




Article

# A Novel Time Constraint-Based Approach for Knowledge Graph Conflict Resolution

Yanjun Wang<sup>1,2</sup>, Yaqiong Qiao<sup>1</sup>, Jiangtao Ma<sup>2,3</sup> , Guangwu Hu<sup>4,\*</sup> , Chaoqin Zhang<sup>2,3</sup>,  
Arun Kumar Sangaiah<sup>5</sup> , Hongpo Zhang<sup>1,6</sup> and Kai Ren<sup>6</sup>

<sup>1</sup> State Key Laboratory of Mathematical Engineering and Advanced Computing, Zhengzhou 450000, China; wyj@zzuli.edu.cn (Y.W.); kitesmile@126.com (Y.Q.)

<sup>2</sup> School Of Computer and Communication Engineering, Zhengzhou University of Light Industry, Zhengzhou 450002, China; majiangtao@zzuli.edu.cn (J.M.); 2000007@zzuli.edu.cn (C.Z.)

<sup>3</sup> National Digital Switching System Engineering & Technological R&D Center, Zhengzhou 450002, China

<sup>4</sup> School of Computer Science, Shenzhen Institute of Information Technology, Shenzhen 518172, China

<sup>5</sup> School of Computer Science and Engineering, VIT University, Vellore 632014, India; arunkumarsangaiah@gmail.com

<sup>6</sup> Cooperative Innovation Center of Internet Healthcare, Zhengzhou University, Zhengzhou 450000, China; zhp@zzu.edu.cn (H.Z.); renkai@zzu.edu.cn (K.R.)

\* Correspondence: hugw@szit.edu.cn; Tel.: +86-755-8922-6342

Received: 20 September 2019; Accepted: 14 October 2019; Published: 17 October 2019



**Abstract:** Knowledge graph conflict resolution is a method to solve the knowledge conflict problem in constructing knowledge graphs. The existing methods ignore the time attributes of facts and the dynamic changes of the relationships between entities in knowledge graphs, which is liable to cause high error rates in dynamic knowledge graph construction. In this article, we propose a knowledge graph conflict resolution method, knowledge graph evolution algorithm based on deep learning (Kgedl), which can resolve facts confliction with high precision by combing time attributes, semantic embedding representations, and graph structure features. Kgedl first trains the semantic embedding vector through the relationships between entities. Then, the path embedding vector is trained from the graph structures of knowledge graphs, and the time attributes of entities are combined with the semantic and path embedding vectors. Finally, Kgedl uses a recurrent neural network to make the inconsistent facts appear in the dynamic evolution of the knowledge graph consistent. A large number of experiments on real datasets show that Kgedl outperforms the state-of-the-art methods. Especially, Kgedl achieves 23% higher performance than the classical method numerical Probabilistic Soft Logic (nPSL).in the metric HITS@10. Also, extensive experiments verified that our proposal possess better robustness by adding noise data.

**Keywords:** knowledge graph; knowledge conflict resolution; neural recurrent network

## 1. Introduction

Automatic construction of knowledge graphs is an active research area [1–3]. Projects such as DBpedia [4], KnowItAll [5], Read The Web [6], and Yet Another Great Ontology (YAGO) [7] have achieved high accuracy and recall rates from structured knowledge representations built from unstructured or semi-structured Web resources. In the process of automatically constructing knowledge graphs, there will be cases of inconsistent knowledge [8–10], which is called knowledge conflicts in knowledge graphs. The recently popular knowledge graph embedding methods embed the entities and relationships into a continuous vector space, while maintaining the structure of the graph. These methods have shown good results in knowledge graph conflict resolution. However, the existing embedding models ignore the time attributes of facts [11], so the performance is not satisfactory

when dealing with knowledge conflicts brought about by time attributes. However, in the real world, many facts are not static but change over time. Therefore, time attributes should play an important role in the conflict resolution of knowledge graphs. This article focuses on the knowledge conflicts of time-sensitive knowledge graphs.

Most of the facts in the real world change dynamically over time, so these facts are time-sensitive in knowledge graphs. The corresponding entities in the knowledge graphs of persons and their relationships also need to be constantly updated to suit constantly changing real facts. The change in entities and their relationships in knowledge graphs is called the knowledge graph evolution process. In the evolution process of knowledge graphs, problems due to noise, inaccuracy, and even errors are often encountered. These errors will cascade during knowledge reasoning and graph evolution [12], which causes inconsistencies and fact conflicts in knowledge graphs. Therefore, to solve the knowledge conflicts that are inconsistent in the process of knowledge graph evolution, it is necessary to propose new methods to ensure the consistency of entities and their relationships.

A major shortcoming of the existing knowledge graphs is the lack of time attributes in presentation models and queries. Therefore, these extraction techniques work well if we evaluate each extracted fact individually, but the time limit plays a major role when we need to reason multiple facts from their temporal context. The main reason is the error detection and time stamping caused by the diversity of timetables in the text as well as the inconsistencies of different data sources.

To solve the shortcomings of the existing knowledge graph conflict resolution methods, we propose a new time-constrained knowledge graph conflict resolution method, knowledge graph evolution algorithm based on deep learning (Kgedl), which uses entity and relationship vector embedding and path-based searches to represent entity relationships. Kgedl proposes to use the time conflict restriction method to resolve the time-conflicting entity relationship. A large number of experimental results have shown the effectiveness of Kgedl. In summary, the main contributions of this article are as follows:

- (1) We propose a knowledge graph conflict resolution method Kgedl, which is based on entity and relationship vector embedding and path finding. Kgedl takes the time attributes of facts as an important feature in the evolution process of knowledge graphs. As far as we know, this is the first attempt to resolve conflicts in time-sensitive knowledge graphs. To incorporate time attributes, we propose a time-sensitive embedding model that encodes and embeds timing information into the geometry of the embedding space;
- (2) We propose to use time disjoint, pre-relationship, and mutual exclusion to constrain the facts of time-based conflicts. The method of the recurrent neural network is used to adjust the inconsistencies in conflict detection, thus ensuring knowledge consistency during the evolution of knowledge graphs;
- (3) We create a time-based knowledge graph based on the Yet Another Great Ontology (YAGO) and Internet Movie Database (IMDB) datasets. A large number of experiments on this knowledge graph show that Kgedl is superior to the existing baseline method, Kgedl's performance at the metric HITS@10 is 23% higher than the classical method numerical Probabilistic Soft Logic (nPSL), which confirms the effectiveness of incorporating time information to resolve knowledge graph conflicts and proves that this method is robust by adding noise data.

## 2. Problem Formulation

Add a valid time  $[b, e]$  to each fact of the knowledge graph to obtain the temporal knowledge graph (TKG), where  $TKG = \langle F, C, Q \rangle$ ,  $C$  is the timing constraint condition,  $Q$  is the query condition of the knowledge graph, and  $F_Q$  is the set of all the facts involved in the query condition  $Q$ ,  $F_Q \subseteq F$ . The timing fact is expressed as the probability that  $f = \langle s, p, o, w, [b, e] \rangle$ ,  $f \in F$ , and  $w$  is the probability of the fact being true. Here is Keith Brian Alexander's example of his career:

1.  $\langle \text{Alexander, retirement from, NSA, 1.0, [2014, now]} \rangle$
2.  $\langle \text{Alexander, served as, the 1st Commander of the United States Cyber Command, 1.0, [2010, 2014]} \rangle$

3. <Alexander, served as, the 16th Director of the National Security Agency, 1.0, [2005, 2014]>
4. <Alexander, served as, the Deputy Chief of Staff G-2 in U.S. Army, 0.9, [2003, 2005]>
5. <Alexander, served as, the Commanding General of the U.S. Army Intelligence and Security Command, 0.7, [2001, 2003]>

If you want to check out Alexander’s job positions from 2010 to 2014, from the above facts we know that although there is overlap between facts (2) and (3), there is no conflict because Alexander served two positions in 2010–2014. This query-based conflict resolution is described by the following method.

For TKG, a query Q based on time limit C is given, and the query result F<sub>Q</sub> is returned, but F<sub>Q</sub> will contain some conflicting facts. The goal of this article is to perform conflict resolutions in F<sub>Q</sub>, thus returning accurate query results. Therefore, it is necessary to maximize the sum of the consistency fact weights to minimize the factual conflict, as shown in the following condition (1):

$$\max \sum_{f \in F_{Q,C}} w(f) \quad \text{s.t. } \forall f_{Q,C} \in F_{Q,C}, \text{Confidence}(C, f_{Q,C}) > \theta. \tag{1}$$

Here, Confidence (C, f<sub>Q,C</sub>) represents the confidence of the fact f<sub>Q,C</sub> under the timing limit C. When this confidence is greater than threshold θ, this fact is added to the set of query results. Therefore, the evolution goal of the knowledge graph is to maximize the sum of the weights for the facts of the query results and resolve the factual conflicts of the query results, ensuring the consistency of knowledge during evolution.

### 3. Knowledge Conflict Resolution Model

In this section, we present a knowledge conflict resolution model for knowledge graph construction. The specific framework is shown in Figure 1. Firstly, this model learns the semantic embedding from the knowledge graph. Then, it learns the path embedding from the graph features of the knowledge graph. Thirdly, it learns the time embedding features from the evolving knowledge graph. Fourth, semantic embedding, path embedding, and time embedding are combined as the input of the recurrent neural network. Finally, the conflict facts are dispelled and the tuple in the knowledge graph can update dynamically in the recurrent neural network. In each layer of the recurrent neural network the conflict fact is replaced with the new relationship with high confidence, therefore the conflict fact is deleted from knowledge graph, and the new fact is added into the knowledge graph. This scheme uses the semantic embedding representation between entities and the path embedding representation between entities, and it adds time series representation according to the appearance of new facts. The recurrent neural network is used to dynamically update the knowledge graph and use the time-limited factual conflict resolution method to solve the knowledge conflict problem in the process of knowledge graph dynamic update.

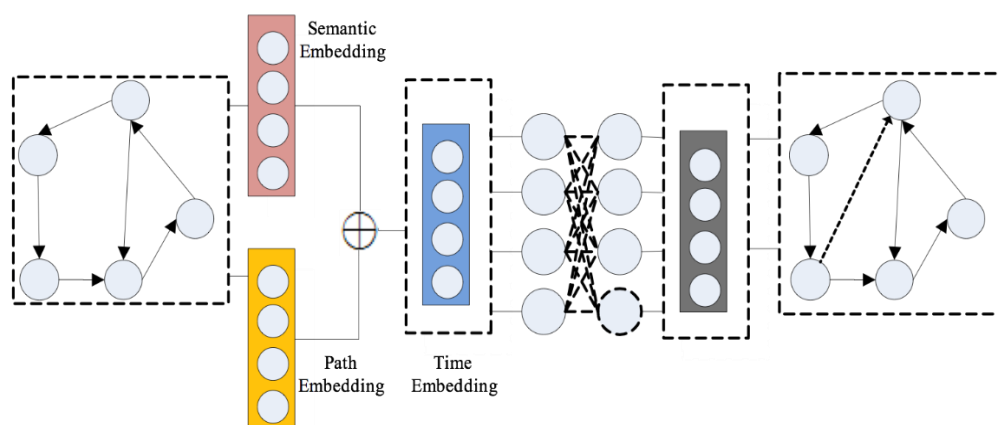


Figure 1. Knowledge conflict resolution model in a knowledge graph.

### 3.1. Uncertainty Temporal Knowledge Graph

The uncertainty temporal knowledge graph is a combination of temporal knowledge graphs and probability graph models. It can represent both the knowledge of uncertainty and the time feature of knowledge. Each fact has a valid time and confidence value. The semantics of the temporal knowledge graph are based on the joint probability distribution of the uncertain parts of the knowledge graph. The fact weight in TKG determines the log-linear probability distribution. As mentioned earlier, the effective time domain of the fact is discrete and finite, so the possible state of the fact is also limited. For the time–history knowledge graph  $TKG = \langle F, C, Q \rangle$ ,  $G = \langle f, c, q \rangle$  is the query subset based on query  $q$  and constraint  $c$ , and  $G \subseteq TKG$  is the probability distribution of the facts in the query  $G$ , based on Equation (2):

$$P(G) = \frac{1}{Z} \exp \left( \sum_{(f_i, w_{f_i}) \in F} w_{f_i} \right), \tag{2}$$

where  $f \subseteq F$ ,  $c \subseteq C$ ,  $q \subseteq Q$ ,  $i$  is the number of facts,  $Z$  is the normalization constant of the log-linear probability distribution. We used the maximum probability inference to find the subset of the temporal knowledge graph with the highest probability and use rules to limit queries on the temporal knowledge graph to identify conflicting facts.

### 3.2. Uncertain Time Conflicts in Knowledge Graphs

Knowledge graphs often contain large amounts of numerical data, such as time, weight, and length. There may be inconsistencies in the corresponding numerical data of facts during the evolution of knowledge graphs. One way to solve this problem is to use a series of probability limits to calculate the maximum a posteriori estimate of the given fact in the knowledge graph, and to derive the weight or confidence of the fact. For example, an uncertain knowledge contains the following two facts: (1)  $\langle \text{Alexander, retirement, 1.0, [2014, now]} \rangle$ , (2)  $\langle \text{Alexander, served as the first US Cyber Force Commander, 1.0, [2010, 2015]} \rangle$ . Since Alexander retired in 2014 and the second fact has a working time range of [2010, 2015], these two facts have time conflicts. To eliminate the conflicts, we can ensure the validity of the data by adding restrictions. This article added constraints to the uncertainty knowledge graph to ensure the validity of numerical properties.

### 3.3. Constrained on Time Conflication Among Facts

We mainly considered three types of time disjoint, temporal precedence, and time mutual exclusion. These three types can handle most of the time-constrained knowledge conflicts.

The time disjoint limit is to limit the time interval between any two relationships, while avoiding the existence of two facts with the same head entity and relationship as well as the overlaps of different tail entities, as shown in condition (3):

$$R(e_i, e_j, t_1) \wedge R(e_i, e_k, t_2) \wedge (e_j \neq e_k) \rightarrow disjoint(t_1, t_2), \tag{3}$$

where  $e_j$  and  $e_k$  are not the same entity, and the times of  $t_1$  and  $t_2$  cannot overlap. For example, in a marriage relationship, a person cannot maintain a marital relationship with two persons at the same time.

The temporal precedence restriction ensures the order in which some pairs of relationships occur. The end time of relationship  $R_1$  is earlier than the start time of  $R_2$  and the temporal precedence constraints can be expressed by condition (4):

$$R_1(e_i, e_j, t_1) \wedge R_2(e_i, e_k, t_2) \rightarrow before(t_1, t_2). \tag{4}$$

For example, one obvious constraint is that a person’s date of birth should precede the time they joined the work. The knowledge graph conflict problem often appears when the query on knowledge

graph is invoked, so the time-constrained confliction will be resolved on the query time with knowledge graph. For example, there are two relationships below, the first is “<Alexander, retirement from, NSA,1.0, [2014, now]>”, the second one is “<Alexander, served as, the 1st Commander of the United States Cyber Command, 1.0, [2010, 2015]>”, there is a time-constrained confliction between [2014, now] and [2010, 2015], which will appear when an Alexander’s position query after 2014 is invoked.

The mutually exclusive relationship of facts means that a relationship  $R$  cannot appear at the same time, which can be described as condition (5):

$$R(e_i, e_j, t) \wedge R(e_i, e_k, t) \wedge (e_j \neq e_k) \rightarrow false. \tag{5}$$

For example, a student cannot attend elementary school in two different cities at the same time. In order to solve the problem of time conflicts in the process of knowledge graph evolution, we use semantic embedding, path embedding, and time embedding methods based on the recurrent neural network to resolve the knowledge conflicts generated during the evolution of knowledge graphs and predict the relationship between entities. Therefore, the knowledge graph can change dynamically.

### 3.4. Semantic Embedding and Path Embedding with RNN

Suppose there is a path  $(r_1, r_2, \dots, r_l)$  between two entities, inspired by the idea of relationship path embedding [13], which is built via path composition of relation embedding. We utilize Equation (6) to get the path embedding between two entities:

$$\mathbf{p} = \mathbf{r}_1 + \dots + \mathbf{r}_l. \tag{6}$$

We utilize semantic embedding to represent the relationship between entities, and semantic embedding is realized with Recurrent Neural Network (RNN), which is a neural network model for learning semantic embedding vector. RNN is utilized to update the conflict of the fact in knowledge graph. The operation is realized using a matrix  $W$  through Equation (7):

$$s_i = w(W[\mathbf{s}_{i-1} \times \mathbf{p}_i]), \tag{7}$$

where  $w$  is a non-linearity function, and  $[\mathbf{s}_{i-1} \times \mathbf{p}_i]$  represents the concatenation of the semantic embedding vector and path embedding vector. The semantic embedding vector is built via semantic composition of relationship embedding. By setting  $\mathbf{s}_1 = \mathbf{p}_1$  and recursively performing RNN following the relation path, we finally obtain the embedding representation of two entities’ relationship  $\mathbf{s}_l$ .

### 3.5. Knowledge Conflict Resolution Model

In this section, the semantic embedding vector between entities, the graph structure embedding in the knowledge graph, and the time attributes of facts are integrated into the recurrent neural network. With the addition of new facts, the relationship of entities in knowledge graphs changes. The network compares the facts that are newly entered into the knowledge graph with the existing facts and selects a more accurate entity relationship to update the original knowledge graph. It uses the latent representation of the knowledge graph and the most recent event to predict the event. In the prediction model, the event tensor  $E$  and TKG are used together to predict the relationship between the entities. Event tensor is composed with relationship embedding with time attribute. The general form is described as Equation (8):

$$\beta_{s,p,o,t}^{predict} = f_{p,o}^{predict}(a_{e_s}, a_{e_{s,t}}, \dots, a_{e_{s,t-T}}). \tag{8}$$

The prediction is based on the subject ( $s$ ), object ( $o$ ), and predicate ( $p$ ) of the knowledge graph tensor and the time-based representation of the event tensor.  $a_{e_s}$  represents the entity’s attribute feature of time  $t$ , which can be gained from the knowledge graph model. In the prediction model, the tuples in

the knowledge graph are updated by events, the changes in the knowledge graph are first reflected in the event tensor, and the prediction of the events is also included in the prediction of the relationships between entities in the knowledge graph. The change in the event affects the change in the state of the knowledge graph and also affects the latent representation of the knowledge graph. The attribute information of the entity will change accordingly, and the relationship between the entities needs to be re-estimated.

The facts that exist in the knowledge graph are positive examples. Therefore, we used the local closed world [14] hypothesis to get a negative example and a negative log-likelihood cost function for the Bernoulli likelihood estimation, which is described as  $-\log P(x|\theta) = \log [1 + \exp\{(1-2x)\theta\}]$ , where  $x$  represents a subset of related facts in the training set:

$$J(\beta)^{predict} = - \sum_{e_{s,p,o,t} \in E} \log P(e_{s,p,o,t} | \beta_{s,p,o,t}^{predict}(L, M, W)). \quad (9)$$

The matrix  $M$  represents the relationship between entities, using the knowledge graph model to initialize the latent representation  $L$ , the latent representation  $L$  and  $M$  mapping to describe the cost function, and  $W$  being the parameter of the function transformation. In the prediction task, the recurrent neural network is used to predict the event model. The cost function of the prediction model is described in Equation (9).

Perozzi et al. [15] found that the frequency distribution of node generated by the random walk method in the network is similar to the frequency of occurrence of word in our method. They all obey the power law distribution. Inspired by this phenomenon, we used the random walk method to sample the neighbors of the nodes in the knowledge graph (similar to the context of the word) so that the graph structure of the knowledge graph still maintained the relationship between the entities in the vector space. Nodes are entities in the knowledge graph, and entity nodes are also used in the following text. We used a semi-supervised approach to learn the structural features of graphs edge between two entity nodes in a knowledge graph. In the link prediction task between knowledge graph entities, the graph structure feature can be used to predict whether there is an edge of two entity nodes in the knowledge graph. Because the random walk method is based on the connected structure of the entity nodes in the knowledge graph, we used an iterative method to extend the features of a single entity node into a pair of entity nodes.

Semantic embedding is acquired through training the relationships of the entities. Path embedding is acquired through training the structure of the knowledge graph. The output of the diagram is a knowledge graph without relationship confliction. The confidence score is the intermediate result of a new relation, when the confidence score is greater than the conflicted relationship's confidence score, the new relationship will replace the conflicted relation. Firstly, the proposed model predicted the confidence between entities in the new event based on the semantic embedding and the path-based embedding vector. Secondly, the proposed model compared with the confidence of the original fact. If the difference between the two confidences was greater than a certain threshold, then the old facts were replaced with the newly added facts to achieve the renewal of the knowledge graph. The proposed algorithm Kgedl (knowledge graph evolution algorithm based on deep learning) is shown in Algorithm 1.

**Algorithm 1.** Knowledge graph evolution algorithm based on deep learning.

Input: F is the set of tuples in TKG and new facts,  $f = \langle s, p, o, w, [b, e] \rangle$ ,  $f \in F$ , query Q, time constraint set C and query results  $F_{Q,C}$ ,  $f' \in F_{Q,C}$ ;

Output: updated TKG.

1: for ( $j = 0$ ;  $j < |f|$ ;  $j++$ )

2: {

3:  $\mathbf{p} = \mathbf{r}_1 + \dots + \mathbf{r}_l$  // get the path embedding between entity  $s$  and  $o$

4:  $s_i = w(W[s_{i-1} \times \mathbf{p}_i])$  // RNN is utilized to get the concatenation of path embedding and semantic embedding

5:  $P_{embedding}(s, p, o, w, b, e) = f_{p,o}^{predict}(a_{e_s}, a_{e_{s-t-b}}, \dots, a_{e_{s,t-e}})$  // Calculate the probability of fact to be truth from the perspective of embedding semantics

6:  $P(o|o', R_n) = \frac{R_n(o', o)}{|R_n(o', \cdot)|}$  // The probability that node  $o'$  jumps to  $o$  along the edge of type  $R_n$

7:  $h_{s,p}(o) = \sum_{o' \in range(P')} h_{s,p'}(o') \cdot P(o|o', R_n)$

8:  $P_{path}(o, s) = \sum_{P \in P_n} h_{s,p(o)} \theta_P$  // Path score probability of node  $s$  to  $o$

9:  $P(s, o) = \alpha P_{embedding}(s, o) + (1 - \alpha) P_{path}(o, s)$  // Probability scores of semantic embedding and path embedding for nodes  $s$  to  $o$

10:  $P(G) = \frac{1}{Z} \exp \left( \sum_{(f_i, w_{f_i}) \in F} w_{f_i} \right)$  // Estimate the probability distribution of the facts to be true in the query subset

11: if  $(|P(f) - P(f')| > \theta)$  // If the confidence of the two facts is greater than the threshold

12: {

13:     deletfromTKG( $f'$ );

14:     addtoTKG( $f$ );

15:     } //updating TKG

16: }

#### 4. Experimental Results and Analysis

The experiment compared the effect of Kgedl and baseline methods on detecting knowledge conflicts in knowledge graphs on two datasets, and it added noise data in the dataset to verify the robustness of Kgedl. We measured the performance of Kgedl using accuracy, recall, and F1 values. Experiments were carried out on a Central Processing Unit (CPU) with Intel Celeron E5-2620 V3, NVIDIA Tesla K80 GPU, 128G main memory, and CentOS6.4 Linux operating system, and the algorithms in the experiment were implemented in TensorFlow with C++.

We used accuracy, recall, and F1 values to evaluate the proposed method of detecting knowledge conflicts. The accuracy rate (P) indicates the fraction of the detected conflict facts that are indeed conflict facts. The recall rate (R) indicates the ratio of the detected conflict facts to all the conflict facts, and F1 is the harmonic mean of P and R,  $F1 = 2P \times R / (P + R)$ . The x-axis is the recall rate, the y-axis is the accuracy, and the composition curve is the Receiver Operating Characteristic (ROC) curve. The area under the ROC curve is named as Area Under the ROC Curve (AUC). And the larger value of AUC, the better the method's performance.

Since there is no public time series knowledge graph dataset, this experiment extracted the timing facts from the IMDB and YAGO datasets: the movie data from 2007 to 2017 were extracted from the IMDB website, including the release time of the movie and the birth time of the actor and related facts. We also extracted time series facts from YAGO's yagoMetaFacts dataset, which contains time and space information from YAGO meta facts. For example, birth time and graduation time are time-sensitive relationships, and location relationships are not time-sensitive information. The timing fact format in YAGO is  $\langle \text{factID}, \text{occurSince}, b \rangle$  and  $\langle \text{factID}, \text{occurUntil}, e \rangle$ , indicating that this fact is true within the time period  $[b, e]$ . Here factID represents a specific fact  $\langle s, p, o \rangle$ . This article directly uses  $\langle s, p, o, w, [b, e] \rangle$  to represent a tuple. In the experiment, 12 commonly used time-sensitive relationships were selected as time series datasets, and then a subset of entities where both entities exist in the time series were selected. This produced 36,754 tuple facts, which were then randomly divided according to a

certain ratio to generate training sets and test sets. The statistics of the datasets YAGO and IMDB are shown in Table 1. All experiments were repeated five times using a re-divided training/validation/test set, and the average results of the five experimental results were finally given.

**Table 1.** YAGO (Yet Another Great Ontology) and IMDB (Internet Movie Database) datasets statistics.

Dataset	#Rel	#Ent	#Train	#Valid	#Test	#Facts
YAGO	12	10,712	29,403	3675	3676	36,754
IMDB	10	11,364	38,836	4854	4856	48,546

We selected three current mainstream knowledge graph evolution methods, i.e., Path Ranking Algorithm (PRA) [16], Translation in the corresponding Relation space (TransR) [17] and nPSL [18], as the baseline methods to compare with Kgedl. PRA is a path-sorting reasoning method based on the random walk method. It mainly uses the graph structure features of knowledge graphs to infer the potential relationship between entities. PRA learns the relationship between entities and classifies the relationship using the logistic regression method. Relationship features are the main features of the classifier. The TransR method maps two relationships in a tuple to the same vector space. It can cluster different entity nodes, learn the vector representation of the relationship in different clusters, and judge the entity relationship according to the inconsistency of the vector, i.e., conflicts that result in consistent entity relationships in knowledge graphs. nPSL is a time series inference method based on the probabilistic soft logic. It numerically expands the probabilistic soft logic to detect the fact that there is time conflict in knowledge graphs.

In the experiment, we used Kgedl and the baseline methods to perform conflict detection in the YAGO and IMDB datasets with uncertain timing facts. In the experiment, 10%, 20%, 30%, 40%, 50%, 60%, 70%, and 80% error facts were injected into the dataset. For example, adding a 10% error fact means that each of the three types of relationships added a 10% error fact. Table 2 gives the AUC of Kgedl under different noises. It can be seen that even if 80% of the error facts were added, the AUC value in Kgedl was still 0.89, which fully demonstrates that the proposed method is robust.

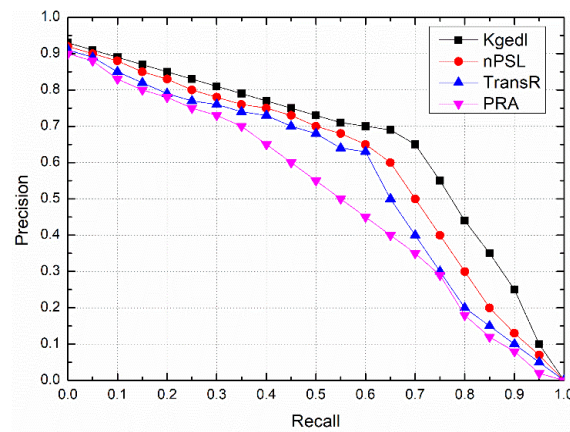
**Table 2.** The Area Under the Receiver Operating Characteristic (ROC) Curve (AUC) of Kgedl with different scale of noise data.

Noise	0%	10%	20%	30%	40%	50%	60%	70%	80%
YAGO'AUC	0.94	0.93	0.92	0.91	0.9	0.9	0.89	0.9	0.89
IMDB'AUC	0.95	0.94	0.94	0.93	0.92	0.91	0.91	0.9	0.89

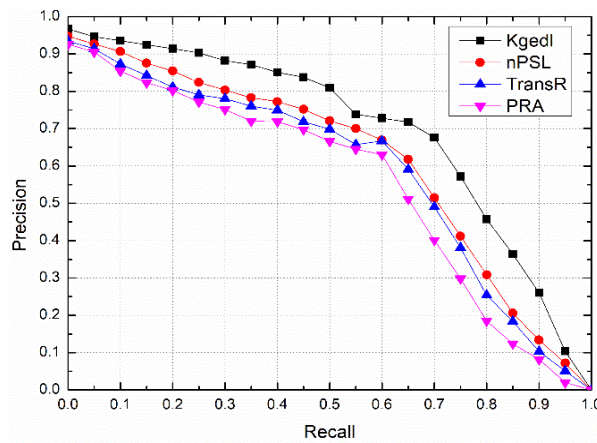
Figure 2 shows the comparison of Kgedl with several baseline methods on the YAGO dataset. It can be seen that Kgedl was obviously superior to the baseline methods. When the recall rate was 5%, the accuracy rates of Kgedl and nPSL, TransR, and PRA were 0.91, 0.9, 0.89, and 0.88, respectively. When the recall rate was 50%, the accuracy rates of Kgedl and nPSL, TransR, and PRA were 0.73, 0.7, 0.68, 0.55, respectively, and when the recall rate was 90%, the accuracy of Kgedl and nPSL, TransR, and PRA were 0.25, 0.13, 0.1, and 0.08, respectively.

The experimental results on the IMDB dataset are shown in Figure 3. The accuracy rates of Kgedl and nPSL, TransR, and PRA were 0.946, 0.927, 0.914, 0.905 when the recall rate was 5%. When the recall rate rose to 50%, the accuracy of Kgedl, nPSL, TransR, and PRA dropped to 0.809, 0.721, 0.698, 0.665, and when the recall rate rose to 90%, the accuracy of Kgedl, nPSL, TransR, and PRA dropped to 0.26, 0.133, 0.102, 0.08.



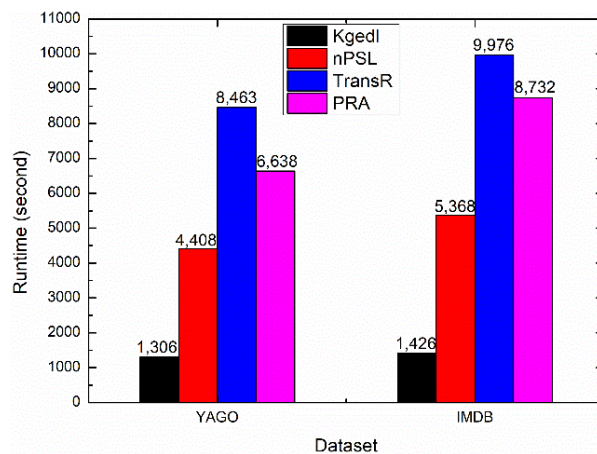


**Figure 2.** The recall rate precision comparison results between Kgedl and baseline methods on the YAGO dataset.



**Figure 3.** The recall rate precision comparison results between Kgedl and baseline methods on the IMDB dataset.

Figure 4 compares the operating efficiency of Kgedl with the baseline methods, from which it can be seen that Kgedl had the best performance. On the YAGO dataset, Kgedl’s runtime was 30% of nPSL’s and 15% of TransR’s. On the IMDB dataset, Kgedl’s runtime was 27% of nPSL’s and 14% of TransR’s.



**Figure 4.** The runtime comparison between Kgedl and three baseline methods on the YAGO and IMDB datasets.

The effect of solving the knowledge conflict problem in the evolution process of the Kgedl knowledge graph has been discussed above, and the method was used to test its performance in link prediction in the following experiment. Given the test tuple  $\langle s, p, o, w, [b, e] \rangle$ , all entities in the YAGO dataset were replaced with the entity of the base facts and relationships with the target entity for all candidate entities in the tuple were generated. The probability was sorted in a descending order, and Mean Absolute Rank (MAR), which is the ranking of the entity corresponding to the link prediction in the candidate entity. The smaller MAR, the better the link prediction method. STandard DEVIation (STDEV) for MAR and HITS@10 were evaluated in the experiment.

Figure 5 shows the effect of Kgedl and the baseline methods on link prediction problem on the YAGO dataset. It can be seen that Kgedl was significantly better than those baseline methods. The MAR value obtained by Kgedl was 22 higher than the average of the second-ranked method nPSL. It can be seen from the figure that the prediction ability of the nPSL method was also good, but it cannot capture dynamic information.

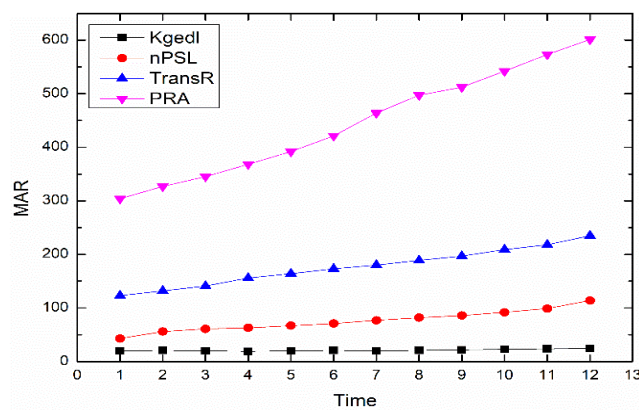


Figure 5. The Mean Absolute Rank (MAR) link prediction comparison result between Kgedl and baseline methods on the YAGO dataset.

Figure 6 shows the standard deviation of MAR. It can be seen from the figure that the deviation of Kgedl was the smallest, which was 154 lower than the following method nPSL.

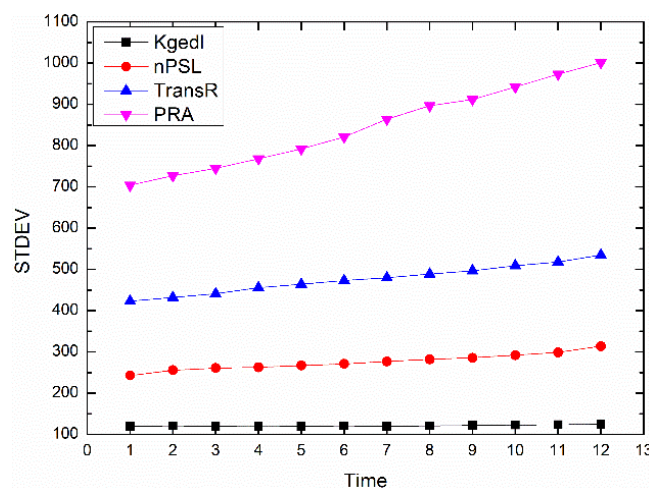


Figure 6. The link prediction comparison of STandard DEVIation (STDEV) of MAR between Kgedl and baseline methods on the YAGO dataset.

In order to evaluate the effect of path embedding and semantic embedding, we tested Kgedl without path embedding and semantic embedding, which was called Kgedl0. Kgedl0 utilizing path embedding was named Kgedl0 + path, and Kgedl0 utilizing semantic embedding was called

Kgedl + semantic. These methods were tested on the YAGO dataset. Figure 7 gives the comparisons of experimental results, the area under the curve for Kgedl was 17.32%, 30.83%, and 33.76% higher than Kgedl0, Kgedl0 + path, and Kgedl0 + semantic, respectively. The area under the curve for Kgedl0 + path was 16.47% higher than Kgedl0. We can reach the conclusion that path embedding plays an important role in the knowledge graph conflict resolution performance.

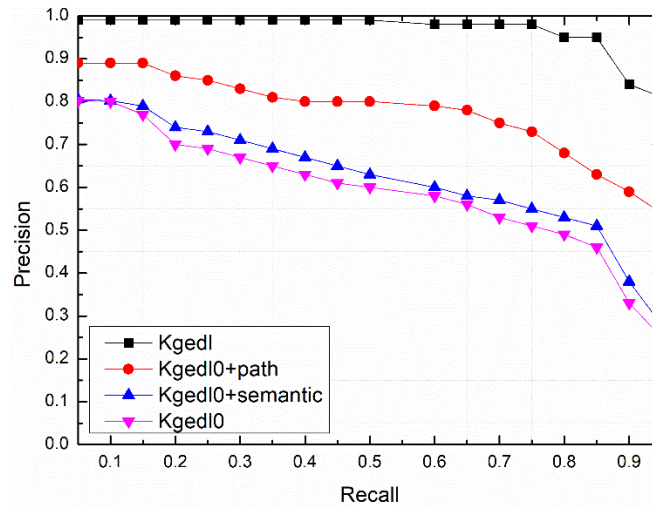


Figure 7. The comparison results of precision and recall between Kgedl and the Kgedl0, Kgedl0 + path, Kgedl0 + semantic methods on the YAGO dataset.

Figure 8 shows Kgedl’s performance at HITS@10 and demonstrates its ability to discriminate, with an average HITS@10 of 0.92, which was 23% higher than the following method nPSL. This shows that in a complex case where there are multiple relationships between entities, other methods can only rely on static entity embedding, and the Kgedl method can update the entity relationships in the knowledge graph to obtain more accurate prediction results.

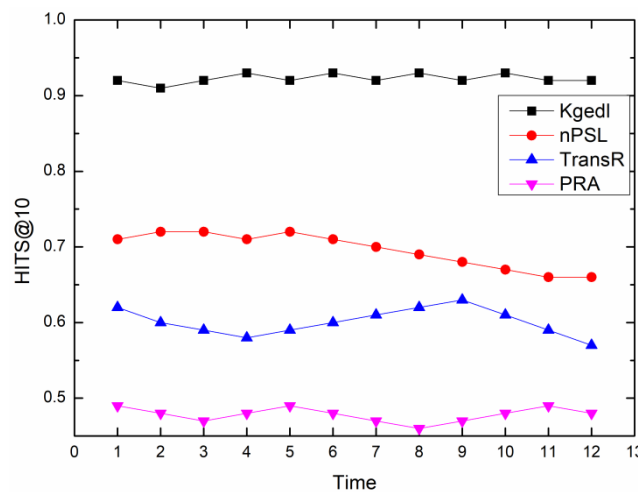


Figure 8. The performance comparison between Kgedl and three baseline methods with the metric HITS@10.

### 5. Conclusions and Future Work

In this article, the knowledge conflicts in the process of knowledge graph evolution were analyzed. Considering that existing studies have ignored the dynamic change of relationships between entities in knowledge graphs, the Kgedl method was proposed, which combines time constraints, semantic embedding representation, and graph structure together. Kgedl is based on the fact that the time

attribute incorporates the semantic embedding representation as well as the relationship and the path representation between entities. It uses a recurrent neural network to embed the time series, so that the inconsistent facts appearing in the dynamic evolution of the knowledge graph pass through the loop. Adjustments were made in the neural network to achieve and ensure consistency during the evolution of knowledge graphs over time. A large number of experiments on real datasets showed that this method is better than the current mainstream methods. By adding noise data, Kgedl proves to be robust. This indicates that it is feasible to use the recurrent neural network method to deal with the knowledge conflict in the process of knowledge graph evolution. The end-to-end graph learning using graph structure and semantic method provides a new idea and method for the dynamic evolution of knowledge graphs. In future work, we will try to apply the proposed time-constrained knowledge conflict resolution method to the dynamic evolution of knowledge graphs.

**Author Contributions:** Y.W. and J.M. conceived and designed the experiments. Y.Q. and G.H. performed the experiments. C.Z., K.R. and H.Z. analyzed the data. Y.W. and J.M. wrote the article. G.H. and A.K.S. reviewed and edited the manuscript. All the authors read and approved the manuscript.

**Funding:** This research was funded by the National Natural Science Foundation of China (No. 61702462, No. 61672470, No. 61802352, No. 61866008), the Doctoral Research Fund of Zhengzhou University of Light Industry (2017BSJJ046, 2018BSJJ039, 13501050045), Scientific & Technological Project of Henan Province (No. 182102210607), Foundation of Henan Province Educational Committee (No. 17A520064) and Fundamental Research Project of Shenzhen Municipality (No. JCYJ20170817115335418).

**Acknowledgments:** The authors would like to thank the editors and the reviewers for their comments on an earlier draft of this article.

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

## References

- Dong, X.L.; Gabrilovich, E.; Heitz, G.; Horn, W.; Murphy, K.; Sun, S.; Zhang, W. From Data Fusion to Knowledge Fusion. *Proc. VLDB Endow.* **2014**, *7*, 881–892. [[CrossRef](#)]
- Dettmers, T.; Minervini, P.; Stenetorp, P.; Riedel, S. Convolutional 2d knowledge graph embeddings. In Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, New Orleans, LA, USA, 2–7 February 2018; pp. 1811–1818.
- Shi, B.; Weninger, T. Open-world knowledge graph completion. In Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, New Orleans, LA, USA, 2–7 February 2018; pp. 1957–1964.
- Lehmann, J.; Isele, R.; Jakob, M.; Jentzsch, A.; Kontokostas, D.; Mendes, P.N.; Hellmann, S.; Morse, M.; van Kleef, P.; Auer, S.; et al. DBpedia—A large-scale, multilingual knowledge base extracted from Wikipedia. *Semant. Web* **2015**, *6*, 167–195.
- Hossain, B.A.; Schwitter, R. A survey on automatically constructed universal knowledge bases. *Sem. Web* **2018**, *6*, 1852–1873.
- Mitchell, T.; Cohen, W.; Hruschka, E.; Talukdar, P.; Betteridge, J.; Carlson, A.; Dalvi, B.; Gardner, M.; Kisiel, B.; Krishnamurthy, J.; et al. Never-Ending Learning. *Commun. ACM* **2018**, *61*, 103–115. [[CrossRef](#)]
- Rebele, T.; Suchanek, F.; Hoffart, J.; Biega, J.; Kuzey, E.; Weikum, G. YAGO: A Multilingual Knowledge Base from Wikipedia, Wordnet, and Geonames. In Proceedings of the 15th International Semantic Web Conference, Kobe, Japan, 17–21 October 2016; pp. 177–185.
- Pradhan, R.; Aref, W.G.; Prabhakar, S. Leveraging Data Relationships to Resolve Conflicts from Disparate Data Sources. In Proceedings of the Database and Expert Systems Applications, Regensburg, Germany, 3–6 September 2018; pp. 99–115.
- Zhao, B.; Rubinstein, B.I.P.; Gemmell, J.; Han, J. A Bayesian Approach to Discovering Truth from Conflicting Sources for Data Integration. *Proc. VLDB Endow.* **2012**, *5*, 550–561. [[CrossRef](#)]
- Kalchgruber, P.; Klas, W.; Jnoub, N.; Momeni, E. FactCheck—Identify and Fix Conflicting Data on the Web. In Proceedings of the Web Engineering, Cáceres, Spain, 5–8 June 2018; pp. 312–320.
- Aljefri, Y.M.; Hipel, K.W.; Fang, L. General hypergame analysis within the graph model for conflict resolution. *Int. J. Syst. Sci. Oper. Logist.* **2018**, *1*, 1–16. [[CrossRef](#)]

12. Chen, Y.; Wang, D.Z. Knowledge Expansion over Probabilistic Knowledge Bases. In Proceedings of the 2014 ACM SIGMOD International Conference on Management of Data, Snowbird, UT, USA, 22–27 June 2014; pp. 649–660.
13. Lin, Y.; Liu, Z.; Luan, H.; Sun, M.; Rao, S.; Liu, S. Modeling relation paths for representation learning of knowledge bases. *arXiv Prepr.* **2015**, arXiv:1506.00379.
14. Nickel, M.; Murphy, K.; Tresp, V.; Gabrilovich, E. A Review of Relational Machine Learning for Knowledge Graphs. *Proc. IEEE* **2016**, *104*, 11–33. [[CrossRef](#)]
15. Perozzi, B.; Al-Rfou, R.; Skiena, S. DeepWalk: Online Learning of Social Representations. In Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, New York, NY, USA, 24–27 August 2014; pp. 701–710.
16. Lao, N.; Mitchell, T.; Cohen, W.W. Random Walk Inference and Learning in a Large Scale Knowledge Base. In Proceedings of the Conference on Empirical Methods in Natural Language Processing, Edinburgh, UK, 27–31 July 2011; pp. 529–539.
17. Lin, Y.; Liu, Z.; Sun, M.; Liu, Y.; Zhu, X. Learning Entity and Relation Embeddings for Knowledge Graph Completion. In Proceedings of the AAAI, Austin, TX, USA, 25–30 January 2015; pp. 2181–2187.
18. Chekol, M.W.; Pirrò, G.; Schoenfish, J.; Stuckenschmidt, H. Marrying Uncertainty and Time in Knowledge Graphs. In Proceedings of the AAAI, San Francisco, CA, USA, 4–9 February 2017; pp. 88–94.



© 2019 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 (<http://creativecommons.org/licenses/by/4.0/>).