies that our attacks cannot identify the inverse of the target triple's relation in Step 1. We investigate this hypothesis further in Appendix D, and evaluate the attacks on WN18 dataset where the inverse relations have not been filtered out. This means that the KGE model can learn the inversion pattern and

the inversion attacks can identify the inverse of the target relation. In this setting, we find that the inversion attacks outperform other attacks against ComplEx on WN18, indicating the sensitivity of ComplEx to the inversion pattern when the dataset contains inverse relations.

An exception in the results is the composition pattern on TransE where the model performance improves instead of degrading on the target triples. This is likely due to the model's sensitivity to composition pattern such that adding this pattern improves the performance on all triples, including target triples. To verify this, we checked the change in ranks of decoy triples and found that composition attacks on TransE improve these ranks too. Results for this experiment are available in Appendix C. This behaviour of composition also indicates that the selection of adversarial entities in Step 3 of the composition attacks can be improved. It also explains why the increase is more significant for WN18RR than FB15k-237 - WN18RR does not have any composition relations but FB15k-237 does; so adding these to WN18RR shows significant improvement in performance. We aim to investigate these and more hypotheses about the proposed attacks in future work.

5 Related Work

KGE models can be categorized into tensor factorization models like DistMult (Yang et al., 2015) and ComplEx (Trouillon et al., 2016), neural archi-

		DistMult		ComplEx		ConvE		TransE	
		MRR	Hits@1	MRR	Hits@1	MRR	Hits@1	MRR	Hits@1
Original		0.61	0.38	0.61	0.45	0.61	0.45	0.63	0.48
Baseline Attacks	Random_n	0.54 (-11%)	0.40	0.54 (-12%)	0.40	0.56 (-8%)	0.41	0.60 (-4%)	0.45
	Random_g1	0.54	0.40	0.55	0.41	0.57	0.43	0.62	0.46
	Random_g2	0.55	0.41	0.55	0.40	0.57	0.42	0.61	0.46
	Zhang et al.	0.53 (-13%)	0.39	0.51 (-16%)	0.38	0.54 (-11%)	0.39	0.57 (-10%)	0.42
	CRIAGE	0.54	0.41	-	-	0.56	0.41	-	-
Proposed Attacks	Sym_truth	0.51	0.36	0.56	0.41	**0.51 (-17%)**	**0.34**	0.62	0.48
	Sym_rank	0.53	0.39	0.53	**0.38**	0.55	0.38	**0.53 (-16%)**	**0.36**
	Sym_cos	**0.46 (-25%)**	**0.31**	**0.51 (-17%)**	**0.38**	0.52	0.37	0.55	0.40
	Inv_truth	0.55	0.41	0.54	0.40	0.56	0.41	0.62	0.46
	Inv_rank	0.56	0.43	0.55	0.40	0.55 (-9%)	0.40	0.58 (-8%)	0.42
	Inv_cos	0.54 (-11%)	0.40	0.53 (-14%)	0.39	0.56	0.42	0.59	0.44
	Com_truth	0.56	0.42	0.55	0.41	0.57	0.43	0.65	0.51
	Com_rank	0.56 (-8%)	0.42	0.55 (-11%)	0.40	0.56 (-8%)	0.41	0.69	0.48
	Com_cos	0.56 (-8%)	0.43	0.56	0.42	0.56	0.42	0.63 (0%)	0.49

Table 5: Reduction in MRR and Hits@1 due to different attacks on the **target split** of FB15k-237. For each block of rows, we report the *best* relative percentage difference from original MRR; computed as $(original - poisoned)/original * 100$. Lower values indicate better results; best results for each model are in bold. Statistics on the target split are in Table 3.

tectures like ConvE (Dettmers et al., 2018) and translational models like TransE (Bordes et al., 2013). We refer the reader to (Cai et al., 2018) for a comprehensive survey. Due to the black-box nature of KGE models, there is an emerging literature on understanding these models. (Pezeshkpour et al., 2019) and (Zhang et al., 2019a) are most closely related to our work as they propose other data poisoning attacks for KGE models.

Minervini et al. (2017) and Cai and Wang (2018) use adversarial regularization in latent space and adversarial training to improve predictive performance on link prediction. But these adversarial samples are not in the input domain and aim to improve instead of degrade model performance. Poisoning attacks have also been proposed for models for undirected and single relational graph data like Graph Neural Networks (Zügner et al., 2018; Dai et al., 2018) and Network Embedding models (Bojchevski and Günnemann, 2019). A survey of poisoning attacks for graph data is available in (Xu et al., 2020). But the attacks for these models cannot be applied directly to KGE models because they require gradients of a dense adjacency matrix.

In the literature besides adversarial attacks, Lawrence et al. (2020), Nandwani et al. (2020) and Zhang et al. (2019b) generate post-hoc explanations to understand KGE model predictions. Trouillon et al. (2019) study the inductive abilities of KGE models as binary relation properties for controlled inference tasks with synthetic datasets. Allen et al. (2021) interpret the structure of knowl-

edge graph embeddings by comparison with word embeddings. On the theoretical side, Wang et al. (2018) study the expressiveness of various bilinear KGE models and Gutiérrez-Basulto and Schockaert (2018) study the ability of KGE models to learn hard rules expressed as ontological knowledge.

The soft-logical model of inference patterns in this work is inspired by the literature on injecting logical rules into KGE models. Guo et al. (2016) and Guo et al. (2018) enforce soft logical rules by modelling the triples and rules in a unified framework and jointly learning embeddings from them. Additionally, our algebraic model of inference patterns, which is used to select adversarial relations, is related to approaches for graph traversal in latent vector space discussed in Yang et al. (2015); Guu et al. (2015); Arakelyan et al. (2021).

6 Conclusion

We propose data poisoning attacks against KGE models based on inference patterns like symmetry, inversion and composition. Our experiments show that the proposed attacks outperform the state-of-art attacks. Since the attacks rely on relation inference patterns, they can also be used to understand the KGE models. This is because if a KGE model is sensitive to a relation inference pattern, then that pattern should be an effective adversarial attack. We observe that the attacks based on symmetry pattern generalize across all KGE models which indicates their sensitivity to this pattern.

In the future, we aim to investigate hypotheses about the effect of input graph connectivity and existence of specific inference patterns in datasets. We note that such investigation of inference pattern attacks will likely be influenced by the choice of datasets. In this paper, we have used benchmark datasets for link prediction. While there are intuitive assumptions about the inference patterns on these datasets, there is no study that formally measures and characterizes the existence of these patterns. This makes it challenging to verify the claims made about the inductive abilities of KGE models, not only by our proposed attacks but also by new KGE models proposed in the literature.

Thus, a promising step in understanding knowledge graph embeddings is to propose datasets and evaluation tasks that test varying degrees of specific inductive abilities. These will help evaluate new models and serve as a testbed for poisoning attacks. Furthermore, specifications of model performance on datasets with different inference patterns will improve the usability of KGE models in high-stake domains like healthcare and finance.

In addition to understanding model behaviour, the sensitivity of state-of-art KGE models to simple inference patterns indicates that these models can introduce security vulnerabilities in pipelines that use knowledge graph embeddings. Thus, another promising direction for future work is towards mitigating the security vulnerabilities of KGE models. Some preliminary ideas for this research can look into adversarial training; or training an ensemble of different KGE scoring functions; or training an ensemble from subsets of the training dataset. Since our experiments show that state-of-art KGE models are sensitive to symmetry pattern, we call for future research to investigate neural architectures that generalize beyond symmetry even though their predictive performance for link prediction on benchmark datasets might not be the best.

Acknowledgements

Broader Impact

We study the problem of generating data poisoning attacks on KGE models. Data poisoning attacks identify the vulnerabilities in learning algorithms that could be exploited by an adversary to manipulate the model's behaviour (Joseph et al., 2019; Biggio and Roli, 2018). Such manipulation can lead to unintended model behaviour and failure. Identifying these vulnerabilities for KGE models is critical because of their increasing use in domains that need high stakes decision making like heathcare (Bendtsen and Petrovski, 2019) and finance (Hogan et al., 2020; Noy et al., 2019). In this way, our research is directed towards minimizing the negative consequences of deploying state-of-art KGE models in our society. This honours the ACM code of Ethics of contributing to societal well-being and acknowledging that all people are stakeholders in computing. At the same time, we aim to safeguard the KGE models against potential harm from adversaries and thus honour the ACM code of avoiding harm due to computing systems.

Arguably, because we study vulnerabilities by attacking the KGE models, the proposed attacks can be used by an actual adversary to manipulate the model behaviour of deployed systems. This paradox of an arms race is universal across security research (Biggio and Roli, 2018). For our research, we have followed the principle of proactive security as recommended by Joseph et al. (2019) and Biggio and Roli (2018). As opposed to reactive security measures where learning system designers develop countermeasures after the system is attacked, a proactive approach *anticipates* such attacks, simulates them and designs countermeasures before the systems are deployed. Thus, by revealing the vulnerabilities of KGE models, our research provides an opportunity to fix them.

Besides the use case of security, our research can be used in understanding the inductive abilities of KGE models, which are black-box and hard to interpret. We design attacks that rely on the inductive assumptions of a model to be able to deceive that model. Thus, theoretically, the effectiveness of attacks based on one inference pattern over another indicates the model's reliance on one inference pattern over another. However, as we discussed in our paper, realistically, it is challenging to make such claims about the inductive abilities of KGE models because the inference patterns in benchmark datasets are not well defined.

Thus, we would encourage further work to evaluate our proposed attacks by designing benchmark tasks and datasets that measure specific inductive abilities of models. This will not only be useful for evaluating the proposed attacks here, but also for understanding the inductive abilities of existing KGE models. This in turn, can guide the community to design better models. In this direction, we encourage researchers proposing new KGE models to evaluate not only the predictive performance on benchmark datasets, but also the claims made on inductive abilities of these models and their robustness to violations of these implicit assumptions.

References

Carl Allen, Ivana Balazevic, and Timothy Hospedales. 2021. Interpreting knowledge graph relation representation from word embeddings. In *International Conference on Learning Representations*.

Erik Arakelyan, Daniel Daza, Pasquale Minervini, and Michael Cochez. 2021. Complex query answering with neural link predictors. In *International Conference on Learning Representations*.

Claus Bendtsen and Slavé Petrovski. 2019. How data and ai are helping unlock the secrets of disease. In *AstraZeneca Blog*.

Battista Biggio and Fabio Roli. 2018. Wild patterns: Ten years after the rise of adversarial machine learning. *Pattern Recognition*, 84:317–331.

Aleksandar Bojchevski and Stephan Günnemann. 2019. Adversarial attacks on node embeddings via graph poisoning. In *International Conference on Machine Learning*, pages 695–704. PMLR.

Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26*, pages 2787–2795. Curran Associates, Inc.

Hongyun Cai, Vincent W Zheng, and Kevin Chen-Chuan Chang. 2018. A comprehensive survey of graph embedding: Problems, techniques, and applications. *IEEE Transactions on Knowledge and Data Engineering*.

Liwei Cai and William Yang Wang. 2018. KBGAN: Adversarial learning for knowledge graph embeddings. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1470–1480, New Orleans, Louisiana. Association for Computational Linguistics.

Chandrahas, Aditya Sharma, and Partha Talukdar. 2018. Towards understanding the geometry of knowledge graph embeddings. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 122–131, Melbourne, Australia. Association for Computational Linguistics.

Hanjun Dai, Hui Li, Tian Tian, Xin Huang, Lin Wang, Jun Zhu, and Le Song. 2018. Adversarial attack on graph structured data. In *International conference on machine learning*, pages 1115–1124. PMLR.

Tim Dettmers, Pasquale Minervini, Pontus Stenetorp, and Sebastian Riedel. 2018. Convolutional 2d knowledge graph embeddings. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32.

Shu Guo, Quan Wang, Lihong Wang, Bin Wang, and Li Guo. 2016. Jointly embedding knowledge graphs and logical rules. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 192–202, Austin, Texas. Association for Computational Linguistics.

Shu Guo, Quan Wang, Lihong Wang, Bin Wang, and Li Guo. 2018. Knowledge graph embedding with iterative guidance from soft rules. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32.

Víctor Gutiérrez-Basulto and Steven Schockaert. 2018. From knowledge graph embedding to ontology embedding? an analysis of the compatibility between vector space representations and rules. In *Principles of Knowledge Representation and Reasoning: Proceedings of the Sixteenth International Conference, KR 2018, Tempe, Arizona, 30 October - 2 November 2018*, pages 379–388. AAAI Press.

Kelvin Guu, John Miller, and Percy Liang. 2015. Traversing knowledge graphs in vector space. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 318–327, Lisbon, Portugal. Association for Computational Linguistics.

Petr Hájek. 1998. *Metamathematics of Fuzzy Logic*, volume 4. Springer Science & Business Media.

Aidan Hogan, Eva Blomqvist, Michael Cochez, Claudia d'Amato, Gerard de Melo, Claudio Gutiérrez, José Emilio Labra Gayo, Sabrina Kirrane, Sebastian Neumaier, Axel Polleres, Roberto Navigli, Axel-Cyrille Ngonga Ngomo, Sabbir M. Rashid, Anisa Rula, Lukas Schmelzeisen, Juan F. Sequeda, Steffen Staab, and Antoine Zimmermann. 2020. Knowledge graphs. *CoRR*, abs/2003.02320.

Anthony D. Joseph, Blaine Nelson, Benjamin I. P. Rubinstein, and J. D. Tygar. 2019. *Adversarial Machine Learning*. Cambridge University Press.

Carolin Lawrence, T. Sztyler, and Mathias Niepert. 2020. Explaining neural matrix factorization with gradient rollback. *ArXiv*, abs/2010.05516.

Pasquale Minervini, Thomas Demeester, Tim Rocktäschel, and Sebastian Riedel. 2017. Adversarial sets for regularising neural link predictors. In *Proceedings of the Thirty-Third Conference on Uncertainty in Artificial Intelligence, UAI 2017, Sydney, Australia, August 11-15, 2017*. AUAI Press.

Yatin Nandwani, Ankesh Gupta, Aman Agrawal, Mayank Singh Chauhan, Parag Singla, and Mausam. 2020. OxKBC: Outcome explanation for factorization based knowledge base completion. In *Automated Knowledge Base Construction*.

Natasha Noy, Yuqing Gao, Anshu Jain, Anant Narayanan, Alan Patterson, and Jamie Taylor. 2019. Industry-scale knowledge graphs: Lessons and challenges. *Commun. ACM*, 62(8):36–43.

Pouya Pezeshkpour, Yifan Tian, and Sameer Singh. 2019. Investigating robustness and interpretability of link prediction via adversarial modifications. In *Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*.

Daniel Ruffinelli, Samuel Broscheit, and Rainer Gemulla. 2020. You can teach an old dog new tricks! on training knowledge graph embeddings. In *International Conference on Learning Representations*.

Kristina Toutanova and Danqi Chen. 2015. Observed versus latent features for knowledge base and text inference. In *Proceedings of the 3rd Workshop on Continuous Vector Space Models and their Compositionality*, pages 57–66, Beijing, China. Association for Computational Linguistics.

Théo Trouillon, Éric Gaussier, Christopher R. Dance, and Guillaume Bouchard. 2019. On inductive abilities of latent factor models for relational learning. *J. Artif. Int. Res.*, 64(1):21–53.

Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. 2016. Complex embeddings for simple link prediction. In *International Conference on Machine Learning*, pages 2071–2080.

Yanjie Wang, Rainer Gemulla, and Hui Li. 2018. On multi-relational link prediction with bilinear models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32.

Han Xu, Yao Ma, Hao-Chen Liu, Debayan Deb, Hui Liu, Ji-Liang Tang, and Anil K Jain. 2020. Adversarial attacks and defenses in images, graphs and text: A review. *International Journal of Automation and Computing*, 17(2):151–178.

Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. 2015. Embedding entities and relations for learning and inference in knowledge bases. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Hengtong Zhang, Tianhang Zheng, Jing Gao, Chenglin Miao, Lu Su, Yaliang Li, and Kui Ren. 2019a. Data poisoning attack against knowledge graph embedding. In *International Joint Conference on Artificial Intelligence*.

Wen Zhang, Bibek Paudel, Wei Zhang, Abraham Bernstein, and Huajun Chen. 2019b. Interaction embeddings for prediction and explanation in knowledge graphs. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*, pages 96–104.

Daniel Zügner, Amir Akbarnejad, and Stephan Günnemann. 2018. Adversarial attacks on neural networks for graph data. In *International Conference on Knowledge Discovery & Data Mining*, pages 2847–2856.

Appendix

A   Computational Complexity Analysis

Lets say $\mathcal{E}$ is the set of entities and $\mathcal{R}$ is the set of relations. The number of target triples to attack is t and the specific target triple is $(s, \mathbf{r}, o)$. Here, we discuss the computational complexity of the three steps of the proposed attacks -

Determine Adversarial Relations: In this step, we determine the inverse relation or the composition relation of a target triple. To select inverse relation, we need $\mathcal{R}$ computations for every target triple. Selecting composition relation requires the composition operation $\mathcal{R}^2$ times per target triple. To avoid repetition, we pre-compute the inverse and composition relations for all target triples. This gives the complexity $\mathcal{O}(\mathcal{R}^2)$ for inverse relation. For composition relation, we compute compositions of all relation pairs and then select the adversarial pair by comparison with target relation. This gives $\mathcal{O}(\mathcal{R}^2 + \mathcal{R})$ complexity for composition.

Determine Decoy Entity: The three heuristics to compute the decoy entity are soft-truth score, KGE ranks and cosine distance. For symmetry and inversion, the soft truth score requires 2 forward calls to the model for one decoy entity. For composition, if the number of clusters is k, the soft truth score requires $3k$ forward calls to the model. To select decoy entities based on KGE ranks, we require one forward call for each decoy entity. For cosine distance, we compute the similarity of s and o to all entities via two calls to Pytorch's `F.cosine_similarity`. Once the heuristic scores are computed, there is an additional complexity of $\mathcal{O}(\mathcal{E})$ to select the entity with minimum

score. Thus, the complexity for decoy selection is $\mathcal{O}(t\mathcal{E})$ for all heuristics except soft truth score on composition where it is $\mathcal{O}(kt\mathcal{E})$.

Determine Adversarial Entity: This step requires three forward calls to the KGE model because the ground truth score needs to be computed. Thus, the complexity for this step is $\mathcal{O}(t\mathcal{E})$.

Based on the discussion above, the overall computational complexity is $\mathcal{O}(t\mathcal{E})$ for symmetry attacks and $\mathcal{O}(\mathcal{R}^2 + t\mathcal{E})$ for inversion attacks. For composition attacks, it is $\mathcal{O}(\mathcal{R}^2 + \mathcal{R} + kt\mathcal{E})$ for soft truth score and $\mathcal{O}(\mathcal{R}^2 + \mathcal{R} + t\mathcal{E})$ for KGE ranks and cosine distance.

B Implementation Details

B.1 Training KGE models

Our codebase[6] for KGE model training is based on the codebase from (Dettmers et al., 2018)[7]. We use the 1-K training protocol but without reciprocal relations. Each training step alternates through batches of (s,r) and (o,r) pairs and their labels. The model implementation uses an if-statement for the forward pass conditioned on the input batch mode.

For TransE scoring function, we use L2 norm and a margin value of 9.0. The loss function used for all models is Pytorch's BCELosswithLogits. For regularization, we use label smoothing and L2 regularization for TransE; and input dropout with label smoothing for remaining models. We also use hidden dropout and feature dropout for ConvE.

We do not use early stopping to ensure same hyperparameters for original and poisoned KGE models. We used an embedding size of 200 for all models on both datasets. For ComplEx, this becomes an embedding size of 400 because of the real and imaginary parts of the embeddings. All hyperparameters are tuned manually based on suggestions from state-of-art implementations of KGE models (Ruffinelli et al., 2020; Dettmers et al., 2018). The hyperparameter values for all model dataset combinations are available in the codebase. Table 6 shows the MRR and Hits@1 for the original KGE models on WN18RR and FB15k-237.

For re-training the model on poisoned dataset, we use the same hyperparameters as the original model. We run all model training, adversarial attacks and evaluation on a shared HPC cluster with Nvidia RTX 2080ti, Tesla K40 and V100 GPUs.

---

[6] https://github.com/PeruBhardwaj/InferenceAttack

[7] https://github.com/TimDettmers/ConvE

	WN18RR		FB15k-237	
	MRR	Hits@1	MRR	Hits@1
DistMult	0.42	0.39	0.27	0.19
ComplEx	0.43	0.40	0.24	0.20
ConvE	0.43	0.40	0.32	0.23
TransE	0.19	0.02	0.34	0.25

Table 6: MRR and Hits@1 results for original KGE models on WN18RR and FB15k-237

	WN18RR		
	Original	High	Low
DistMult	0.90	0.82	0.83
ComplEx	0.89	0.76	0.79
ConvE	0.92	0.90	0.90
TransE	0.36	0.25	0.24
	FB15k-237		
	Original	High	Low
DistMult	0.61	0.55	0.53
ComplEx	0.61	0.51	0.52
ConvE	0.61	0.54	0.54
TransE	0.63	0.57	0.57

Table 7: MRR of KGE models trained on original datasets and poisoned datasets from the attack in Zhang et al. (2019a). High, Low indicate the high and low percentage of candidates used for attack.

B.2 Baseline Implementation Details

One of the baselines in our evaluation is the attack from (Zhang et al., 2019a). It proposed edits in the neighbourhood of subject of the target triple. We extend it for both subject and object to match our evaluation protocol. Since no public implementation is available, we implement our own.

The attack is based on computing a perturbation score for all possible candidate additions. Since the search space for candidate additions is of the order $\mathcal{E} \times \mathcal{R}$, the attack uses random down sampling to filter out the candidates. The percent of triples down sampled are not reported in the original paper and the implementation is not available. So, in this paper, we pick a high and a low value of the percentage of triples down sampled and generate adversarial edits for both fractions. The high and low percent values that were used to select candidate adversarial additions for WN18RR are DistMult: (20.0, 5.0); ComplEx: (20.0, 5.0); ConvE: (2.0, 0.1); TransE: (20.0, 5.0). For FB15k-237, these values are DistMult: (20.0, 5.0); ComplEx: (15.0, 5.0); ConvE: (0.3, 0.1); TransE: (20.0, 5.0)

Thus, we generate two poisoned datasets from the attack - one that used a high number of candi-
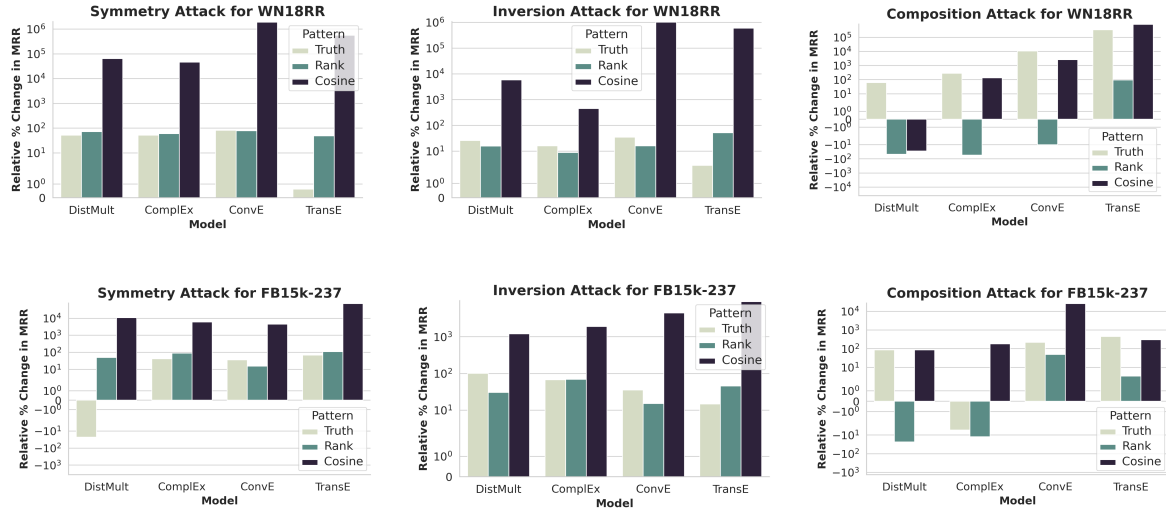
Figure 2: Mean of the relative increase in MRR of object and subject side **decoy triples** due to proposed attacks on WN18RR and FB15k-237. The increase is computed relative to original MRR of decoy triples as $(poisoned - original)/original$. The scale on y-axis is symmetric log scale. Higher values are better; as they show the effectiveness of attack in improving decoy triples' ranks relative to their original ranks.

dates and another that used a low number of candidates. We train two separate KGE models on these datasets to assess attack performance. Table 7 shows the MRR of the original model; and poisoned KGE models from attack with high and low downsampling percents. The results reported for this attack's performance in Section 4.1 are the better of the two results (which show more degradation in performance) for each combination.

B.3 Attack Implementation Details

Our proposed attacks involve three steps to generate the adversarial additions for all target triples. For step1 of selection of adversarial relations, we pre-compute the inversion and composition relations for all target triples. Step2 and Step3 are computed for each target triple in a *for* loop. These steps involve forward calls to KGE models to score adversarial candidates. For this, we use a vectorized implementation similar to KGE evaluation protocol. We also filter out the adversarial candidates that already exist in the training set. We further filter out any duplicates from the set of adversarial triples generated for all target triples.

For the composition attacks with soft-truth score, we use the KMeans clustering implementation from $scikit - learn$. We use the elbow method on the grid [5, 20, 50, 100, 150, 200, 250, 300, 350, 400, 450, 500] to select the number of clusters. The number of clusters selected for WN18RR are Dist-Mult: 300, ComplEx: 100, ConvE: 300, TransE: 50. For FB15k-237, the numbers are DistMult:

200, ComplEx: 300, ConvE: 300, TransE: 100.

C Analysis on Decoy Triples

The proposed attacks are designed to generate adversarial triples that improve the KGE model performance on decoy triples $(s, \mathbf{r}, o')$ and $(s', \mathbf{r}, o)$. In this section, we analyze whether the performance of KGE models improves or degrades over decoy triples after poisoning. For the decoy triples on object side $(s, \mathbf{r}, o')$, we compute the change in object side MRR relative to the original object side MRR of these triples. Similarly, for the decoy triples on subject side $(s', \mathbf{r}, o)$, we compute the change in subject side MRR relative to the original subject side MRR of these decoy triples. Figure 2 shows plots for the mean change in MRR of object and subject side decoy triples.

We observed in Section 4.1 that the composition attacks against TransE on WN18RR improved the performance on target triples instead of degrading it. In Figure 2, we notice that composition attacks against TransE are effective in improving the ranks of decoy triples on both WN18RR and FB15k-237. This evidence supports the argument made in the main paper - it is likely that the composition attack does not work against TransE for WN18RR because the original dataset does not contain any composition relations; thus adding this pattern improves model's performance on *all* triples instead of just the target triples because of the sensitivity of TransE to composition pattern.

		DistMult		ComplEx		ConvE		TransE	
		MRR	Hits@1	MRR	Hits@1	MRR	Hits@1	MRR	Hits@1
Original		0.82	0.67	0.99	0.99	0.80	0.63	0.65	0.45
Baseline Attacks	Random_n	0.80 (-2%)	0.63	0.99 (0%)	0.98	0.79 (-2%)	0.61	0.46 (-29%)	0.18
	Random_g1	0.82	0.66	0.99	0.98	0.80	0.62	0.57	0.33
	Random_g2	0.81	0.65	0.99	0.98	0.79	0.62	0.50	0.22
	Zhang et al.	0.77 (-6%)	0.59	0.97 (-3%)	0.95	0.77 (-3%)	0.61	0.43 (-33%)	0.16
	CRIAGE	0.78	0.61	-	-	0.78	0.63	-	-
Proposed Attacks	Sym_truth	0.62	0.30	0.90	0.82	**0.58 (-17%)**	**0.27**	0.74	0.60
	Sym_rank	0.59	0.27	0.89 (-10%)	0.79	0.62	0.33	0.52	0.34
	Sym_cos	**0.50 (-38%)**	**0.17**	0.92	0.85	0.60	0.35	**0.41 (-37%)**	**0.13**
	Inv_truth	0.81	0.66	0.86	0.74	0.78 (-3%)	0.61	0.59	0.34
	Inv_rank	0.82	0.66	**0.84 (-16%)**	**0.68**	0.79	0.61	0.55	0.34
	Inv_cos	0.79 (-3%)	0.64	0.87	0.75	0.80	0.63	0.51 (-22%)	0.25
	Com_truth	0.79	0.62	0.98	0.97	0.77	0.62	0.53 (-18%)	0.25
	Com_rank	0.80	0.64	0.98	0.96	0.75 (-6%)	0.58	0.67	0.47
	Com_cos	0.78 (-5%)	0.61	0.97 (-2%)	0.95	0.77	0.62	0.58	0.32

Table 8: Reduction in MRR and Hits@1 due to different attacks on the **target split** of WN18. For each block of rows, we report the *best* relative percentage difference from original MRR; computed as $(original - poisoned)/original * 100$. Lower values indicate better results; best results for each model are in bold.

D Analysis on WN18

The inversion attacks identify the relation that the KGE model might have learned as inverse of the target triple's relation. But the benchmark datasets WN18RR and FB15k-237 do not contain inverse relations, and a KGE model trained on these clean datasets would not be vulnerable to inversion attacks. Thus, we perform additional evaluation on the WN18 dataset where triples with inverse relations have not been removed. Table 8 shows the results for different adversarial attacks on WN18.

We see that the symmetry based attack is most effective for DistMult, ConvE and TransE. This indicates the sensitivity of these models to the symmetry pattern even when inverse relations are present in the dataset. For DistMult and ConvE, this is likely due to the symmetric nature of their scoring functions; and for TransE, this is likely because of the translation operation as discussed in Section 4.1. On the ComplEx model, we see that though the symmetry attacks are more effective than random baselines, the inversion attacks are the most effective. This indicates that the ComplEx model is most sensitive to the inversion pattern when the input dataset contains inverse relations.

E Analysis of Runtime Efficiency

In this section, we compare the runtime efficiency of the baseline and proposed attacks. Table 9 shows the time taken (in seconds) to select the adversarial triples using different attack strategies for all

	DistMult	ComplEx	ConvE	TransE
Random_n	10.08	10.69	8.76	7.83
Random_g1	**8.28**	**8.16**	**7.64**	**6.49**
Random_g2	16.01	15.82	18.72	13.33
Zhang et al.	94.48	**255.53**	666.85	**81.96**
CRIAGE	**21.77**	-	21.96	-
Sym_truth	**19.63**	35.40	**22.76**	31.59
Sym_rank	23.47	**27.25**	25.82	25.03
Sym_cos	22.52	28.62	25.69	**23.13**
Inv_truth	**11.43**	**15.69**	24.13	31.89
Inv_rank	15.27	18.14	30.99	21.82
Inv_cos	14.96	20.47	**23.02**	**20.63**
Com_truth	2749.60	1574.44	6069.79	470.34
Com_rank	**22.04**	**31.53**	37.81	20.88
Com_cos	34.78	68.06	**32.37**	**19.86**

Table 9: Time taken in seconds to generate adversarial triples using baseline and proposed attacks on WN18

models on WN18 dataset. Similar patterns were observed for attack execution on other datasets.

For CRIAGE, the reported time does not include the time taken to train the auto-encoder model. Similarly, for soft-truth based composition attacks, the reported time does not include the time taken to pre-compute the clusters. We observe that the proposed attacks are more efficient than the baseline Zhang et al. attack which requires a combinatorial search over the canidate adversarial triples; and have comparable efficiency to CRIAGE. Among the different proposed attacks, composition attacks based on soft-truth score take more time than others because they select the decoy entity by computing the soft-truth score for multiple clusters.