applied sciences

MDPI

Article

A Cybersecurity Knowledge Graph Completion Method Based on Ensemble Learning and Adversarial Training

Peng Wang [1,2], Jingju Liu [1,2,*], Dongdong Hou [1,2] and Shicheng Zhou [1,2]

[1] College of Electronic Engineering, National University of Defense Technology, Hefei 230037, China
[2] Anhui Province Key Laboratory of Cyberspace Security Situation Awareness and Evaluation, Hefei 230037, China
* Correspondence: jingjul@aliyun.com

Abstract: The application of cybersecurity knowledge graphs is attracting increasing attention. However, many cybersecurity knowledge graphs are incomplete due to the sparsity of cybersecurity knowledge. Existing knowledge graph completion methods do not perform well in domain knowledge, and they are not robust enough relative to noise data. To address these challenges, in this paper we develop a new knowledge graph completion method called CSEA based on ensemble learning and adversarial training. Specifically, we integrate a variety of projection and rotation operations to model the relationships between entities, and use angular information to distinguish entities. A cooperative adversarial training method is designed to enhance the generalization and robustness of the model. We combine the method of generating perturbations for the embedding layers with the self-adversarial training method. The UCB (upper confidence bound) multi-armed bandit method is used to select the perturbations of the embedding layer. This achieves a balance between perturbation diversity and maximum loss. To this end, we build a cybersecurity knowledge graph based on the CVE, CWE, and CAPEC cybersecurity databases. Our experimental results demonstrate the superiority of our proposed model for completing cybersecurity knowledge graphs.

Keywords: cybersecurity knowledge graph; knowledge graph completion; ensemble learning; adversarial training

Citation: Wang, P.; Liu, J.; Hou, D.; Zhou, S. A Cybersecurity Knowledge Graph Completion Method Based on Ensemble Learning and Adversarial Training. *Appl. Sci.* **2022**, *12*, 12947. https://doi.org/10.3390/app122412947

Academic Editors: Cheng Huang, Weina Niu and Wang Yang

Received: 1 November 2022
Accepted: 14 December 2022
Published: 16 December 2022

1. Introduction

As the application of computer networks becomes more widespread, cybersecurity issues are posing a serious threat to people's work and life. In recent years, cybersecurity attacks such as denial of service (DoS), phishing, and others have increased. Realizing cyberspace security situational awareness [1] by using existing cyber-threat knowledge is of great significance. Vulnerabilities are critical to cybersecurity, as they can allow hackers to gain unauthorized access or destroy computer systems. Hackers take advantage of vulnerabilities as soon as they discover them. This results in enormous losses, because individuals and enterprises are not be able to use effective protective measures immediately. Therefore, comprehensive vulnerability information can help individuals and enterprises to reduce cybersecurity risks. Many cyberattacks are implemented against software or system weaknesses. Software reliability is often affected by a variety of complex factors [2]. Mastering software weakness information can help individuals and enterprises to understand the possible cyberattacks they may encounter. Individuals and enterprises can use effective preventive measures to reduce potential losses by mastering the attack patterns of hackers. In the process of dealing with cyber-threats, people have accumulated much valuable experience. To better address existing cybersecurity threats, the MITRE Corporation maintains the CVE (common vulnerability and exposure), CWE (common weakness enumeration), and CAPEC (common attack pattern enumerations and classifications) cybersecurity knowledge databases for describing cybersecurity threats in real-world

scenarios. CVE contains a variety of computer security vulnerabilities that have been disclosed publicly, while CWE contains a number of software defect descriptions and classifications and is a community-developed list of software and hardware weakness types. CAPEC provides a public classification of common attack patterns used by hackers, and has become an important public standard for the software community to share attack patterns. These cybersecurity knowledge databases contain important empirical information. For example, CWE-266 (incorrect privilege assignment) may be threatened by CVE-2005-2741 (product allows users to grant themselves certain rights that can be used to escalate privileges). CAPEC-234 (hijacking a privileged process) can follow CAPEC-242 (code injection). Despite the fact that existing cyber-knowledge databases are already very large, new vulnerabilities, software weaknesses, and cyberattack patterns continue to emerge. At the same time, important relationships in existing knowledge databases have been discovered. For example, the mapping of CAPEC-698 (install malicious extension) to CWE-507 (trojan horse) was not added until CAPEC List Version 3.8, while CWE-1357 (reliance on uncontrolled component) was not added until CWE List Version 4.7. This is an important reason knowledge databases must be constantly updated and maintained.

In recent years, knowledge graph technology has developed rapidly. A knowledge graph is a semantic network composed of entities, relationships, and their related attributes. It has been widely used in the medical [3], financial [4], and transportation fields [5], among others [6–9]. However, due to the variety of network states and low data value density, the development of cybersecurity knowledge graphs remains in its infancy. Introducing knowledge graphs to the field of cybersecurity can make effective use of previous experiences [10], help to show the cybersecurity situation, and support security decision making and early warning predictions. Because people have limited cognitive abilities, the existing cybersecurity knowledge is clearly not flawless.

To address this problem, we propose a model for cybersecurity knowledge graph completion. This model can infer unknown facts from existing facts to realize and complement existing cybersecurity knowledge. First, we define ontologies and relationships based on existing cybersecurity knowledge. The knowledge graph can then be used to store cybersecurity entities, relationships, and associated attributes. By mining associations on the basis of existing security vulnerabilities, software weaknesses, and attack patterns, a model familiar with the relevant rules can be obtained. If software weaknesses are known, it becomes possible to predict potential attack patterns and the relationships between various software weaknesses based on learned relationship patterns, such as sequential and parent–child relationships. When the model learns abstract and complex relationships between existing entities and relationships, it is able to realize unknown knowledge by mining information on known entities and relationships, then use them to provide assistance with situation awareness.

In summary, the main contributions of this paper are as follows:

- We propose a new knowledge graph completion model that uses a variety of projection and rotation operations to model relationships between entities. In addition, it represents entities and relationships based on angle information. The utilization of ensemble learning improves the generalization of the model.
- We design a cooperative adversarial training method, which plays an important role in the training process of the model. The UCB multi-armed bandit method is used to set the adversarial perturbations, which improves the robustness of the model.
- A cybersecurity knowledge graph is built based on the CVE, CWE, and CAPEC databases. We measure the performance of our model using experiments, finding that it can effectively discover unknown facts through the creation of a cybersecurity knowledge graph.

The remainder of this paper is organized as follows: Section 2 describes the related work on cybersecurity knowledge and knowledge graph completion; Section 3 introduces our method of completing the knowledge graph construction task; Section 4 describes the completed cybersecurity knowledge graph and demonstrates the performance of our

model by comparing it with other excellent models. Finally, our conclusions are provided in Section 5.

2. Related Work

Cybersecurity knowledge. It is very important to master knowledge on cybersecurity. There is a gap in knowledge between cybersecurity education and industrial needs [11]. The use of existing cybersecurity knowledge can provide strong knowledge support for cybersecurity operations. With the continuous accumulation of cybersecurity knowledge, many researchers have conducted valuable studies based on the existing knowledge. Xiang Li et al. [12] designed an effective datamining algorithm to obtain the basic characteristics of vulnerabilities based on the CVE and CWE databases. Zhuobing Han et al. [13] designed a knowledge graph embedding method that combined descriptions and structural knowledge. Their method embeds the relationship between software weaknesses in a low-dimensional vector space in order to help infer the consequences of common software weaknesses. Hongbo Xiao et al. [14] embedded the relationship information and descriptive information of software security entities into the continuous vector space. This method combines the translation-based embedding model and a CNN encoder to predict the relationship of software security entities based on the CVE, CWE, and CAPEC databases. Liu Yuan et al. [15] proposed a text-enhanced graph attention network model. They built a cybersecurity knowledge graph based on the CVE, CWE, and CAPEC databases and predicted the relationship between entities in the knowledge graph. Yichao Zang et al. [16] used association rule mining technology to mine the semantic knowledge of a penetration test hidden in a large amount of original penetration test data, which is very helpful for the automated penetration test. In addition to using existing structured cybersecurity data, researchers have extracted information from unstructured cyber-threat intelligence. Robert A. Bridges et al. [17] designed a method of labeling cybersecurity texts, and provided a corpus for information extraction in the field of cybersecurity. T. Satyapanich et al. [18] proposed the CASIE model, which can extract events from cybersecurity texts. Peipei Liu et al. [19] designed multifeature-based semantic augmentation networks able to extract cybersecurity entities from unstructured texts for further analyses. Information extraction models based on neural networks have made great progress in processing threat intelligence. All of the above studies excavate existing security knowledge from different perspectives, and can provide important guidance for future network protection measures.

Knowledge graph completion. Due to the limitations of knowledge extraction and other technologies, most advanced knowledge graphs are usually incomplete. However, incomplete knowledge graphs usually contain rich semantic information. Knowledge graph completion technology can be used to supplement this knowledge. By mining the existing information in knowledge graph, it is usually possible to find unknown facts. This process can predict whether there are missing relationship edges between entities in the knowledge graph in order to expand the knowledge graph and correct errors [20]. This procedure is helpful for further applications of the knowledge graph. Rule-based models are basic knowledge graph completion methods with good interpretability. Simon Ott et al. [21] proposed a rule-based model called SAFRAN. However, it is difficult for a rule-based model to learn abstract features. As such, distance-based models have an important influence on the completion of knowledge graphs, as they map entities and relationships into low-dimensional space vectors and then perform relevant calculations. TransE [22] uses the classical calculation expression, $h + r = t$, to carry out the knowledge graph completion task; h, r, and t represent the embedding of the head entity, relationship, and tail entity, respectively. This method has obvious advantages, including fewer parameters and low computational complexity, and it has powerful abilities in solving knowledge graphs with various relationship attributes. In recent years, researchers began to map models to different complex vector spaces for modeling in order to better cope with the diversity of relationship attributes. TransH [23] uses a hyperplane to solve the above problems. TransR [24] uses a projection matrix to project the source vector to the

entity space with specific relationships on the basis of TransH. RotatE [25] completes the rotation from head entities to tail entities based on the complex vector space, although the rotation ability is limited. CHolE [26] models the entities and relationships with their type constraints in the complex vector space. A few effective models [27–29] even project entities and relationships into quaternion or octet spaces, which greatly increases computational complexity. HAKE [30] uses a polar coordinate system to solve the problem and to model semantic hierarchies. Because its rotation of the radial coordinate is limited, simlar to RotatE, its hierarchical modeling abilities are limited. HousE [31] uses householder transformation to realize projections and rotations. Although this method has good interpretability, it has shortcomings when mining potential semantic knowledge. Compared with the above models, semantic matching models can be used to mine potential semantic knowledge. Semantic matching models calculate semantic similarity to complete the knowledge graph. RESCAL [32] uses low-dimensional vectors to represent entities and a matrix to represent relationships. The scoring function $f_r(h,t) = h^T M_r t$ is used to evaluate semantic similarity. Distmult [33] simplifies the matrix into a diagonal matrix on the basis of RESCAL, reducing the number of parameters and maintaining good performances. ComplEx [34] enhances Distmult in a complex vector space, although it incurs a high cost. Yihong Chen et al. [35] used relation prediction as an auxiliary training objective; however, the performance of this approach is limited when the number of relationships is small. With the development of neural networks, researchers have used neural networks to complete knowledge graphs. ConvE [36], ConvKB [37], and ConvR [38] all use convolutional neural networks to represent entities and relationships for information interaction. R-Men [39] uses a relational memory neural network to encode the internal dependencies of a triple. RGCN [40], SACN [41], and KBGAT [42] use graph neural networks to fuse the information of the surrounding entities. Rui Wang et al. [43] used graph-attenuated attention networks to complete the knowledge graph, although this was insufficient to deal with relationships. With the development of pretrained language models, many researchers have used such models to complete knowledge graphs. SimKGC [44] introduces contrastive learning and pretrained language models into the knowledge graph completion process. However, when the distribution difference between the pretrained language models and the target is large, the experimental effect is limited. Other knowledge graph completion models [45–47] obtain information from texts. Therefore, when there is no text that can accurately describe entities or relationships, the models cannot obtain sufficient reasoning basis, and the reasoning effect is poor.

Although the above models have achieved fine results, they ignore the comprehensive utilization of entity information in the modeling process. In practice, a single model usually has limitations. Ibomoiye Domor Mienye et al. [48] demonstrated that a single model tends to produce large fluctuations in variance, bias, and accuracy during the training process, while the ensemble method can effectively reduce the probability of errors. Thomas G. Dietterich [49] found that a single model has important deficiencies in the statistical, computational, and representational aspects. A single model is more prone to fall into a local optimal solution than an ensemble model. Motivated by the success of widely used ensemble models, we propose an ensemble model in which the base models perform different projection and rotation operations. This model makes good use of angle information to complete the knowledge graph. The model is often disturbed during the training process, which results in unexpected results. Adversarial training is one of the most promising defense methods for improving the robustness of a model. The idea of adversarial training is to establish a robust model that can be well extended to samples with small perturbations [50]. Researchers have used adversarial training to solve many problems during the training process, such as reducing overfitting and improving generalizations [51]. Thus, our model enhances generalization and robustness using the cooperative adversarial training method. Our model shows good performances when processing cybersecurity data.

3. Methodology

3.1. Overview

Most existing knowledge graphs are incomplete, and it is necessary to supplement the unknown facts in knowledge graphs [52]. Figure 1 shows the basic process of knowledge graph completion. Here, we have a knowledge graph G, entity set H, and relationship set R. The knowledge graph G contains a large number of triples, i.e., $< head, relation, tail >$, where $head \in H$, $tail \in H$, and $relation \in R$. When the triple is incomplete, the missing head entity or tail entity needs to be predicted. This section introduces CSEA—a **C**yber **S**ecurity knowledge graph completion model based on **E**nsemble learning and **A**dversarial training. CSEA uses multiple projections to handle sophisticated relation-mapping properties; this method allows relational patterns to be modeled based on rotations due to its superior capacity. It uses angle information to distinguish entities. In addition, cooperative adversarial training methods are used to enhance the robustness of the model.

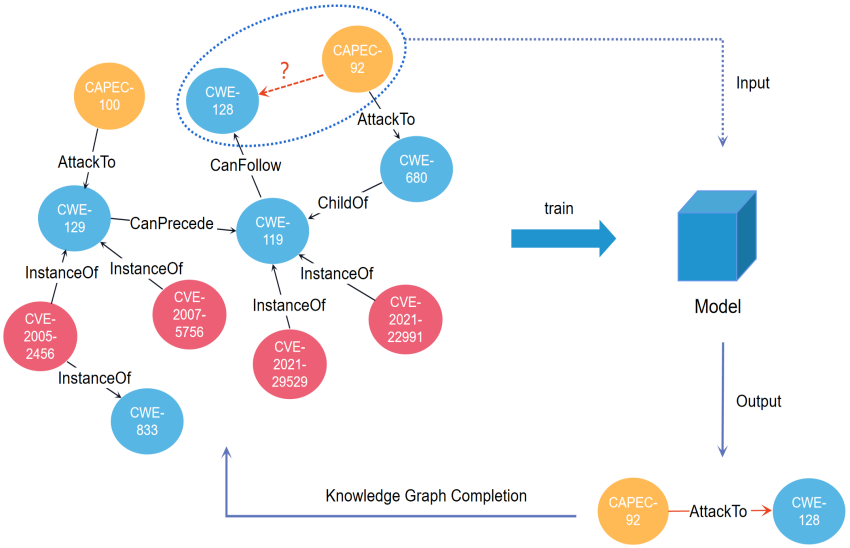

Figure 1. An illustration of the knowledge graph completion process.

3.2. Ensemble Model

In the first part of the model, we use householder transformation to realize projections and rotations. The householder transformation is an orthogonal transformation that can transform an n-dimensional vector x to any n-dimensional vector $\hat{x}$. From a geometric perspective, the householder transformation reflects x relative to $\hat{x}$ via the hyperplane between x and $\hat{x}$. Assuming that the source vector is x, given the unit vector e and identity matrix E, a new vector, $\hat{x}$, can be obtained using householder matrix $Matrix(e)$, where $Matrix(e) = E - 2ee^T$. The matrix $Matrix(e)$ takes e as a variable. The basic transformation of the householder transformation is described as follows:

$$\hat{x} = Matrix(e)x = x - 2 < x, e > e, \tag{1}$$

where $< x, e >$ represents the dot product of x and e. Many research studies have been conducted with respect to entity projection, and entity projections allow entities to produce relationship-specific representations in order to solve complex relationship-mapping properties. We use householder transformations to realize householder projections. The householder projection can be realized by fine-tuning the householder matrix $Matrix(e)$. We add a scalar, ω, to control the distance amplitude of the projection. The projection can better change the relative distance between two points and ensure that it produces a robust relationship-specific representation. The projection matrix is modified to $PMatrix(e, \omega) = E - \omega ee^T$. We denote a head entity in the first part as h_1, $h_1 \in \mathbb{R}^{k \times m}$, where k represents the embedding size and m represents the dimension of each row vector,

while t_1 represents a tail entity in the first part, $t_1 \in \mathbb{R}^{k \times m}$. For each relationship, two types of parameters are used to represent the projection of the relationship, namely, $P \in \mathbb{R}^{k \times c \times m}$ and $W \in \mathbb{R}^{k \times c}$. Each row $P[i] \in \mathbb{R}^{c \times m}$ comprises c m-dimensional unit vectors. Each row of W consists of c real numbers. Each dimension of the projected head entity $h_{project}$ and tail entity $t_{project}$ can be calculated in the following form:

$$\begin{cases} h_{project}[i] = Project(h_1[i], P_1[i], W_1[i]) = \prod_{j=1}^{c} PMatrix(P_1[i][j], W_1[i][j]) h_1[i] \\ t_{project}[i] = Project(t_1[i], P_2[i], W_2[i]) = \prod_{j=1}^{c} PMatrix(P_2[i][j], W_2[i][j]) t_1[i], \end{cases} \tag{2}$$

where $1 \leq i \leq k$ and $1 \leq j \leq c$. Based on Formula (1), the rotation matrix is set to $RMatrix(e) = E - 2ee^T$. The householder rotation operation can be performed on the source vector. Rui Li et al. [31] proved that the composition of $2\lfloor m/2 \rfloor$ householder reflections can realize any m-dimensional transformation; thus, we denote a relation in the first part as r_1, $r_1 \in \mathbb{R}^{k \times 2n \times m}$, where $n = \lfloor m/2 \rfloor$ and each row of r_1 consists of a series of m-dimensional unit vectors. Formula (3) shows the rotation of each row vector $h_{project}[i]$:

$$h_{rotate}[i] = Rotate(h_{project}[i], r_1[i]) = \prod_{j=1}^{2n} RMatrix(r_1[i][j]) h_{project}[i], \tag{3}$$

where $1 \leq i \leq k$ and $1 \leq j \leq 2n$, while h_{rotate} represents the rotation of the projected entity $h_{project}$. Relation-specific householder rotations are applied to each row of the projected head embedding. There is an expectation that the rotated result h_{rotate} should be close to the projected tail embedding $t_{project}$; thus, $dist_1$ can be calculated as follows:

$$dist_1 = \sum_{i=1}^{k} \| h_{rotate}[i] - t_{project}[i] \|_2. \tag{4}$$

In addition to using householder reflections, we achieve projection and rotation operations in a complex vector space. In the second part, we share similar operations with Rot-Pro [53]. The projection matrix for each dimension of an entity is defined as follows:

$$Pro\_Matrix(i) = \begin{bmatrix} cos\theta_i & -sin\theta_i \\ sin\theta_i & cos\theta_i \end{bmatrix}^{-1} \begin{bmatrix} x_i & 0 \\ 0 & y_i \end{bmatrix} \begin{bmatrix} cos\theta_i & -sin\theta_i \\ sin\theta_i & cos\theta_i \end{bmatrix}, \tag{5}$$

where $1 \leq i \leq k$, $x_i \in \{0, 1\}$ and $y_i \in \{0, 1\}$, while θ_i is the rotation phase in each dimension. We use a complex number to represent an entity; thus, the projection is described as follows:

$$\begin{bmatrix} Re(h_{c\_pro}[i]) \\ Im(h_{c\_pro}[i]) \end{bmatrix} = Pro\_Matrix(i) \begin{bmatrix} Re(h_2[i]) \\ Im(h_2[i]) \end{bmatrix}, \tag{6}$$

where $1 \leq i \leq k$, h_2 is a head entity embedding in the second part, $h_2 \in \mathbb{C}^k$; $Re(h_2[i])$ represents the real number of its ith dimension, and $Im(h_2[i])$ represents the imaginary number of its ith dimension. Tail entity embedding t_2 in the second part performs similar projection operations, i.e., $t_2 \in \mathbb{C}^k$. After carrying out these projections, we obtain the projected head entity h_{c_pro} and projected tail entity t_{c_pro}. Then, we rotate the projected head entity h_{c_pro}. The rotation for each dimension of h_{c_pro} is as follows:

$$\begin{bmatrix} Re(h_{c\_rot}[i]) \\ Im(h_{c\_rot}[i]) \end{bmatrix} = \begin{bmatrix} cos\phi_i & -sin\phi_i \\ sin\phi_i & cos\phi_i \end{bmatrix} \begin{bmatrix} Re(h_{c\_pro}[i]) \\ Im(h_{c\_pro}[i]) \end{bmatrix}, \tag{7}$$

where $1 \leq i \leq k$ and ϕ_i is relational rotation phase in each dimension. The rotated result h_{c_rot} is expected to be close to the projected tail embedding t_{c_pro}, and the distance function in the second part is defined as follows:

$$dist_2 = \| h_{c\_rot} - t_{c\_pro} \|_2. \tag{8}$$

Semantic hierarchies usually exist in the field of cybersecurity. For example, for entities at different levels of the hierarchy, CWE-345 (efficient verification of data authentication) is the parent of CWE-352 (cross-site request forgery). For entities at the same level of the hierarchy, CWE-345 (dfficient verification of data authentication) is the peer of CWE-20 (improve input validation). While rotations can distinguish entities at different levels of the hierarchy, there are nonetheless inadequacies in distinguishing entities at a hierarchy of the same level. HAKE demonstrates that angular information is well able to capture differences between entities at the same level of the hierarchy. For entities at the same level of the hierarchy, their measurement scores at different levels of the hierarchy can be considered very similar. Due to different points on the same circle possessing different phases, the difference in phases is used to distinguish them. The phase mapping of the head entity h_3, relationship r_3, and tail entity t_3 satisfies the following relationship:

$$(h_3 + r_3) mod\ 2\pi = t_3, \tag{9}$$

where $h_3, r_3, t_3 \in [0, 2\pi)^k$. Because the phase difference in the same plane changes periodically, we use periodic function $sin(x/2)$ to calculate the difference between the head entity and tail entity at the same level of the hierarchy. The formula is as follows:

$$dist_3 = \|sin(\frac{h_3 + r_3 - t_3}{2})\|_1. \tag{10}$$

The dynamic adjustment of the weights of various rotation and projection methods during training can improve the effectiveness of ensemble learning. The process of dynamically adjusting weights is shown in Algorithm 1.

Algorithm 1: Weight optimization algorithm

Input: The weights of previous training step $Initial\_\alpha, Initial\_\beta, Initial\_\lambda$
The weight growth directions $Direct\_\alpha, Direct\_\beta, Direct\_\lambda$
The big fixed constant T
The small fixed constant η, $0 \leq \eta \leq 1$
The loss of previous training step $loss_{t-1}$
The loss of current training steps $loss_t$

Output: The optimized weight α, β, λ

if $loss_{t-1} < loss_t$ **then**
  $\sigma = 1 - e^{\frac{-|loss_t - loss_{t-1}|}{T}}$
  $\theta_\alpha \leftarrow$ Generate a random number from 0 to 1
  if $\theta_\alpha > \sigma$ **then**
    if $Direct\_\alpha == " + "$ **then**
      $Direct\_\alpha = " - "$
      α=$Initial\_\alpha - \eta \cdot Initial\_\alpha$
    else
      $Direct\_\alpha = " + "$
      α=$Initial\_\alpha + \eta \cdot Initial\_\alpha$
    end
  else
    α=$Initial\_\alpha$
  end
  β and λ are obtained by performing operations similar to α
else
  α=$Initial\_\alpha$, β=$Initial\_\beta$, λ=$Initial\_\lambda$.
end
return α, β, λ

Inspired by the simulated annealing algorithm, we use information from the training process to continuously optimize the weight. In the training process, the goal is to minimize loss; thus, we use the change in loss as the judgment condition of weight changes. At the beginning of the training process, we initialize the weights; in addition, we randomly set the value growth direction (+ or −) for each weight. If the loss of the previous step is greater than the loss of the current step, this means that the weight's settings and the value's growth direction are appropriate, and the current weights do not need to be changed. If the loss of the previous step is less than the loss of the current step, the value's growth direction should be changed, then the weight's value in the new value growth direction should be changed. To prevent falling into the local optimal solution, there is a small probability that the weight and direction may remain unchanged when the loss of the current training step is greater than that of the previous training step; this probability is controlled by the loss change in two steps.

The basic process of CSEA is shown in Algorithm 2. The final evaluation function of the triple is described as follows:

$$score = \gamma - (\alpha.dist_1 + \beta.dist_2 + \lambda.dist_3), \tag{11}$$

where γ is a fixed constant and α, β, and λ are weights.

Algorithm 2: Forward procedure of CSEA

Input: The triple $(head, relation, tail)$
The fixed constant γ
The weights: α, β, λ
Parameters of size: k, c, m, n

Output: The score of triple $(head, relation, tail)$

Initialize h_1, r_1, t_1, P_1, W_1, h_2, t_2, θ, x, y, ϕ, t_2, h_3, r_3, t_3 for triple $(head, relation, tail)$.

for $i = 1$ *to* k **do**

$h_{project}[i] = Project(h_1[i], P_1[i], W_1[i]) = \prod_{j=1}^{c} PMatrix(P_1[i][j], W_1[i][j])h_1[i]$

$t_{project}[i] = Project(t_1[i], P_2[i], W_2[i]) = \prod_{j=1}^{c} PMatrix(P_2[i][j], W_2[i][j])t_1[i]$

$h_{rotate}[i] = Rotate(h_{project}[i], r_1[i]) = \prod_{j=1}^{2n} RMatrix(r_1[i][j])h_{project}[i]$

end

$dist_1 = \sum_{i=1}^{k} \|h_{rotate}[i] - t_{project}[i]\|_2$

for $i = 1$ *to* k **do**

$$\begin{bmatrix} Re(h_{c\_pro}[i]) \\ Im(h_{c\_pro}[i]) \end{bmatrix} = Pro\_Matrix(i) \begin{bmatrix} Re(h_2[i]) \\ Im(h_2[i]) \end{bmatrix}$$

$$\begin{bmatrix} Re(t_{c\_pro}[i]) \\ Im(t_{c\_pro}[i]) \end{bmatrix} = Pro\_Matrix(i) \begin{bmatrix} Re(t_2[i]) \\ Im(t_2[i]) \end{bmatrix}$$

$$\begin{bmatrix} Re(h_{c\_rot}[i]) \\ Im(h_{c\_rot}[i]) \end{bmatrix} = \begin{bmatrix} cos\phi_i & -sin\phi_i \\ sin\phi_i & cos\phi_i \end{bmatrix} \begin{bmatrix} Re(h_{c\_pro}[i]) \\ Im(h_{c\_pro}[i]) \end{bmatrix}$$

end

$dist_2 = \|h_{c\_rot} - t_{c\_pro}\|_2$

$dist_3 = \|sin(\frac{h_3 + r_3 - t_3}{2})\|_1$

$score = \gamma - (\alpha.dist_1 + \beta.dist_2 + \lambda.dist_3)$

return *score*

General distance-based models usually treat the relationships between entities as shifts between vectors, which is too idealized to accommodate complex and variable relationships. Rotations have stronger representations than normal translations. The householder transformation effectively expands the rotation to a higher-dimensional space, and it combines projection operations to produce relationship-specific representations. In addition, projection and rotation in complex spaces are considered to be simple and effective, and

can cooperate with projections and rotations in higher-dimensional space. Ensemble learning can implement multiple projections and rotations in parallel. When a single learning algorithm produces calculation errors, other learning algorithms can reduce the error range and retain final results that are stable.

3.3. Cooperative Adversarial Training

In order to improve the generalization and robustness of the model, many adversarial training methods [54–57] were used. In this section, we design an effective model training method, namely, the cooperative adversarial training method. Most current knowledge graph completion algorithms set and use negative sampling, which improves the recognition ability of a model in the face of negative samples. For this reason, researchers have designed a general framework for self-adversarial training using negative samples. Formula (12) describes the self-adversarial training process:

$$\begin{aligned} self\_adv\_loss = &- logsigmoid(\mu - score(h, r, t)) \\ &- \sum_{i=1}^{l} p(h_i', r, t_i') logsigmoid(score(h_i', r, t_i') - \mu), \end{aligned} \tag{12}$$

where (h, r, t) is a positive triple, μ is a constant, (h_i', r, t_i') is one of the negative triples, and $p(h_i', r, t_i')$ is the probability distribution of negative sampling. However, it is insufficient to only set negative samples. Most current methods use representation learning to represent entities and relationships. Therefore, accurate embedded expressions have important impacts on the performance of a model. During the training process, it is easy for poor-quality samples to prevent a model from achieving the best fit. Because the embedding layer can directly represent entities and relationships, adding perturbations to the embedding layer trains the model in being able to accurately represent them in the presence of perturbations. Adversarial training needs to maximize the internal loss, and this form of training tries its best to reduce external losses to ensure that the model can effectively resist perturbations. Taking the classification task as an example, we can suppose that x is the input sample and y is the corresponding label; the well-known min–max formula [58] provides a good description of this idea:

$$Min\{Max\{Loss(x + \eta, y; \psi)\}\}, \tag{13}$$

where η is the value of the interference and ψ includes the parameters of the model.

The machine learning model mainly relies on the gradient descent method to help the model converge to the optimal value. Inspired by the FGSM (Fast Gradient Sign Method) [54], we adjusted the perturbations based on the same backpropagation gradient in order to maximize the loss. In other words, we considered adding perturbations to the embedding layers of the model along the direction of the negative gradient. The perturbations are expressed as follows:

$$\eta = \tau \cdot sign(\nabla_x Loss(x, y; \psi)), \tag{14}$$

where η is the added perturbation, τ denotes the numerical value of the perturbation, and $sign()$ is a symbolic function.

The above-mentioned adversarial training method generates perturbations for the embedding layer. This method can cooperate with the self-adversarial training method, thereby effectively realizing cooperative adversarial training and improving the anti-interference ability of the model. The final loss can be calculated as follows:

$$\begin{aligned} loss = &\rho \cdot self\_adv\_loss + \omega \cdot FGSM\_self\_adv\_loss \\ &+ \frac{\varepsilon}{|H|} \cdot \sum_{e \in H} (L2(House\_emb(e)) + L2(Comp\_emb(e)) + L2(Phase\_emb(e))), \end{aligned} \tag{15}$$

where $self\_adv\_loss$ is the loss of the triples, $FGSM\_self\_adv\_loss$ is the loss of the triples after adding perturbations, ρ and ω are weights, ε is the regularization coefficient, H is the entity set, $L2$ is the L2 normalization, $House\_emb$ is the householder entity embedding, $Comp\_emb$ is the complex entity embedding, and $Phase\_emb$ is the phase entity embedding.

The selection of perturbations. The selection of perturbations has an important influence on the effect of adversarial training. If the perturbation is too small to produce effective interference, achieving the purpose of adversarial training is difficult. If the perturbation is too large, the model does not encounter similar intensity perturbations during training, and the anti-interference ability of the model is not effectively enhanced. Here, we determine the amplitude of perturbations according to the parameter range of the embedding layer and select the best perturbation within this range. If a fixed perturbation value is used, determining the optimal parameter becomes difficult. Even small perturbations may sometimes cause greater loss, and the perturbations are irregular and diverse in the actual training process. In order to simulate this type of irregularity, we use the UCB multi-armed bandit method to select perturbations and obtain the dynamic balance between loss maximization and perturbation diversity to ensure that the model can resist both large and small perturbations.

The multi-armed bandit method [59] is a classical reinforcement learning algorithm that effectively solves the balance problem of exploration and utilization. Figure 2 shows the basic process of the multi-armed bandit method. The traditional multi-armed bandit method has multiple rocker arms, and the user can obtain certain benefits by shaking each rocker arm. However, as the user does not know the specific values of the rewards, they must estimate the rewards of the rocker arm based on previous experience. Because our goal is to maximize loss, we regard each perturbation value as a rocker arm and we treat the loss generated by each perturbation value as a profit. We record the mean loss produced by each perturbation as the estimated reward, and the user selects the perturbation with the largest estimated reward.

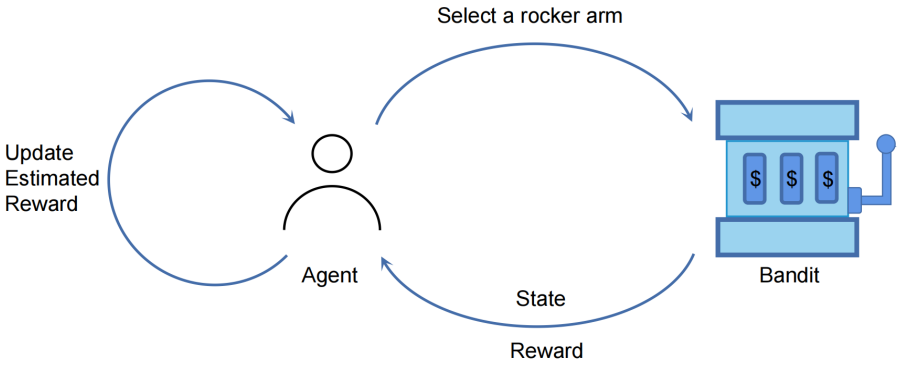

Figure 2. An illustration of the multi-armed bandit method. The agent can sense the state and obtain rewards by shaking the rocker arms. This process updates the estimated reward according to past rewards.

The number of times each rocker arm is used varies, and the more frequently used rocker arm tends to estimate gains more accurately; thus, we need to make each arm application as balanced as possible while ensuring that the maximum loss is selected. Moreover, in the actual training processes the model is subject to a variety of interference types. In order to obtain accurate estimated benefits and simulate the irregularity of perturbations, we use the UCB (upper confidence bound) algorithm to optimize the multi-armed bandit method. We set a factor ε to balance the loss maximization with the disturbance diversity. The balance formula is as follows:

$$Estimated\_Q_i = Avg\_loss_i + \varepsilon \sqrt{\frac{logT}{2(t_i+1)}}, \tag{16}$$

where T is the total number of times all rocker arms are used, t_i denotes the number of times a single rocker arm is used, and Avg_loss_i is the average loss value considering the previous instances. To enhance the exploratory ability of the algorithm and avoid obtaining local optimal solutions, we set a small factor, δ; when the probability is smaller than this factor, we randomly select the perturbation. When the probability is larger than this factor, we select the perturbation according to Formula (16). The perturbation selection process is shown in Algorithm 3.

Algorithm 3: Perturbation selection algorithm

Input: A set of perturbed values S and its set size K
A balance factor ε and a small random factor δ
The total number of times all perturbations are used T
The set of times each perturbation is used $t_i, 1 \leq i \leq K$
Output: Perturbation value P

Initialize $T = 0, t_i = 0, 1 \leq i \leq K$.
$q \leftarrow$ Generate a random number from 0 to 1
if $q < \delta$ **then**
  Random select s_i *in* S
  $index \leftarrow Get\_index(s_i)$
else
  for $s_i \in S$ **do**
    $Estimated\_Q_i = Avg\_loss_i + \varepsilon\sqrt{\frac{logT}{2(t_i+1)}}$
  end
  $index \leftarrow Get\_index(max(Estimated\_Q_i)), 1 \leq i \leq K$
end
$t_{index} \leftarrow t_{index} + 1$
$T \leftarrow T + 1$
$P \leftarrow s_{index}$
return P

4. Experiment

4.1. Cyber Security Knowledge Graph

Because the CVE, CWE, and CAPEC databases are highly standardized and are open sources for the public, they have been widely recognized by cybersecurity researchers. The CVE, CWE, and CAPEC cybersecurity databases each contain a hierarchical structure and sequential structure. These databases focus on the characteristics of cybersecurity data in the cybersecurity space; thus, they are highly representative. The knowledge graph completion model designed according to the characteristics of cybersecurity data can be further extended to other cybersecurity data analysis tasks. Compared with other cybersecurity data sources, these databases have more complete information and accurate ground truth, and these properties are helpful for evaluating the quality of knowledge graph completion models. Low-quality dirty data contain more error information, and they are unsuitable for model evaluation. Therefore, information mining research using the CVE, CWE, and CAPEC databases can help cybersecurity researchers in their studies.

Due to the popularity of computer networks the number of security vulnerabilities is increasing anually, and more software weaknesses are constantly exposed; it has been observed that the attack methods of hackers are gradually becoming more complex and diversified. The CVE, CWE, and CAPEC databases are being constantly updated as well. The completion of knowledge graphs can effectively help researchers to supplement unknown knowledge in attack patterns, vulnerabilities, and software weaknesses. Therefore, in this paper we build knowledge graphs based on CVE, CWE, and CAPEC databases.

The design of ontologies and relationships is of great significance for cybersecurity knowledge graphs. The ontologies and relationships of network security should better reflect the network environment and the technologies used by hackers. Because the CVE, CWE, and CAPEC databases already contain relationships, many previous studies [14,15] have constructed connections for the three data sources stated above; we reuse these ontologies and relationships here to connect multi-source data. At the same time, we select rich relationships and discard relationships with low values and small numbers. Selected entities and relationships in the cybersecurity knowledge graph are shown in Figure 3.

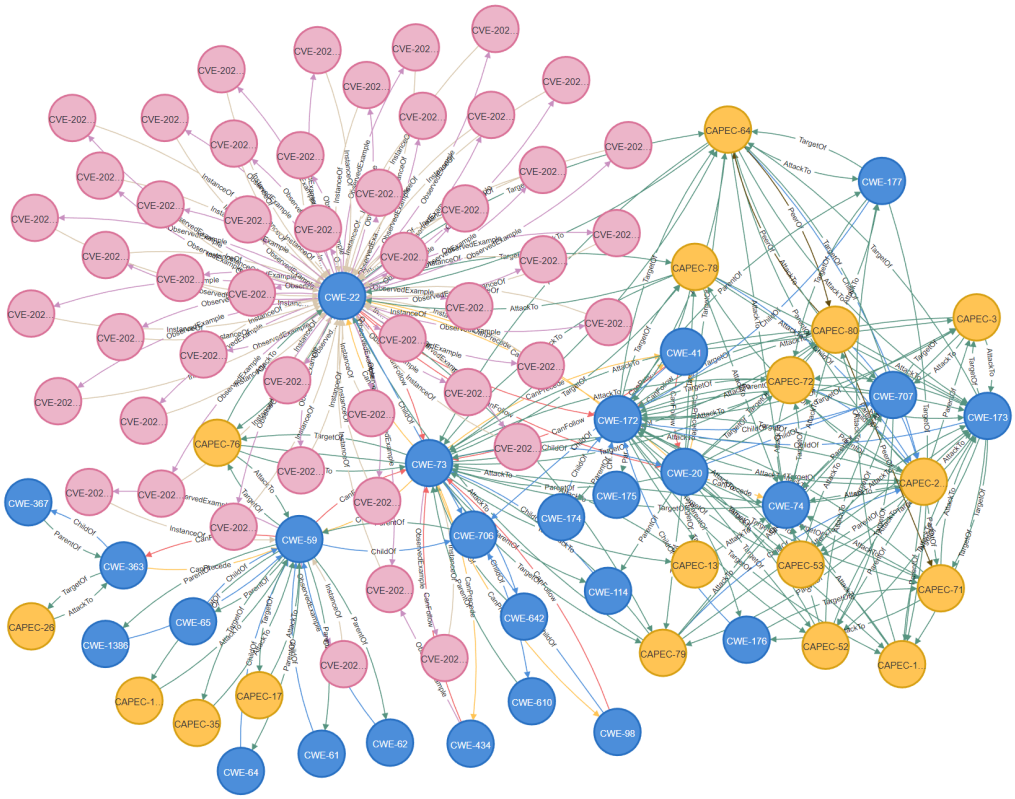

Figure 3. Partial display of the cybersecurity knowledge graph.

Because the number of existing vulnerabilities is much larger than the number of software weaknesses and attack patterns, we only select critical-level CVE data from 2021 in order to keep the number of relationships as balanced as possible. In summary, there are nine relationships and 4095 entities for a final collection of 10,986 triples. The number of triples for each relationship in the knowledge graph is shown in Table 1.

Table 1. Attribute description and quantity statistics of each relationship in the cybersecurity knowledge graph.

Relationships	Head → Tail	Number
InstanceOf	CVE → CWE	2298
ObservedExample	CWE → CVE	2298
PeerOf	CWE → CWE,CAPEC → CAPEC	206
AttackTo	CAPEC → CWE	1145
TargetOf	CWE → CAPEC	1145
CanFollow	CWE → CWE,CAPEC → CAPEC	293
CanPrecede	CWE → CWE,CAPEC → CAPEC	293
Childof	CWE → CWE,CAPEC → CAPEC	1654
ParentOf	CWE → CWE,CAPEC → CAPEC	1654

In order to prevent imbalance in the number of different relationships in the training set, validation set, and test set, we disordered the triples and divided them into the training set, validation set and test set according to a ratio of 7:1:2 relative to each relationship. Table 2 describes the details of the cybersecurity dataset.

Table 2. The quantity statistics of CybersecurityKG; #E and #R represent the number of entities and relations, respectively, while #TR,#VA, and #TE represent the size of the training set, validation set, and test set, respectively.

Dataset	#E	#R	#TR	#VA	#TE
CybersecurityKG	4095	9	7686	1100	2200

4.2. Results

In this section, we measure the performance of CSEA through experiments. We compare CSEA with several recent excellent baselines: TransE is a representative distance-based model; Distmult and ComplEx are excellent semantic matching models.; HAKE possesses good hierarchical modeling abilities; and both Rot-Pro and HousE are excellent models based on projection and rotation.

Evaluation metrics. In the cyber security link prediction task, we use MRR, MR, and Hits@k as the evaluation metrics; MRR is the mean reciprocal rank of the correct entities, MR is the mean rank of the correct entities, andHit@K is the proportion of correct entities ranking in the top K. For Hits@K, we use Hits@1/3/10 to evaluate the performance of the models. A higher MRR, lower MR, and higher Hits@1/3/10 indicate better performances.

Hyperparameters. The hyperparameter settings have an important impact on the applied methods. We control the hyperparameters of each method to ensure the fairness of the experiment. Table 3 shows common experimental hyperparameters.

Table 3. List of hyperparameter values of the models.

Hyperparameter	Value
Max Steps	20,000
Valid Steps	1000
Learning Rate	0.00005
Batch Size	128
γ	6.0

Cybersecurity knowledge graph evaluation. Although our current cybersecurity knowledge graph is relatively complete, it is intended to simulate the phenomenon of having a lack of facts in the field of cybersecurity. We remove head entities or tail entities of triples in the test set to form prediction tasks ($<?, relation, tail>$ and $<head, relation, ?>$). The removed head entities or tail entities can be used as standard answers to test the completion performance of the models for the cybersecurity knowledge graph. Table 4 summarizes the experimental results of our CSEA model compared to the other models.

Table 4. Average evaluation results on CybersecurityKG; results in bold are the best.

Model	MRR	MR	Hits@1	Hits@3	Hits@10
TransE [22]	0.713	308	0.673	0.734	0.777
DistMult [33]	0.690	622	0.639	0.735	0.760
ComplEx [34]	0.735	673	0.720	0.745	0.762
HAKE [30]	0.741	411	0.727	0.749	0.769
Rot-Pro [53]	0.745	377	0.730	0.753	0.770
HousE [31]	0.747	314	0.732	0.753	0.775
CSEA	**0.756**	**148**	**0.735**	**0.764**	**0.793**

As shown in the Table 4, our model achieves good results in all indicators. The above experimental results show that CSEA can complete missing information better than a single model. Missing entities can be found more quickly among individuals with the highest scores. CSEA produces more accurate predictions in practical applications.

In addition, we performed the predict head and predict tail tasks for our model and the other models. Table 5 records the average prediction results of the predict head and predict tail tasks. Compared with the other models, our method achieves similar improvements in the predict head and predict tail tasks.

Table 5. Average evaluation results in predicting head and tail tasks; results in bold are the best.

Prediction	Model	MRR	MR	Hits@1	Hits@3	Hits@10
Head Entity <?, r, t>	TransE [22]	0.713	302	0.674	0.734	0.777
	DistMult [33]	0.690	623	0.638	0.738	0.760
	ComplEx [34]	0.735	670	0.720	0.745	0.760
	HAKE [30]	0.740	400	0.725	0.750	0.769
	Rot-Pro [53]	0.743	373	0.728	0.753	0.769
	HousE [31]	0.747	309	0.731	0.754	0.774
	CSEA	**0.756**	**147**	**0.735**	**0.764**	**0.794**
Tail Entity <h, r, ?>	TransE [22]	0.713	313	0.672	0.733	0.777
	DistMult [33]	0.689	621	0.640	0.732	0.759
	ComplEx [34]	0.735	677	0.719	0.745	0.765
	HAKE [30]	0.742	422	0.729	0.747	0.770
	Rot-Pro [53]	0.746	382	0.731	0.754	0.770
	HousE [31]	0.748	319	0.732	0.753	0.776
	CSEA	**0.756**	**148**	**0.736**	**0.765**	**0.793**

Different models have different sensitivities to each relationship. The experimental performance of the model may be too good for one relationship in the knowledge graph completion task, while the performance of the model may be poor for other relationships, greatly affecting the average prediction results. In this case, the average prediction results are not be sufficiently representative; thus, we carried out knowledge graph completion tasks for each relationship. Table 6 shows the average evaluation results (MRR) for each relationship.

Table 6. Average evaluation results (MRR) for each relationship; results in bold are the best.

	InstanceOf	Observed Example	PeerOf	AttackTo	TargetOf	CanFollow	CanPrecede	ChildOf	ParentOf
TransE [22]	0.682	0.668	0.194	0.820	0.827	0.692	0.811	0.706	0.725
DistMult [33]	0.693	0.683	0.638	0.781	0.802	0.634	0.730	0.606	0.647
ComplEx [34]	0.694	0.683	0.553	0.777	0.794	0.734	0.827	0.752	**0.784**
HAKE [30]	0.689	0.679	**0.655**	0.810	0.821	0.742	0.848	0.753	0.778
Rot-Pro [53]	0.697	0.681	0.624	**0.822**	**0.835**	0.744	**0.860**	0.750	0.775
HousE [31]	0.699	0.688	0.614	**0.822**	0.832	**0.746**	0.856	**0.754**	0.780
CSEA	**0.721**	**0.713**	0.646	0.819	0.830	0.739	0.850	0.751	0.773

From the above experiments, it can be seen that the CSEA model achieves good results in all tasks, indicating that it can help greatly in expressing and inferring knowledge. The increase in MRR indicates that the knowledge expression learned by CSEA is more universal. The increase in Hits@K indicates that the CSEA model has a higher recommendation quality for knowledge graph completion tasks. Rot-Pro and HousE are effective models based on projections and rotations; Rot-Pro mainly implements projection and rotation in complex spaces, while HousE does so in higher dimension spaces. However, the experimental results show that there are limitations to a single operation. Our CSEA ensemble model can make more comprehensive use of information. For the overall task, CSEA adopts ensemble learning to further reduce the fluctuations of information, achieving good average results. In the relationship evaluation task, certain entity types appear more frequently in the cybersecurity knowledge graph, meaning that the relationship types

connected with the above entities appear more frequently as well. It can be observed that when there are substantial amounts of relational data, many single models have a propensity to fall into a local optimal solution; our results show that CSEA is better able to avoid falling into a local optimal solution in scenarios involving large-scale data. Even with insufficient amounts of data, CSEA is able to maintain its overall performance via ensemble Learning.

5. Conclusions

In this paper, we design a cybersecurity knowledge graph completion model, which we call CSEA. The model uses multiple projections to handle sophisticated relation-mapping properties, and it has a superior capacity for modeling relation patterns based on rotations. Angle information enhances the ability of CSEA to distinguish entities. In the training process, the cooperative adversarial training method enhances the generalization and robustness of the model. Our experiments prove that the model can be used for knowledge graph completion. The CVE, CWE, and CAPEC databases we used can collectively reflect the characteristics of security data in cyberspace, and have strong representativeness.

Currently, the cybersecurity knowledge graph designed by our model focuses on the relationship between vulnerabilities, software weaknesses, and attack patterns. However, certain attributes of vulnerabilities, software weaknesses, and attack patterns are incomplete. We can rely on named entity recognition technologies to extract relevant attributes from the description text of CVE, CWE, and CAPEC. This method can enrich the types of relationships and improve the quality of cybersecurity knowledge graphs. In addition, this model can be migrated to other cybersecurity data analysis tasks, potentially playing an important role in threat intelligence and other fields.

Author Contributions: Conceptualization, P.W. and J.L.; methodology, P.W.; software, P.W.; validation, P.W., J.L. and S.Z.; formal analysis, D.H. and S.Z.; investigation, J.L.; resources, P.W.; data curation, P.W.; writing—original draft preparation, P.W.; writing—review and editing, P.W., J.L., D.H. and S.Z.; visualization, D.H. and S.Z.; supervision, J.L.; project administration, P.W. and J.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Endsley, M.R. *Toward a Theory of Situation Awareness in Dynamic Systems*; Human Factors and Ergonomics Society: Santa Monica, CA, USA, 1995.
2. Özcan, A.; Çatal, Ç.; Togay, C.; Tekinerdogan, B.; Dönmez, E. Assessment of environmental factors affecting software reliability: A survey study. *Turk. J. Electr. Eng. Comput. Sci.* **2020**, *28*, 1841–1858. [CrossRef]
3. Gao, M.; Lu, J.; Chen, F. Medical Knowledge Graph Completion Based on Word Embeddings. *Information* **2022**, *13*, 205. [CrossRef]
4. Wang, W.; Xu, Y.; Du, C.; Chen, Y.; Wang, Y.; Wen, H. Data Set and Evaluation of Automated Construction of Financial Knowledge Graph. *Data Intell.* **2021**, *3*, 418–443. [CrossRef]
5. Tan, J.; Qiu, Q.; Guo, W.; Li, T. Research on the Construction of a Knowledge Graph and Knowledge Reasoning Model in the Field of Urban Traffic. *Sustainability* **2021**, *13*, 3191. [CrossRef]
6. Chen, J.; Yang, Y.; Peng, L.; Chen, L.; Ge, X. Knowledge Graph Representation Learning-Based Forest Fire Prediction. *Remote Sens.* **2022**, *14*, 4391. [CrossRef]
7. Liu, P.; Tian, B.; Liu, X.; Gu, S.; Yan, L.; Bullock, L.; Ma, C.; Liu, Y.; Zhang, W. Construction of Power Fault Knowledge Graph Based on Deep Learning. *Appl. Sci.* **2022**, *12*, 6993. [CrossRef]
8. Jin, Y.; Liu, J.; Wang, X.; Li, P.; Wang, J. Technology Recommendations for an Innovative Agricultural Robot Design Based on Technology Knowledge Graphs. *Processes* **2021**, *9*, 1905. [CrossRef]
9. Jiang, S.; Liu, Y.; Zhang, Y.; Luo, P.; Cao, K.; Xiong, J.; Zhao, H.; Wei, J. Reliable Semantic Communication System Enabled by Knowledge Graph. *Entropy* **2022**, *24*, 846. [CrossRef]

10. Liu, K.; Wang, F.; Ding, Z.; Liang, S.; Yu, Z.; Zhou, Y. Recent Progress of Using Knowledge Graph for Cybersecurity. *Electronics* **2022**, *11*, 2287. [CrossRef]
11. Catal, C.; Ozcan, A.; Donmez, E.; Kasif, A. Analysis of cyber security knowledge gaps based on cyber security body of knowledge. *Educ. Inf. Technol.* **2022**. [CrossRef]
12. Li, X.; Chen, J.; Lin, Z.; Zhang, L.; Wang, Z.; Zhou, M.; Xie, W. A Mining Approach to Obtain the Software Vulnerability Characteristics. In Proceedings of the Fifth International Conference on Advanced Cloud and Big Data, Shanghai, China, 13–16 August 2017; pp. 296–301. [CrossRef]
13. Han, Z.; Li, X.; Liu, H.; Xing, Z.; Feng, Z. DeepWeak: Reasoning common software weaknesses via knowledge graph embedding. In Proceedings of the 25th International Conference on Software Analysis, Evolution and Reengineering, SANER 2018, Campobasso, Italy, 20–23 March 2018; pp. 456–466. [CrossRef]
14. Xiao, H.; Xing, Z.; Li, X.; Guo, H. Embedding and Predicting Software Security Entity Relationships: A Knowledge Graph Based Approach. In Proceedings of the Neural Information Processing—26th International Conference, ICONIP 2019, Sydney, NSW, Australia, 12–15 December 2019; pp. 50–63. [CrossRef]
15. Yuan, L.; Bai, Y.; Xing, Z.; Chen, S.; Li, X.; Deng, Z. Predicting Entity Relations across Different Security Databases by Using Graph Attention Network. In Proceedings of the IEEE 45th Annual Computers, Software, and Applications Conference, COMPSAC 2021, Madrid, Spain, 12–16 July 2021; pp. 834–843. [CrossRef]
16. Zang, Y.; Hu, T.; Zhou, T.; Deng, W. An Automated Penetration Semantic Knowledge Mining Algorithm Based on Bayesian Inference. *Comput. Mater. Contin.* **2021**, *66*, 2573–2585. [CrossRef]
17. Bridges, R.A.; Jones, C.L.; Iannacone, M.D.; Goodall, J.R. Automatic Labeling for Entity Extraction in Cyber Security. *arXiv* **2013**, arXiv:1308.4941.
18. Satyapanich, T.; Ferraro, F.; Finin, T. CASIE: Extracting Cybersecurity Event Information from Text. In Proceedings of the Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, New York, NY, USA, 7–12 February 2020; pp. 8749–8757.
19. Liu, P.; Li, H.; Wang, Z.; Liu, J.; Ren, Y.; Zhu, H. Multi-features based Semantic Augmentation Networks for Named Entity Recognition in Threat Intelligence. *arXiv* **2022**, arXiv:2207.00232.
20. Rossi, A.; Firmani, D.; Matinata, A.; Merialdo, P.; Barbosa, D. Knowledge Graph Embedding for Link Prediction: A Comparative Analysis. *arXiv* **2020**, arXiv:2002.00819.
21. Ott, S.; Meilicke, C.; Samwald, M. SAFRAN: An interpretable, rule-based link prediction method outperforming embedding models. In Proceedings of the 3rd Conference on Automated Knowledge Base Construction, AKBC 2021, Virtual, 4–8 October 2021. [CrossRef]
22. Bordes, A.; Usunier, N.; García-Durán, A.; Weston, J.; Yakhnenko, O. Translating Embeddings for Modeling Multi-relational Data. In Proceedings of the Advances in Neural Information Processing Systems, Lake Tahoe, NV, USA, 5–8 December 2013; pp. 2787–2795.
23. Wang, Z.; Zhang, J.; Feng, J.; Chen, Z. Knowledge Graph Embedding by Translating on Hyperplanes. In Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence, Québec City, QC, Canada, 27–31 July 2014; pp. 1112–1119.
24. Lin, Y.; Liu, Z.; Sun, M.; Liu, Y.; Zhu, X. Learning Entity and Relation Embeddings for Knowledge Graph Completion. In Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, Austin, TX, USA, 25–30 January 2015; pp. 2181–2187.
25. Sun, Z.; Deng, Z.H.; Nie, J.Y.; Tang, J. RotatE: Knowledge Graph Embedding by Relational Rotation in Complex Space. *arXiv* **2019**, arXiv:1902.10197.
26. Li, X.; Wang, Z.; Zhang, Z. Complex Embedding with Type Constraints for Link Prediction. *Entropy* **2022**, *24*, 330. [CrossRef]
27. Zhang, S.; Tay, Y.; Yao, L.; Liu, Q. Quaternion Knowledge Graph Embeddings. In Proceedings of the Advances in Neural Information Processing Systems, NeurIPS 2019, Vancouver, BC, Canada, 8–14 December 2019; pp. 2731–2741.
28. Gao, L.; Zhu, H.; Zhuo, H.H.; Xu, J. Dual Quaternion Embeddings for Link Prediction. *Appl. Sci.* **2021**, *11*, 5572. [CrossRef]
29. Yu, M.; Bai, C.; Yu, J.; Zhao, M.; Xu, T.; Liu, H.; Li, X.; Yu, R. Translation-Based Embeddings with Octonion for Knowledge Graph Completion. *Appl. Sci.* **2022**, *12*, 3935. [CrossRef]
30. Zhang, Z.; Cai, J.; Zhang, Y.; Wang, J. Learning Hierarchy-Aware Knowledge Graph Embeddings for Link Prediction. In Proceedings of the Thirty-Fourth AAAI Conference on Artificial Intelligence 2020, New York, NY, USA, 7–12 February 2020; pp. 3065–3072.
31. Li, R.; Zhao, J.; Li, C.; He, D.; Wang, Y.; Liu, Y.; Sun, H.; Wang, S.; Deng, W.; Shen, Y.; et al. HousE: Knowledge Graph Embedding with Householder Parameterization. In Proceedings of the International Conference on Machine Learning, ICML 2022, Baltimore, MD, USA, 17–23 July 2022; Volume 162, pp. 13209–13224.
32. Nickel, M.; Tresp, V.; Kriegel, H. A Three-Way Model for Collective Learning on Multi-Relational Data. In Proceedings of the 28th International Conference on Machine Learning, ICML 2011, Bellevue, WA, USA, 28 June–2 July 2011; pp. 809–816.
33. Yang, B.; Yih, W.; 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, ICLR 2015, San Diego, CA, USA, 7–9 May 2015.
34. Trouillon, T.; Welbl, J.; Riedel, S.; Gaussier, É.; Bouchard, G. Complex Embeddings for Simple Link Prediction. In Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York, NY, USA, 19–24 June 2016; Volume 48, pp. 2071–2080.

35. Chen, Y.; Minervini, P.; Riedel, S.; Stenetorp, P. Relation Prediction as an Auxiliary Training Objective for Improving Multi-Relational Graph Representations. In Proceedings of the 3rd Conference on Automated Knowledge Base Construction, AKBC 2021, Virtual, 4–8 October 2021. [CrossRef]
36. Dettmers, T.; Minervini, P.; Stenetorp, P.; Riedel, S. Convolutional 2D Knowledge Graph Embeddings. In Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), New Orleans, LA, USA, 2–7 February 2018; pp. 1811–1818.
37. Nguyen, D.Q.; Nguyen, T.D.; Nguyen, D.Q.; Phung, D.Q. A Novel Embedding Model for Knowledge Base Completion Based on Convolutional Neural Network. In Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT, New Orleans, LA, USA, 1–6 June 2018; pp. 327–333. [CrossRef]
38. Jiang, X.; Wang, Q.; Wang, B. Adaptive Convolution for Multi-Relational Learning. In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, 2–7 June 2019; pp. 978–987. [CrossRef]
39. Nguyen, D.Q.; Nguyen, T.; Phung, D. A Relational Memory-based Embedding Model for Triple Classification and Search Personalization. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, 5–10 July 2020; pp. 3429–3435. [CrossRef]
40. Schlichtkrull, M.S.; Kipf, T.N.; Bloem, P.; van den Berg, R.; Titov, I.; Welling, M. Modeling Relational Data with Graph Convolutional Networks. In Proceedings of the The Semantic Web–15th International Conference, ESWC 2018, Heraklion, Greece, 3–7 June 2018; Volume 10843, pp. 593–607. [CrossRef]
41. Shang, C.; Tang, Y.; Huang, J.; Bi, J.; He, X.; Zhou, B. End-to-End Structure-Aware Convolutional Networks for Knowledge Base Completion. In Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, Honolulu, HI, USA, 27 January–1 February 2019; pp. 3060–3067. [CrossRef]
42. Nathani, D.; Chauhan, J.; Sharma, C.; Kaul, M. Learning Attention-based Embeddings for Relation Prediction in Knowledge Graphs. In Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, 28 July–2 August 2019; pp. 4710–4723. [CrossRef]
43. Wang, R.; Li, B.; Hu, S.; Du, W.; Zhang, M. Knowledge Graph Embedding via Graph Attenuated Attention Networks. *IEEE Access* **2020**, *8*, 5212–5224. [CrossRef]
44. Wang, L.; Zhao, W.; Wei, Z.; Liu, J. SimKGC: Simple Contrastive Knowledge Graph Completion with Pre-trained Language Models. *arXiv* **2022**, arXiv:2203.02167.
45. Shen, J.; Wang, C.; Gong, L.; Song, D. Joint Language Semantic and Structure Embedding for Knowledge Graph Completion. In Proceedings of the Proceedings of the 29th International Conference on Computational Linguistics, COLING 2022, Gyeongju, Republic of Korea, 12–17 October 2022; pp. 1965–1978.
46. Wang, B.; Shen, T.; Long, G.; Zhou, T.; Wang, Y.; Chang, Y. Structure-Augmented Text Representation Learning for Efficient Knowledge Graph Completion. In Proceedings of the WWW '21: The Web Conference 2021, Ljubljana, Slovenia, 19–23 April 2021; pp. 1737–1748. [CrossRef]
47. Clouâtre, L.; Trempe, P.; Zouaq, A.; Chandar, S. MLMLM: Link Prediction with Mean Likelihood Masked Language Model. In Proceedings of the Findings of the Association for Computational Linguistics: ACL/IJCNLP 2021, Online Event, 1–6 August 2021; Volume ACL/IJCNLP, pp. 4321–4331. [CrossRef]
48. Mienye, I.D.; Sun, Y. A Survey of Ensemble Learning: Concepts, Algorithms, Applications, and Prospects. *IEEE Access* **2022**, *10*, 99129–99149. [CrossRef]
49. Dietterich, T.G. Ensemble Methods in Machine Learning. In *Multiple Classifier Systems*; Springer: Berlin/Heidelberg, Germany, 2000; pp. 1–15.
50. Qian, Z.; Huang, K.; Wang, Q.; Zhang, X. A survey of robust adversarial training in pattern recognition: Fundamental, theory, and methodologies. *Pattern Recognit.* **2022**, *131*, 108889. [CrossRef]
51. Zhao, W.; Alwidian, S.; Mahmoud, Q.H. Adversarial Training Methods for Deep Learning: A Systematic Review. *Algorithms* **2022**, *15*, 283. . [CrossRef]
52. Zamini, M.; Reza, H.; Rabiei, M. A Review of Knowledge Graph Completion. *Information* **2022**, *13*, 396. [CrossRef]
53. Song, T.; Luo, J.; Huang, L. Rot-Pro: Modeling Transitivity by Projection in Knowledge Graph Embedding. In Proceedings of the Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, Virtual, 6–14 December 2021; pp. 24695–24706.
54. Goodfellow, I.J.; Shlens, J.; Szegedy, C. Explaining and Harnessing Adversarial Examples. In Proceedings of the 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, 7–9 May 2015.
55. Miyato, T.; Dai, A.M.; Goodfellow, I.J. Adversarial Training Methods for Semi-Supervised Text Classification. In Proceedings of the 5th International Conference on Learning Representations, ICLR 2017, Toulon, France, 24–26 April 2017.
56. Shafahi, A.; Najibi, M.; Ghiasi, A.; Xu, Z.; Dickerson, J.P.; Studer, C.; Davis, L.S.; Taylor, G.; Goldstein, T. Adversarial training for free! In Proceedings of the Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, Vancouver, BC, Canada, 8–14 December 2019; pp. 3353–3364.
57. Zhu, C.; Cheng, Y.; Gan, Z.; Sun, S.; Goldstein, T.; Liu, J. FreeLB: Enhanced Adversarial Training for Natural Language Understanding. In Proceedings of the 8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, 26–30 April 2020.

58. Madry, A.; Makelov, A.; Schmidt, L.; Tsipras, D.; Vladu, A. Towards Deep Learning Models Resistant to Adversarial Attacks. In Proceedings of the 6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, 30 April–3 May 2018.
59. Vermorel, J.; Mohri, M. Multi-armed Bandit Algorithms and Empirical Evaluation. In Proceedings of the Machine Learning: ECML 2005, 16th European Conference on Machine Learning, Porto, Portugal, 3–7 October 2005; Volume 3720, pp. 437–448. [CrossRef]