


Article

A Cybersecurity Knowledge Graph Completion Method for Penetration Testing

Peng Wang^{1,2}, Jingju Liu^{1,2,*}, Xiaofeng Zhong^{1,2} and Shicheng Zhou^{1,2} ¹ College of Electronic Engineering, National University of Defense Technology, Hefei 230037, China² Anhui Province Key Laboratory of Cyberspace Security Situation Awareness and Evaluation, Hefei 230037, China

* Correspondence: jingjul@aliyun.com

Abstract: Penetration testing is an effective method of making computers secure. When conducting penetration testing, it is necessary to fully understand the various elements in the cyberspace. Prediction of future cyberspace state through perception and understanding of cyberspace can assist defenders in decision-making and action execution. Accurate cyberspace detection information is the key to ensuring successful penetration testing. However, cyberspace situation awareness still faces the following challenges. Due to the limited detection capability, the information obtained from cyberspace detection intelligence is incomplete. There are some errors in the cyberspace detection intelligence, which may mislead the penetration testing workers. The knowledge graph can store and manage the cybersecurity data. In order to ensure the integrity and accuracy of cyberspace information, we design a knowledge graph completion model called CSNT to complete cybersecurity data. CSNT uses the BiLSTM to capture the interaction information between entities and relationships. It models the relationship between entities by combining the neural network and tensor decomposition. The Pearson Mix Net is designed to control the generation of joint vectors. We also design a novel self-distillation strategy to reduce catastrophic forgetting during model training. After learning the relationship pattern between entities in the cyberspace detection intelligence, the model can be used to mine the knowledge not found in the cybersecurity detection intelligence and correct the erroneous records. Experiments show that our method has certain advantages for the knowledge graph completion.



Citation: Wang, P.; Liu, J.; Zhong, X.; Zhou, S. A Cybersecurity Knowledge Graph Completion Method for Penetration Testing. *Electronics* **2023**, *12*, 1837. <https://doi.org/10.3390/electronics12081837>

Academic Editor: Cheng-Chi Lee

Received: 28 March 2023

Revised: 10 April 2023

Accepted: 10 April 2023

Published: 12 April 2023



Copyright: © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

Keywords: penetration testing; cyberspace situation awareness; cybersecurity; knowledge graph completion

1. Introduction

With the continuous popularity of computer networks, people have become extremely dependent on them in their work and life. However, the cybersecurity situation is not optimistic. Cybersecurity accidents occur frequently all over the world and have caused enormous losses to individuals, enterprises and governments. Currently, the means of hacker attack are increasing, which poses a great challenge for maintaining cybersecurity. Penetration testing can evaluate computer system security by simulating a malicious hacker's attack, which enables enterprises and others to patch computer systems in a timely manner. In the process of penetration testing, more valuable information is needed, mainly divided into the following types. Firstly, there is target cyberspace state information, including IP address, open ports, services, domain name system, vulnerabilities, etc. The above elements can better describe the current cyberspace state, so that penetration testing workers can fully understand the current environment. Secondly, vulnerability utilization tool information can enable penetration testing workers to grasp the information of tool modules that can be used directly, such as Exploit and Auxiliary modules which are currently available. Finally, cybersecurity knowledge information is also very

important. Because of the fast update of cybersecurity knowledge, penetration testing workers may not be able to grasp the latest cybersecurity knowledge in time. However, as people's experience in dealing with cybersecurity challenges accumulates, many previous experiences can provide guidance for penetration testing workers, such as CWE (common weakness enumeration) which stores existing knowledge of software and hardware defects. CAPEC (common attack pattern enumerations and classifications) stores common hacker attack patterns.

Cyberspace situation awareness [1] is proposed to perceive and understand the elements in the current network environment. It can support the development of defense strategies and provide effective information support for penetration testing. In order to make full use of all kinds of information in cyberspace to resolve cybersecurity risks, the cybersecurity knowledge graph has received more attention. Since the concept of a knowledge graph was proposed by Google, many researchers have studied it in depth. The knowledge graph has been widely used in medical [2], financial [3], education [4] and other fields [5–7]. It has a profound impact. Applying a knowledge graph to the field of cybersecurity can enable one to effectively manage the data in the cyberspace and discover knowledge from it. This will better realize situation awareness in the cyberspace and help people achieve cognitive intelligence in the field of cybersecurity more quickly.

Cyberspace detection intelligence is of great importance to defenders, but the current network environment is complex and changeable, the amount of cybersecurity data is large and the value density is low, and people have limited ability to detect the network environment. The information obtained from cyberspace detection intelligence is incomplete and may have errors, which makes it difficult to achieve cyberspace situation awareness and make defense strategies.

To solve these problems, we propose a knowledge graph completion method. This method can mine implicit relationships from existing cybersecurity data and it can be better used to complete and correct cybersecurity intelligence information, making the intelligence information more accurate and complete.

In conclusion, our contributions are mainly as follows:

- We design a knowledge graph completion model called CSNT. It uses recurrent neural network to enhance interaction. It models entities and relationships in cyberspace based on neural networks and tensor decomposition. At the same time, it uses the Pearson correlation coefficient between them to design Pearson Mix Net to obtain joint vectors.
- We design the Progressive-Replay-SA self-distillation strategy for model training. This strategy adopts the methods of sample replay and progressive learning to solve the catastrophic forgetting problem and prevent model degradation. At the same time, the simulated annealing algorithm is used to adaptively adjust the distillation temperature to gradually increase the difficulty of the course learning.
- We use the real cybersecurity data to build the cybersecurity knowledge graph to provide support for the penetration testing. We carry out the completion experiment based on the cybersecurity knowledge graph. The experiment shows that our model has a good effect on the cybersecurity knowledge graph completion and can be better used to assist the penetration testing.

The rest of this article is as follows. Section 2 introduces the related research work on cybersecurity data and knowledge graph completion. Section 3 describes in detail how our method is used in the cybersecurity knowledge graph completion. In Section 4, we describe the cybersecurity knowledge graph that we have built. We have completed the experiment and compared it with other popular knowledge graph completion methods. Finally, the conclusions are given in Section 5.

2. Related Work

2.1. Research on Cybersecurity Data

In order to formulate a reasonable cybersecurity defense strategy, it is necessary to have a full understanding of the cyberspace environment. Cyberspace detection technology can effectively understand various elements such as asset information in cyberspace and help people perceive the current cyberspace situation.

Cyberspace mapping effectively supports cyberspace situational awareness. It uses digital communication, cybersecurity and other technologies to digitally map the current network environment and provides corresponding detection intelligence for defenders. In order to fully understand the cyberspace environment, people have carried out in-depth exploration and research on the cyberspace detection technology. Unlike traditional geographic information detection, the detection of cyberspace has both physical and virtual resources. At the same time, the cyberspace environment has more changes and higher uncertainty. At present, there are many tools for cyberspace detection, such as Nmap, Zenmap, ZMap, IPSonar, SolarWinds, etc. In the process of penetration testing, workers often use the above tools to collect information in cyberspace and prepare for the next step. The full name of Nmap is Network Mapper. It can detect whether the host in the network is online and obtain the open port and services of the host. Zoomeye is an efficient cyberspace search engine, which can detect and identify devices and websites in cyberspace and effectively help users achieve cyberspace mapping. By obtaining basic information such as topological connections in cyberspace, penetration testing workers can be helped to realize a good perception of the current cyberspace situation.

In addition to various elements in the current cyberspace, basic cybersecurity knowledge is also indispensable for penetration testing. Especially with the increasing trend of intelligence and automation, mastering certain cybersecurity knowledge not only helps penetration testing workers to formulate good defense strategies, but also is the basis for intelligent penetration testing. In the process of dealing with the challenges of cybersecurity, people have accumulated a lot of valuable experience and knowledge. At present, the well known cybersecurity knowledge databases include CVE (Common Vulnerability and Exposure), CWE (Common Weakness Enumeration) and CAPEC (Common Attachment Pattern Enumeration and Classifications), maintained by MITRE. CVE covers a lot of cybersecurity vulnerability information, CWE includes a lot of software and hardware defect information, CAPEC lists common hacker attack methods, etc. Combining the above cybersecurity knowledge and the current cyberspace situation, people can effectively formulate corresponding defense strategies. For example, people can know what vulnerabilities affect current assets. Moreover, people can know how an attack might occur when a system has a defect.

2.2. Research on Knowledge Graph Completion

A knowledge graph is a semantic network connecting things in the real world. However, most of the knowledge graphs are incomplete and even some information is wrong. In order to improve the quality of the knowledge graph, the knowledge graph completion algorithm can be used to complete and correct the knowledge graph. The knowledge graph completion algorithm can be divided into algorithms based on distance model, semantic matching model and neural network model.

Most distance-based models map entities and relationships in the knowledge graph to low-dimensional vectors and then calculate the relationship between vectors. TransE [8] is a classical algorithm in the distance model. Through the translation between vectors, the relationship vector becomes the transfer vector between the head entity vector and the tail entity vector. Its corresponding scoring function is $h + r = t$, where h is the head entity, r is the relationship and t is the tail entity. This method has low computational complexity and fewer parameters. It can give good consideration to efficiency and results. In order to deal with more complex relationship types, TransH [9] projects vectors into the hyperplane to realize the transformation between entities. TransR [10] maps entities

to different relational semantic spaces through matrices. TransD [11] uses the dynamic mapping matrix to complete the knowledge graph and further reduces the amount of calculation. RotatE [12] realizes the transformation from head entity to tail entity through rotation and also uses self-adversarial training to improve the experimental performance. Many knowledge graph completion methods also extend the representation of entities and relationships to other spaces, such as quaternion space [13], octonion space [14], hyperbolic space [15], etc.

The distance model still has great shortcomings in capturing the potential semantic relationship between entities, so many researchers have designed knowledge graph completion methods based on the semantic matching model. Semantic matching models mainly rely on tensor decomposition to capture the internal interaction of triples and realize the mining of potential semantic relationships. RESCAL [16] uses the scoring function $h^T M_r t$ to evaluate whether the triple is true. The matrix M_r represents the relationship, vector h represents the head entity and t represents the tail entity. DistMult [17] limits the relational matrix to diagonal matrix on the basis of RESCAL, which simplifies the calculation and reduces the operation cost. ComplEx [18] extends DistMult to the complex space and can model more abundant relationship types. TuckER [19] uses Tucker decomposition to achieve knowledge graph completion, which can better mine the potential semantic relationships in the knowledge graph.

Neural network has developed rapidly in recent years and has made great achievements in image processing and natural language processing. Many researchers use neural networks to complete the knowledge graph. ConvE [20] uses the convolution neural network to extract the entity and relationship features. It takes the feature matrix obtained as the input of the convolution layer and generates the evaluation score for each triple by the inner product of the output and all object entity vectors. InteractE [21] performs feature replacement, reshape operation and circular convolution on the basis of convolution operation, which promotes the interactive integration of entities and relationships, but also consumes a huge amount of computation. ParamE [22] regards head entity embeddings, relation embeddings and tail entity embeddings as the input, parameters and output of a neural network, respectively. This makes ParamE much more expressive. This method has achieved good experimental results.

In addition to innovation in the scoring function, many model training methods have also been used to improve performance. Knowledge distillation is considered as an effective way to improve learning efficiency and it has been widely used in the field of deep learning. Hinton et al. [23] proposed the teacher–student structure and transferred the knowledge trained by the teacher network to the student network. It improves the effectiveness of neural network training. In order to reduce the amount of additional network computation, the self-distillation strategy has received more attention. Kyungyu Kim et al. [24] proposed a simple and effective method of knowledge distillation, which transfers the knowledge from the previous round to the current round, effectively improving the generalization ability. Linfeng Zhang et al. [25] performed knowledge distillation based on the neural network structure and they used the deep part of the network as a teacher to perform distillation learning on the shallow part of this network. Yiqing Shen et al. [26] proposed the DLB self-distillation strategy, this method provides soft targets to realize knowledge distillation in the next round. It extracts smooth labels from the previous round, which consumes less computation.

3. Our Method

3.1. Problem Description

In order to enhance the capability of situation awareness in cyberspace, people need to detect the current network environment. Due to the limited detection technology of the existing tools and people's cognitive ability, the information obtained by the detection is often incomplete or even wrong. We introduce knowledge graph technology into the cybersecurity field to construct the cybersecurity knowledge graph based on cybersecurity data. The basic process of knowledge graph completion is shown in Figure 1.

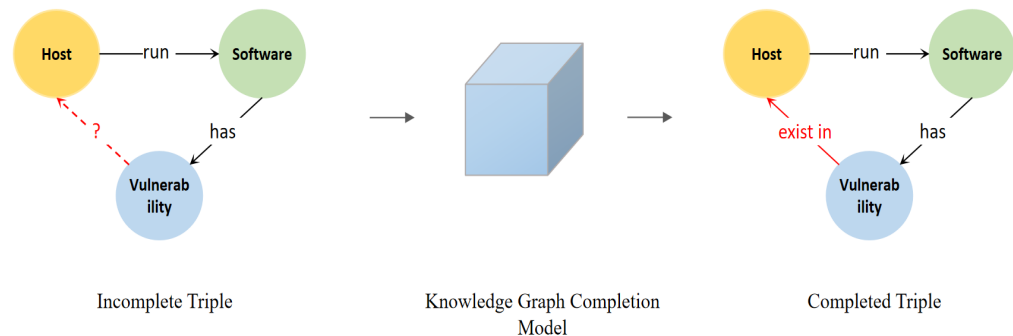


Figure 1. The basic process of cybersecurity knowledge graph completion.

For a knowledge graph $G = (H, R)$, where H is the set of entities and R is the set of relations. The acquired intelligence knowledge is stored in the knowledge graph in the form of triples (h, r, t) , where h and t belong to H and r belongs to R . When a triple is incomplete, such as $(?, r, t)$ or $(h, r, ?)$, we expect it to be completed using knowledge graph completion algorithms. When there is an error in the knowledge graph, the knowledge graph completion algorithm detects the conflict in the stored knowledge by learning the existing relational schema to achieve the purpose of error correction. Therefore, we design CSNT, a Cyber Security knowledge graph completion model based on Neural network and Tensor decomposition.

3.2. Problem Modeling

3.2.1. Knowledge Graph Completion Model

For triples in the knowledge graph, we expect an accurate embedded representation of entities and relationships. The head entity vectors generate the output vectors through the transformation of relations. Then it generates the evaluation score for each triple by the inner product of the output vectors and all object entity vectors.

Most existing studies usually represent head entities and relations independently before relation transformation. However, we find that head entities can better interact with information in relations for relation transformation. Some previous works, such as InteractE [21], consumed too many computing resources to capture the interaction information between entities and relations. However, we found that simple recurrent neural networks can effectively facilitate the information interaction. LSTM (long short-term memory) [27] effectively controls the flow of information through the gate structure, which has certain advantages for modeling context information. BiLSTM (Bi-directional Long Short-Term Memory) enhances context information interaction in two directions based on LSTM. Therefore, we first use BiLSTM to carry out interactions between entities and relationships and capture more interaction information for subsequent modeling. The basic process of information interaction is shown in Figure 2.

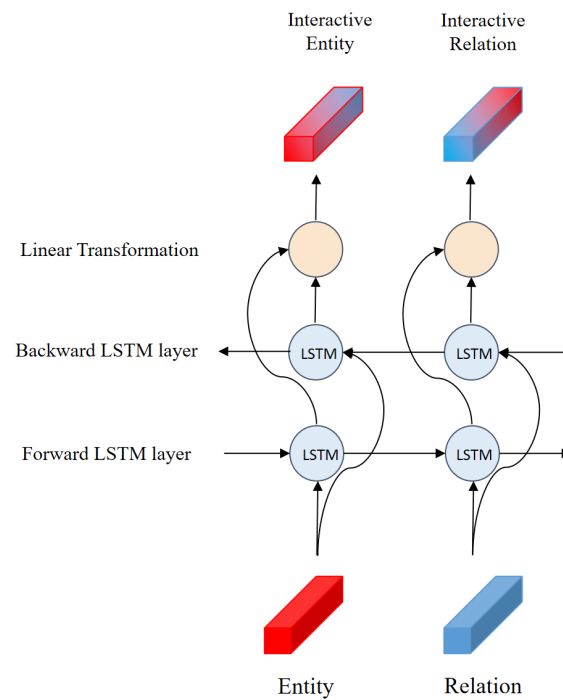


Figure 2. BiLSTM is used to realize the information interaction between entity and relation.

The relationships in the cybersecurity knowledge graph are complex. In order to further mine the implicit semantic relationships in the knowledge graph, we use tensor decomposition and neural network to jointly mine the relationships. The interactive head entity and relation encoding are combined into a 3D tensor. By first decomposing the 3D tensor, it is expected that the trained representation of entities and relations can lead to higher scores for true triples. Inspired by DistMult, we use the form of multiplying head entities with relational matrices to generate vectors.

In addition to tensor decomposition of the head entity and relationship after interaction, we also input it into the neural network model. With the strong self-learning and inductive ability of the neural network model, it can better approximate the complex non-linear relationship. Figure 3 shows the process of generating the vector representation by the neural network.

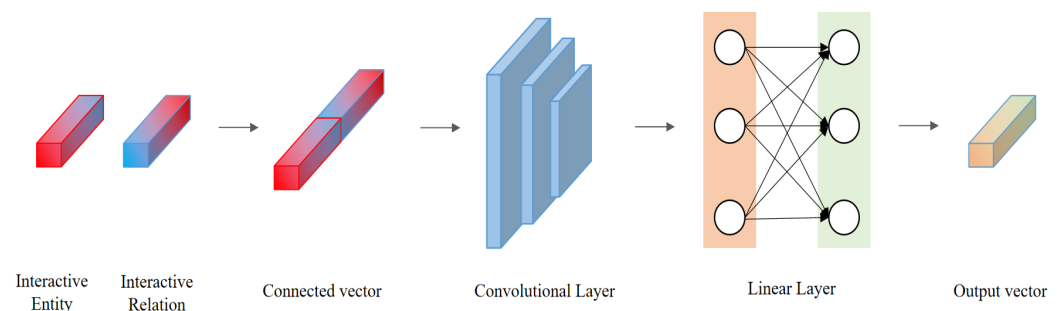


Figure 3. The feature vector is obtained via convolutional layers and linear layers.

In order to promote effective feature fusion, inspired by multi-agent reinforcement learning, QMIX [28] was successfully applied to the multi-agent reinforcement learning algorithm, which effectively solves the reliability allocation problem between agents. In a similar way, we introduce the mix net into the process of entity and relationship feature fusion. We combine and stitch the feature vectors of each module and map the mix net to obtain the embedded representation of the end-approaching entity. To further explore the deeper meaning of the relationship, we designed Pearson Mix Net to promote the

information fusion between the two modules. First, Pearson correlation coefficients of the above vectors are calculated. The Pearson correlation coefficients are calculated as follows.

$$corr(P, Q) = \frac{Cov(P, Q)}{\sigma_P \sigma_Q} = \frac{\sum_{i=1}^n (P_i - P_{mean})(Q_i - Q_{mean})}{\sqrt{\sum_{i=1}^n (P_i - P_{mean})^2} \sqrt{\sum_{i=1}^n (Q_i - Q_{mean})^2}}, \quad (1)$$

where P and Q represent the output vectors of the two modules, $Cov(P, Q)$ is the covariance between P and Q , σ_P and σ_Q are the variances. P_{mean} and Q_{mean} are two mean vectors. After the Pearson correlation coefficient is obtained, we input Pearson correlation coefficients into a parameter generator, which is constantly optimized during training. The parameter generator is a shallow convolutional neural network. The Pearson correlation coefficient is used to control the parameter weights of the Pearson Mix Net. The joint vector can better fuse the characteristics of the two modules, which further promotes the information interaction between the neural network and the tensor decomposition. The basic structure of Pearson Mix Net is shown in Figure 4.

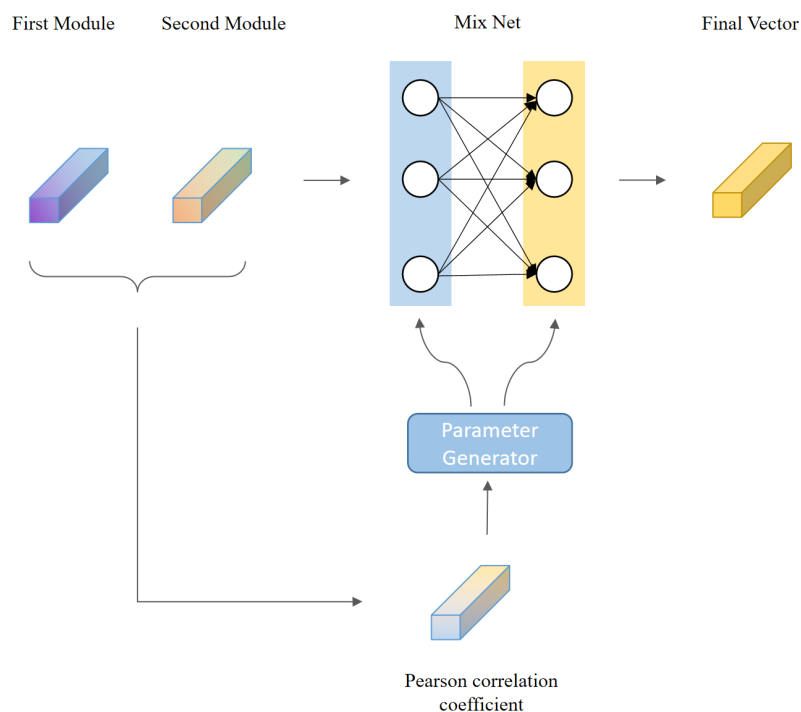


Figure 4. Simple illustration of Pearson Mix Net. The Pearson correlation coefficient is used to control the parameter weights of the Pearson Mix Net.

Finally, the Pearson Mix Net will output the final embedded vector. The inner product of this vector and all object entity vectors is used to determine whether the triple is true or not.

3.2.2. Progressive-Replay-SA Self-Distillation

Due to the huge amount of cybersecurity data and the different frequency of each entity, the catastrophic forgetting problem is more likely to occur. To solve this problem, we adopt a knowledge distillation strategy in the model training process. Most of the current knowledge distillation strategies use the teacher–student structure; however, some knowledge distillation strategies need to consume large amounts of computation. Therefore, we use the self-distillation strategy, which does not require additional networks to participate in the training. It can use the model of the previous round as the teacher of the current round.

In the training process of the knowledge graph completion model, a large number of samples need to be accepted, which is prone to undergo the catastrophic forgetting problem. To avoid the catastrophic forgetting problem, we design a replay self-distillation strategy. We use the updated model to review the samples of the previous batch again and transfer the empirical knowledge obtained from the previous round of prediction.

We calculate the $loss_t$ of the new model for the last batch of samples and the loss l^t of the new model for the current batch of samples. At the same time, we calculate the Kullback–Leibler divergence (KL divergence) l_{kd} of the predicted value $last_v_{t-1}$ of the last round and the predicted value now_v_{t-1} of the new model for the last round of batch samples. This replay method can alleviate the forgetting problem and effectively realize the transfer of knowledge. It is calculated as follows.

$$\begin{cases} loss_t = l^t + l^{t-1} + \beta \cdot l_{kd} \\ l_{kd} = \frac{1}{n} \sum_{i=1}^n \tau^2 KL(\sigma(now_v_{t-1}/\tau), \sigma(last_v_{t-1}/\tau)), \end{cases} \quad (2)$$

where τ is the distillation temperature, $\sigma(\cdot)$ is the softmax function. KL divergence is a good way to describe the difference between two distributions. Consider two probability distributions M and N and their corresponding probability density functions $m(x)$ and $n(x)$, respectively. The computation from $n(x)$ to $m(x)$ can be expressed in the following form.

$$KL(M||N) = \sum m(x) \log \frac{m(x)}{n(x)} = \sum m(x) \log m(x) - \sum m(x) \log n(x), \quad (3)$$

where $\log(\cdot)$ is the logarithm. If the KL divergence is larger, the difference between the two is larger. When the KL divergence is smaller, it means that the difference between the two is smaller. This mathematical form can help us measure the difference between students and teachers very well. The process of Progressive-Replay-SA self-distillation is shown in Figure 5.

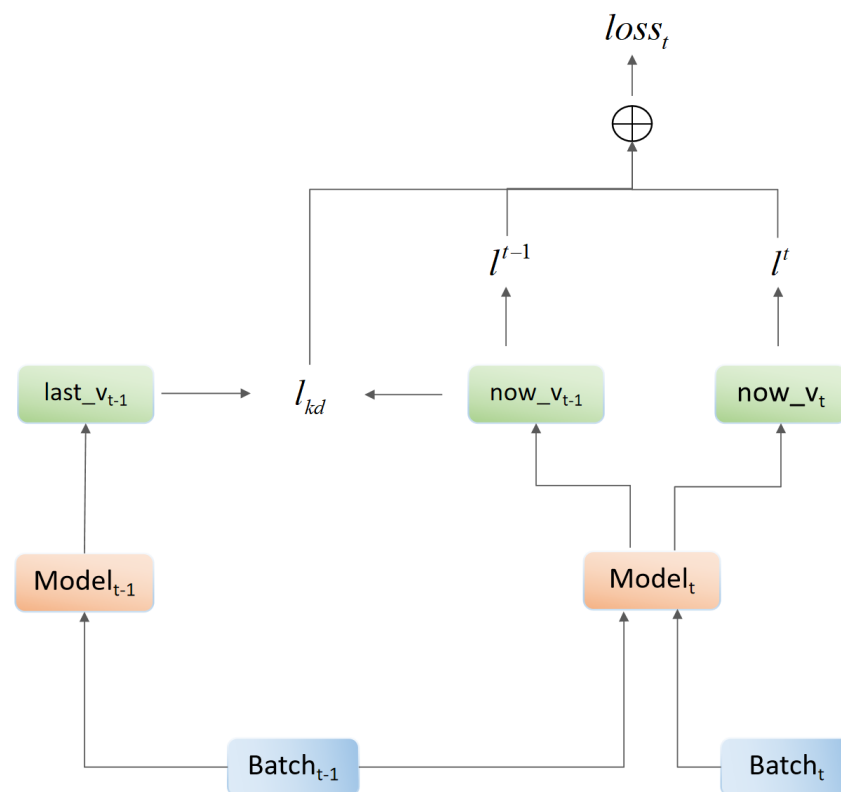


Figure 5. The overall architecture of Progressive-Replay-SA self-distillation.

At the same time, with the increase in training rounds, the accuracy and robustness of the model are continuously enhanced and the reliability is also continuously enhanced. In order to effectively prevent the occurrence of model degradation, we also adopt a progressive strategy for distillation to ensure the stationarity of the training process. The basic calculation of the progressive policy is as follows.

$$loss = \left(1 - \frac{t}{\gamma}\right) \times loss_t + \frac{t}{\gamma} \times loss_{t-1}, \quad (4)$$

where t is the current number of iterations, γ is a large constant, $loss_t$ is the total loss value of the current round and $loss_{t-1}$ is the total loss value of the previous round.

In the process of knowledge distillation, distillation temperature is an important factor that affects the distillation effect. In most of the existing distillation strategies, the distillation temperature is usually set to a fixed value, which seriously limits the performance of distillation. A lower temperature τ will sharpen the distribution and widen the difference between the two distributions. It will concentrate the distillation on the largest logits predicted by the teacher. While a higher temperature τ will flatten the distribution and close the gap between the two models [29]. In a realistic teaching scenario, students should learn the course from easy to difficult and temperature τ will affect the difficulty level of the loss minimization process. We expect the course to be increasingly difficult to learn. The problem can be viewed as consisting of an inner maximization problem and an outer minimization problem. The purpose of the internal maximization problem is to gradually increase the difficulty of the course learning. On the other hand, the goal of the external minimization problem is to find the model parameters that minimize the internal loss. The increase in course difficulty is reflected in the process of self-distillation as the loss value of the current iteration round is greater than the loss value of the previous round. Globally, the model needs to minimize the overall loss. The simulated annealing algorithm is used to realize the adaptive adjustment of distillation temperature to achieve the above purpose.

The simulated annealing algorithm is an effective intelligent optimization algorithm that is widely used in engineering technology and management science. In the process of adjusting the distillation temperature, the initial distillation temperature and the temperature growth direction are initialized. If the difficulty of the course in the current round is greater than that of the last round, the distillation temperature and the temperature growth direction are kept unchanged. If the difficulty of the course in the current round is less than the difficulty of the course in the previous round, that is, $loss_t < loss_{t-1}$, the direction of temperature growth is changed and the distillation temperature is adjusted in the new temperature growth direction. At the same time, in order to prevent falling into the local optimal solution, there is a certain probability that the temperature growth direction and the distillation temperature are still unchanged and the probability value is determined by the loss difference of the two rounds.

The basic flow of distillation temperature optimization is shown in Algorithm 1.

Algorithm 1: Adaptive Distillation Temperature Optimization Algorithm

Input: The initialized temperature τ
 The temperature growth directions ω
 The big fixed constant T
 The small fixed constant $\eta, 0 \leq \eta \leq 1$
 The loss of previous training step $loss_{t-1}$
 The loss of current training steps $loss_t$

Output: The temperature τ

```

1 if  $loss_{t-1} > loss_t$  then
2    $\sigma = 1 - e^{\frac{-|loss_t - loss_{t-1}|}{T}}$ 
3    $\theta_\alpha \leftarrow$  Generate a random number from 0 to 1
4   if  $\theta_\alpha > \sigma$  then
5     if  $\omega == "+"$  then
6        $\omega = "-"$ 
7        $\tau = \tau - \eta \cdot \tau$ 
8     else
9        $\omega = "+"$ 
10       $\tau = \tau + \eta \cdot \tau$ 
11     end
12   else
13     Keep  $\tau$  and  $\omega$  unchanged.
14   end
15 else
16   Keep  $\tau$  and  $\omega$  unchanged.
17 end
18 return  $\tau$ 

```

4. Experiment

4.1. Cybersecurity Knowledge Graph

In order to make penetration testing work smoothly, it is necessary to construct an effective cybersecurity knowledge graph to provide information support. We mainly construct the cybersecurity knowledge graph from three aspects: cyberspace situation intelligence, available tools and basic cybersecurity knowledge.

By probing cyberspace, various elements in cyberspace can be obtained effectively. Information such as IP address, DNS, subdomain name and geographical location in cyberspace is of great significance for penetration testing work and we take the above information as part of cyberspace situation intelligence.

In addition, the use of vulnerability exploitation tools usually requires workers to have some experience. In order to make it more convenient for penetration testing workers to use vulnerability exploitation tools based on the current cyberspace environment, we also include some available tool information as part of the knowledge graph. Metasploit is a well known vulnerability exploitation tool, which contains important modules such as Exploit, Auxiliary and Post. We also integrate it in the knowledge graph as a penetration testing tool part.

As we all know, penetration testing is a job that requires workers have a high skill level, which requires workers to have more penetration testing experience and knowledge. However, the rapid update of cybersecurity technology has brought great challenges to penetration testing workers. At present, the trend of intelligence is increasing. In order to better save human resources, intelligent penetration testing has become the trend of future development. Therefore, it is particularly important to save a large amount of valuable penetration testing experience and knowledge to use, which will make up for the lack of experience of penetration testing workers and provide the possibility of intelligent penetration testing. We choose to incorporate part of the empirical knowledge in CVE,

CWE and CAPEC into the cybersecurity knowledge graph as the cybersecurity empirical knowledge part.

The number of entities in the cybersecurity knowledge graph is 103,993, the number of relations is 30 and it is divided into training set, validation set and test set. The information for the training set, validation set and test set is shown in Table 1.

Table 1. The quantity statistics of cybersecurity knowledge graph. #E and #R represent the number of entities and relations, respectively. #TR, #VA and #TE represent the size of the train set, validation set and test set, respectively.

#E	#R	#TR	#VA	#TE
103,993	30	285,295	15,850	15,866

The cybersecurity knowledge graph we constructed contains ontology types such as IP address, service and geographical location and the relationship between each ontology is shown in Table 2.

Table 2. The attribute description and quantity statistics of each relationship in the cybersecurity knowledge graph.

Relationships	Head→Tail
Address	DNS→IP, subdomain→IP
IsAddressOf	IP→DNS, IP→subdomain
BePlatformOf	OS→EXP
AffectPlatform	EXP→OS
BeAuxOf	AUX→EXP
BeExpOf	EXP→CVE
BePostOf	Post→EXP
Connect	subdomain→subdomain
Control	DNS→subdomain
ControlledBy	subdomain→DNS
Exist	CVE→IP, CVE→subdomain
HasAux	EXP→AUX
HasCve	IP→CVE, subdomain→CVE
HasExp	CVE→EXP
HasPost	EXP→Post
LocArea	IP→Region
LocContinent	IP→Continent
LocatedIn	Region→Continent
OpenPort	IP→Port
OpenedBy	Port→IP
RelatedTo	Port→Service, Service→Port
InstanceOf	CVE→CWE
ObservedExample	CWE→CVE
PeerOf	CWE→CWE, CAPEC→CAPEC
AttackTo	CAPEC→CWE
TargetOf	CWE→CAPEC
CanFollow	CWE→CWE, CAPEC→CAPEC
CanPrecede	CWE→CWE, CAPEC→CAPEC
Childof	CWE→CWE, CAPEC→CAPEC
ParentOf	CWE→CWE, CAPEC→CAPEC

4.2. Experimental Evaluation Metrics and Settings

In order to evaluate the performance of the knowledge graph completion model, we use MRR (Mean Reciprocal Rank) and Hits@K as evaluation metrics. MRR is the mean reciprocal ranking of correct entities. Hits@K is the proportion of correct entities ranked in the top K. For Hits@K, we use Hits@1/3/10 to evaluate the performance of the model. Higher MRR and Hits@1/3/10 indicate better performance.

Since the cybersecurity information we collect is accurate and complete, we expect to simulate the phenomenon of lack of cybersecurity data. We remove the head entity or tail entity of the triples in the test set to form the knowledge graph completion task ($\langle ?, \text{relation}, \text{tail} \rangle$ and $\langle \text{head}, \text{relation}, ? \rangle$). The head entity or tail entity that we will remove can be used as a standard answer to test the experimental performance of the cybersecurity knowledge graph completion model.

4.3. Experimental Results and Analysis

In order to illustrate the performance superiority of our designed knowledge graph completion model, this model is used to conduct experimental comparison with the existing advanced models, including TransE, DistMult, TuckER and ConvE. TransE is an excellent distance-based model with good interpretability. DistMult is a classical semantic matching model, which has a good effect in mining the latent semantic relations in knowledge graph. TuckER mines entities and relations based on the theory of tucker decomposition. ConvE uses convolutional neural networks to encode head entities and relations to achieve knowledge graph completion.

We conduct knowledge graph completion experiments on each model on the cybersecurity knowledge graph we constructed. We record the best results within 100 iteration rounds for each model. The experimental results are shown in Table 3.

Table 3. The experimental results on cybersecurity knowledge graph. The results in bold are the best.

Model	MRR	Hits@1	Hits@3	Hits@10
TransE [8]	0.487	0.450	0.509	0.547
DistMult [17]	0.481	0.462	0.490	0.514
TuckER [19]	0.629	0.584	0.653	0.695
ConvE [20]	0.689	0.670	0.704	0.715
CSNT	0.767	0.728	0.825	0.835

The experimental results show that our knowledge graph completion model has good performance for the completion of cybersecurity data. Compared with the existing models, CSNT has a certain improvement in MRR and Hits@K, which also proves the effectiveness of CSNT.

Different embedding dimensions have a great impact on the experimental performance and generally most methods struggle to maintain good performance in lower dimensions. We use the knowledge graph completion models in both high-dimensional (dim = 200) and low-dimensional (dim = 150) embedded spaces in order to analyze the influence of embedded dimensions on the experimental results. Table 4 records the average predictions in low-dimensional embedded space.

Table 4. Cybersecurity knowledge graph completion results in low-dimensional space. The results in bold are the best.

Model	MRR	Hits@1	Hits@3	Hits@10
TransE [8]	0.477	0.449	0.494	0.525
DistMult [17]	0.472	0.451	0.485	0.505
TuckER [19]	0.554	0.511	0.576	0.621
ConvE [20]	0.668	0.640	0.691	0.711
CSNT	0.751	0.735	0.762	0.778

The experimental results show that our model maintains good experimental performance in both high-dimensional and low-dimensional embedded spaces. Because our model uses a variety of forms to mine the potential semantic relationships, when a single way is insufficient, it can be corrected to avoid the interference of noise and further reduce the fluctuation of information. Finally, we use Pearson Mix Net to combine the

vectors. Pearson Mix Net has good adaptive ability and the ideal experimental results are achieved by adaptively adjusting the parameter weights of the neural network in the training process.

5. Conclusions

This paper designs a knowledge graph completion model, which can effectively capture the interactive information of entities and relationships. It uses tensor decomposition and neural network to mine the hidden relationships in knowledge graph. At the same time, it uses Pearson Mix Net to realize the fusion of abstract features. In order to deal with the problem of catastrophic forgetting in the process of model training, an incremental simulated annealing self-distillation strategy is proposed, which effectively suppresses the model degradation through the incremental strategy. At the same time, the simulated annealing algorithm is used to adjust the distillation temperature adaptively, so that the learning process can be from simple to difficult. Experiments show that our method has certain advantages in realizing cybersecurity knowledge graph completion and can be better used to assist penetration testing work.

In the future, the model can be used for error correction, completion and unknown knowledge discovery of cybersecurity intelligence, which can effectively improve the quality of cybersecurity intelligence. Knowledge graph can effectively realize intelligent penetration testing by combining with recommender systems, question answering systems and other modules.

Author Contributions: Conceptualization, P.W. and J.L.; methodology, P.W.; software, P.W.; validation, P.W., J.L. and X.Z.; formal analysis, P.W.; investigation, S.Z.; resources, X.Z.; data curation, J.L.; writing—original draft preparation, P.W.; writing—review and editing, P.W.; visualization, P.W.; supervision, J.L., X.Z. and S.Z.; project administration, P.W. 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. *Hum. Factors J. Hum. Factors Ergon. Soc.* **1995**, *37*, 32–64. [[CrossRef](#)]
2. Guo, Q.; Cao, S.; Yi, Z. A medical question answering system using large language models and knowledge graphs. *Int. J. Intell. Syst.* **2022**, *37*, 8548–8564. [[CrossRef](#)]
3. Zehra, S.; Mohsin, S.F.M.; Wasi, S.; Jami, S.I.; Siddiqui, M.S.; Raazi, S.M.K. Financial Knowledge Graph Based Financial Report Query System. *IEEE Access* **2021**, *9*, 69766–69782. [[CrossRef](#)]
4. Li, N.; Shen, Q.; Song, R.; Chi, Y.; Xu, H. MEduKG: A Deep-Learning-Based Approach for Multi-Modal Educational Knowledge Graph Construction. *Information* **2022**, *13*, 91. [[CrossRef](#)]
5. Chhetri, T.R.; Kurteva, A.; Adigun, J.G.; Fensel, A. Knowledge Graph Based Hard Drive Failure Prediction. *Sensors* **2022**, *22*, 985. [[CrossRef](#)] [[PubMed](#)]
6. Sakurai, K.; Togo, R.; Ogawa, T.; Haseyama, M. Controllable Music Playlist Generation Based on Knowledge Graph and Reinforcement Learning. *Sensors* **2022**, *22*, 3722. [[CrossRef](#)] [[PubMed](#)]
7. Xing, X.; Wang, S.; Liu, W. An Improved DDPG and Its Application in Spacecraft Fault Knowledge Graph. *Sensors* **2023**, *23*, 1223. [[CrossRef](#)] [[PubMed](#)]
8. 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. 787–795.
9. 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.

10. 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.
11. Ji, G.; He, S.; Xu, L.; Liu, K.; Zhao, J. Knowledge Graph Embedding via Dynamic Mapping Matrix. In Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, ACL 2015, Beijing, China, 26–31 July 2015; Volume 1: Long Papers, pp. 687–696. [[CrossRef](#)]
12. Sun, Z.; Deng, Z.H.; Nie, J.Y.; Tang, J. RotatE: Knowledge Graph Embedding by Relational Rotation in Complex Space. In Proceedings of the International Conference on Learning Representations, New Orleans, LA, USA, 6–9 May 2019.
13. 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.
14. 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](#)]
15. Balazevic, I.; Allen, C.; Hospedales, T.M. Multi-relational Poincaré Graph Embeddings. 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. 4465–4475.
16. 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, Washington, DC, USA, 28 June–2 July 2011; pp. 809–816.
17. 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.
18. Trouillon, T.; Welbl, J.; Riedel, S.; Gaussier, É.; Bouchard, G. Complex Embeddings for Simple Link Prediction. In Proceedings of the 33rd International Conference on Machine Learning, ICML 2016, New York, NY, USA, 19–24 June 2016; Volume 48, pp. 2071–2080.
19. Balazevic, I.; Allen, C.; Hospedales, T.M. TuckER: Tensor Factorization for Knowledge Graph Completion. In Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, 3–7 November 2019; pp. 5184–5193. [[CrossRef](#)]
20. 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.
21. Vashishth, S.; Sanyal, S.; Nitin, V.; Agrawal, N.; Talukdar, P.P. InteractE: Improving Convolution-Based Knowledge Graph Embeddings by Increasing Feature Interactions. In Proceedings of the Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, 7–12 February 2020; pp. 3009–3016.
22. Che, F.; Zhang, D.; Tao, J.; Niu, M.; Zhao, B. ParamE: Regarding Neural Network Parameters as Relation Embeddings for Knowledge Graph Completion. In Proceedings of the AAAI Conference on Artificial Intelligence, New York, NY, USA, 7–12 February 2020.
23. Hinton, G.; Vinyals, O.; Dean, J. Distilling the Knowledge in a Neural Network. *Comput. Sci.* **2015**, *14*, 38–39.
24. Kim, K.; Ji, B.; Yoon, D.; Hwang, S. Self-Knowledge Distillation with Progressive Refinement of Targets. In Proceedings of the 2021 IEEE/CVF International Conference on Computer Vision, ICCV 2021, Montreal, QC, Canada, 10–17 October 2021; pp. 6547–6556. [[CrossRef](#)]
25. Zhang, L.; Song, J.; Gao, A.; Chen, J.; Bao, C.; Ma, K. Be Your Own Teacher: Improve the Performance of Convolutional Neural Networks via Self Distillation. In Proceedings of the 2019 IEEE/CVF International Conference on Computer Vision, ICCV 2019, Seoul, Republic of Korea, 27 October–2 November 2019; pp. 3712–3721. [[CrossRef](#)]
26. Shen, Y.; Xu, L.; Yang, Y.; Li, Y.; Guo, Y. Self-Distillation from the Last Mini-Batch for Consistency Regularization. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, 18–24 June 2022; pp. 11933–11942. [[CrossRef](#)]
27. Hochreiter, S.; Schmidhuber, J. Long Short-Term Memory. *Neural Comput.* **1997**, *9*, 1735–1780. [[CrossRef](#)] [[PubMed](#)]
28. Rashid, T.; Samvelyan, M.; de Witt, C.S.; Farquhar, G.; Foerster, J.N.; Whiteson, S. QMIX: Monotonic Value Function Factorisation for Deep Multi-Agent Reinforcement Learning. In Proceedings of the Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholm, Sweden, 10–15 July 2018; Proceedings of Machine Learning Research; Volume 80, pp. 4292–4301.
29. Li, Z.; Li, X.; Yang, L.; Zhao, B.; Song, R.; Luo, L.; Li, J.; Yang, J. Curriculum Temperature for Knowledge Distillation. *arXiv Preprint* **2022**, arXiv:2211.16231.

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.