Time Series Attention Based Transformer Neural Turing Machines for Diachronic Graph Embedding in Cyber Threat Intelligence

Binghua Song[1,2], Rong Chen[3], Baoxu Liu[1,2(✉)], Zhengwei Jiang[1,2], and Xuren Wang[3]

[1] Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China
[2] School of Cyber Security, University of Chinese Academy of Sciences, Beijing, China
{songbinghua,liubaoxu,jiangzhengwei}@iie.ac.cn
[3] Information Engineering College, Capital Normal University, Beijing, China
{2201002025,wangxuren}@cnu.edu.cn

Abstract. The cyber threats are often found to threaten individuals, organizations and countries at different levels and evolve continuously over time. Cyber Threat Intelligence (CTI) is an effective approach to solve cyber security problems. However, existing processes are considered inherent responses to known threats. CTI experts recommend proactively checking for emerging threats in existing knowledge. In addition, most researches focus on static snapshots of the CTI knowledge graph, while ignoring the temporal dynamics. To this end, we create a novel framework TSA-TNTM (Time Series Attention based Transformer Neural Turing Machines) for diachronic graph embedding framework, which uses time series self-attention mechanism to capture the non-linearly evolving entity representations over time. We demonstrate significantly improved performance over various approaches. A series of benchmark experiments illustrate that TSA-TNTM could generate higher quality than the state-of-the-art word embedding models in tasks pertaining to semantic analogy, clustering, threat classification and proactively identify emerging threats in CTI fields.

Keywords: Threat Intelligence · Dynamic Knowledge Graph · Time Series Attention · Graph Embedding.

1 Introduction

1.1 backgrounds

Due to the dynamically rise of the cyber attacks such as the attack against GitHub lasted 72 hours in 2015 [20], cyber security is becoming a vital research area to our society. Attackers exploit vulnerabilities to attack systems, for example, a hacker can exploit SQL-related injections to illegally break into the system and obtain user information. Cyber Threat Intelligence (CTI) based Knowledge Graphs (KGs) is an effective approach to reveal and predict the latent attack

ICCS Camera Ready Version 2022
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-08760-8_2

behaviors. Especially, KGs that represent structural relations between entities have become an increasingly popular research direction towards exploration and prediction of CTI. Most of advances in research based on knowledge graph focus on Knowledge Representation Learning (KRL) or Knowledge Graph Embedding(KGE) by mapping entities and relations into low-dimensional vectors, while capturing their static semantic meanings [17] [3].

The existing neural network models have achieved good performance in processing graph data. For example, the graph convolution network (GCN [14]) uses the CNN [21] mode to process the first-order, second-order or even more higher-order neighbors aggregation on nodes in graphs. Convolution algorithm extracts the similar features of nodes and graph topological features, Recurrent Neural Networks(RNN [35]) such as LSTM [10] can extract the input information at the current moment and the state information at the previous moment through a gating mechanism.

However, these algorithms can easily cause gradients disappearing or exploding, which make it difficult to extract features depend on the long time distances. In the field of threat intelligence, the span of cyber security behaviors is usually very long. For example, the data set [26] which we used in this paper has a very large time span from 1996 to 2010. Many hacker terms and semantics have changed with the development of new IT technology over time. So, these current existing models face a lot of challenges.

1.2 contribution

Recently, there has been a growing research interest for researches in temporal KG embeddings, where the edges of a KG are also endowed with information about the time period(s) in which the relationship is considered valid. Many temporal knowledge graph models have been proposed such as RE-NET [13], Know-Evolve [30], ATiSE [34], JODIE [15], DGNN [32], GC-LSTM [4], D-GEF [26]. However, each of these models cannot adequately capture temporal and spatial information about the network and lack sufficient capability of spatio-temporal dependent feature extraction. It is difficult to efficiently realize the joint extraction of temporal and spatial features.

To this end, we leverage the existing NTM (Neural Turing Machines) [8] memory enhancement neural network model. The controller of existing NTM models mostly use traditional graph neural network methods such as GCN and RNN. Our TSA-TNTM model replace the controller with Transformer [31] model, and use the write head selectively write to memory block and use the read head to read from it again. In addition, we further propose a time series-based attention mechanism, focus on the time and space features of dynamic knowledge graph. Overall, our contributions are as follows:

1) We propose a time series attention based differentiable neural Turing machine model for dynamic CTI Knowledge Graph so as to promote the processing seed and accuracy. We use the transformer model for the component of controller rather than LSTM to capture unlimited long distance spatio-temporal

ICCS Camera Ready Version 2022
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-08760-8_2

dependencies, and further modify the structure of transformer such as temporal position encoding(TPE) to improve the ability to capture the topological dynamics of the graph.

2) We propose an adaptive multi-head self-attention mechanism based on time series database which can speed up the spatio-temporal embeddings and prediction according to the different scenarios which focus on disparate concerns of attack behaviors in each snapshot.

3) Based on a real hacker forum data set across about 23 years, we made a series of experiments and achieved the remarkable performance over other related approaches.

2 related work

Many researchers considered the study of time series or dynamic knowledge graph. However, these studies are always focused on **general** fields, and they are mostly **static** knowledge graph.

2.1 General Static and Time-based KG Models

In general fields, the most fundamental and widely used model is TransE [2], which treats the relationship in knowledge graph as some kind of translation vector between entities. However, TransE only works well for the one-to-one relationship. TransH [33] regards a relation as a translation operation on a hyperplane. TransR [17] projects the entities from the entity space to the corresponding relational space, and then transforms the projected entities, TransD [11] considers the diversity of entities and relationships, constructing a dynamic mapping matrix for each entity-relation pair.

The following combination of Trans family models and time series solved the problems of embedding entities and relationships of data at different times in low-dimensional vector space. Jin W et.al [13] proposed Recurrent Event Network (RE-NET), which employed a recurrent event encoder to encode past facts, and uses a neighborhood aggregator to model the connection of facts at the same timestamp in order to infer future facts. Trivedi R et.al [30] presented Know-Evolve, which learned non-linearly evolving entity representations over time. The occurrence of a fact is modeled as a multivariate point process. A García-Durán et.al [6] proposed a method to utilize recurrent neural networks to learn time-aware representations of relation types. Tingsong J et.al [12] proposed a time-aware KG embedding model using temporal order information among facts, specifically, using temporal consistency information as constraints to incorporate the happening time of facts. Julien L et al. [16] focused on the task of predicting time validity for unannotated edges. Shib S D et.al [5] proposed HyTE, which explicitly incorporates time in the entity relation space by associating each timestamp with a corresponding hyperplane. Xu C et.al [34] proposed ATiSE which incorporates time information into entity or relation representations by using

ICCS Camera Ready Version 2022
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-08760-8_2

additive time series decomposition. Goel R et.al [7] built novel models for temporal KG completion through equipping static models with a diachronic entity embedding function.

These models can only handle **static** information and cannot handle complex **dynamic** graphs.

2.2 General Fields Dynamic KG Models

Due to the limitations of the static KG algorithms, many researchers began to study **dynamic** KGC techniques. Rakshit T et.al [29] proposed a model named DyRep, which is a latent mediation process bridging two observed dynamics on the network. Srijan K et.al [15] proposed JODIE, a coupled recurrent neural network model which employed two recurrent neural networks to update the embedding. Ma Y et.al [32] proposed DGNN, which could model the dynamic information as the graph evolving. Manessi F et.al [19] combined LSTM and GCN to learn long -short term dependencies together with graph structure, so that jointly exploiting structured data and temporal information. Chen J et.al [4] proposed GC-LSTM, a GCN embedded LSTM network for dynamic link prediction. Maheshwari A et.al [18] proposed a novel adversarial algorithm DyGAN to learn representation of dynamic networks, who leverage generative adversarial networks and recurrent networks to capture temporal and structural information.

These dynamic KG models are less studied and ignore **time** information and can not capture changes in semantic information over time.

2.3 Cyber Threat Intelligence Fields KG Models

Xiaokui S et.al [28] introduced threat intelligence computing as a graph computation problem, presenting a threat intelligence computing platform through the design and implementation of a domain-specific graph language. Samtani S et.al [26] created a novel Diachronic Graph Embedding Framework(D-GEF), which operated on a Graph of Words(GoW) representation of hacker forum text to generate word embeddings across time-spells. Rastogi N et al. [25] introduced an open-source malware ontology model named MALOnt, which allows the structured extraction of information and knowledge graph generation, especially for threat intelligence. Pingle A et al.[24] proposed a system to create semantic triples over cyber security text. Sarhan I et al. [27] presented OpenCyKG, which is constructed using an attention-based neural Open Information Extraction model to extract valuable cyber threat information from unstructured Advanced Persistent Threat (APT) reports.

In the field of cyber threat intelligence, there are fewer researches on knowledge graphs, especially with time information. Most of these studies focus on how to construct a knowledge graph, unable to predict potential threats, and ignore the changes in some threat terminology over time.

ICCS Camera Ready Version 2022
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-08760-8_2

3 method

3.1 Dynamic Graph Embedding Models

Most dynamic graph embedding models given a graph $G = G_1, G_2, \cdots, G_T$, where $G_T = (V_T, E_T)$, V is the node set, T is different timestamp and E is the edge set. Nodes have an edge if they are adjacent. Define mapping $f : V \to R^d$, function f preserves some proximity measure defined on the graph G, graph embedding is a time series of mappings $F = \{f_1, \cdots, f_T\}$ such that mapping f_T is a graph embedding for G_T and all mappings preserve the proximity measure for their respective graphs.

Current existing graph embedding methods cannot capture embedding shifts and changes in the temporal datasets mostly. Graph data is splitted into time-spells and the embeddings are created in each time-spell. However, each time-spell has a different semantic embedding space, which makes it difficult to directly compare the graph embeddings in different time-spells.

3.2 Attention Model

The attention model has become an important concept in neural networks. The Transformer model replaces convolution entirely with the attention mechanism and can be processed in parallel. There are many ways to calculate attention. We use the dot product formula, which is as shown:

$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_k}})V \tag{1}$$

$$MultiHead(Q, K, V) = Concat(head_1, ..., head_h)W^Q \tag{2}$$

where Q, K, V represents the query, the key, and the value respectively, $\sqrt{d_k}$ is represented for normalization and $head_i = attention(QW_i^Q, KW_i^K, VW_i^V)$. Each position of output is the vector weighted average of all positions of the value, the weight is calculated by attention from all positions of the query and key.

Attention maps the query and key to a same high-dimensional space for calculating similarity, while multi-head attention allows the model to jointly attend to information from different representation sub-spaces at different positions [1]. For Example, the temporal and spatial features can be weighted in different head simultaneously.

3.3 TSA-TNTM Model

Based on the previous two steps, we will build our TSA-TNTM model. Firstly we describe the high-level structure of our model TSA-TNTM. The major novel component of the whole model is transformer-based controller, which can be utilized to capture unlimited spatio-temporal features for the dynamic CTI knowledge graph over time with high speed. The Neural Turing Machine (NTM) [8] had

ICCS Camera Ready Version 2022
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-08760-8_2

a similar structure, which extend the capabilities of neural networks by coupling them to external memory resources, which they can interact with each other by attention mechanism to read from and write to the memory block selectively according to different weights.

Our proposed model TSA-TNTM extend and modify the controller of NTM, which is equivalent to the CPU part of the computer. The extension of controller could be realized by graph neural network in the neural Turing machine [8]. We abandon the existing RNN [35], GCN [14] and other graph neural networks, adopt the prevalent transformer model, and use the attention mechanism to improve the dynamic time feature extraction capability and parallel processing capability of the threat intelligence knowledge graph.

In addition, we propose time series hybrid attention and realize seamless docking with memory in key-value mode. In time series hybrid attention, the three Tensors of Q (Query) value, K(Key) value, and V(Value) value are all from the same memory input. We calculate the dot product between Q value and K value firstly, and then divided by a scale $\sqrt{d_k}$ in order to prevent the result from being too large, where d_k is the dimension of a query and key tensor. Then use the softmax operation to normalize the result to a probability distribution, and multiply the result by the tensor V so that getting the weighted summation. This operation can be expressed as formula (1), where K and V establish a mapping with the key and value of the underlying time series database such as OpenTSDB [23] by parallel processing technology, so as to make the best of massive data processing of the underlying big data technology.

Finally, we introduce a positive and negative feedback mechanism. Memory is not only used for general data reading and writing, but also forms the input and output of the transformer. Among them, memory provides dynamic preprocessing capabilities based on attention for the input, and automatically learns the distribution and spatio-temporal characteristics of the graph data. This further reduces the burden on transformer, and improve the processing performance. In contrast to most models of working memory, our model is analogous to a Turing Machine or Von Neumann architecture but is differentiable end-to-end, allowing it to be efficiently trained with gradient descent. The model is as shown in Figure 1.

As illustrated in Figure 1, TSA-TNTM consist of a memory block followed by a controller block according to the architecture of Neural Turing Machines. Each block may contain multiple stacked layers of the same type. The memory block extract spatio-temporal features from the dynamic graph with unfixed-length through the multi-head self-attention aggregation. The details of this attention approach will be described in the next section. In this memory block, there are two ways to ingest the graph data: one way is to employee the online method to capture the topological evolution and node interactions of the dynamic graph continuously, the other way is to employee the offline method to capture the long-time historical temporal and topological evolution of multiple graphs. During the real time ingestion of dynamic graph, we can also use multi-head self-attention to extract multi-grained long span features to store into the time series database

ICCS Camera Ready Version 2022
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-08760-8_2

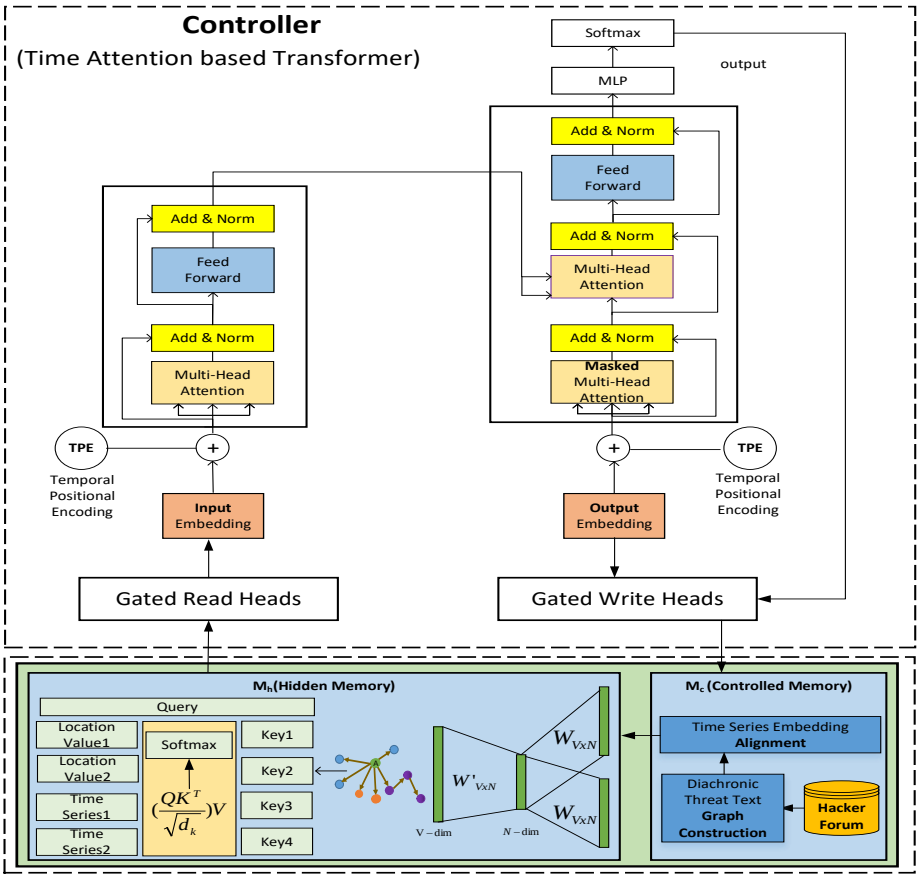

Fig. 1. Simple illustration of TSA-TNTM.

so as to deal with the long-distance correlation such as a new threat intelligence first appear in 1996 and changed representation in 1998, 2001, 2012 respectively.

The processing flow of the TSA-TNTM are as follows:

Firstly, Memory Block Construction. The main function of Memory Block is collecting CTI threat intelligence corpus in real time and dynamically constructing a knowledge graph. The Memory module takes the acquisition of the dynamic knowledge graph as its main function, uses the attention mechanism to optimize the storage performance of the Memory Block. The Graph Construction process in Memory Block is as follows:

1) Collect threat intelligence corpus text, preprocess and store it into time series database such as OpenTSDB[23]. In this paper, the hacker forum data we used is one of the corpus, we would consider to add other sources later.

2) Split data into time-spell and construct Threat Text Graph. According to the order of words in the sentence, build two adjacent words in accordance

ICCS Camera Ready Version 2022
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-08760-8_2

with first-order and second-order neighbors, adopt the CBOW model to learn the weight of word embedding, construct the time information before and after the word as a separate entity structure and add it to the edge set list uniformly. The figure 2 is illustration of the constructed threat text graph in the time-spell 2019. From the graph, we can see that hackers exploit remote administration tools(RATs), email hacks, denial of service(DOS), and others to attack the system.

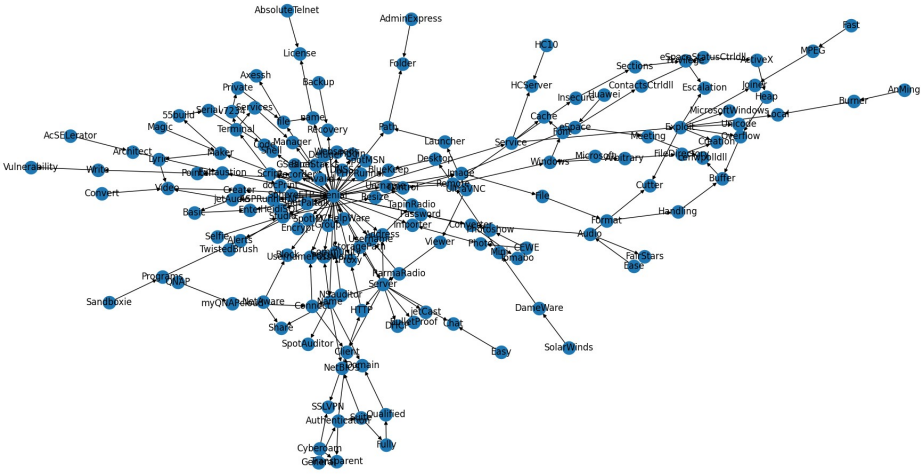

Fig. 2. The Diachronic Threat Text Graph in Time-Spell 2019.

3) Embedding alignment. Embedding spaces across time-spells are aligned while retaining cosine similarities by optimizing the following objective function:

$$R^{(t)} = \underset{Q^T Q=I}{\arg\min} ||W^{(t)}Q - W^{(t+1)}||_F \tag{3}$$

where $||\cdot||_F$ denotes the Forbnius norm.

4) Read the dynamic edge set list in real time and construct the graph in real time.

5) Calculate the attention score of the new input side information, and dynamically modify the weights. The related details can be found in Formula (1) and Formula (2).

6) Realize the initial embedding of the spatio-temporal features of the graph according to the attention weight, and adjust the relevant value in $L1$ and $L2$ norm.

Secondly, Controller Block Graph Embedding and Computation. Controller Block selectively read the word embedding vector of Memory Block through Gated Read Head, encoded by the Encoder, further extracts and processes

ICCS Camera Ready Version 2022
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-08760-8_2

spatio-temporal features, and outputs processing results through the Decoder, and then stores the results into Memory Block.

In the TSA-TNTM model, we divide the memory block into two parts: the **controller memory** M_c and the **hidden memory** M_h. The M_c is controlled by the controller block and store the controller's result into time series database for persistent store. The M_h is not memory by the controller and is connected with the M_c. The hidden memory saves the accumulated content into the controller, the procedure of read and write in time t is:

$$M_c(t) = h(M_c(t-1), w(t-1), c(t)) \tag{4}$$

$$M_{\mathrm{h}}(t) = aM_h(t-1) + bM_c(t) \tag{5}$$

$$r(t) = W_r(t)M_h(t) \tag{6}$$

where M_c is updated at time t-1 by the write head $w(t-1)$ and generate the output of M_c at time t and update the M_h which is used to generate read head $r(t)$ at time t. $c(t)$ is the output of controller, h is the function of update the controlled memory and the write weight in order to realize the function of erasion and addition. The a and b is hybrid weight of the scalar and read head reads from the $M_h(t)$.

Based on the above analysis, it can be seen that TSA-TNTM expands the attention mechanism from the controller as a neural network to memory, thereby realizing the lightweight attention of the entire Turing model from storage to computing. Meanwhile, it realize unlimited spatio-temporal feature extraction by the expansion of memory.

4 Experiment and Evaluation

4.1 dataset

We adopt the real hacker forum data set from D-GEF model [26] to complete the experiment in this paper. In order to analysis the temporal changes about hacker threats, the author collected a large and long-standing international hacker forum. According to the descriptions, collection procedures resulted in data set with 32766 threats posts made by 8429 hackers between January 1, 1996 and July 10, 2019 (across 23 years), but actually we only find 6,767 posts in their publicly opened data at the GitHub repository. The statistics of the data set are summarized into Table 1. The detail descriptions of hacker forum data set in this table can be viewed from the D-GEF model [26].

4.2 Evaluation Method and Computational Setup

We evaluate our model performances by using well-established metrics of accuracy, precision, recall, and F1-score (i.e., F-measure) as following:

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \tag{7}$$

ICCS Camera Ready Version 2022
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-08760-8_2

Table 1. Statistics for Hacker Forum DataSet.

Threat Type	Target Platform	Date Range	Exploit Numbers	Total
Remote	Windows	03/23/2009 – 07/05/2019	1,418	1,864
	Linux	06/24/2000 – 07/02/2019	446	
Local	Windows	09/28/2004 – 06/20/2019	1,818	2,429
	Linux	01/01/1996 – 07/02/2019	611	
Total	-	01/01/1996 – 07/05/2019	4,293	4,293

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{8}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{9}$$

$$\text{F1-score} = \frac{2 * \text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}} \tag{10}$$

where the True Positive (TP) means the percent of correctly classified normal samples; the False Positive (FP) means the percent of classifying abnormal samples as normal samples; the True Negative (TN) means the percent of classifying abnormal samples as abnormal samples; the False Negative (FN) means the percent of classifying normal samples as abnormal samples.

We use a mobile workstation notebook equipped with an Intel® Core i7-8750H@2.20GHz processor, 32 GB of RAM, and an NVIDIA® Quadro P2000 Graphical Processing Unit(GPU). The experiments use Python language, version 3.9. All word embedding approaches were implemented using the Genism package and Open Network Embedding(OpenNE) package. We use scikit-learn to process performance metrics and statistical tests. All of the experiments were trained over 50 epochs, and generated 128 dimensions embeddings.

4.3 Benchmark Methods

We select prevailing word and graph embedding approaches to compare with our TSA-TNTM model. In order to compare with D-GEF [26] in the same data set, considering almost no model predict the emerging threat in threat intelligence, our TSA-TNTM model just compare the classification of attack type and platform with six prevailing word embedding models and four famous graph embedding models.

Although the two classification tasks are based on the past facts, they contain temporal information about the threat facts, the subsequent happened facts are also the their future. So, the classification experiments also check the ability of prediction about the future **threat type** (local or remote attack) and **platform** (attack from Linux or windows). The happened facts are also the labels for prior facts. The methods include classic **word embedding** models and **graph embedding** methods.

ICCS Camera Ready Version 2022
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-08760-8_2

Table 2. Summary of Results from **Attack Type** Classification Experiments.

Category	Method	Accuracy	Precision	Recall	F1	AUC
TF-IDF	X^2	87.6±1.3%	88.1±1.2%	77.6±1.2%	80.6±1.5%	0.862
word2vec	SGNS	84.6±1.3%	82.6±1.3%	74.6±1.3%	76.1±2.3%	0.899
	CBOW	83.1±1.3%	81.6±1.3%	70.6±1.3%	73.6±1.3%	0.868
FastText	SGNS	85.6±1.3%	83.6±1.3%	76.5±1.3%	78.2±2.1%	0.901
	CBOW	83.1±1.3%	81.6±1.3%	70.6±1.3%	73.6±1.3%	0.868
Doc2Vec	DM	82.5±1.2%	79.5±1.3%	68.5±1.5%	71.6±1.8%	0.845
	DBOW	85.7±1.2%	81.8±1.3%	74.6±1.3%	76.9±1.5%	0.898
Graph Factorization	GF	87.6±1.3%	85.5±1.3%	79.5±1.5%	81.2±2.1%	0.915
	HOPE	84.5±1.3%	82.1±1.2%	72.7±1.2%	75.6±1.3%	0.881
	GraRep	86.5±1.2%	83.8±1.3%	77.6±1.3%	80.6±1.3%	0.912
Random Walk	DeepWalk	88.6±1.3%	86.6±1.3%	80.5±1.3%	82.2±2.1%	0.925
	node2vec	87.9±1.5%	85.6±2.1%	80.6±1.3%	82.2±2.3%	0.921
AutoEncoder	SDNE	83.2±1.3%	81.3±1.2%	71.5±1.2%	74.7±3.3%	0.872
EdgeReconstruction	LINE	86.1±1.3%	83.3±1.2%	77.8±3.2%	78.5±3.5%	0.912
TSA-TNTM	NTM	91.7±1.5%	92.5±2.6%	87.6±2.3%	86.6±2.5%	0.961

We conduct two experiments, which include:

1) **experiment 1**: **attack type (Local vs Remote)** classification benchmark. The experiment 1 is an attack type classification benchmark of the hacker forum data set from January 1, 1996 to July 10, 2019, which contains the attack type information for the past 23 years period. It is a typical use case to check word and graph embedding and prediction capability for dynamic graph across time-spells. In this experiment, we will use the above 10 benchmark methods and our TSA-TNTM model to compare the attack type classification prediction results by using the metrics of accuracy, precision, F1-score and AUC.

2) **experiment 2**: **platform (Linux vs Windows)** classification benchmark. The experiment 2 is a platform classification benchmark about the hacker forum dataset from January 1, 1996 and July 10, 2019, which contains the platform information for the past 23 years period. In this experiment, we will use the above ten benchmark methods and our TSA-TNTM model to compare the platform classification prediction results by using the metrics of accuracy, precision, F1-score and AUC.

4.4 Experiment Results

We report the accuracy, precision, recall, and F1 scores with a confidence interval for each algorithm. Besides our TSA-TNTM model, results across both classification tasks indicate that the random walk-based methods outperform the competing graph and word embedding approaches. In terms of F1 scores, the DeepWalk model has achieved a score of 82.6%. However, our TSA-TNTM model achieves the best scores in all metrics both experiment 1 and experiment 2. The experiment 1 results are as shown in Table 2 and the experiment 2 results are as shown in Table 3.

ICCS Camera Ready Version 2022
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-08760-8_2

Table 3. Summary of Results from **Attack Platform** Experiments.

Category	Method	Accuracy	Precision	Recall	F1	AUC
TF-IDF	X^2	84.6±1.3%	88.1±1.2%	77.6±1.2%	80.6±1.5%	0.916
word2vec	SGNS	85.6±1.3%	82.5±1.5%	74.6±1.3%	76.1±2.3%	0.925
	CBOW	80.1±1.3%	81.6±1.3%	70.5±1.3%	73.7±1.2%	0.896
FastText	SGNS	85.6±1.3%	83.5±1.3%	76.5±1.3%	78.2±2.1%	0.922
	CBOW	79.5±1.2%	80.7±1.3%	70.6±1.2%	73.5±1.3%	0.882
Doc2Vec	DM	80.6±1.3%	79.5±1.3%	68.5±1.3%	71.5±2.1%	0.881
	DBOW	85.6±1.3%	81.8±1.3%	74.5±1.3%	76.9±1.3%	0.921
Graph Factorization	GF	86.6±1.3%	85.1±1.3%	79.5±1.3%	81.2±2.1%	0.936
	HOPE	84.5±1.3%	82.1±1.3%	72.6±1.2%	75.5±1.3%	0.922
	GraRep	86.5±1.3%	83.6±1.3%	78.5±1.3%	80.7±1.3%	0.937
Random Walk	DeepWalk	87.6±1.3%	86.6±1.3%	80.5±1.3%	82.5±2.1%	0.937
	node2vec	87.9±1.3%	85.6±1.3%	80.6±1.3%	82.5±1.3%	0.938
AutoEncoder	SDNE	83.5±1.3%	81.3±1.2%	71.5±1.2%	74.7±3.3%	0.916
EdgeReconstruction	LINE	86.1±1.3%	83.3±2.2%	77.6±3.2%	79.5±3.2%	0.938
TSA-TNTM	NTM	91.6±1.2%	92.6±2.1%	86.5±2.3%	85.5±2.3%	0.959

5 Conclusion

In this paper, we propose a model named TSA-TNTM based on time series attention. Our model can capture the temporal and spatial features assisted by time series database which breaking the long time dependencies limitation of previous models. Meanwhile, we propose an adaptive multi head self attention mechanism based on time series big data, which can promote the speed of temporal prediction across across long timescales. Our model achieved the best results among the mentioned methods based on the data set of a real hacker forum.

In the future work, we consider improving our model on more larger datasets and more data sources such as the the National Vulnerability Database (NVD)[22] which gather and store many Common Vulnerabilities and Exposures (CVEs) across long time span so as to find more unseen latent threats and predict emerging threats in the future. In addition, we also introduce the Hawkes process [9] for modeling sequential discrete events occurring in continuous time where the time intervals between neighboring events may not be identical. Thus, we can build a enhanced **continuous** time series based attention mechanism to further promote our TSA-TNTM model.

Acknowledgements This research is supported by Key Laboratory of Network Assessment Technology, Chinese Academy of Sciences and Beijing Key Laboratory of Network Security and Protection Technology. We thank the anonymous reviewers for their insightful comments on the paper.

ICCS Camera Ready Version 2022
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-08760-8_2

References

1. Bahdanau, D., Cho, K., Bengio, Y.: Neural machine translation by jointly learning to align and translate. arXiv preprint arXiv:1409.0473 (2014) 3.2
2. Bordes, A., Usunier, N., Garcia-Duran, A., Weston, J., Yakhnenko, O.: Translating embeddings for modeling multi-relational data. Advances in neural information processing systems **26** (2013) 2.1
3. Cao, Z., Xu, Q., Yang, Z., Cao, X., Huang, Q.: Dual quaternion knowledge graph embeddings. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 35, pp. 6894–6902 (2021) 1.1
4. Chen, J., Wang, X., Xu, X.: Gc-lstm: Graph convolution embedded lstm for dynamic link prediction. arXiv preprint arXiv:1812.04206 (2018) 1.2, 2.2
5. Dasgupta, S.S., Ray, S.N., Talukdar, P.: Hyte: Hyperplane-based temporally aware knowledge graph embedding. In: Proceedings of the 2018 conference on empirical methods in natural language processing. pp. 2001–2011 (2018) 2.1
6. García-Durán, A., Dumančić, S., Niepert, M.: Learning sequence encoders for temporal knowledge graph completion. arXiv preprint arXiv:1809.03202 (2018) 2.1
7. Goel, R., Kazemi, S.M., Brubaker, M., Poupart, P.: Diachronic embedding for temporal knowledge graph completion. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 34, pp. 3988–3995 (2020) 2.1
8. Graves, A., Wayne, G., Danihelka, I.: Neural turing machines. arXiv preprint arXiv:1410.5401 (2014) 1.2, 3.3
9. Han, Z., Ma, Y., Wang, Y., Günnemann, S., Tresp, V.: Graph hawkes neural network for forecasting on temporal knowledge graphs. arXiv preprint arXiv:2003.13432 (2020) 5
10. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural computation **9**(8), 1735–1780 (1997) 1.1
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 (volume 1: Long papers). pp. 687–696 (2015) 2.1
12. Jiang, T., Liu, T., Ge, T., Sha, L., Chang, B., Li, S., Sui, Z.: Towards time-aware knowledge graph completion. In: Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers. pp. 1715–1724 (2016) 2.1
13. Jin, W., Qu, M., Jin, X., Ren, X.: Recurrent event network: Autoregressive structure inference over temporal knowledge graphs. arXiv preprint arXiv:1904.05530 (2019) 1.2, 2.1
14. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. arXiv preprint arXiv:1609.02907 (2016) 1.1, 3.3
15. Kumar, S., Zhang, X., Leskovec, J.: Predicting dynamic embedding trajectory in temporal interaction networks. In: Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining. pp. 1269–1278 (2019) 1.2, 2.2
16. Leblay, J., Chekol, M.W., Liu, X.: Towards temporal knowledge graph embeddings with arbitrary time precision. In: Proceedings of the 29th ACM International Conference on Information & Knowledge Management. pp. 685–694 (2020) 2.1
17. Lin, Y., Liu, Z., Sun, M., Liu, Y., Zhu, X.: Learning entity and relation embeddings for knowledge graph completion. In: Twenty-ninth AAAI conference on artificial intelligence (2015) 1.1, 2.1

ICCS Camera Ready Version 2022
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-08760-8_2

18. Maheshwari, A., Goyal, A., Hanawal, M.K., Ramakrishnan, G.: Dyngan: generative adversarial networks for dynamic network embedding. In: Graph representation learning workshop at NeurIPS (2019) 2.2
19. Manessi, F., Rozza, A., Manzo, M.: Dynamic graph convolutional networks. Pattern Recognition **97**, 107000 (2020) 2.2
20. Nestor, M.: Github has been under a continuous ddos attack in the last 72 hours (2015) 1.1
21. Niepert, M., Ahmed, M., Kutzkov, K.: Learning convolutional neural networks for graphs. In: International conference on machine learning. pp. 2014–2023. PMLR (2016) 1.1
22. NIST: National vulnerability database. `https://nvd.nist.gov/` (2018) 5
23. opentsdb: Opentsdb, `http://opentsdb.net/` 3.3, 3.3
24. Pingle, A., Piplai, A., Mittal, S., Joshi, A., Holt, J., Zak, R.: Relext: Relation extraction using deep learning approaches for cybersecurity knowledge graph improvement. In: Proceedings of the 2019 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining. pp. 879–886 (2019) 2.3
25. Rastogi, N., Dutta, S., Zaki, M.J., Gittens, A., Aggarwal, C.: Malont: An ontology for malware threat intelligence. In: International Workshop on Deployable Machine Learning for Security Defense. pp. 28–44. Springer (2020) 2.3
26. Samtani, S., Zhu, H., Chen, H.: Proactively identifying emerging hacker threats from the dark web: A diachronic graph embedding framework (d-gef). ACM Transactions on Privacy and Security (TOPS) **23**(4), 1–33 (2020) 1.1, 1.2, 2.3, 4.1, 4.3
27. Sarhan, I., Spruit, M.: Open-cykg: An open cyber threat intelligence knowledge graph. Knowledge-Based Systems **233**, 107524 (2021) 2.3
28. Shu, X., Araujo, F., Schales, D.L., Stoecklin, M.P., Jang, J., Huang, H., Rao, J.R.: Threat intelligence computing. In: Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security. pp. 1883–1898 (2018) 2.3
29. Trivedi, R., Farajtabar, M., Biswal, P., Zha, H.: Dyrep: Learning representations over dynamic graphs. In: International conference on learning representations (2019) 2.2
30. Trivedi, R., Farajtabar, M., Wang, Y., Dai, H., Zha, H., Song, L.: Know-evolve: Deep reasoning in temporal knowledge graphs. arXiv preprint arXiv:1705.05742 (2017) 1.2, 2.1
31. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. Advances in neural information processing systems **30** (2017) 1.2
32. Wang, J., Song, G., Wu, Y., Wang, L.: Streaming graph neural networks via continual learning. In: Proceedings of the 29th ACM International Conference on Information & Knowledge Management. pp. 1515–1524 (2020) 1.2, 2.2
33. Wang, Z., Zhang, J., Feng, J., Chen, Z.: Knowledge graph embedding by translating on hyperplanes. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 28 (2014) 2.1
34. Xu, C., Nayyeri, M., Alkhoury, F., Yazdi, H.S., Lehmann, J.: Temporal knowledge graph embedding model based on additive time series decomposition. arXiv preprint arXiv:1911.07893 (2019) 1.2, 2.1
35. Zaremba, W., Sutskever, I., Vinyals, O.: Recurrent neural network regularization. arXiv preprint arXiv:1409.2329 (2014) 1.1, 3.3

ICCS Camera Ready Version 2022
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-08760-8_2