frontiers | Frontiers in Physics

TYPE Original Research
PUBLISHED 23 February 2023
DOI 10.3389/fphy.2023.1140733

OPEN ACCESS

EDITED BY
Leilei Chen,
Huanghuai University, China

REVIEWED BY
Xin Zhang,
Southwest Jiaotong University, China
Fang-Yuan Shi,
Ningxia University, China

*CORRESPONDENCE
Zili Zhang,
zhangzl@swu.edu.cn

SPECIALTY SECTION
This article was submitted to Statistical and Computational Physics, a section of the journal Frontiers in Physics

RECEIVED 09 January 2023
ACCEPTED 23 January 2023
PUBLISHED 23 February 2023

CITATION
Wu Y and Zhang Z (2023), Refining large knowledge bases using co-occurring information in associated KBs.
Front. Phys. 11:1140733.
doi: 10.3389/fphy.2023.1140733

Refining large knowledge bases using co-occurring information in associated KBs

Yan Wu and Zili Zhang*

College of Computer and Information Science, Southwest University, Chongqing, China

To clean and correct abnormal information in domain-oriented knowledge bases (KBs) such as DBpedia automatically is one of the focuses of large KB correction. It is of paramount importance to improve the accuracy of different application systems, such as Q&A systems, which are based on these KBs. In this paper, a triples correction assessment (TCA) framework is proposed to repair erroneous triples in original KBs by finding co-occurring similar triples in other target KBs. TCA uses two new strategies to search for negative candidates to clean KBs. One triple matching algorithm in TCA is proposed to correct erroneous information, and similar metrics are applied to validate the revised triples. The experimental results demonstrate the effectiveness of TCA for knowledge correction with DBpedia and Wikidata datasets.

KEYWORDS

abnormal information, matching algorithm, knowledge correction, Q&A systems, negative candidates

1 Introduction

Domain-oriented knowledge bases (KBs) such as Wikidata [1] and DBpedia [2] are extracted from Wikipedia articles. Since KBs are constructed automatically, some errors are imported from Wikipedia, including inconsistencies, typing errors, and numerical outliers [3–6]. One of the major errors is a range violation of triples in KBs. The problem arises when triples contain some abnormal information. For example, one triple < dbr:Andreas_Baum, dbo:nationality, dbr:Germans > is extracted from the sentence "Andreas Baum is a German politician" in DBpedia. The erroneous nationality of Andreas_Baum is "Germans," and the correct target is "Germany". Some facts use language values as the target of predicate "nationality," such as < dbr:Ami_Haruna, dbo:nationality, dbr:Japanese_language >, < dbr:Amelia_Rosselli, dbo:nationality, dbr:Italian_language >, and < dbr:Diederik_Grit, dbo:nationality, dbr:Dutch_language >. Some triples consider the value of their ethnic group or language as the object for nationality, and they violate the range value of the predicate. These incorrect triples are called abnormal information in KBs. Some triples with abnormal information have some implicit features in KBs. Usually, these abnormal triples are removed during data cleaning. Therefore, some interesting details are ignored in the application of KBs. The accuracy of knowledge greatly affects the results of question and answer (Q&A) systems with these KBs. Several published datasets explore the balance of natural language questions and SPARQL queries, ignoring errors in answers [7]. In SQuAD 2.0 [8] from extractive reading comprehension systems, there are some questions about "nationality" with erroneous answers, such as "question": "Along with German immigrants, immigrants of what nationality supported Tammany Hall?", "answers": ["Irish"] and "question": "What was Diogo Cao's nationality?", "answers": ["text": "Portuguese"]. The incorrect answers,

Irish and Portuguese, are replaced by the correct items (Ireland and Portugal). These answers have similar triples in KBs constructed from Wikipedia. KBs are used effectively in the backend of question-answering systems, e.g., IBM Watson System [9] containing YAGO [10] KBs. In order to improve the accuracy of answers in Q&A systems, our work is shifted to refine large KBs at the backend of the Q&A systems. The task focuses on cleaning and correcting errors by finding co-occurring similar triples in KBs.

Fact validation and a rule-based model are applied to detect erroneous information by searching candidates in KBs [11–15]. These cleaning algorithms are designed to look for existing errors in training datasets, but they cannot search for more errors in KBs. This study analyzes the characteristics of incorrect information and extracts the featurization of triples to improve the effectiveness of mining incorrect triples in KBs. For correcting these errors [6, 16, 17], some semantic embedding methods were designed to build a correction framework. The accuracy of the model depends on the pre-training model. For these methods, some pre-trained parameters are applied to make the correction decision. Every triple is checked for consistency. The framework is not suitable for tons of errors, i.e., for large KBs. Correction rules are acquired by rule models [18] for solving large KBs. However, positive and negative rules are generated before constructing correction rules. Correction rules are applied to solve a batch of errors. For a single error, it takes a lot of time to obtain the correction rule. Similarly, for errors without redundant information, the corresponding correction rules are not obtained.

In this study, an automatic framework, triples correction assessment (TCA), is developed to clean abnormal triples and revise these facts for refining large KBs. First, statements of erroneous triples are analyzed to acquire some new negative candidates and more negative sampling by small erroneous triples with range violations. After the process of data cleaning in TCA, small samples are used to obtain a large amount of abnormal information to clean up a large knowledge base. In our framework, the abnormal information in data cleaning is transmitted to mine interesting features for data correction. So, one triple matching method is proposed to find some repairs in target KBs by matching co-occurring triples between original and target KBs in the part of data correction. Other parts assist the whole framework to screen better correction results by similarity measures. Here, one new correction similarity is designed to acquire final repair to perform the alteration in incorrect triples. Our TCA framework is designed to correct range violations of the triple by discovering evidence triples from an external knowledge base. There are already a large number of Wikipedia-related knowledge bases, and they are quite mature and have a higher quality of triples. Our framework skips the pre-training part and further explores the relationship between KBs with the original source to correct the knowledge base. Also, our framework bridges sample inconsistencies between data cleaning and data rectification, further refining large knowledge bases.

1.1 Contributions

The novel contributions are as follows:

- An automatic framework, TCA, is developed to clean abnormal information and find consensus from other knowledge bases to correct the range errors of RDF triples.
- Some negative candidate search strategies are collected to filter abnormal information, and cross-type negative sample methods are applied to clean erroneous knowledge. Here, correction similarity metrics are designed to evaluate candidates for gathering final repairs.
- One co-occurring triple matching algorithm is designed to match similar triples to find candidates for correcting abnormal information in two different KBs.

The organization of this paper is as follows: In Sections 2 and 3, related work and preliminary materials are presented. Section 4 introduces the proposed framework containing negative candidate searching strategies and a correction model, respectively. Section 5 shows the experiments and analysis of our model. At last, the conclusion is presented in Section 6.

2 Related work

Some mistaken tails of the triples are recognized by wrong links between different KBs, and each link is embedded into a feature vector in the learning model [19]. In addition, the PaTyBRED [20] method incorporated type and path features into local relation classifiers to search triples with incorrect relation assertions in KB. Integrity rules [21] and constraints of functional dependencies [22–24] are considered to solve constraint violations in KBs. Preferred update formulations are designed to repair ABox concepts in KBs through active integrity constraints [25]. Data quality is improved with statistical features [26] or graph structure [27] by type. Liu et al. [11] proposed consensus measures to crawl and clean subject links in data fact validation. Usually, a fact-checking model is trained to detect erroneous information in KBs. Some rules are generated to perform correctness checking by searching candidate triples [13]. So, candidate triples are leveraged to find more erroneous triples for cleaning KBs. Wang et al. [14] used relational messages for passing aggregate neighborhood information to clean data. It seems inevitable that knowledge acquisition [28] is strongly affected by the noise that exists in KBs. Triples accuracy assessment (TAA) [12] is used to filter erroneous information by matching triples between the target KB and the original KBs.

Piyawat et al. [29] correct the range violation errors in the DBpedia for data cleaning. The Correction of Confusions in Knowledge Graphs [16] model was designed to correct errors with approximate string matching. The correction tower [30] was designed to recognize errors and repair knowledge with embedding methods. The incorrect facts are removed by the embedding models with the Word2vec method in KBs [17]. Embedding algorithms, rule-based models, edit history, and other approaches are leveraged to correct errors in KBs. A new family of models to predict corrections has received increasing attention in the domain of embedding methods, such as TransE [31], RESCAL [32], TransH [33], TransG [34], DistMult [35], HolE [36], or ProjE [37]. Our work focuses on associated KBs to search for similar triples and connections for KB repairs. Bader et al. [38] considered previous repair methods to correct abnormal knowledge with source codes. One error correction system [39] contains the majority of fault values in the tables and leverages the correction values as the sample

repairs. Baran et al. [39] without these prerequisites was designed for data correction in tabular data. The edit history [40] of KBs was considered in the correction models for repairing Wikidata. They ignored contextual errors in the edit history of KBs.

Mahdavi et al. [41] designed an error detection system (Raha) and updated a system (Baran) for error correction by transfer learning. Other studies correct entity type [5, 16, 42] in the task of cleaning KBs. The work of fixing bugs is carried out by checking whether the KB violates the constraints of the schema [6, 43] automatically. Some erroneous structured knowledge in Wikipedia is repaired by using pre-trained language model (LM) probes [44]. Natural language processing methods are combined with knowledge-correction algorithms [45]. Some models were designed to validate the syntax of knowledge and clean KBs, such as ORE [46], RDF:ALERTS [47], VRP [48], and AMIE [49]. Some clean systems were proposed to solve inconsistencies in tabular data [50–53]. Also, some correction systems [30, 41] are designed to refine KBs. Usually, some correction methods focus on solving specific problems [5, 6, 16, 42, 43]. Extending these studies, natural language processing methods are combined with knowledge correction algorithms [44, 45]. To solve the errors existing in structured knowledge, pre-trained models are trained to set parameters and a framework to correct errors or eliminate them [54, 55]. In these correction models, errors are predefined in the training datasets and not in the KBs. Such models ignore the process of exploring errors and fail to achieve good correction results in large KBs.

These methods are used when there is a lack of association between KBs, and these cannot be scaled to multiple large KBs. While the problem of correcting errors has been neglected in the field of knowledge application, the available repair methods mainly result in the undesired knowledge loss caused by the data removal. Triples with the correct subject are considered in this study. A method to correct these errors is posited by a post factum investigation of the KB.

3 Preliminaries

A KB (such as Wikidata) following Semantic Web standards covering RDF (Resource Description Framework), RDF Schema, and the SPARQL Query Language [56] is considered in our experiment. A KB is composed of a TBOX (terminology) and an ABox (assertions). Through the TBox level, the KB defines classes, a class hierarchy (*via* rdfsLsubClassOf), properties (relations), and property domains and ranges. The ABox contains a set of facts (assertions) describing concrete entities represented by a Uniform Resource Identifier (URI). Let K_1 and K_2 represent two KBs. K_1 is the original knowledge base for validation, and K_2 is the additional KB that is leveraged to provide matching information or correction features. The entities of two KBs are represented as E_1 and E_2, respectively. The predicates are R_1 and R_2, and the type sets of entities are T_1 and T_2 which include the domain and range of relation, respectively.

3.1 Overlapping type of entity

Two entities $e_1 \in E_1$ and $e_2 \in E_2$ are selected: e_i, $(i = 1, 2)$ is an entity with overlapping type, if e_1 and e_2 denote the same real-world facts. The connection of e_1 and e_2, can be represented as $e_1 = e_2$, and the connection of types in two entities, τ_{e_1} and τ_{e_2}, can be represented as $\tau_{e_1} = \tau_{e_2}$. Here, the entities of the KB are represented as E and the predicate as R. The KB can be symbolized as a set of triples (e_s, r, e_o) indicated as S, where e_s and $e_o \in E$ mark head and tail, respectively. $r \in R$ expresses the predicate name (relation/property) between them. For every fact (e_s, r, e_o), the formulation ϕ of KB-embedding models assigns a score, $\phi(e_s, r, e_o) \in R$, showing whether this triple is correct or not.

Most of the KB-embedding algorithms [31, 33] follow the open-world assumption (OWA), stating that KBs include only positive samples and that non-observed knowledge is either false or just missing. The negative samples (i.e., $(\cdot, r, e_o)$ or $(e_s, r, \cdot)$) are found by applying the type property of source triple (e_s, r, e_o). For instance, $(\cdot, r, e_o)$ has wrong domain property of relation and $(e_s, r, \cdot)$ has wrong range property of predicate name.

3.2 Overlapping type pair of entities

Given two triples and type pair, (e_s^1, r^1, e_o^1) and $(e_s^2, r^2 e_o^2)$ are from different KBs. If $e_s^1 = e_s^2$ and $e_o^1 = e_o^2$, (e_s^1, e_o^1) and (e_s^2, e_o^2) are defined as a strict overlapping entity pair for r_1 and r_2. The pair group of entities for r_1 and r_2 is written as $O(r_1, r_2)$ strictly.

If $\tau_{e_s^1} = \tau_{e_s^2}$ and $\tau_{e_o^1} = \tau_{e_o^2}$, $\tau_{(e_s^1, e_o^1)}$ and $\tau_{(e_s^2, e_o^2)}$ are described as a rough overlapping type pair for r_1 and r_2. The pair group of type for r_1 and r_2 is written as $O_\tau(r_1, r_2)$.

Example 1. (Monte_Masi, nationality, Australia), (Person, Country) are in K_1. (Monte Masi, country of citizenship, Egypt), (Person, country) are in K_2. For the relations "nationality" and "country of citizenship," they share the overlapping entities "Monte_Masi" and "Egypt" and the overlapping type pair (Person, country). Hence, the overlapping entity pair of predicates "nationality" and "country of citizenship" is (Monte_Masi, Australia), i.e., O (nationality, country of citizenship) = (Monte_Masi, Australia). At the same time, the overlapping type pair of relations "nationality" and "country of citizenship" is (Person, Country), i.e., O_τ(nationality, country of citizenship) = (Person, Country).

Example 1. In Figure 1, (Berlin, locatedat, Germany), (Germany, city), (Germany, country) are in the target base, and (Berlin, locatedin, Germany), (Germany, city), (Germany, country) is in the external base. The overlapping entities ("Berlin", "Germany") and the overlapping type pair (city, country) are shared in the predicates "locatedat" and "locatedin." Therefore, the overlapping entities group of predicates "locatedat" and "locatedin" is (Berlin, Germany), i.e., O (locatedat, locatedin) = (Berlin, Germany). At the same time, the overlapping type group of predicates "locatedat" and "locatedin" is (city, country), i.e., O_τ(locatedat, locatedin) = (city, country).

3.3 Evaluation measures

To fairly validate the performance of algorithms, three classical evaluation measures are used in our experiment, i.e., ***Mean_Raw_Rank***, **Precision@K**, and **Recall** [57]. To mathematically explain the measures, the evaluation set is defined as D, consisting of positive/negative feedback set D^+/D^-. For the i_{th} triple, the rank i represents

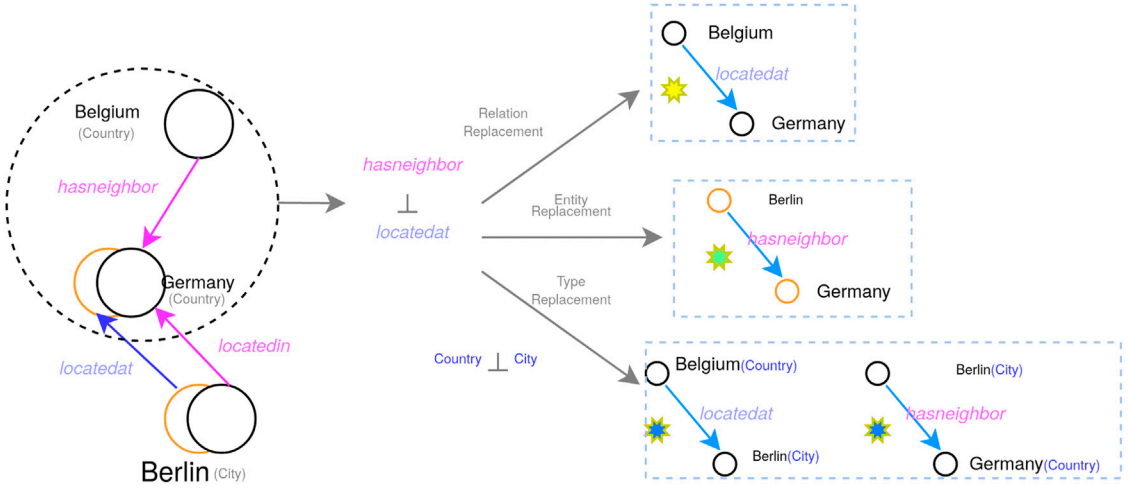

FIGURE 1
Three conditions of cross-KB negative sampling.

its rank in the evaluation set D. Triples with higher scores are filtered out as positive feedback. The rank of incorrect triples has lower values with better performance.

3.3.1 Triple semantic similarity

Word-to-word similarity is leveraged to calculate the consensus confidence of two entities in triples. By the confidence, some co-similar entities have near confidences and they are leveraged in matching methods.

3.3.2 Correction similarity

For calculating the correction similarity for repairs, a harmonic average similarity is proposed to validate the revised triples. The d_L denotes the distance in similarity of words for entities. Also, some special features are considered in similarity measures, e.g., the predicate *wikiPageWikiLink* discovers the same parts of two triples in the original sources, regarded as *semantic_measure*(e_0, e_i). The outer semantic measure calculates the quantity of matching parts in (P_{e_i}, *wikiPageWikiLink*) to acquire the common source, as explained in Func. 1. The part *semantic_measure*(e_0, e_i) considers the best similarity of two entities with soft cardinality [58]. Here, some similarity algorithms are leveraged to validate matching methods, considering their inner features, such as *theLevenshtein_distance*, *Cosine_similarity*, *Sorensen_Dice*, and *Jaro_Winkler*. Last, the harmonic correction similarity is shown in Func. 2.

$$semantic\_measure\,(e_0, e_i) = \frac{|P_{e_0}|_{soft} \cap |P_{e_i}|_{soft}}{|P_{e_0}|_{soft}}, \tag{1}$$

$$s\,(e_0, e_i) = 1 - \frac{d_L\,(e_0, e_i)}{max\,(|e_0|, |e_i|)} + semantic\_measure\,(e_0, e_i). \tag{2}$$

3.3.3 Soft harmonic similarity

A new soft harmonic means function is generated with character-level measure and semantic-relatedness in Func. 1, in order to balance the features of semantics and characters. The consensus is acquired by searching repair similarity of the optimal correction. Let single word T be a set of n tokens: $T = \{T^1, T^2 \ldots T^n\}$. $d(T^i, T^j)$ is a character-level similarity measure scaled in the interval [0,1]. The soft cardinality of the single word T is calculated as in Function 3.

$$|T|_{soft} = \sum_{i=1}^{n} \left[\frac{1}{\sum_{i=1}^{n} d\,(T^i, T^j)}\right] \tag{3}$$

$$f\_sim = \frac{2 \times character - level\,(e_0, e_i)_{soft} \times semantic - related\,(e_0, e_i)}{character - level\,(e_0, e_i)_{soft} + semantic - related\,(e_0, e_i)}. \tag{4}$$

Cross-similarity measures are leveraged to validate repairs of erroneous triples in KBs. After our model operations, some mistaken assertions are matched with multiple values in the process of repairs. Here, a new cross-similarity measure is proposed to analyze final revised assertions of triples in KBs, aiming to discover common features between original entities and repairs after correction. In Eq. (6), the *Jaro–Winkler* distance [59] is suitable for calculating the similarity between short strings such as names, where d_j is the *Jaro–Winkler* string similarity between e_0 and e_i, m is the number of strings matched, and t is the number of transpositions. Then *sim_external*(,) analyzes the external similarity probability, matching co-occurrence Wikipages in the (*wikiPageWikiLink*) property. $s(e_0, e_i)$ is a pair of compared objects. A new cross-function, f_{cross}, in Eq. (7) is the harmonic mean of distance and external similarity, which is designed to cover all correlations of assertions and candidate repairs.

$$sim\_external\,(e_0, e_i) = \left(|P_{e_0}| \cap |P_{e_i}|\right) / |P_{e_0}|, \tag{5}$$

$$d_j = \frac{1}{3}\left(\frac{m}{|e_0|} + \frac{m}{|e_i|} + \frac{m-t}{m}\right), \tag{6}$$

$$f_{cross} = \frac{2 \times d_j \times sim\_external\,(e_0, e_i)}{d_j + sim\_external\,(e_0, e_i)}. \tag{7}$$

3.3.4 Relation semantic similarity

The framework uses a method to calculate the semantic similarity between two relations based on word-to-word

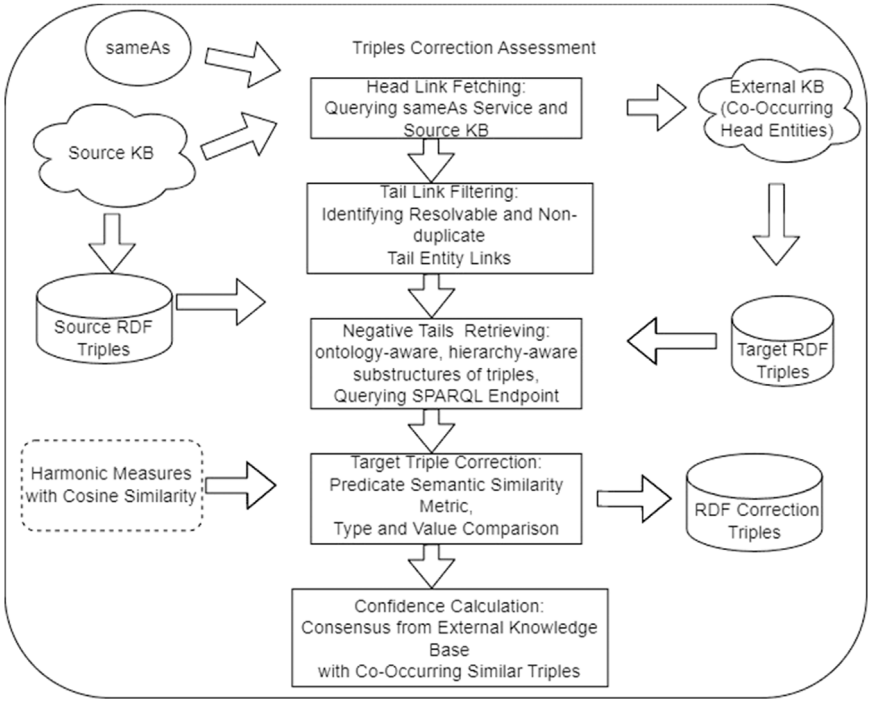

FIGURE 2
TCA framework.

similarity and the abstract-based information content (IC) of words, which is a measure of concept specificity. More specific type concepts (e.g., scientist) have higher values of IC over some type concepts (e.g., person). Generally, types of entities have underlying hierarchy concepts and structures, such as the structure among types with sub-concepts {*actor*, *award_winner*, *person*} in types of Freebase. Given the weights of hierarchy-based concepts [60], entity e and its type set are denoted as T_e. A hierarchy structure among concepts is presented as $C = /t_1/t_2/\ldots/t_i/\ldots/t_n$, where $t_i \in T_e$, n is the counts of hierarchy levels, t_n is the most specific semantic concept, and t_1 is the most general semantic concept. Usually, the range concept of a relation picks t_1 as the value.

4 The proposed framework

The TCA framework comprises five units (Figure 2). The first two elements recognize equivalent head entity links for a group of source triples, while the middle two parts select negative candidates with erroneous ranges from the source triples and perform the correction. The last item calculates a confidence score for each repair, representing the level of accuracy of the corrected entities.

The Head Link Fetching (HLFetching) is used to attain similar links of the candidate instance of a source entity. Since there may be duplicate and non-resolvable tails for different head entities, the second part, Tail Link Filtering (TLFiltering), makes a genuine attempt to find these tail links of tuples co-occurring in two KBs. Then, the Negative Tails Retrieving (NTR) accumulates target values including the identified candidate property links from external KBs. The third component, target triple correction (TTC), integrates a set of functions to identify repaired triples semantically similar to the source triple. The last component, confidence calculation (CC), calculates the confidence score for corrected triples from external KBs.

4.1 Problem statements

In knowledge bases, there is some noisy and useless information. Before the utilization of the knowledge base, some invalid data are removed and some knowledge is corrected for reuse in the application of KBs. So, knowledge base completion (KBC) is a hot research topic in the field of web science. Most research studies of KBC focus on predicating new information. Here, removing some invalid data and correcting some erroneous facts are our tasks. Aiming at the abnormal information in the knowledge base, this topic filters out invalid data and corrects error information for cleaning and completing KBs. In our approach, the first step is to find more error triples in KBs. Then, some valid erroneous triples are corrected to expand KBs.

Even when the selected entities are correct in KBs, incorrect relations between entities can still cause these triples to go wrong. Here, some other problem statements are explained.

4.1.1 Triple with conflict range type.

For instance, one selected triple < *dbr:Hiro_Arikawa*, *dbo:nationality*, *dbr:Japanese_people* > has the correct predicate

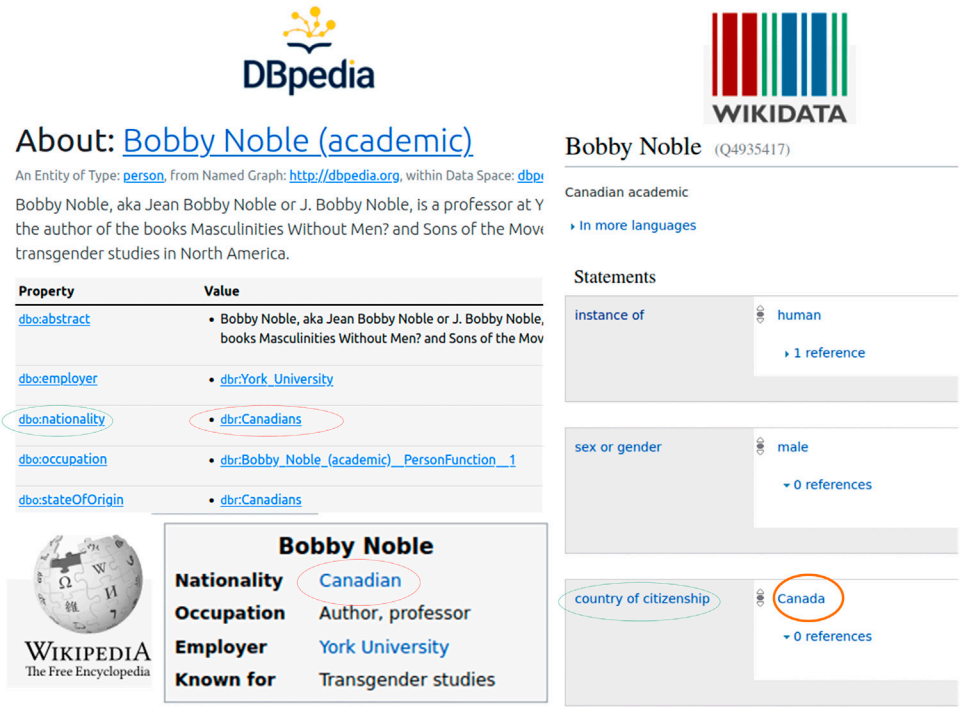

FIGURE 3
Structured information about Bobby Noble (academic).

range property (dbo:Country), but the *dbr:Japanese_people* has a conflicting rdf:type (dbo:EthnicGroup). Here, dbo:EthnicGroup and dbo:Country has strict conflict (*dbo*: *EthnicGroup* ⊥ *dbo*: *Country*). The incorrect triple is revised to < *dbr:Hiro_Arikawa, dbo: nationality, dbr:Japan* >. In the entity errors, the "nationality" specifies that a particular person comes from a particular country. The errors violate inconsistencies of type. The correct triple based on the type should be (dbr:Hiro_Arikawa, dbo: nationality, dbr:Japan). After analysis of predicate errors, the new correct triple based on the type should be (dbr:Hiro_Arikawa, *ethnic group*, dbr:Japan).

4.1.2 Error information in original source

The following two triples (illustrated in Figure 3) are about professor Bobby Noble: (Bobby Noble (academic), nationality, Canadians) in DBpedia as of September, 2022, and (Bobby Noble, nationality, Canadian) in Wikipedia. The triples from the two associated knowledge bases have the same errors since their original source contains incorrect information. Referring to the Wikidata database, the corrected triple (Bobby Noble, country of citizenship, Canada) equals (Bobby Noble (academic), nationality, Canada), since the predicate name *nationality* has the equivalent property of "*country of citizenship*."

4.1.3 Type errors

Given a fixed relation "birthplace" in the DBpedia as the sample, the noise type information is detected by the TBox property. Here, the hierarchical property *rdfs*: *subClassOf* is considered in the experiment to find the erroneous types. By the manual evaluation, the precision of corrected type is 95% in the relation of *birthplace*. Similarly, the quantity of the incorrect type (*dbo*: *Organisation*, *dbo*: *SportsClub*, *dbo*: *Agent*, *etc.*) is small. The corrected type contains some more subcategories, i.e., *dbo*: *City* < *dbo*: *Settlement* < *dbo*: *PopulatedPlace* < *dbo*: *Place*. So, searching the errors of types refers to the range of type and their inner property. In the closed-world assumption (CWA), negative triples with erroneous type are found by the type property, i.e., the range of the predicate. Then, in the open world assumption, the tail of the triple is replaced with another type of property.

For example, the positive triple: < Albert_Einstein, birthPlace, Ulm > and type pair < Person, birthPlace, Place >. Here, we remove the premise of *dbo*: all examples exist in the DBpedia. CWA: < Javed_Omar, birthPlace, Bangladesh_national_cricket_team > exists in the KB. OWA: a. < Albert_Einstein, birthPlace, University_of_Zurich >, < Person, birthPlace, Organization >. The negative type for the range of birthplaces is replaced. b. < Balquhain_Castle, birthPlace, Ulm >, < Building, birthPlace, Place >. Both of these triples are not in the KB, but in general knowledge: Albert_Einstein graduated from the University_of_Zurich. The triple *a* is regarded as unknown knowledge in the DBpedia or similar triples are not extracted from Wikipedia. But the triple *b* is actually false. Finally, the study exclusively uses the tail type replacement in the process of negative triples detection.

TABLE 1 Some examples of conflict feedbacks.

Fact	Feedback
< dbr: Wang_Zeng,dbo: birthPlace, dbr: Song_dynasty >	×
< dbr: Wang_Zeng,dbo: birthPlace, dbr: Qingzhou >	✓
< dbr: Novotel,dbo: locationCountry, dbr: Évry,_Essonne >	×
< dbr: Novotel,dbo: locationCountry, dbr: France >	✓
< dbr: Averroes, dbo: birthPlace, dbr: Almohad_Caliphate >	×
< dbr: Averroes, dbo: birthPlace, dbr: Córdoba,_Andalusia >	✓
< dbr: Pope_Telesphorus, dbo: birthPlace, dbr: Calabria >	×
< dbr: Pope_Telesphorus, dbo: birthPlace, dbr: Greece >	✓

4.1.4 Conflict feedback

Conflict feedback is assumed to consist of binary *true*/*false* assessments of facts that have the same subjects contained in the KB. Two different triples have the same subject and predicate but different objects. Not all positive examples can find corresponding counterexamples; conflict feedback cannot be obtained with a small number of examples. Two different paths are proposed to find the conflict feedback. First, range violation errors of triples are considered to search abnormal facts. The default settings are that subjects are always correct and objects have range violations. For example, the triple *< dbr: Wang_Zeng, dbo: birthPlace, dbr: Song_dynasty >* in DBpedia is incorrect since the predicate *dbo: birthPlace* requires a tail with the *dbo:Place* property (the best type following the characteristic distribution), which *dbr:Song_dynasty* is devoid of since Song_dynasty was an era of Chinese history, not a place. The inconsistency damages the effectiveness of any applications in KBs. To correct the instance, the *dbr: Song_dynasty* should be removed and *dbr:Qingzhou*, where *Wang_Zeng* was born, is saved in KBs. In Table 1, some examples are acquired from conflict feedback in DBpedia 2016 version. Such conflict feedback strictly disturbs information for further predictions, causes data distortion, and increases noise. The conflict errors are removed after searching all abnormal facts, and erroneous triples of one-to-many attributes are corrected in our proposed method for knowledge base correction.

4.2 Generated erroneous entities

Negative statements are regarded as incorrect triples. One major problem statement is that an object of triple has a type without a matching range of predicate. This error is also called a range violation of relation [61]. For the erroneous triples, cross-type negative sampling is used to generate erroneous entities. Also, the convenient way of error generation is to refer to TBox property, such as a class hierarchy (*via* rdfs:subClassOf) and owl:equivalentClass In the incorrect examples, the subject is not unique. For some conflict feedback, the same subject and the same property have different objects. Conflict feedback is considered to clean KBs, since some conflict feedback contains negative statements obfuscating facts in the real world.

4.2.1 Cross-type negative sampling

The model presents how to produce cross-KB negative samples over two KBs based on cross-KB negative predicates. The cross-KB negative samples can be caused by three strategies: predicate replacement, entity substitution, and type replacement.

4.2.1.1 Cross-KB negative type of predicate

There are two predicates: $r_1 \in R_1$ and $r_2 \in R_2$. r_i, $i = 1, 2$ has an empty overlapping type pair, i.e., $O_\tau(r_1, r_2) = \varnothing$; then the predicates r_1,r_2 are shown as $\tau_{r_1} \perp \tau_{r_2}$, called as generalized cross-KB negative type of predicate. The cross-KG negative relation [57] is defined by the strict cross-KB negative relation. For a given relation $r_\star^1 \in K_1$ and the type $\tau r_\star^1 \in K_1$, the cross-KB negative type of predicate set $N(\tau_{r_\star^1})$ of $r_\star^1$ is expressed as $N(\tau_{r_\star^1}) = \{\tau_{r_2} | \tau_{r_2} \perp \tau_{r_\star^1}, \tau_{r_2} \in K_2\}$, and the cross-KB negative set $N(\tau_{r_\star^2})$ of the predicate $\tau_{r_2} \in K_2$ is described as $N(\tau_{r_\star^2}) = \{\tau_{r_1} | \tau_{r_1} \perp \tau_{r_\star^2}, \tau_{r_1} \in K_1\}$. All the types of entities in the set of T_i, $i = 1, 2$.

Example 2. Let us assume that K_1 = {Germany, Berlin, Albert_Einstein, Belgium} and R_1 = {locatedat, livesin}. Three observed triples are (Berlin, locatedat, Germany), (Berlin, locatedin, Germany), and (Albert_Einstein, livesin, Berlin). The predicate "livesin" in Figure 1 is taken as an instance. The pair of entities on this predicate is (Albert_Einstein, Berlin). This pair of entities does not fulfill any predicate in the additional links. Thus, all predicates in the external links are its cross-KB negative type of predicates, i.e., N (livesin) = {locatedin, hasneighbor}. For the property "hasneighbor" in another knowledge base, its cross-KB negative type of predicate is N (livesin, locatedat).

4.2.1.2 Predicate replacement

Let us assume Q_2 represents the set of triples in the other KB K_2. For a triple $(e_s^2, r_2, e_o^2) \in Q_2$, if r_2 is replaced by any predicate $r_1 \in N(r_2)$, new triple (e_s^2, r_1, e_o^2) is regarded as a cross-KB negative sample. This new negative candidate is composed of entities $e_s^2, e_o^2 \in K_2$ and $r_1 \in R_1$. S_r' is denoted as the set of cross-KB negative samples acquired by predicate replacement. The intuition of predicate replacement is that if a triple (e_s^2, r_2, e_o^2) is correct, r_1 and r_2 do not have any overlapping entity pair, i.e., no triples can fulfill predicates r_1 and r_2 simultaneously and the new incorrect triple is (e_s^2, r_1, e_o^2).

Example 3. As shown in Figure 1, since *hasneighbor* $\perp$ *locatedat*, "hasneighbor" is alternated by "locatedat" between the entities "Belgium" and "Germany" to obtain a negative sample (Belgium, locatedat, Germany).

4.2.1.3 Entity substitution

Given a triple $(e_s^2, r_2, e_o^2) \in Q_2$ and $r_1 \in N(r_2)$, (e_s^2, e_o^2) is replaced with any entity pair (e_s^1, e_o^1) of triples satisfying r_1, the new (e_s^1, r_2, e_o^1) is seen as a cross-KB negative sample.

Example 4. Since (Berlin, Germany) contains the predicate "locatedat" shown in Figure 1, and *hasneighbor* $\perp$ *locatedat*, substituting the negative predicate "locatedat," the entity pairs have alternates on the predicate "hasneighbor." So, a new negative candidate is acquired, i.e., (Berlin, hasneighbor, Germany).

The cross-KB negative sampling efficiently acquires validation knowledge from additional KB for the source KB. Although tons of negative samples are produced without semantic similarity, such negative samples are still very instructive for embedding learning.

Since the method needs to learn from easy examples (e.g., negative relations "hasneighbor" and "hasPresident") to difficult instances (e.g., "hasneighbor" and "locatedat"), negative sample sets containing many simple conditions are beneficial for simple model learning. Difficult negative triples are more informative for complex models.

4.2.1.4 Type replacement

There are $(e_s^2, r_2, e_o^2) \in Q_2$ and its type $(T_{e_s^2}, r_2, T_{e_o^2}) \in T_2$. The positive triple and type pair is the $(e_s^2, r_2, e_o^2) \in Q_2$ and $(T_{r_{domain}}, r_2, T_{r_{range}}) \in T_2$. If the new samples satisfy the condition that $T_{e_i} is \in T_2, \neq T_{r_{domain}}$, the set of triples are new negative samples, i.e., $(T_{e_i}, r_2, T_{r_{range}})$. In the same assumption, the type of target entity is replaced by other types. The new negative samples $((T_{r_{domain}}, r_2, T_{e_i}))$ satisfies the condition that $T_{e_i} is \in T_2, \neq T_{r_{range}}$.

$$
\begin{aligned}
&positivetriples:\\
&(e_s, r, e_o) \in K, \quad type \in \left(T_{r_{domain}}, r, T_{r_{range}}\right).\\
&negativetriples:\\
&a.(e_s, r, e_o^*) \notin K,\\
&type \in (T_{r_{domain}}, r, T^*), \quad T^* \in T, T^* \neq T_{r_{range}}.\\
&b.(e_s', r, e_o) \notin K,\\
&type \in \left(T^*, r, T_{r_{range}}\right), \quad T^* \in T,\ T^* \neq T_{r_{domain}}.
\end{aligned}
\tag{8}
$$

$r_1 \in N(r_2)$ and $t_1 \in N(r_2)$, (e_s^2, e_o^2) is replaced with any entity group (e_s^1, e_o^1) of triples which includes r_1; the new (e_s^1, r_2, e_o^1) is regarded as a cross-KB negative sample.

4.2.2 Search strategy to generate negative candidates

In the CHAI model [13], they regard the candidate triples as true when the original triples are correct. Extending this idea; the negative candidates are also false. Considering the criteria from the CHAI model and the RVE model [29], a new search strategy is defined to explore more negative candidates. In short, $< s, p, o >$ is a triple in K and one erroneous triple is taken as negative feedback.

4.2.2.1 Existing subject and object

The criterion collects all candidates whose subject and object appear as such for some triples in K; p' and p have the same *ObjectPropertyRange*:

$$exist_{KB1}(s, p, o) \Leftrightarrow \exists p' \in \xi | (s, p', o) \in K. \tag{9}$$

4.2.2.2 Existing subject and predicate:

The criterion collects all candidates whose subject and predicate occur as such for some triples in K. There exists no candidate with the correct property type:

$$exist_{KB2}(s, p, o) \Leftrightarrow \exists o' \in \xi | (s, p, o') \in K. \tag{10}$$

4.2.2.3 Existing predicate and object

The criterion collects all candidates whose object entity replaces the subject one or more times in a triple that has another predicate p' or the object entity appears at least once as the object in a triple that has another predicate p':

$$exist_{KB3}(s, p, o) \Leftrightarrow \exists s' \in \xi | (s', p, o) \in K. \tag{11}$$

For instance, one negative triple *(Bobby Noble (academic), nationality, Canadians)* is chosen as the example. In criterion *a*, one candidate *(Bobby Noble (academic), dbo:stateOfOrigin, Canadians)* can be generated. In criterion *b*, one erroneous triple is *(Bonipert, nationality,French_people)* and the candidate is *(Bonipert, nationality,Italians)*. In criterion *c*, there are erroneous objects *Canadians, French_people, Italians, etc.* The number of candidate samples about (?a, nationality, Canadians) is over 4,900. The number of candidates about French_people is over 1,300 and the quantity about Italians is near 1,000. For positive triples, the results of candidates have a lower number of incorrect or noisy candidates, which also exist in the original KB. So, sparsity negative examples can be crawled by some features, and then our previous work produced a GILP model [15] to acquire more negative examples in iterations.

Combining the search strategy of negative candidates with the method of cross-type negative sampling, erroneous entities, and their triples can be generated for cleaning. Also, some interesting negative statements are selected to be corrected as new facts for knowledge base completion.

4.3 Fetching and filtering erroneous tails links

The HLFetching part acquires the tail of a source triple as input by the http://sameas.org*sameAs* service and equivalent links of the candidate instances are fetched in external KB. The *sameAs* property supplies service to quickly get equivalent links with arbitrary URIs, and 200 million URIs are served, currently. The SameAs4J API is used to fetch equivalent tails links from the *sameAs* service [62].

In a KB, a target predicate P_r, $< s, o >$ is used to detect a negative example if $< s, P_r', o > \in KB$, with $P_r' \neq P_r$, for every $< s, o >$ is semantically connected by at least one predicate. To refine the quality of training triples and delete cases of mixed types, all the subjects must have the same type, and the same is true for the object values. For example, the pseudo-SPARQL query is leveraged to present how to get negative examples in the predicate of child in the DBpedia database. Such as the pseudo-SPARQL query: *select distinct ?head ?tail where { ?head rdf:type dbr: Person. ?tail rdf:type dbr: Person. ?subject ?relation ?tail. {{ ?head dbr: child ?realTail. } UNION { ?realHead dbr: child ?tail.}} FILTER NOT EXISTS{?head dbr: child ?tail. }}*

4.4 Target triple correction

For target triple correction, the model takes co-occurring similar entities into consideration. One fixed predicate name is chosen as the sample to illustrate the process of correction. In the CWA, some simple queries can be serviced to find erroneous entities without correct *ObjectPropertyRange*, i.e., *< subject, predicate, object >* and *< object, a, wrong_ObjectPropertyRange >*. For example, the correction type of the "nationality" range is *Country*. The DBpedia contains over 1,800 different values of objects with the correct type. Also, there are some false positive items, e.g., *dbr:Canadians*, *dbr:Germans*, *dbr:*

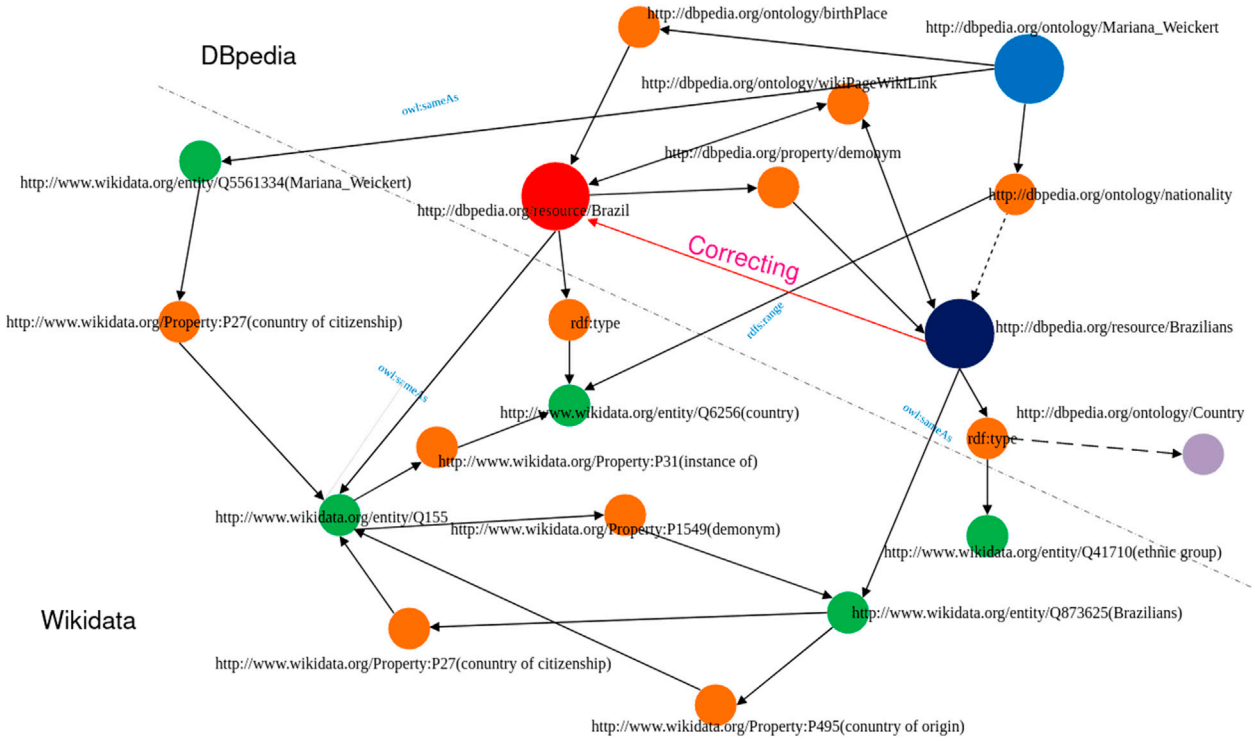

FIGURE 4
Evidence graph as displayed in the TCA (the dark dot denotes erroneous entity and the red one is corrected. Orange dots denote predicate property, and other colors show the entities between DBpedia and Wikidata).

French_language, and *dbr:Pakistanis* Comparatively, the KB holds over 1,000 different incorrect entities of triples. Next, the co-occurring similar entities in the Wikidata are leveraged to validate the repairs in the DBpedia. The algorithms assess the correctness of entity values by cross-checking them with properties of type from a new KB, shown in Algorithm 1. The system automatically checks the conformity of the entity inside the old KB (DBpedia) to all the same entities inside the Wikidata with the property of *sameAs*. In the CWA, the YAGO has the precise information of type by the property of *wordnet*. Referring to Wikidata, we can also leverage the features to verify the repairs of YAGO in the rule correction algorithms.

Algorithm 1 describes the triple matching algorithm to correct negative candidates. First, in the former methods, it is proposed to generate erroneous triples. Then, conflict feedback is removed from sets of erroneous entities. The predicate name is extracted from one erroneous triple. True *ObjectPropertyRange* τ is leveraged to find candidate property p'in associated KB K'. Also, p' can be found by overlapping type pairs of entities. At the same time, corresponding candidate instance s' is acquired by *owl*: *sameAs* relation from original subject s of $<s, p, o>$. Next, new objects are found in K' from $<s', p', ?>$ and stored in $set\{obj\}$. Finally, some similarity measures are used to filter consensus and make the final correction. The TCA iterations are terminated either when no triples are in E or when $Corr_n$ remains unchanged among two iterations.

Our problem is simplified to finding the corresponding property in Wikidata based on a co-occurring similar triple in DBpedia. Especially, the equivalent property of the predicate name of triples is selected to find repairs for the wrong entity. One entity *Mariana_Weickert* extracted from DBpedia is regarded as an example of a correcting task. An evidence graph is shown in the TCA, in Figure 4. For erroneous triple $<Mariana\_Weickert, dbo: nationlity, Brazilians>$, it violates the range constraint of a predicate name. The dashed lines represent wrong relations.

```
Input: pand <s, p, o>;
Output: Corr_n;
Corr_i = null, i = 0;
Erroneous_entities_sets: (s, p, o) ∈ E;
while K ≠ ∅ or Corr_i changed do
  p → ObjectPropertyRange(τ(true)orτ'(false));
  (p, τ, owl: equivalentProperty) → candidate_
  property: p';
  (s, owl: sameAs) -> candidate_instance: s';
  <s', p', ?> → property_value: o';
  (o', owl: sameAs, ?) → repairs: set{obj};
  correction(obj) = filterConsensus(set{obj});
  <s, p, o> → <s', p', o'> → <s, p, obj>;
  Corr_i := < s, p, obj > ∪ Corr_i; i = ++;
end
return Corr_n;
```

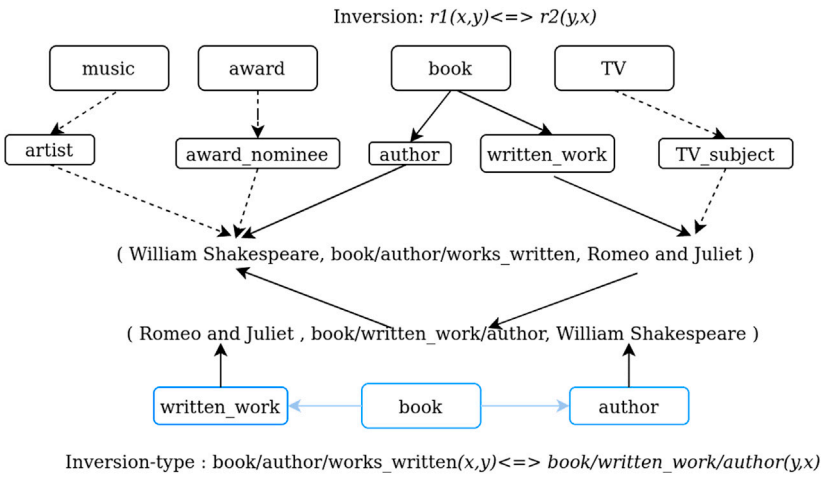

FIGURE 5
Example of logic relations with entity hierarchical types in Freebase.

Algorithm 1Co-occurring Triple Matching Algorithm.

Two major paths are expressed in the process of repairing the wrong range constraint. First, based on subject *Mariana_Weickert*, a similar entity in Wikidata is filtered by *owl*: *sameAs* and the equivalent property of nationality is replaced by Wikidata:P27 (country of citizenship). So, the repair entity is *wikidata*: Q155, and the corresponding entity is Brazil in DBpedia. *dbr*: *Brazilians* has wrong type *dbo*: *Country*. Second, referring to the wrong object and the correct range type, *Brazilians* and *Brazil* are related by properties *wikidata*: *P*495 (country of origin) and *wikidata*: *P*27. Finally, $< Mariana\_Weickert, dbo: nationlity, Brazilians >$ can be corrected to < Mariana_Weickert, dbo:nationlity, Brazil >. Before application in the answer–question system, some results are validated by our algorithm. Some constructed KBs, such as DBpedia or YAGO, have high precision. For these KBs, our approach can be used to validate the final results in the question–answer system.

4.4.1 Hierarchy information for knowledge correction

The taxonomy and hierarchy of knowledge can be applied to many downstream tasks. Hierarchical information originated from concept ontologies, including semantic similarity [63, 64], facilitating classification models [65], knowledge representation learning models [66], and question–answer systems [67]. Well-organized algorithms or attentions of hierarchies are widely applied in the works of relation extraction, such as concept hierarchy, relation hierarchy with semantic connections, a hierarchical attention scheme, and a coarse-to-fine-grained attention [68, 69].

4.4.2 Hierarchical type

In Freebase and DBpedia, selecting one hierarchical type c with k layers as example, $c(i)$ is the i_{th} sub-type of c. The most precise sub-type is considered the first layer, and the most general sub-type is regarded as the last layer, while each sub-type $c(i)$ has only one parent sub-type $c(i+1)$. Taking a bottom-up path in the hierarchy, the form of hierarchical type is represented as $c = c(1), c(2), \ldots, c(k)$. In YAGO, subclass Of is used to connect the concepts (sub-types). In logic rules, like the inversion, $r_1(x, y) < => r_2(y, x)$ and the variables x, y can be the entities in general. Here, we expand the logic relations with entity hierarchical types and acquire the fixed domain entities.

As shown in Figure 5, the inversion-type logic are r_1(author, written_work) < = > r_2(written_work, author). So, the relations r_1 and r_2 are book/author/works_written and book/written_work/author. Especially, the entity of freebase contains the type information in the label of the entity. One negative triple is inversion-type, so negative candidates can be acquired by inversion relations. For instance, nationality has *InversePath* (*is nationality of*. In DBpedia, an entity page displays statements in which an entity may be not only a subject but also an object. In the latter case, the respective property appears as "is . . .of." If one negative triple $< s, p, o >$ has inverse path, all candidates extracted from the condition satisfies $< o, is\_p\_of, s' >$ are incorrect.

For example, the object of irthplace in entity *Nick_Soolsma* follows the type path: *Andijk*(*dbo*: *Village*) < *Medemblik*(*dbo*: *Town*) < *North_Holland*(*dbo*: *Region*) < *Netherlands*(*dbo*: *Country*). One logic path: country containing one birthplace of a person is the person's nationality. By hierarchical property, dbr:Nick_Soolsma acquires one new nationality, dbr: Netherlands. Repair results can be obtained by predicting erroneous information by hierarchical type. The correction method was proposed in our previous work [18]. For the explanation of hierarchical correction, related paths, and relationships can be used to acquire corrections for negative triples.

5 Experiments

Our approach is tested by using four datasets from four predicate names. Here, mean reciprocal ranking (MRR), HITS@1, and HITS@10 [6] are selected to measure the confidence

calculation of corrected triples in the knowledge base. All training datasets are leveraged in the experiments from http://ri-www.nii.ac.jp/FixRVE/Dataset8. Some baseline algorithms were realized in Python, using Ref. 6. Our framework is constructed in the Ubuntu 20.04.5 system and Java 1.8.0, and experimental analysis is run on a notebook with a 12th Gen Intel Core i9-12900KF × 24 and 62.6 GB memory.

5.1 Negative feedback generation

P is given a constraint predicate. A constraint has several lines when it leverages a specified relation. *#constr* is the total quantity of constraints of the *errors type* in Dbpedia. *#triple* is the number for calculating all these constraints of triples with the predicate *P*. *#violations* is the quantity of violations for this constraint in Dbpedia in October 2016. *#current_cor* is the quantity of current corrections collected from Dbpedia in 2020.

In type classification of nationality, objects with the *country* property are up to 67%, and entities with *ethnic group* is 31%. Other types are less than 2%, such as language, island, and human settlement. After analysis of negative constraints of *nationality*, there are duplicate triples between problem statements. In Dbpedia, the type of the entity is a parallel relationship in the SPARQL query results, and the hierarchical relationship between the attributes cannot be obtained from the query results. Therefore, there are overlapping parts among all these errors because the object value of the predicate "*nationality*" is not unique. Nearly 20% of the triples determined as can be corrected to complete KBs since the objects can have multiple values for *nationality*, explained in Table 2. For the relation *birthplace*, the conflict feedback is removed because the predicate objects have a single value. Also, there are over 70% conflict types in error types for nationality. Here, some examples extracted from nationality are applied to validate our correction model.

For a single incorrect triple, a search strategy is proposed to generate negative candidates. Following strategy *a* for *nationality*, some new predicate names *isCitizenOf, stateOfOrigin* are acquired from KBs. In strategy *b*, the object types of triples are all exception properties. Negative candidates are obtained by determining the type of a multi-valued object. In search *c*, the set of all errors for such a predicate name can be found with a single incorrect entity object.

5.2 Discussion

Some examples of repairs with predicate *nationality* are shown in Table 3. Most subjects have word similarity of repair and tail. The results of some samples about nationality are shown in Figure 6. For predicate *nationality*, there are a large number of different subjects for one incorrect object. Therefore, for triples with the same erroneous object, such subjects from triples are aggregated into a set, which can ignore the quantity of subjects. Incorrect triples are revised from the perspective of the object. For each pair of error object and repair, the correction similarity is calculated by harmonic correction similarity with different distance methods. In TCA framework, the confidence calculation component holds maximum similarity to filter corrections. The precision of repairs is focused on the interval of [0.3, 0.6], since the great majority of incorrect objects have few connections. In our validation part, the precision of repairs is over 0.5, and these revised triples are regarded as final corrections.

TABLE 2 Negative constraints of *nationality* in DBpedia.

Items	Numbers	Rates (%)
#triple	150,332	
#current_cor	92,995	61.86
#constr	57,337	38.14
conflict_type	45,405	30.20
conflict_feedback	32,594	21.68

TABLE 3 Repairs examples.

Subject	Tail	Repair	Correction
Walter_Mignolo	Argentinians	Argentina	Argentina
Moira_Gatens	Australians	Australia	Australia
Bobby_Noble	Canadian	Canada	Canada
Oisín_Kelly	Dublin	Australia; Belarus	Ireland
Jerry_Weyer	Luxembourgish	Germany; Luxembourg	Luxembourg

In Figure 7, string similarity methods are leveraged to replace distance methods in harmony correction similarity. String similarity measures are extracted from two aspects, i.e., character-level measures and token-level measures. Nine repair examples are randomly used to validate the correction rates. Fourteen similarity measures are separated by their values. By the nature of repairs, TCA only focuses on the words, not the sentences. So, the results show the Qgram(2) and NGram(i), NormalizedLevenshtein has the better performance. Compared with word and string features, correction similarity is suitable to acquire repairs with word similarity.

Some similarity measures are used to compare these repairs in TCA, as shown in Figure 8. The mistaken entities have single values as the final correction. For multiple values as repairs, cross-similarity is proposed to discover final corrections. Distance similarity measures are leveraged to validate repairs, such as the longest common subsequence (LCS), Optimal String Alignment (OSA), and normalized Levenshtein distance (NLD). Compared to DBpedia, the similarity of repairs in Wikidata focuses on word similarity. For a single erroneous triple, Jaro–Winkler similarity is used to validate repairs, and the revised correction has an interval with high precision. In the experiment, 2,000 negative entities were randomly selected to verify the TCA model. The best performance of cross-similarity is shown in Figure 8 and Eq. (7). So, cross-similarity is leveraged to filter final repairs in the EILC model. The final pairs of errors and corrections exhibit unique characteristics that have a high degree of word similarity. Here, multiple repairs indicate that some examples have over 90% similarity probability, i.e., Jaro–Winkler similarity.

The traditional measures, e.g., Mean_Raw_Rank, Precision, and Recall, are used to evaluate the effect of our correction model and to make comparisons with other classic algorithms. The bold value of **M** stands for Mean_Raw_Rank, explained in evaluation measures. And the

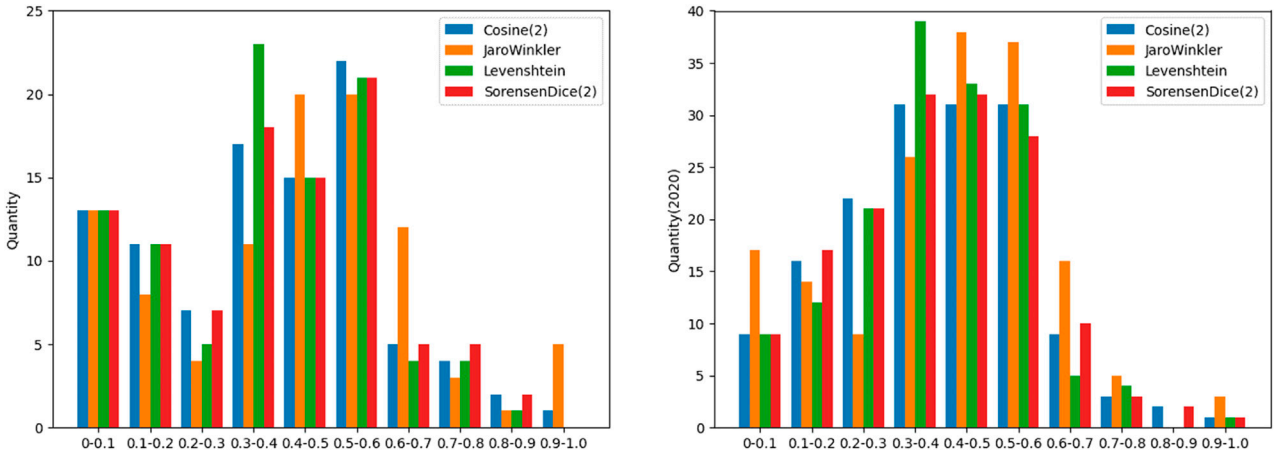

FIGURE 6
Correction similarity with distance methods.

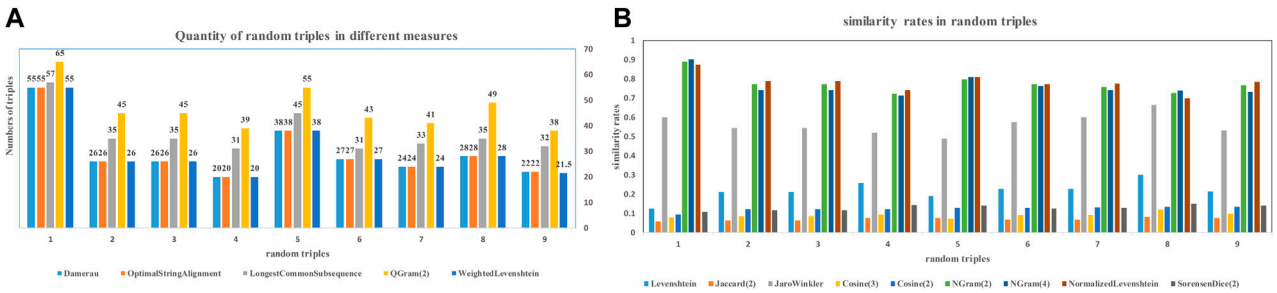

FIGURE 7
Comparison of similarity measures.

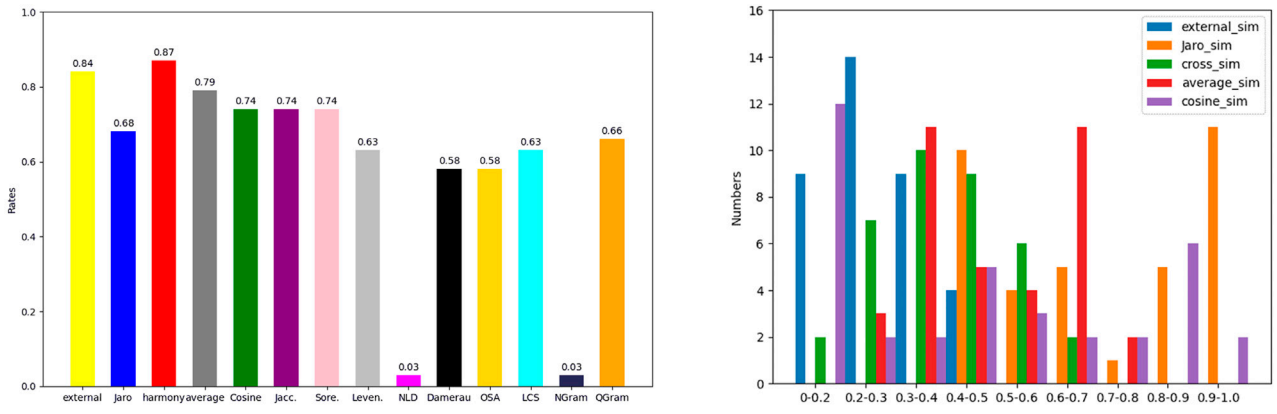

FIGURE 8
Correction rates and intervals based on different similarity measures.

TABLE 4 Comparison of baseline methods.

Method	locationCountry			formerTeam			Employer			birthPlace		
	M	**@1**	**@10**	**M**	**@1**	**@10**	**M**	**@1**	**@10**	**M**	**@1**	**@10**
DBpedia lookup	0.02	1	3	0.06	6	6	0.07	6	10	0.04	3	5
WikiDisambiguates	0.04	3	4	0.04	3	4	0.04	3	7	0.11	8	18
TransE	0.19	11	37	0.02	1	1	0.00	0	3	0.24	20	35
AMIE+	0.42	40	43	0.06	6	6	0.00	0	0	0.01	1	1
Graph method	0.89	88	91	0.37	25	61	0.62	53	74	0.44	24	75
Keyword method	0.58	48	77	0.60	49	78	0.36	33	45	0.48	36	72
TCA	0.95	87	98	0.38	32	46	0.76	64	87	0.55	54	61

The bold value of **M** stands for Mean_Raw_Rank, explained in evaluation measures. And the **@1** and **@10** present the value of **precison @K**.

TABLE 5 Some examples overlapping type pair of entities.

DBpedia (predicate)	Wikidata (property)
dbo: locationCountry	P17(country); P27 (country of citizenship); P495 (country of origin)
dbo:formerTeam	P5138(season of club or team); P463(member of); P664(organizer)
dbo:employer	P108(employer); P127(owned by); P112(founded by); P123 (publisher)
dbo: birthPlace	P27 (country of citizenship); P495 (country of origin)

@1 and **@10** present the value of **precison @K**. The comparison results are shown in Table 4 Our approach is compared to six baseline methods. Two are normally leveraged for entity search (DBpedia lookup and dbo: wikiPageDisambiguates) to find entities with the correct range type of predicate name and object. Two baseline methods were originally created for knowledge graph completion (TransE [70] and AMIE+ [49]) for finding the correct object from a given subject and a predicate name. Also, the graph method and keyword method [2] are leveraged to correct triples with range violations.

For positive examples in DBpedia and Wikidata, one example of overlapping type pair is O_τ(dbo: locationCountry, country of citizenship) = (person, country). The negative triple follows the equation: $O_\tau(r_1, r_2)$=(?a, country). Here, ?a does not equal country. Following an overlapping type pair of entities, corresponding predicates are acquired from positive examples in target KBs. Predicate comparisons from DBpedia and Wikidata are explained in Table 5. By the comparisons, some properties are used to search the repairs from co-occurring similar subjects. For these type pairs, some predicate names in external KBs are acquired for correcting negative candidates.

Three evaluation measures are used to calculate the correct object provided for each method. It is evident that our model outperforms common algorithms for all training sets. One condition is that the incorrect object of an erroneous triple has a unique corresponding subject (e.g., locationCountry). TCA and graph methods work closely, since the pair of object and subject has more connections and the paths of triples contain more details. In another condition, one incorrect object has multiple subjects and a graph method. There is a lot of redundant and ambiguous information provided by the graph algorithm with graph structure, which makes it impossible to find the correct object. In this condition (e.g., formerTeam), the keyword method is more effective because it takes advantage of external information from abstracts of triples, including subject and object. In order to be faster and more efficient in the algorithm, TCA explores knowledge correction methods from different perspectives.

TCA is more effective than other basic methods and the keyword method. For these basic methods, they can only correct some single error entity. To make up for such shortcomings and save time complexity, TCA is leveraged to correct range violations by using co-occurring similar entities. By making full utilization of other related knowledge bases for knowledge correction, it is beneficial to think about linked open data. The predefined paths are applied for hierarchy correction. The paths are derived from positive examples. In AMIE+, some paths can be provided by AMIE+. Not all predicates have a logical relationship, and hierarchical learning is very dependent on path information. The final result is close to AMIE+. After analysis of all methods, our proposed TCA model has better performance in base methods. If the source is not Wikipedia, or if the target is not DBpedia or YAGO, the original data sets need to do some changes. While the correction model is applied to other background knowledge bases, the training sets are changed to a triple formulation. All testing facts are transferred to $<subject, predicate, object>$. Also, the corresponding knowledge is matched by the associated knowledge bases with the same conditions. Our correction algorithm is, indeed, applicable to Wikipedia-linked knowledge bases.

6 Conclusion

This paper proposed a TCA framework to detect abnormal information and correct negative statements that exist in Wikipedia automatically by co-occurring similar facts in external KBs. Based on ontology-aware substructures of triples, fixing extracted errors is a significant research topic for KB curation. Additionally, our framework is executed post factum, with no changes in the process of KB construction. Two new strategies are applied to search for negative candidates for cleaning KBs. One triple matching algorithm in TCA is proposed to correct erroneous information. Our compared experimental results show that TCA is effective over some baseline methods and widely applied in large knowledge bases. Our framework is straightforwardly adapted to detect erroneous knowledge on other KBs, such as YAGO and Freebase.

In the future, conflicting feedback facts or predictions can be used to refine the KBs. Also, our framework will focus on the search space of triples with other similar contents, such as the abstracts, the labels, and the derived peculiarities. Moreover, more features of similar facts with logic rules are detected in the hub research of knowledge base completion. In our next work plan, a neural network is added to explore more paths for searching for mistakes in KBs. Next, the number of associated knowledge bases can be expanded and the problem of completing large knowledge bases can be solved by associating and matching more effective information toward the goal of completing large KBs.

Data availability statement

The original contributions presented in the study are included in the article/Supplementary Material; further inquiries can be directed to the corresponding author.

Author contributions

YW and ZZ contributed to the design and implementation of the research, to the analysis of the results, and to the writing of the manuscript.

Conflict of interest

The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Publisher's note

All claims expressed in this article are solely those of the authors and do not necessarily represent those of their affiliated organizations, or those of the publisher, the editors, and the reviewers. Any product that may be evaluated in this article, or claim that may be made by its manufacturer, is not guaranteed or endorsed by the publisher.

References

1. Vrandečić D, Krötzsch M. Wikidata: A free collaborative knowledgebase. *Commun ACM* (2014) 57:78–85. doi:10.1145/2629489

2. Auer S, Bizer C, Kobilarov G, Lehmann J, Cyganiak R, Ives Z. Dbpedia: A nucleus for a web of open data. *The semantic web*. Springer) (2007). p. 722–35.

3. Raimbault F, Ménier G, Marteau PF, et al. *On the detection of inconsistencies in rdf data sets and their correction at ontological level* (2011). p. 1–11. Available at: https://hal.archives-ouvertes.fr/hal-00635854 (Oct 26, 2011).

4. Zaveri A, Kontokostas D, Sherif MA, Bühmann L, Morsey M, Auer S, et al. User-driven quality evaluation of dbpedia. Proceedings of the 9th International Conference on Semantic Systems (2013). p. 97–104.

5. Paulheim H, Bizer C. Type inference on noisy rdf data. *International semantic web conference*. (Springer) (2013). p. 510–25.

6. Lertvittayakumjorn P, Kertkeidkachorn N, Ichise R. *Resolving range violations in dbpedia*. Joint international semantic technology conference. Springer) (2017). p. 121–37.

7. Dubey M, Banerjee D, Abdelkawi A, Lehmann J. Lc-quad 2.0: A large dataset for complex question answering over wikidata and dbpedia. Springer). Proceedings of the 18th International Semantic Web Conference (ISWC) (2019). p. 1–8.

8. Rajpurkar P, Jia R, Liang P. "Know what you don't know: Unanswerable questions for squad," in Proceedings of the 56th annual meeting of the association for computational linguistics (short papers), Melbourne, Australia, July 15 - 20, 2018. arXiv preprint arXiv:1806.03822 (2018). p. 784–9.

9. Ferrucci D, Brown E, Chu-Carroll J, Fan J, Gondek D, Kalyanpur AA, et al. Building watson: An overview of the deepqa project. *AI Mag* (2010) 31:59–79. doi:10.1609/aimag.v31i3.2303

10. Suchanek FM, Kasneci G, Weikum G. Yago: A core of semantic knowledge. Proceedings of the 16th international conference on World Wide Web (2007). p. 697–706.

11. Liu S, d'Aquin M, Motta E. *Towards linked data fact validation through measuring consensus*. Springer, Cham. Portoro, Slovenia: LDQ@ ESWC (2015). p. 21.

12. Liu S, d'Aquin M, Motta E. Measuring accuracy of triples in knowledge graphs. *International conference on language, data and knowledge*. Springer) (2017). p. 343–57.

13. Borrego A, Ayala D, Hernández I, Rivero CR, Ruiz D. Generating rules to filter candidate triples for their correctness checking by knowledge graph completion techniques. Proceedings of the 10th International Conference on Knowledge Capture (2019). p. 115–22.

14. Wang H, Ren H, Leskovec J. *Entity context and relational paths for knowledge graph completion*. arXiv preprint arXiv:2002.06757 (2020).

15. Wu Y, Chen J, Haxhidauti P, Venugopal VE, Theobald M. Guided inductive logic programming: Cleaning knowledge bases with iterative user feedback. *EPiC Ser Comput* (2020) 72:92–106.

16. Melo A, Paulheim H. An approach to correction of erroneous links in knowledge graphs. *CEUR Workshop Proc (Rwth)* (2017) 2065:54–7.

17. Chen J, Chen X, Horrocks I, B Myklebust E, Jimenez-Ruiz E. Correcting knowledge base assertions, Proceedings of The Web Conference (2020). p. 1537–47.

18. Wu Y, Zhang Z, Wang G. Correcting large knowledge bases using guided inductive logic learning rules. *Pacific rim international conference on artificial intelligence*. Springer (2021). p. 556–71.

19. Paulheim H. Identifying wrong links between datasets by multi-dimensional outlier detection. *WoDOOM* (2014) 27–38.

20. Melo A, Paulheim H. Detection of relation assertion errors in knowledge graphs. Proceedings of the Knowledge Capture Conference (2017). p. 1–8.

21. Zhang L, Wang W, Zhang Y. Privacy preserving association rule mining: Taxonomy, techniques, and metrics. *IEEE Access* (2019) 7:45032–47. doi:10.1109/access.2019.2908452

22. Fan W, Geerts F. *Foundations of data quality management*. Springer, Cham: Morgan and Claypool Publishers (2012). doi:10.1007/978-3-031-01892-3

23. Galárraga LA, Teflioudi C, Hose K, Suchanek F. "*Amie: Association rule mining under incomplete evidence in ontological knowledge bases*," in WWW '13: Proceedings of the 22nd international conference on World Wide Web. Rio de Janeiro, Brazil: WWW (2013). p. 413–22. doi:10.1145/2488388.2488425

24. Zeng Q, Patel JM, Page D. QuickFOIL: Scalable inductive logic programming. *PVLDB* (2014) 8:197–208. doi:10.14778/2735508.2735510

25. Rantsoudis C, Feuillade G, Herzig A. "Repairing aboxes through active integrity constraints. 30th international workshop on description logics (DL 2017)," In 30th international workshop on description logics (DL workshop 2017), 18 July 2017 - 21 July 2017. Montpellier, France: CEUR-WS: Workshop proceedings (2017). p. 1–13. https://oatao.univ-toulouse.fr/22739/.

26. Paulheim H, Bizer C. Improving the quality of linked data using statistical distributions. *Int J Semantic Web Inf Syst (Ijswis)* (2014) 10:63–86. doi:10.4018/ijswis.2014040104

27. Liang J, Xiao Y, Zhang Y, Hwang SW, Wang H. "Graph-based wrong isa relation detection in a large-scale lexical taxonomy," In Proceedings of the thirty-first AAAI conference on artificial intelligence. San Francisco California USA (2017). p. 1–6. doi:10.1609/aaai.v31i1.10676

28. Manago M, Kodratoff Y. Noise and knowledge acquisition. *IJCAI* (1987) 348–54.

29. Lertvittayakumjorn P, Kertkeidkachorn N, Ichise R. Correcting range violation errors in dbpedia. *International semantic web conference*. Kobe, Japan. (Springer): Posters, Demos and Industry Tracks (2017). p. 1–4. https://ceur-ws.org/Vol-1963/.

30. Abedini F, Keyvanpour MR, Menhaj MB. Correction tower: A general embedding method of the error recognition for the knowledge graph correction. *Int J Pattern Recognition Artif Intelligence* (2020) 34:2059034. doi:10.1142/s021800142059034x

31. Bordes A, Usunier N, Garcia-Duran A, Weston J, Yakhnenko O. Translating embeddings for modeling multi-relational data. *Adv Neural Inf Process Syst* (2013) 26.

32. Nickel M, Tresp V, Kriegel HP. A three-way model for collective learning on multi-relational data. *Icml* (2011) 1–8.

33. Wang Z, Zhang J, Feng J, Chen Z. Knowledge graph embedding by translating on hyperplanes. *Proc AAAI Conf Artif Intelligence* (2014) 28:1112–9. doi:10.1609/aaai.v28i1.8870

34. Xiao H, Huang M, Hao Y, Zhu X. "Transg: A generative mixture model for knowledge graph embedding," in Proceedings of the 54th annual meeting of the association for computational linguistics (Germany) (2015). p. 2316–25. doi:10.48550/arXiv.1509.05488

35. Yang B, Yih WT, He X, Gao J, Deng L. "Embedding entities and relations for learning and inference in knowledge bases. Proceedings of the international conference on learning representations. San Diego, CA, USA: ICLR (2014). p. 1–13. doi:10.48550/arXiv.1412.6575

36. Nickel M, Rosasco L, Poggio T. Holographic embeddings of knowledge graphs. *Proc AAAI Conf Artif Intelligence* (2016) 30:1955–61. doi:10.1609/aaai.v30i1.10314

37. Shi B, Weninger T. Proje: Embedding projection for knowledge graph completion. *Proc AAAI Conf Artif Intelligence* (2017) 31:1236–42. doi:10.1609/aaai.v31i1.10677

38. Bader J, Scott A, Pradel M, Chandra S. Getafix: Learning to fix bugs automatically. *Proc ACM Programming Languages* (2019) 3:1–27. doi:10.1145/3360585

39. Mahdavi M, Baran AZ. Effective error correction via a unified context representation and transfer learning. *Proc VLDB Endowment* (2020) 13:1948–61.

40. Pellissier Tanon T, Suchanek F. *Neural knowledge base repairs*. European Semantic Web Conference (Springer) (2021). p. 287–303.

41. Mahdavi M, Abedjan Z, Castro Fernandez R, Madden S, Ouzzani M, Stonebraker M, et al. Raha: A configuration-free error detection system. Proceedings of the 2019 International Conference on Management of Data (2019). p. 865–82.

42. Zhao Y, Hou J, Yu Z, Zhang Y, Li Q. Confidence-aware embedding for knowledge graph entity typing. *Complexity* (2021) 2021:1–8. doi:10.1155/2021/3473849

43. Chen J, Jiménez-Ruiz E, Horrocks I, Chen X, Myklebust EB. *An assertion and alignment correction framework for large scale knowledge bases*. Semantic Web Pre-press, IOS Press (2021). p. 1–25. doi:10.3233/SW-210448

44. Arnaout H, Tran TK, Stepanova D, Gad-Elrab MH, Razniewski S, Weikum G. Utilizing language model probes for knowledge graph repair. *Wiki Workshop* (2022) 2022:1–8.

45. Petroni F, Rocktäschel T, Lewis P, Bakhtin A, Wu Y, Miller AH, et al. "Language models as knowledge bases?," 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). Hong Kong, China: Association for Computational Linguistics (2019). p. 2463–73. doi:10.18653/v1/D19-1250

46. Lehmann J, Bühmann L. Ore-a tool for repairing and enriching knowledge bases. *International semantic web conference*. Springer) (2010). p. 177–93.

47. Knuth M, Hercher J, Sack H. "Collaboratively patching linked data," In Proceedings of 2nd international workshop on usage analysis and the web of data (USEWOD 2012) p. 1–6. doi:10.48550/arXiv.1204.2715

48. Ma Y, Qi G. *An analysis of data quality in dbpedia and zhishi. me*. China Semantic Web Symposium and Web Science Conference. Springer (2013). p. 106–17.

49. Lajus J, Galárraga L, Suchanek F. *Fast and exact rule mining with amie 3*. European Semantic Web Conference. Springer (2020). p. 36–52.

50. Chu X, Morcos J, Ilyas IF, Ouzzani M, Papotti P, Tang N, et al. Katara: Reliable data cleaning with knowledge bases and crowdsourcing. *Proc VLDB Endowment* (2015) 8:1952–5. doi:10.14778/2824032.2824109

51. Krishnan S, Franklin MJ, Goldberg K, Wang J, Wu E. Activeclean: An interactive data cleaning framework for modern machine learning. Proceedings of the 2016 International Conference on Management of Data (2016). p. 2117–20.

52. Rekatsinas T, Chu X, Ilyas IF, Holoclean RC. Holistic data repairs with probabilistic inference. *Proc. VLDB Endow.* (2017) 10(11):1190–201. arXiv preprint arXiv:1702.00820.

53. Krishnan S, Franklin MJ, Goldberg K, Wu E. *Boostclean: Automated error detection and repair for machine learning*. arXiv preprint arXiv:1711.01299 (2017).

54. De Melo G. *Not quite the same: Identity constraints for the web of linked data*. Twenty-Seventh AAAI Conference on Artificial Intelligence (2013). p. 1092–8.

55. Ngonga Ngomo AC, Sherif MA, Lyko K. Unsupervised link discovery through knowledge base repair. *European semantic web conference*. Springer) (2014). p. 380–94.

56. Domingue J, Fensel D, Hendler JA. *Handbook of semantic web technologies*. Springer Science and Business Media (2011).

57. Wang Y, Ma F, Gao J. Efficient knowledge graph validation via cross-graph representation learning. Proceedings of the 29th ACM International Conference on Information and Knowledge Management (2020), 1595–604.

58. Vargas SGJ. *A knowledge-based information extraction prototype for data-rich documents in the information technology domain*. [Dissertation/master's thesis]. Columbia: National University (2008).

59. Wang Y, Qin J, Wang W. Efficient approximate entity matching using jaro-winkler distance. *International conference on web information systems engineering*. Springer (2017). p. 231–9.

60. Cui Z, Kapanipathi P, Talamadupula K, Gao T, Ji Q. "Type-augmented relation prediction in knowledge graphs," in The thirty-fifth AAAI conference on artificial intelligence (AAAI-21). Vancouver, Canada (2020). doi:10.1609/aaai.v35i8.16879

61. Dimou A, Kontokostas D, Freudenberg M, Verborgh R, Lehmann J, Mannens E, et al. *International semantic web conference*. Springer (2015). p. 133–49.Assessing and refining mappings to rdf to improve dataset quality

62. Fiorentino A, Zangari J, Darling MM. DaRLing: A datalog rewriter for owl 2 RL ontological reasoning under SPARQL queries. *Theor Pract Logic Programming* (2020) 20:958–73. doi:10.1017/s1471068420000204

63. Leacock C, Chodorow M. Combining local context and wordnet similarity for word sense identification. *WordNet: Electron lexical database* (1998) 49: 265–83.

64. Zhang K, Yao Y, Xie R, Han X, Liu Z, Lin F, et al. Open hierarchical relation extraction. Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (2021). p. 5682–93.

65. Weinberger KQ, Chapelle O. Large margin taxonomy embedding for document categorization. *Adv Neural Inf Process Syst* (2009) 1737–44.

66. Xie R, Liu Z, Sun M, et al. Representation learning of knowledge graphs with hierarchical types. *IJCAI* (2016) 2965–71.

67. Toba H, Ming ZY, Adriani M, Chua TS. Discovering high quality answers in community question answering archives using a hierarchy of classifiers. *Inf Sci* (2014) 261:101–15. doi:10.1016/j.ins.2013.10.030

68. Han X, Yu P, Liu Z, Sun M, Li P. Hierarchical relation extraction with coarse-to-fine grained attention. Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (2018). p. 2236–45.

69. Zhang N, Deng S, Sun Z, Wang G, Chen X, Zhang W, et al. "Long-tail relation extraction via knowledge graph embeddings and graph convolution networks," In Proceedings of the 2019 conference of the north american chapter of the association for computational linguistics: Human language technologies, Volume 1 (Long and Short Papers). Minneapolis, Minnesota: Association for Computational Linguistics (2019). p. 3016–25. doi:10.18653/v1/N19-1306

70. Lin Y, Liu Z, Sun M, Liu Y, Zhu X. *Learning entity and relation embeddings for knowledge graph completion*. Twenty-ninth AAAI conference on artificial intelligence (2015). p. 1–7.