An Enhanced Knowledge Graph Embedding for Small-scale Sparse Knowledge Graph

Yushun Xie[1], Haiyan Wang[4(✉)], Runnan Tan[3], Xiangyu Song[4], and Zhaoquan Gu[2,4(✉)]

[1] Shenzhen Institute for Advanced Study, University of Electronic Science and Technology of China, Shenzhen, China
yshxie@std.uestc.edu.cn
[2] School of Computer Science and Technology, Harbin Institute of Technology, Shenzhen, China
guzhaoquan@hit.edu.cn
[3] Cyberspace Institution of Advanced Technology, Guangzhou University, Guangzhou, China
1112106007@e.gzhu.edu.cn
[4] Department of New Networks, Peng Cheng Laboratory, Shenzhen, China
{wanghy01,songxy02}@pcl.ac.cn

Abstract. Knowledge graph embedding (KGE) learns to map entities and relationships to a continuously dense, high-dimensional representation in a vector space, capturing and utilizing semantic relationships between entities through numerical operations. As the key for realizing intelligent inference for deep learning, it has garnered widespread attention in recent years. However, existing KGE techniques generally perform poorly on small-scale datasets, because small-scale knowledge graphs are easy to cause model overfitting due to their incompleteness and sparsity. To address these issues, in this paper, we propose an enhanced knowledge graph embedding for small-scale knowledge graphs that can extremely efficiently achieve mutual improvement of rule-based data augmentation and neighborhood-based embedding enhancement. Additionally, to make the method applicable to the conducted Cybersecurity Attack Knowledge Graph, we abstract temporal attributes into knowledge, effectively preserving temporal dependencies of the attack steps. Furthermore, we correct the KGE ranking by leveraging spatial rule scores. Extensive experiments show that our method outperforms existing KGE techniques with 15% improvement on MRR and 6% improvement on Hits@10.

Keywords: Knowledge graph embedding · Small-scale knowledge graph · Cybersecurity attack.

1 Introduction

Researchers increasingly emphasize knowledge graph (KG) as an efficient graph data model representing associative information between data. Knowledge graph embedding (KGE) maps the facts of KG into a continuous dense high-dimensional

space [4], reflecting the semantic information between entities and relations, this technique enables deep learning to understand the data of the knowledge graph. So the KGE models are regarded as the key for realizing automatic inference. KGE has been widely used in KG-related tasks, such as recommendation system [8], semantic search [29], question answering system [12], etc.

Although KGE has achieved excellent achievements in many fields, it is not mature enough for practical applications. There is a huge gap because the data used in the research are generally similar to large-scale KGs such as FB15K237 [25], WN18RR [27]. However, the knowledge graphs in actual tasks are small-scale datasets in specialized domains, characterized by small size, strong specialization, and high sparsity. A large number of researches have also shown that existing KGE models cannot be applied to small-scale KGs [21].

Data on cybersecurity attacks has rich correlations, so it is particularly suitable for KGE. However, because of the specialization and complexity of the cybersecurity domain, we only extract a small number of facts from the specific corpus, so the Cybersecurity Attack Knowledge Graph has poor completeness and connectivity sparsity, it is a small-scale knowledge graph. Although the application value of the Cybersecurity Attack Knowledge Graph is high, it is challenging to learn, the traditional KGE techniques tend to perform poorly on this KG. Through careful analysis of the cybersecurity task, it was discovered that we can utilize the temporal and spatial attributes of the attack process. Specifically, the temporal attributes indicate the sequence of the attack behaviors during the attack incident, and the spatial attributes indicate the IP information involved in the cybersecurity attack.

Although temporal knowledge graph embedding (TKGE) [11] fully utilizes the temporal attributes in the KGs, it is unable to model complex causal relationships and temporal dependency. However, there are strong temporal logical dependencies between the attack steps of a cybersecurity attack incident, so the existing TKGE techniques are not suitable to the cybersecurity field. Meanwhile, existing KGE models do not utilize spatial attributes, namely, IP information.

To address the above problems, we collect data from multiple advanced persistent threat (APT) [1] complex attack incidents to construct a Cybersecurity Attack Knowledge Graph, and we propose an enhanced knowledge graph embedding for small-scale KG. Our method abstracts temporal attributes and extracts spatial rules, iteratively using rule-based data augmentation and neighborhood-based embedding enhancement techniques, as shown in Fig. 1. Our approach enriches the representation space of the KGE model, the contributions are summarized as follows:

1) We propose an enhanced knowledge graph embedding model that applies to small-scale KGs like Cybersecurity Attack Knowledge Graph.
2) Our method enriches the representation space of the KGE model. First, we use the rule mining system to mine Horn rules for inferring potential facts, then, we use the neighborhood information to enrich the embedding vectors of entities for predicting massive examples, and finally, we use a

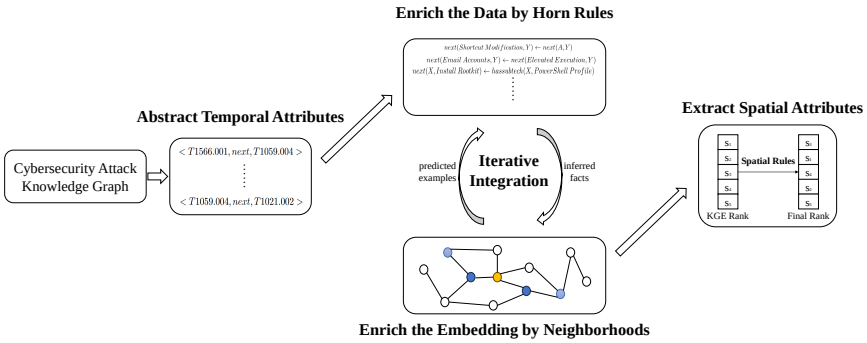

Fig. 1: An Enhanced Knowledge Graph Embedding for Small-scale Sparse Knowledge Graph.

simple yet effective iterative mechanism to let the above two techniques efficiently enhance each other.

3) Our method abstracts temporal attributes, we abstract the attack knowledge in the Cybersecurity Attack Knowledge Graph, preserving the temporal dependencies between attack behaviors and making our method suitable to this KG.
4) Our method extracts spatial attributes, we use AnyBURL to mine rules related to IP information, meanwhile, we aggregate spatial rule scores and predicted scores during the model testing, correcting the KGE ranking.
5) We demonstrate that our method applies to small-scale knowledge graphs. More specifically, we conduct experiments on three small-scale datasets, and the experimental results show that our method gains significant improvement over existing KGE techniques, with 15% improvement on MRR and 6% improvement on Hits@10.

The rest of this paper is organized as follows. Section 2 introduces related work. Section 3 describes the preliminary, which includes the Cybersecurity Attack Knowledge Graph, the KGE models and Horn rules. In Section 4, we introduce our method and proposed dataset. In Section 5, we carry out the empirical study of our model and we conclude our work in Section 6.

2 Related work

Knowledge graph embedding methods are embedding facts into a high dimensional space, depending on the scoring function, they can be roughly classified into distance-based models, semantic-based models and neural network-based models. Distance-based models measure the plausibility of facts by calculating the distance between entities, e.g. TransE [4], TransR [15], HousE [14], etc.

Semantic-based models judge factual credibility by measuring semantic similarity, such as DistMult [30], QuatRE [18], CompilE [16], etc. Neural network-based models output factual reliability through nonlinear activation functions and complex network structures, e.g., ConvE [5], AdaProp [35], etc.

Current research in KGE mainly focuses on static knowledge graphs, i.e., facts do not change over time, but knowledge graphs in practical applications are usually dynamic, so there are also some dynamic knowledge graph embedding methods. Jiang et al. [11] encoded temporal information into the representation space of knowledge graphs for the first time, and directly extended the triples into the temporal quadruple. Because of the excellent performance of TKGE, more and more TKGE methods have been proposed, such as TA-TransE [9], EvoKG [19], HyIE [33], etc.

However, many studies have demonstrated that KGE with excellent performance on generic large-scale datasets often performs poorly on small-scale sparse KGs [21]. To solve this problem, researchers have proposed many methods. On the one hand, rule mining systems like AMIE [7,6] and AnyBURL [17] are proposed for mining more training examples, and other researchers combined rule mining and graph embedding with methods such as KALE [10], IterE [34], SimRE [32], etc. On the other hand, RelaGraph [23] modified StarGraph [13] to enrich the embedding of entities using neighborhood information to solve the problem of lacking examples.

Cybersecurity Attack Knowledge Graph is a small-scale knowledge graph. Our method fully utilizes temporal and spatial information, abstracts temporal attributes into attack sequences, and extracts spatial attributes into judging rules, which significantly improves the performance of the KGE model. The difference between our method with other typical KGE methods is shown in Table 1.

Table 1: Comparison the our method with other typical KGE methods.

	Small-scale	Temporal information	Spatial information
TransE	✘	✘	✘
TA-TransE	✘	✔	✘
RelaGraph	✔	✘	✘
Our model	✔	✔	✔

3 Preliminary

The objective of our paper is to propose a reinforced KGE model suitable for small-scale sparse KGs, such as the Cybersecurity Attack Knowledge Graph.

Notations. The most used symbols and their descriptions are given in Table 2. Scalars are denoted normally, and vectors and matrices are denoted by boldface.

Table 2: Symbols and notations.

Symbol	Description
$\mathcal{E}$, $\mathcal{R}$	the set of entities and set of relations
$s, o \in \mathcal{E}$, $r \in \mathcal{R}$	head and tail entity, relation
$\mathbf{s}, \mathbf{o}, \mathbf{r} \in \mathbb{R}^d$	embedding of head entity, tail entity and relation
$G = \{(s, r, o)\}$	the set of examples in knowledge graph
$\{(\bar{s_i}, r_i, \bar{o_i})\}$	the negative example of (s_i, r_i, o_i)
$time, space$	temporal and spatial attributes
$(s, r, o, time, space)$	the quintuple of Cybersecurity Attack Knowledge Graph
$f(s, r, o)$	the scoring function of (s, r, o)
$\mathcal{H} = \{H \leftarrow B_i \mid i \in I\}$	the set of Horn rules
$\mathcal{S} = \{(S_i, c_i) \mid i \in I\}$	the set of spatial rules
$\mathcal{N} = \{(s_i, l_i) \mid i \in I\}$	the neighborhood entity and path length of target entity
G_1, G_2	the set of infered examples inferring by rules and neighborhoods

3.1 Cybersecurity Attack Knowledge Graph

A knowledge graph consists of a large number of facts, each fact can be denoted as (s, r, o), which represents two entities $s, o \in \mathcal{E}$ linked together by a specific relation $r \in \mathcal{R}$. So a KG can be expressed as $G = \{(s, r, o) \mid s, o \in \mathcal{E}$ and $r \in \mathcal{R}\}$. Cybersecurity Attack Knowledge Graph has obvious temporal and spatial attributes, each fact is extended from a triple (s, r, o) to a quintuple $(s, r, o, time, space)$, where *time* denotes temporal information and *space* denotes as spatial information.

3.2 Knowledge Graph Embedding technique

As there are only positive examples in the KGs, negative examples are generated for training by negative samplings. Several negative examples $(\bar{s_i}, r_i, o_i)$ or $(s_i, r_i, \bar{o_i})$ are obtained by randomly replacing s_i or o_i ($\bar{s_i}, \bar{o_i} \in \mathcal{E}$ and $\bar{s_i} \neq s_i$, $\bar{o_i} \neq o_i$) for each triple $(s_i, r_i, o_i) \in G$.

KGE maps s, r, o into a continuous and dense space through an embedding layer (a kind of neural network). The scoring function $f(s, r, o)$ is the core of KGE, which evaluates the plausibility of each positive and negative example in a certain way. By minimizing the loss function, the embedding vectors $\mathbf{s}, \mathbf{r}, \mathbf{o}$ in the high-dimensional space are continuously optimized, thus accomplishing the task of KGE.

3.3 Horn rules

Plenty of rules exist in the KGs, which can be mined from the data and help to complete the KG, so rules are especially suitable for small-scale KGs. Horn rules can be extracted automatically and efficiently via rule mining systems, such as AMIE and AnyBURL. They use heuristic search to pick high-quality rules, which are widely used in practice. Horn rules are formulas of the form:

$$H \leftarrow B_1 \wedge \cdots \wedge B_n \tag{1}$$

where B_i is the body of the rule and H is the head of the rule, the body of a rule is a conjunction of atoms, separated by the symbol $\wedge$ or by a comma.

For example, consider the following rule:

$$\text{Organization}(X, Y) \leftarrow \text{Attack}(X, A) \wedge \text{Attack}(Y, A) \tag{2}$$

if we have the knowledge that $Attack(harker_1, asset)$ and $Attack(harker_2, asset)$, we can intuitively deduce that $Organization(harker_1, harker_2)$.

4 Data and methodology

4.1 Data preprocessing and temporal attributes abstraction

The data used in our paper comes from the 38 APT incidents published in the Attack Flow project [1]. Attack Flow provides a common language and toolset for describing complex adversarial behaviors. For every APT, Attack Flow demonstrates a sequence of behaviors employed by the adversary to achieve the goal. These behaviors can be corresponded to specific tactics, techniques, or dub-techniques of ATT&CK.

Because each APT is a complex multi-step attack with an obvious temporal attribute, i.e., there is a sequential order between the attack steps. We abstract the temporal attributes as the relation "*next*", linking two ATT&CK behaviors. As an example, there are three important steps in the OceanLotus incident. First, the hackers sent phishing emails to the target on October 8, 2019 exploiting vulnerability CVE-2011-3415; then the hackers deployed remotely assess trojan to the target on October 10, 2019 exploiting vulnerability CVE-2002-0840, after that the hackers achieved lateral movement on November 1, 2020 exploiting vulnerability CVE-2018-16509. We can extract the following knowledge:

$$\begin{aligned} &< T1566.001, exploite, CVE\text{-}2011\text{-}3415, 2019.10.08 > \\ &< T1059.004, exploite, CVE\text{-}2002\text{-}0840, 2019.10.10 > \\ &< T1021.002, exploite, CVE\text{-}2018\text{-}16509, 2020.11.01 > \end{aligned} \tag{3}$$

where $T1566.001$, $T1059.004$, $T1021.002$ are the ids of the techniques in ATT&CK, $T1566.001$ represents sending phishing emails, $T1059.004$ represents deploying remote assess trojan, $T1021.002$ represents achieving lateral movement.

After abstracting the temporal attributes, we obtain the following two attack knowledge:

$$\begin{aligned} &< T1566.001, next, T1059.004 > \\ &< T1059.004, next, T1021.002 > \end{aligned} \tag{4}$$

We use the ATT&CK_id as the entity of the attack behavior and extract 625 entities. Because the number of entities is not enough to build a knowledge graph.

We query the cybersecurity knowledge base, containing entities like CAPEC_id, CWE_id and CVE_id, for entities connected to ATT&CK_id and preserve the internal relations between them.

Finally, the Cybersecurity Attack Knowledge Graph we constructed contains 1285 entities and 5 relations, consisting of 3311 examples. Cybersecurity Attack Knowledge Graph is a small-scale KG in the field of cybersecurity, which has the characteristics of specialization, small size, and sparsity compared with the commonly used large-scale KGs. Therefore, applying the existing KGE and TKGE methods directly to the Cybersecurity Attack Knowledge Graph will lead to underlearning and overfitting.

4.2 Enrich the representation of Cybersecurity Attack KG

Since small-scale KGs have few examples, we use rule-based data augmentation and neighborhood-based embedding enhancement techniques to enrich representation and avoid overfitting.

Enrich the data by AnyBURL. Rule mining systems learn explicit patterns, also known as Horn rules, from raw data. Based on Horn rules, we generate many examples for the small-scale KGs. In this paper, we use AnyBURL to mine the rule set $\mathcal{H} = \{H \leftarrow B_i \mid i \in I\}$, which has been shown to achieve very good KGC results [22]. The following rules are shown that can be learned from the Cybersecurity Attack KG.

$$\begin{aligned} next(Shortcut\ Modification, Y) &\leftarrow next(A, Y) \quad 0.047[2/42] \\ next(Email\ Accounts, Y) &\leftarrow next(Elevated\ Execution, Y) \quad 0.67[2/3] \\ next(X, Install\ Rootkit) &\leftarrow hassubtech(X, PowerShell\ Profile) \quad 1[3/3] \end{aligned} \tag{5}$$

where the part behind the arrow is the body of the rule, the part before the arrow is the conclusion of the rule, and the confidence score of the rule is given, indicating the possibility that the rule is true.

Given the query $r(s, ?)$, we apply the learned rule set $\mathcal{H}$ in the knowledge graph G to predict new examples that complete the query. Specifically, we collect all rules in $\mathcal{H}$ that relate to r. After that, for each rule, we replace the variable with an entity, e.g., we set $X = s$, and traverse all entities to evaluate the plausibility when $Y = o^{'}$ based on the confidence of the rule. When the confidences exceed the threshold θ_1, the newly set of inferred examples $G_1 = \{(s, r, o^{'}), (s, r, o^{''}), ..., (s, r, o^{'''})\}$ is added to KG by updating $G = G \cup G_1$.

Enrich the embedding by neighborhood. AnyBURL merely increases the number of positive examples, to further improve the performance of the KGE model, we use the neighborhood information in the KG. Traditional KGE models transform entities into feature space through the entity embedding layer, the model can only implicitly learn weak associations in KG from multiple links

of the same entity. We enrich the representation capability of the model by aggregating the neighborhood information of entities through the self-attention mechanism.

For each target entity s, we use the breadth-first search (BFS) to perform a k-hop search for the neighborhood entities. Because entities with lower degree can not provide value, to speed up the computation, we filter the entities with degree more than the threshold δ into the search space. $\mathcal{N}$ means the results and records the path length l_i between target entity s and neighborhood entity s_i. The formula is as follows:

$$\mathcal{N} = \{(s_i, l_i) \mid i \in I\} \tag{6}$$

After the sampling above, we use the updated KG to train the KGE model obtaining the embedding vectors of entities and relations. Then we take the path length l_i as positional encoding and input the embedding vectors $\mathbf{s}$, $\mathbf{s_i}$ into the self-attention network. The result of updating $\mathbf{s}$ is calculated with the following formula:

$$\boldsymbol{s} = \frac{1}{n} \sum \mathit{self\text{-}attention}\Big(\boldsymbol{s}, \{(\boldsymbol{s_1}, l_1), (\boldsymbol{s_2}, l_2), \cdots, (\boldsymbol{s_n}, l_n)\}\Big) \tag{7}$$

We predict potential examples based on the updated KGE model, and when the predicted score of the example exceeds the threshold θ_2, the example is added to the set $G_2 = \{(s, r, o^{'}), (s^{'}, r, o), ..., (s^{'''}, r, o), (s, r, o^{'''})\}$ and KG has updated again $G = G \cup G_2$.

Iterative Integration. Since rule-based data augmentation and neighborhood-based embedding enhancement can enrich the representational capability of the KGE model and they can mutually enhance each other, we integrate the two techniques iteratively with the following Algorithm 1:

Algorithm 1 Integrate the rule-based data augmentation and neighborhood-based embedding enhancement

Input: the facts in knowledge graph G, the scoring function f
Output: the embedding vectors of entities and relations

```
for epoch = 1, ..., Max-iter do
    Enrich the data by AnyBURL based on G
    Obtain the inferred set G_1 and update G = G ∪ G_1
    Train the KGE model based on updated data G
    Enrich the embedding by neighborhood information
    Obtain the predicted set G_2 and update G = G ∪ G_2
end for
```

Max-iter represents the number of iteration of the Algorithm 1 and it is a hyperparameter. In each iteration, first, we use AnyBURL to infer the set

of positive examples G_1 and add them to the knowledge graph $G = G \cup G_1$; second we learn KGE model based on the updated data; finally, we update the embedding representations of the entities with the neighborhood information, and we predict the set of new examples G_2 by using the updated embedding space, the knowledge graph again with $G = G \cup G_2$. In the next iteration, AnyBURL mines more rules based on the updated KG.

4.3 Overfitting risk reduction

Cybersecurity Attack Knowledge Graph is a small-scale KG, which is especially susceptible to overfitting. To reduce the risk of overfitting, we use the hard negative sampling for generating negative examples, i.e., Negative Example Sampling by mIxup (NESI) [28], and choose TripleRE [31] as the scoring function.

Hard negative sampling filters negative examples beneficial to the model as input data, which can greatly improve the generalization and accuracy of the KGE models. NESI generates hard negative examples by mixing positive and negative examples so that the hard negative examples are infinitely close to the positive example. In this way, the model is more difficult to train, thus reducing the risk of overfitting. The core formula is as follows:

$$\bar{G}_i = \alpha_i * (s_i, r_i, o_i) + (1 - \alpha_i) * (\bar{s_i}, r_i, o_i); \bar{G}_i = \frac{\bar{G}_i}{\|\bar{G}_i\|} \tag{8}$$

where (s_i, r_i, o_i) and $(\bar{s_i}, r_i, o_i)$ respectively represent the positive and negative example, $\bar{G}_i$ represents the hard negative example, $\alpha_i \in (0, 1)$ is a hyperparameter.

In small-scale KGs, the relations appear more frequently compared with entities. TripleRE focuses on the expressiveness of relations, it extracts more features from relation vectors to improve the performance of the model. At the same time, TripleRE decreases the dimension of entity vectors, thus addressing an overfitting problem in small-scale KGs.

Specifically, TripleRE sets the dimensions of the relation vector to three times that of the entity. When evaluating the plausibility of examples, the relation vector $\mathbf{r}$ is split into three parts $\mathbf{r_1}$, $\mathbf{r_2}$ and $\mathbf{r_3}$, which operate with entity vectors $\mathbf{s}$ and $\mathbf{o}$, respectively. The formula is as follows:

$$f(s, r, o) = - \parallel \boldsymbol{s} \circ (\beta \times \boldsymbol{r}_1 + 1) - \boldsymbol{o} \circ (\beta \times \boldsymbol{r}_3 + 1) + \boldsymbol{r}_2 \parallel \tag{9}$$

where $\beta \in (0, 1)$ is a hyperparameter.

4.4 Spatial attributes extraction

During the experiments, we found that the next attack behavior predicted by the KGE model is significantly inconsistent with the facts. For example, in the OceanLotus incident, the KGE model predicts the next step after "Lateral Movement" is "Phishing". Although this is feasible, the next behavior of "Lateral Movement" is more likely to be "Remote File Copy", "Data Destruction" and "Indicator Removal on Host ".

The mentioned example indicates that the Cybersecurity Attack KGE model can be improved, so we utilize the spatial attributes, i.e., IP information involved in the attack behavior. Each attack behavior has two addresses, source ip (s_ip) and target ip (t_ip), the s_ip in the next step and t_ip for "Lateral Movement" is in the same subnet, while satisfying $t\_ip(LateralMovement) = s\_ip(nextstep)$. We design a spatial rule operator to aggregate the prediction of the KGE model. This operator firstly generalizes spatial rules based on the attack background, then mines more potential rules using AnyBURL, and finally obtains a collection of spatial rules $\mathcal{S} = \{(S_i, c_i) \mid i \in I\}$. Notably, each rule S_i in $\mathcal{S}$ corresponds to a confidence score c_i, which is given by AnyBURL and indicates the availability and reliability of the rule.

For each prediction sample $r(s, ?)$, we traverse all entities and use the scoring function f to predict the scores of the top-m candidate entities to get $score_{kge}$, then we use the spatial rules in $\mathcal{S}$ generated by the operator to correct and augment the ranked entities for obtaining $score_{rule}$. The scores are normalized and aggregated to get the final score:

$$score_{aggre} = \gamma * score_{rule} + (1 - \gamma) * score_{kge} \tag{10}$$

where $\gamma \in (0, 1)$ is a hyperparameter.

Based on the aggregated scores, we create a reordered ranking. Once we have computed all the aggregated rankings for the tail prediction for a particular γ in $r(s, ?)$, we compute the MRR for these rankings.

5 Experiments

In this section, we evaluate the performance of our method through experimental analysis conducted on small-scale KGs. All experiments are run in Python with PyTorch framework [20] on RTX 2080 Ti.

5.1 Datasets and compared methods

We use typical small-scale datasets Top 250 Films [3] and Characters in A Dream in Red Mansions [2]. Meanwhile, we conduct the Cybersecurity Attack Knowledge Graph which is a small-scale sparse KG in the field of cybersecurity. Their statistics are shown in Table 3. Note: Red Mansions - A Dream in Red Mansions.

Table 3: Detailed information of the datasets.

Dataset	#Entity	#Relation	#Train	#Vaild	#Test	#Total
Top 250 Films	1061	5	1365	171	171	1707
Red Mansions	381	98	275	34	35	344
Cybersecurity Attack KG	1285	8	2648	331	332	3311

We select six baselines from existing KGE techniques, including four classical KGE models: TransE [4], DistMult [30], ComplEx [26], RotatE [24]; TA-TransE [9], which utilizes temporal attributes, and RelaGraph [23], which is suitable for small-scale knowledge graphs.

We use the link prediction task to measure the improvement after using our enhanced KGE method, the following two metrics are generally used to evaluate different KGE models: Mean reciprocal ranking (MRR) and Hits@n. We use grid search to find the optimal parameters, and we use the same configurations and hyperparameters for both the baseline and our model.

5.2 Results and analysis

The performance of link prediction is compared in Table 4. Overall, our model gains significant improvement on the Cybersecurity Attack Knowledge Graph and other small-scale KGs. Our model has the best performance when applied to the three small-scale KGs, with 15.3% improvement on MRR and 6.1% improvement on Hits@10. The improvement on Cybersecurity Attack KG and Top 250 Films are more obvious than that on Characters in A Dream in Red Mansions, because these two datasets have more entities but fewer relations. More precisely, the degree of entities is greater and better able to utilize Horn rules and neighborhood information.

It can be seen that the traditional KGE models are at a low level, which indicates that the models performing well on large KGs do not work well on small sparse KGs and have a poorly capable ability to mine unknown knowledge. Although TA-TransE and RelaGraph have improved compared to traditional KGE models, there are still some gaps compared to our model. Our model far exceeds baselines in all metrics, which shows that our model effectively improves the utilization of low-frequency entities. Experiments show that our model not only mines useful information through neighborhood information, but also uses Horn rules to mine more potential examples, making the vectors in the embedding space more reasonable. Meanwhile, during the testing process, our model aggregates the spatial rule scores and the KGE scores, which makes the performance of the model improve again, and the prediction ability of our model for new knowledge is even better.

5.3 Ablation study

Table 4 shows that our model achieves significant enhancement on all three small-scale KGs. Here, we use Cybersecurity Attack Knowledge Graph to analyze the effect of rule-based data augmentation, neighborhood-based embedding enhancement, and aggregated spatial rule scores on the results, respectively. Note: Rule-based - Rule-based data augmentation; Neigh-based - Neighborhood-based embedding enhancement.

As shown in Table 5, the neighborhood-based embedding enhancement is more obvious compared to the rule-based data augmentation. It is reasonable because there are fewer examples in small-scale KGs, and the capability of rule

Table 4: Comparison of various KGE models on the three small-scale datasets. The bold number means the best result.

KGE models	Cybersecurity Attack KG		Top 250 Films		Red Mansions	
	MRR	Hits@10	MRR	Hits@10	MRR	Hits@10
TransE	0.179	0.357	0.164	0.321	0.125	0.389
DistMult	0.123	0.162	0.047	0.082	0.035	0.136
ComplEx	0.131	0.191	0.065	0.132	0.068	0.136
RotatE	0.208	0.379	0.172	0.326	0.031	0.111
TA-TransE	0.293	0.423	0.221	0.398	0.136	0.412
RealGraph	0.368	0.503	0.415	0.500	0.251	0.464
Our model	**0.459**	**0.544**	**0.488**	**0.573**	**0.283**	**0.496**

mining systems such as AnyBURL is highly dependent on the size of the original data, so rule-based data augmentation techniques cannot be fully capable in small-scale KGs. However, we combine the two techniques and use neighborhood information to enhance the embedding representation of the KG, which is used to infer more new examples G_2 to update the facts of KG, greatly expanding the size of the dataset. AnyBURL extracts more rules on the updated dataset and mines more new examples G_1, these mined new examples enrich the neighborhood information of the KG in turn. Therefore, the rule-based data augmentation and neighborhood-based embedding enhancement enhance each other, and the experimental results also show that the model combining the two techniques works best, the whole is greater than the sum of its parts.

Table 5: Effects of rule-based data augmentation, neighborhood-based embedding enhancement, and aggregated spatial rule scores.

	Rule-based		Neigh-based		Our model	
	MRR	Hits@10	MRR	Hits@10	MRR	Hits@10
No aggregate	0.229	0.389	0.329	0.446	0.402	0.520
Aggregate	0.275	0.405	0.368	0.503	**0.459**	**0.544**

In the test, we aggregate spatial rule scores to improve the performance of the model. The knowledge graph in the link prediction task gives a score ranking rather than the final answer, meanwhile, there is improvement space for the small-scale KGE model. We find that Cybersecurity Attack KG contains rich spatial attributes (IP information), so we use spatial rules to correct and augment the prediction ranked entities, the experimental results show that aggregating spatial rule scores can increase the model's performance by 8%.

5.4 Parameter analysis

To investigate the effect of the iteration on the performance of our model, we conduct experiments on the Cybersecurity Attack Knowledge Graph and record

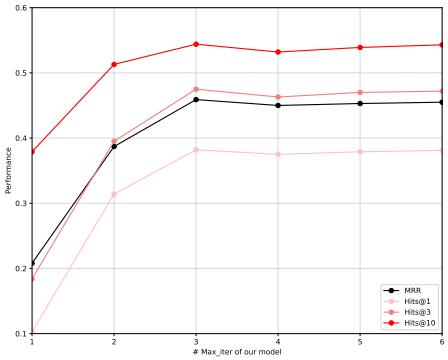

Fig. 2: Impact of $\#Max\text{-}iter$ on our model for Cybersecurity Attack Knowledge Graph.

the metrics of our model on the test data in each iteration as presented in Fig. 2. The experiments demonstrate that with increasing iterations, the MRR, Hits@1, Hits@3, and Hits@10 of the model improve rapidly at first, then tend to converge and reach the highest values in the third iteration. It shows that too many iterations will overfit the KGE model.

We explore how the hyperparameters help improve the performance of our model, to search for the best configures. Since α is different in each negative sampling, we only discuss hyperparameters β and γ. The metrics of models with different β and γ are shown in Table 6 and Table 7, the bold number means the best result.

The hyperparameter β appears in the scoring function TripleRE, mainly to balance the operation of relation vectors and entity vectors. As shown in the Table 6, the model's performance is optimized when $\beta = 0.6$, a larger or a smaller β is unnecessary for our model. γ balances the weights of spatial rule scores and predicted scores in the aggregation process, as shown in Table 7, when $\gamma = 0.4$, the model works best. When γ is too small, the model's prediction is mainly based on the predicted scores, but the scores are uncertain in some cases, so the weight of the spatial rule scores needs to be increased. But when γ is too large, the model's prediction focuses on the spatial rule score, but the spatial rules just correct the ranking. Therefore, the spatial rule scores and the predicted scores are equally important, γ should be taken to the middle of a range of values.

Table 6: Comparison of indicators with different values of β.

β	MRR	Hits@1	Hits@3	Hits@10
0.2	0.457	0.380	0.469	0.540
0.4	0.458	**0.381**	**0.472**	0.542
0.6	**0.459**	0.380	0.471	**0.544**
0.8	0.457	0.380	0.470	0.539

Table 7: Comparison of indicators with different values of γ.

γ	MRR	Hits@1	Hits@3	Hits@10
0.2	0.450	0.375	0.463	0.532
0.4	**0.459**	**0.380**	0.469	**0.544**
0.6	0.456	0.376	**0.470**	0.539
0.8	0.451	0.375	0.462	0.530

6 Conclusion

In this paper, considering the spatiotemporal specific attributes of the Cybersecurity Attack Knowledge Graph, we propose an enhanced knowledge graph embedding method for small-scale knowledge graphs by abstracting temporal attributes and extracting spatial rules. Meanwhile, to enrich the embedding space of small-scale knowledge graphs, we iteratively integrate rule-based data augmentation and neighborhood-based embedding enhancement. Our method has been proven applicable through experimental tests on three small-scale datasets and outperforms existing knowledge graph embedding techniques in all metrics. In the future, we will consider using Large Language Models to expand our knowledge in the field of cybersecurity.

Acknowledgments. This work is supported in part by the Major Key Project of PCL (PCL2024A05), the Shenzhen Science and Technology Program (No. KJZD20231023094701003), and the National Natural Science Foundation of China (Grant No. 62372137).

References

1. Attack flow project, https://center-for-threat-informed-defense.github.io/attack-flow/
2. A dream in red mansions character relationship knowledge graph, www.openkg.cn/dataset/ the-dream-of-the-red-chamber-main
3. Top250 film works knowledge graph at home and abroad, www.openkg.cn/dataset/ top250film

4. Bordes, A., Usunier, N., Garcia-Duran, A., Weston, J., Yakhnenko, O.: Translating embeddings for modeling multi-relational data. Advances in neural information processing systems 26: Annual Conference on Neural Information Processing Systems 2013 pp. 2787–2795 (2013)
5. Dettmers, T., Minervini, P., Stenetorp, P., Riedel, S.: Convolutional 2d knowledge graph embeddings. In: Proceedings of the AAAI conference on artificial intelligence. vol. 32 (2018)
6. Galárraga, L., Teflioudi, C., Hose, K., Suchanek, F.M.: Fast rule mining in ontological knowledge bases with amie +. The VLDB Journal **24**(6), 707–730 (2015)
7. Galárraga, L.A., Teflioudi, C., Hose, K., Suchanek, F.: Amie: association rule mining under incomplete evidence in ontological knowledge bases. In: Proceedings of the 22nd international conference on World Wide Web. pp. 413–422 (2013)
8. Gao, M., Li, J.Y., Chen, C.H., Li, Y., Zhang, J., Zhan, Z.H.: Enhanced multi-task learning and knowledge graph-based recommender system. IEEE Transactions on Knowledge and Data Engineering **35**(10), 10281–10294 (2023)
9. García-Durán, A., Dumančić, S., Niepert, M.: Learning sequence encoders for temporal knowledge graph completion. arXiv preprint arXiv:1809.03202 (2018)
10. Guo, S., Wang, Q., Wang, L., Wang, B., Guo, L.: Jointly embedding knowledge graphs and logical rules. In: Proceedings of the 2016 conference on empirical methods in natural language processing. pp. 192–202 (2016)
11. Jiang, T., Liu, T., Ge, T., Sha, L., Li, S., Chang, B., Sui, Z.: Encoding temporal information for time-aware link prediction. In: Proceedings of the 2016 conference on empirical methods in natural language processing. pp. 2350–2354 (2016)
12. Jin, W., Zhao, B., Yu, H., Tao, X., Yin, R., Liu, G.: Improving embedded knowledge graph multi-hop question answering by introducing relational chain reasoning. Data Mining and Knowledge Discovery **37**(1), 255–288 (2023)
13. Li, H., Gao, X., Feng, L., Deng, Y., Yin, Y.: Stargraph: Knowledge representation learning based on incomplete two-hop subgraph. arXiv preprint arXiv:2205.14209 (2022)
14. Li, R., Zhao, J., Li, C., He, D., Wang, Y., Liu, Y., Sun, H., Wang, S., Deng, W., Shen, Y., Xie, X., Zhang, Q.: House: Knowledge graph embedding with householder parameterization. In: International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA. Proceedings of Machine Learning Research, vol. 162, pp. 13209–13224. PMLR (2022)
15. Lin, Y., Liu, Z., Sun, M., Liu, Y., Zhu, X.: Learning entity and relation embeddings for knowledge graph completion. In: Proceedings of the 29th AAAI Conference on Artificial Intelligence. pp. 2181–2187. AAAI Press (2015)
16. Mai, S., Zheng, S., Yang, Y., Hu, H.: Communicative message passing for inductive relation reasoning. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 35, pp. 4294–4302 (2021)
17. Meilicke, C., Chekol, M.W., Betz, P., Fink, M., Stuckeschmidt, H.: Anytime bottom-up rule learning for large-scale knowledge graph completion. The VLDB Journal **33**(1), 131–161 (2024)
18. Nguyen, D.Q., Vu, T., Nguyen, T.D., Phung, D.: Quatre: Relation-aware quaternions for knowledge graph embeddings. In: Companion Proceedings of the Web Conference 2022. pp. 189–192 (2022)
19. Park, N., Liu, F., Mehta, P., Cristofor, D., Faloutsos, C., Dong, Y.: Evokg: Jointly modeling event time and network structure for reasoning over temporal knowledge graphs. In: Proceedings of the fifteenth ACM international conference on web search and data mining. pp. 794–803 (2022)

20. Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L., Lerer, A.: Automatic differentiation in pytorch. In: Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017 (2017)
21. Pujara, J., Augustine, E., Getoor, L.: Sparsity and noise: Where knowledge graph embeddings fall short. In: Proceedings of the 2017 conference on empirical methods in natural language processing. pp. 1751–1756 (2017)
22. Rossi, A., Barbosa, D., Firmani, D., Matinata, A., Merialdo, P.: Knowledge graph embedding for link prediction: A comparative analysis. ACM Transactions on Knowledge Discovery from Data (TKDD) **15**(2), 1–49 (2021)
23. Shi, B., Wang, H., Li, Y., Deng, S.: Relagraph: Improving embedding on small-scale sparse knowledge graphs by neighborhood relations. Information Processing & Management **60**(5), 103447 (2023)
24. Sun, Z., Deng, Z.H., Nie, J.Y., Tang, J.: Rotate: Knowledge graph embedding by relational rotation in complex space. In: International Conference on Learning Representations
25. Toutanova, K., Chen, D.: Observed versus latent features for knowledge base and text inference. In: Proceedings of the 3rd Workshop on Continuous Vector Space Models and their Compositionality. pp. 57–66. Association for Computational Linguistics (2015)
26. Trouillon, T., Welbl, J., Riedel, S., Gaussier, É., Bouchard, G.: Complex embeddings for simple link prediction. In: International conference on machine learning. pp. 2071–2080. PMLR (2016)
27. Wang, Y., Ruffinelli, D., Gemulla, R., Broscheit, S., Meilicke, C.: On evaluating embedding models for knowledge base completion. In: Proceedings of the 4th Workshop on Representation Learning for NLP (RepL4NLP-2019). pp. 104–112. Association for Computational Linguistics (2019)
28. Xie, Y., Wang, H., Wang, L., Luo, L., Li, J., Gu, Z.: Reinforced negative sampling for knowledge graph embedding. In: International Conference on Database Systems for Advanced Applications. pp. 358–374. Springer (2024)
29. Xiong, C., Power, R., Callan, J.: Explicit semantic ranking for academic search via knowledge graph embedding. In: Proceedings of the 26th international conference on world wide web. pp. 1271–1279 (2017)
30. Yang, B., Yih, S.W.t., He, X., Gao, J., Deng, L.: Embedding entities and relations for learning and inference in knowledge bases. In: Proceedings of the 3rd International Conference on Learning Representations (2015)
31. Yu, L., Luo, Z., Liu, H., Lin, D., Li, H., Deng, Y.: Triplere: Knowledge graph embeddings via tripled relation vectors. arXiv preprint arXiv:2209.08271 (2022)
32. Zhang, D., Rong, Z., Xue, C., Li, G.: Simre: Simple contrastive learning with soft logical rule for knowledge graph embedding. Information Sciences **661**, 120069 (2024)
33. Zhang, S., Liang, X., Tang, H., Guan, Z.: Hybrid interaction temporal knowledge graph embedding based on householder transformations. In: Proceedings of the 31st ACM International Conference on Multimedia. pp. 8954–8962 (2023)
34. Zhang, W., Paudel, B., Wang, L., Chen, J., Zhu, H., Zhang, W., Bernstein, A., Chen, H.: Iteratively learning embeddings and rules for knowledge graph reasoning. In: The world wide web conference. pp. 2366–2377 (2019)
35. Zhang, Y., Zhou, Z., Yao, Q., Chu, X., Han, B.: Adaprop: Learning adaptive propagation for graph neural network based knowledge graph reasoning. In: Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining. pp. 3446–3457 (2023)