Fact Checking in Knowledge Graphs by Logical Consistency

Ji-Seong Kim [a] and Key-Sun Choi [a,*]
[a] *Semantic Web Research Center, School of Computing, KAIST, Daejeon, Korea*
E-mails: jiseong@kaist.ac.kr, kschoi@kaist.ac.kr

Abstract. Misinformation spreads across media, community, and knowledge graphs in the Web by not only human agents but also information extraction systems that automatically extract factual statements from unstructured textual data to populate existing knowledge graphs. Traditional fact checking by experts is increasingly difficult to keep pace with the volume of newly created information in the Web. Therefore, it is important and necessary to enhance the computational ability to determine whether a given factual statement is truthful or not. In this paper, our goal is to 1) mine weighted logical rules from a knowledge graph, 2) to find positive and negative evidential paths in a knowledge graph for a given factual statement by the mined rules, and 3) to calculate a truth score for a given statement by an unsupervised ensemble of the found evidential paths. For example, we can determine the statement "The United States is the birth place of Barack Obama" as truthful since there is the positive evidential path `(Barack Obama, birthPlace, Hawaii)` $\wedge$ `(Hawaii, country, United States)` in a knowledge graph, and it is logically consistent with the given statement. On the contrary, we can determine the factual statement "Canada is the nationality of Barack Obama" as untruthful since there is the negative evidential path `(Barack Obama, birthPlace, Hawaii)` $\wedge$ `(Hawaii, country, United States)` $\wedge$ `(United States,` $\neq$ `, Canada)` in a knowledge graph, and it is logically contradictory to the given statement. For evaluation, we constructed a novel evaluation dataset by labeling true or false labels on the factual statements extracted from Wikipedia texts by the state-of-the-art BERT-based relation extractor. Our evaluation results show that the proposed weighted logical rule-based approach outperforms the state-of-the-art unsupervised approaches significantly by up to 0.12 AUC-ROC, and even outperforms the supervised approach by up to 0.05 AUC-ROC not only in our dataset but also in the two publicly available datasets. The source code and evaluation dataset proposed in this paper is open-source and available at https://github.com/machinereading/KV-rule and https://github.com/machinereading/KV-eval-dataset each.

Keywords: Semantic Web, Knowledge Graph, Fact Checking, Rule Mining, Weighted Logical Rule

1. Introduction

Misinformation in the Web. Misinformation in the Web creates a situation in which false factual statements compete for attention to true factual statements necessary for users and applications. Misinformation in media and community makes difficult for users to search the information they need, and misinformation in knowledge graphs (DBpedia [1], YAGO [2], Freebase [3], K-Box [4]) makes difficult for applications (Open Knowledge Base and Question Answering [5–9], Google Knowledge Panel[1], Apple Siri[2], IBM Watson[3], Wolfram Alpha[4]) to get the outputs they expect. This problem is common and getting worse in modern digital society. Although a lot of information in the Web is still a good resource, there is certainly no guarantee that a given factual statement is true or not. In order not to be fooled by false statements, it is necessary to separate true statements from false ones by assessing truthfulness of factual statements.

*Corresponding author. E-mail: kschoi@kaist.ac.kr.

[1] https://support.google.com/knowledgepanel/
[2] https://www.apple.com/ios/siri/
[3] https://www.ibm.com/watson/
[4] https://www.wolframalpha.com/

Fact Triple and Knowledge Graph. In this paper, we represent a factual statement as a fact triple `(subject, predicate, object)` where `subject` and `object` are entities that have a relationship between them as indicated by `predicate`. For example, the factual statement "Leonardo da Vinci is known for Mona Lisa" can be represented as the fact triple `(Leonardo da Vinci, knownFor, Mona Lisa)`. A set of such triples is called a knowledge graph where nodes represent the entities and directed edges represent the predicates. Different predicates can be represented by edge types. If a knowledge graph was complete to know all the facts, the fact checking problem would be as easy as checking whether a given statement is contained in a knowledge graph or not. In reality, the information contained in a knowledge graph is incomplete.

Misinformation in Knowledge Graphs by Information Extraction. Some information extraction systems [10–12] try to populate incomplete knowledge graphs by finding new fact triples missing in a knowledge graph from unstructured textual data in the Web. However, as the information extraction task is challenging and an accuracy of such algorithms is not yet complete, they often produce incorrect outputs which result in corrupting a knowledge graph with false fact triples. Our dataset constructed in this paper indicates that 83.51% of the fact triples extracted from Wikipedia texts by the state-of-the-art BERT-based relation extractor [10] are actually false.

Embedding-based Approaches. Some embedding-based approaches (KBCNN [13], ConvKB [14]) try to verify newly found fact triples using knowledge graph embedding techniques. The advantage of such approaches is to be able to verify the entity pairs unlinked in a knowledge graph; i.e., there is an advantage in the coverage of the verifiable triples. On the contrary, according to the comprehensive evaluation in some studies [15], it is shown that the embedding-based approaches are weak in verifying some fact triples that can be easily verified using logical reasoning.

Rule-based Approaches. Some rule-based approaches (KStream and KLinker [16], COPPAL [17], RUDIK [18, 19], PredPath [20]) try to verify newly found fact triples using rule mining techniques. The advantage of such approaches is to be able to use logical reasoning in verification by Horn rules, and verified results are interpretable by the explicit representation of the rules. The disadvantage is that the rule-based approaches are not able to verify the entity pairs unlinked in a knowledge graph; i.e., there is a disadvantage in the coverage of the verifiable triples.

Embedding-based Approaches vs. Rule-based Approaches. The pros and cons of embedding-based and rule-based approaches are summarized in Table 1. The embedding-based approaches are able to verify the entity pairs unlinked in a knowledge graph (i.e., more verifiable), but the rule-based approaches are not able to verify the unlinked entity pairs (i.e., less verifiable). The embedding-based approaches are strong in statistical reasoning, while the rule-based approaches are strong in logical reasoning. The embedding-based approaches use numerical vectors (i.e., less interpretable), while the rule-based approaches use expressive rules (i.e., more interpretable). Putting it all together, the embedding-based and rule-based approaches are in a relationship in which the strength of each of the two approaches can compensate for the weakness of the other. According to our experiments [21], using an ensemble of embedding-based approach [13] and rule-based approach [21] shows better performance in a fact checking task than using each alone, as shown in Table 2. This result indicates that the embedding-based and rule-based approaches are complementary each other in a fact checking task.

Goal. Our goal in this paper is to verify truthfulness of fact triples which can serve as a key clue to separate true triples from false ones. We view this problem as a truth scoring task in a knowledge graph, which is to assign a truth score ranging from 0.0 (false) to 1.0 (true) to a given fact triple based on supporting evidential paths that can be found in a knowledge graph. Specifically, our goal is to 1) mine weighted logical positive and negative rules from a knowledge graph, 2) to find positive and negative evidential paths in a knowledge graph for a given triple by the mined weighted logical rules, and 3) to calculate a logical consistency-based truth score for a given triple by an unsupervised ensemble of the found evidential paths. An example of truth scoring by weighted logical positive and negative rules are shown in Fig. 1; Given the fact triple `(Wozniak, education, UC Berkely)`, we can determine that the given triple is close to true since there is the positive evidential path `(Wozniak, almaMater, UC Berkely)` in a knowledge graph, which is logically consistent with the given triple. On the contrary, given the fact triple `(Wozniak, birthPlace, Florida)`, we can determine that the given triple is close to false since there is the negative evidential path `(Wozniak, birthPlace, California)` $\wedge$

Table 1
Pros and Cons of Embedding-based and Rule-based Approaches

	Verifiability	Reasoning Capability	Interpretability
Embedding-based FC	▲ Unlinked Entities Verifiable	Strong in Statistical Reasoning	▼ Numerical Vector Reprsentation
Rule-based FC	▼ Only Linked Entities Verifiable	Strong in Logical Reasoning	▲ Expressive Rule Representation

Table 2
Performance of Embedding-based and Rule-based Approaches

Performance in a Fact Checking Task			
	Precision	Recall	F1
Embedding-based Approach	0.8078	0.3261	0.4646
Rule-based Approach	**0.914**	0.5786	0.7086
Ensemble (Embedding + Rule)	0.8644	**0.742**	**0.7985**

(California, $\neq$, Florida) in a knowledge graph, which is logically contradict to the given triple.

Contributions. Our contributions in this paper are summarized as follows:

- An unsupervised weighted logical positive and negative rule-based approach that outperforms the state-of-the-art unsupervised rule-based approaches significantly by up to 12.68%, and even outperforms the supervised rule-based approach by up to 5.67% in the three evaluation datasets.
- To learn weighted logical positive and negative rules, we reimplemented and improved the existing negative sampling method and rule weighting measure proposed by Ortona et al. [18] by averagely 5.09% and 7.85% each. In addition, the difference from Ortona's approach is that we use an ensemble of both positive and negative rules to solve a fact checking problem. We believe that these improvements are the key factor for outperforming the state-of-the-art approaches.
- A novel evaluation dataset for the fact checking problem, which comprises new fact triples missing in a KG. It makes our dataset more suitable and challenging than the publicly available datasets when evaluating the fact checking capability on newly found fact triples.

2. Related Studies in Fact Checking

Approaches to solve a fact checking problem can be broadly classified into the two types according to the source of evidence used to verify a given fact triple. The first type is a text-based approach [22–24] that finds evidential sentences from unstructured textual data in the Web to check whether a given fact triple is true or not. The second type is a knowledge graph-based approach that uses schema-level constraints (SHACL[5], ShEx[6]) or instance-level fact triples in a knowledge graph as evidence to check whether a given fact triple is true or not. Knowledge graph-based approaches can be further classified into embedding-based and rule-based approaches. Our approach is more relevant to the rule-based approach in the knowledge graph-based approach.

2.1. Embedding-based Approaches

Some embedding-based approaches (KBCNN [13], ConvKB [14], TransE [25]) try to verify fact triples using knowledge graph embedding techniques. Specifically, the embedding-based approaches verify a given fact triple by checking whether an embedding representation of a given fact triple is plausible in the embedding space of a knowledge graph; i.e., they check

[5]https://www.w3.org/TR/shacl/
[6]https://shex.io/shex-semantics/index.html/

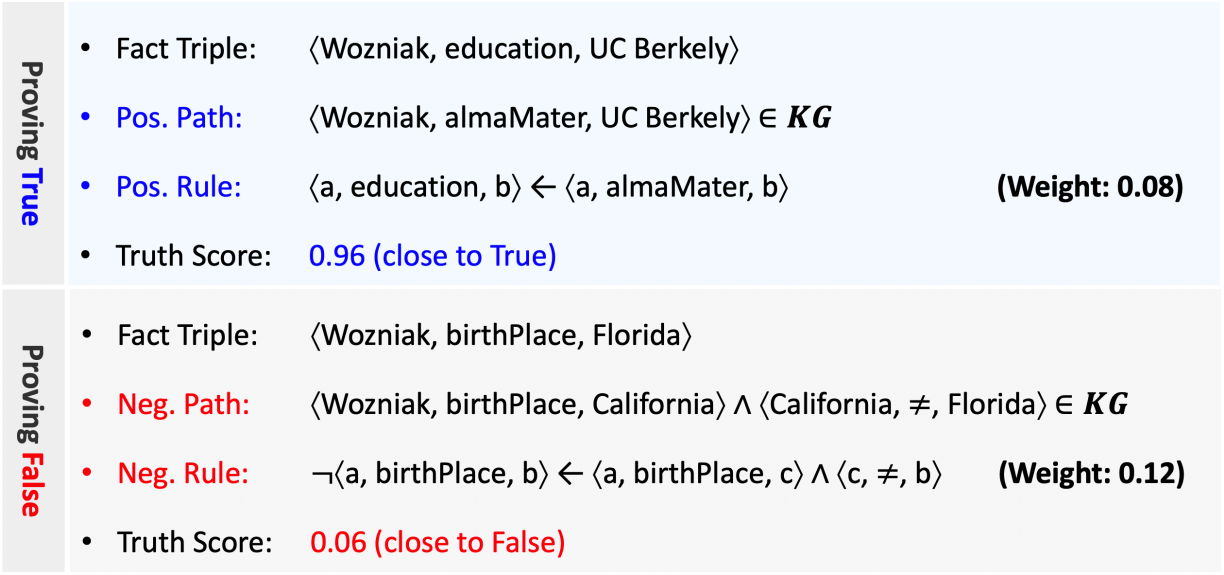


Fig. 1. An Example of Truth Scoring by Weighted Logical Positive and Negative Rules.

whether a given fact triple is contextually consistent with the fact triples in a knowledge graph. The advantage is to be able to verify the entity pairs unlinked (i.e., there is no path between two entities) in a knowledge graph; i.e., there is an advantage in the coverage of the verifiable triples. On the contrary, according to the comprehensive evaluation in some studies [15], it is shown that the embedding-based approaches are weak in verifying some fact triples that can be easily verified using logical reasoning.

2.2. *Rule-based Approaches*

Some rule-based approaches (KStream and KLinker [16], COPPAL [17], RUDIK [18, 19], PredPath [20]) try to verify fact triples using rule mining techniques. The advantage of the rule-based approaches is to be able to use logical reasoning in verification by Horn rules, and verified results are interpretable by the explicit representation of the rules. The disadvantage is that the rule-based approaches are not able to verify the entity pairs unlinked in a knowledge graph; i.e., there is a disadvantage in the coverage of the verifiable triples.

The rule-based approaches can be classified into the three types. The first type uses only positive rules to find positive evidential paths in a knowledge graph to verify a given fact triple. For example, the fact triple `(Leonardo da Vinci, knownFor, Mona Lisa)` is determined to be close to true if there is the positive evidential fact triple `(Leonardo da Vinci, author, Mona Lisa)` in a knowledge graph. Shiralkar et al. [16] proposed a network flow-based unsupervised approach, called Knowledge Stream (KStream) and Relational Knowledge Linker (KLinker), which use a stream of knowledge to find positive evidential paths in a knowledge graph, which support that a given fact triple is true. Syed et al. [17] proposed a meta path-based unsupervised approach, called COPPAL, which uses a corroborative meta path to find the positive evidential paths in a knowledge graph, which support that a given fact triple is true.

Although positive evidential paths are still a good clue for determining truthfulness of a given fact triple, in some cases, negative evidential paths are necessary because of the incompleteness of a knowledge graph. For example, if there are the fact triples `(Barack Obama, birthYear, 1961)` and `(Ann Dunham, birthYear, 1942)` in a knowledge graph, and the knowledge graph does not contain any more information about the two people, we are only able to determine the given fact triple `(Barack Obama, child, Ann Dunham)` as false by the negative evidential path `(Barack Obama, birthYear, 1961)` ∧ `(1961, >, 1942)` ∧ `(Ann Dunham, birthYear, 1942)`, which means "The birth year of Barack Obama is later than that of Ann

Dunham", which implies Ann Dunham cannot be a child of Barack Obama.

On the other side, the second type of the rule-based approaches uses only negative rules to find negative evidential paths in a knowledge graph to verify a given fact triple. This type of approaches suffers from the afore-mentioned same issue with the first type of approaches. Ortona et al. [18, 19] proposed a Horn rule-based unsupervised approach, called RUDIK, which uses negative rules to find the negative evidential paths in a knowledge graph, which support that a given fact triple is false. To make false fact triples used for generating and validating negative rules, the authors of RUDIK proposed a negative sampling method based on Extended Local Closed World Assumption (E-LCWA). The E-LCWA-based negative sampling method generates false fact triples by wrongly connecting the entities directly connected in a knowledge graph. By doing that, the E-LCWA-based method is able to generate more probable false fact triples between semantically related entities that are connected by more meaningful paths. Ortona et al. [18] states that E-LCWA always true for functional predicates (e.g., capital), but it might not hold for non-functional predicates (e.g., child). In this paper, our purpose is to verify all the functional and non-functional predicates that have one or more instances in a knowledge graph. To make a negative sampling method that works on both functional and non-functional predicates, we reimplemented Ortona's approach as a baseline and applied it on non-functional predicates. According to our analysis, using directly adjacent entities works in functional predicates as presented by Ortona et al. [18], but is probable to make true fact triples, not false ones for non-functional predicates. For example, among the false fact triples about the predicate `relative` generated by using directly adjacent entities as the replacement, at least 47.54% of them are indeed true, not false.

The last type of the rule-based approaches uses both positive and negative rules to find both types of evidential paths in a knowledge graph to verify a given fact triple. These approaches suffer less from the afore-mentioned knowledge graph incompleteness issue than the first and second type of approaches. For example, our ablation study in this paper shows that using both positive and negative rules is more effective by up to 0.25 AUC-ROC in a truth scoring task than using only a single type of rules. Shi and Weninger [20] proposed a predicate path-based supervised approach, called PredPath, which uses a positive and negative discriminative predicate path to find positive and negative evidential paths in a knowledge graph, which support that a given fact triple is true or false. PredPath weights a discriminative predicate path by only considering the correct examples to be covered by the path, and ignoring counter examples not to be covered by the path. The discriminative predicate path that they proposed is a kind of rule, and a rule can only be properly weighted by checking whether a rule covers correct examples as well as checking whether a rule does not cover counter examples. For example, our ablation study in this paper shows that the rule weighting by both correct and counter examples has an average of 0.07 AUC-ROC more effective in a truth scoring task than the rule weighting by only correct examples.

3. Fact Checking by Logical Consistency

3.1. Overview

In this paper, we address a truth scoring problem in a knowledge graph: Given a knowledge graph `G` and a fact triple `(s, p, o)`, a truth scoring task is to calculate a truth score ranging from 0.0 (false) to 1.0 (true) for the given fact triple. A truth score gets closer to 1.0 if the given fact triple becomes more truthful.

To calculate a truth score for the given fact triple, we find positive evidential paths in a knowledge graph, which support the given triple is true, as well as negative evidential paths in a knowledge graph, which support the given triple is false. To find positive and negative evidential paths in a knowledge graph, we generate and use weighted logical positive and negative rules.

A positive rule comprises a generalized positive evidential path in a body part of the rule and a generalized fact triple in a head part of the rule. For example, the rule `(x, education, y)` ← `(x, almaMater, y)` is a positive rule, which means that a fact triple `(x, education, y)` is truthful if there is a corresponding positive evidential path `(x, almaMater, y)` in a knowledge graph.

A weighted logical positive rule is a positive rule that is weighted according to its logical validity. For example, the positive rule `(x, education, y)` ← `(x, almaMater, y)` is logically valid, and thus should be strongly weighted since if a person graduated from a certain school, that person must be educated at that school. In contrast, the positive rule `(x, education, y)` ← `(x, residence, z)` ∧ `(y, location, z)` is log-

ically less valid, and thus should be weakly weighted since a person and a school that are in the same place does not mean that the person is graduated from that school.

A negative rule comprises a generalized negative evidential path in a body part of the rule and a generalized and negated fact triple in a head part of the rule. For example, the rule $\neg$(x, birthPlace, y) $\leftarrow$ (x, birthPlace, z) $\wedge$ (z, $\neq$, y) is a negative rule which means that a fact triple (x, birthPlace, y) is untruthful if there is a negative evidential path (x, birthPlace, z) $\wedge$ (z, $\neq$, y) in a knowledge graph.

A weighted logical negative rule is a negative rule that is weighted according to its logical validity. For example, the rule $\neg$(x, birthPlace, y) $\leftarrow$ (x, birthPlace, z) $\wedge$ (z, $\neq$, y) is logically valid, and thus should be strongly weighted since no one can be born in two places. In contrast, $\neg$(x, birthPlace, y) $\leftarrow$ (x, residence, z) $\wedge$ (z, $\neq$, y) is logically less valid, and thus should be weakly weighted since the place of birth is probable to be different from the place of residence.

To assess logical validity of rules, false fact triples are required as counter examples for positive rules and correct examples for negative rules. The challenge is that a knowledge graph follows *open world assumption*, and *closed world assumption* does not longer hold, which means that a knowledge graph contains only true fact triples, and the fact triples missing in a knowledge graph are not able to be labeled as false. To deal with this challenge, we use a negative sampling method to generate false fact triples.

3.2. Overall Workflow

Our approach is a pipeline that comprises mainly three steps: (1) generating false fact triples by a negative sampling method, (2) generating positive and negative rules, and then weighting them according to their logical validity, (3) calculating a truth score for a given fact triple by an ensemble of the most valid positive and negative rules that imply the given fact triple. Fig. 2 shows the overall workflow of our approach in this paper.

In the first step, we generate true and false fact triples to generate positive and negative rules. Since a knowledge graph itself is a set of true fact triples, we use the existing true fact triples contained in knowledge graph to generate positive rules. In contrast, a knowledge graph follows *open world assumption*, and is not intended to contain false fact triples. Therefore, we generate false fact triples from existing true fact triples in a knowledge graph by a negative sampling method, and use the generated false fact triples to generate negative rules.

In the second step, positive and negative rules are generated and weighted using the generated true and false fact triples. In rule generation, positive rules are generated using true fact triples. In contrast, negative rules are generated using false fact triples. To weight a positive rule according to its logical validity, true fact triples are used as correct examples for the given rule to cover, and false fact triples are used as counter examples for the given rule not to cover. In contrast, to weight a negative rule according to its logical validity, false fact triples are used as correct examples for the given rule to cover, and true fact triples are used as counter examples for the given rule not to cover.

In the last step, to calculate a truth score for a given fact triple, we find the most valid positive and negative rules that covers (implies) the given fact triple, and use an ensemble of the found most valid positive and negative rules in calculating a truth score for the given fact triple.

3.3. Negative Sampling

In our approach, false fact triples are required to assess logical validity of positive and negative rules. Some approaches rely on the presence of both true and false facts [26, 27], but a knowledge graph contains only true fact triples, and it follows *open world assumption*, which states that the truth of fact triples missing in a knowledge graph is unknown, not false. Since *closed world assumption* (CWA) does not hold longer in a knowledge graph, we cannot assume that a fact triple missing in a knowledge graph is false, and without CWA, we cannot derive false fact triples. Therefore, we have to generate false fact triples from the existing true fact triples in a knowledge graph by a negative sampling method.

Some studies [28–30] used a negative sampling method based on *local closed world assumption* (LCWA). LCWA states that if a knowledge graph contains one or more object values for a given subject and predicate, then it contains all possible values (i.e., closed world). For example, if a knowledge graph contains one or more children of Clint Eastwood, then it contains all his children; i.e., if the other people not Clint Eastwood's children in a knowledge graph are linked

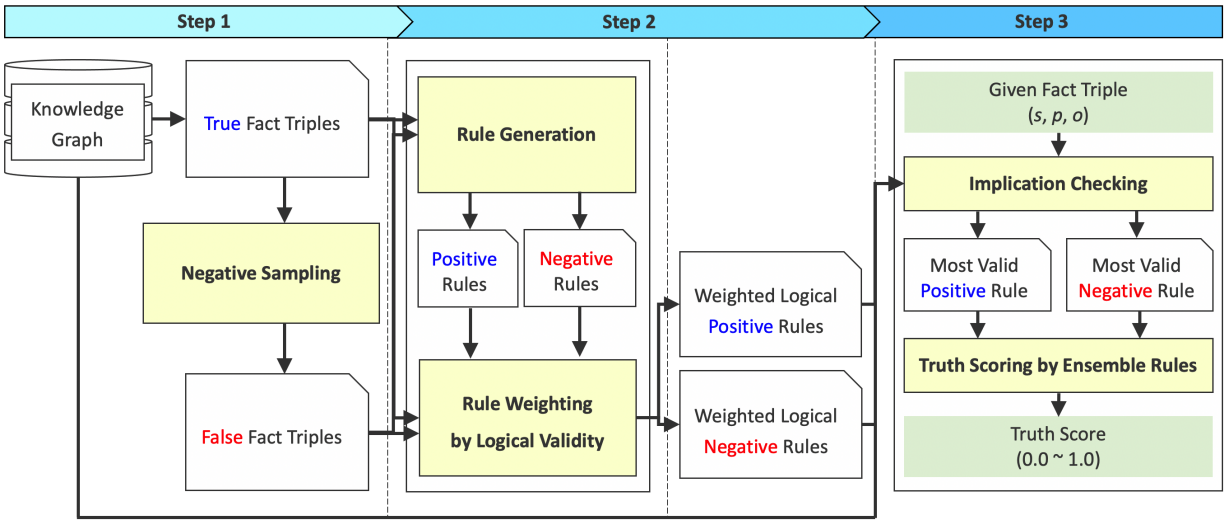


Fig. 2. The Overall Workflow of The Proposed Approach.

by the predicate `child` with Clint Eastwood, then the links can be labeled as false.

Specifically, the LCWA-based negative sampling method generates false fact triples as follows: given a true fact triple `(s, p, o)`, a false fact triple `(s, p, o')` is generated by replacing `o` with `o'` that has the same type with `o`. Likewise, given a true fact triple `(s, p, o)`, a false triple `(s', p, o)` is generated by replacing `s` with `s'` that has the same type with `s`. This process gives false fact triples with a very high precision since the replacement `o'` (or `s'`) can be chosen at random, and the likelihood of two randomly selected entities being a true fact triple is extremely low. However, only a very small fraction of these entity pairs are semantically related, and unrelated entities have less meaningful paths between them.

To overcome this issue, Ortona et al. [18] proposed a negative sampling method based on *extended local closed world assumption* (E-LCWA). The E-LCWA-based negative sampling method generates false fact triples as follows: given a true fact triple `(s, p, o)`, a false fact triple `(s, p, o')` is generated by replacing `o` with `o'` that has the same type with `o`, and is directly adjacent to `s` in a knowledge graph. Likewise, given a true fact triple `(s, p, o)`, a false triple `(s', p, o)` is generated by replacing `s` with `s'` that has the same type with `s`, and is directly adjacent to `o` in a knowledge graph.

The difference between the LCWA-based and E-LCWA-based methods is that the E-LCWA-based negative sampling method chooses the replacement `o'` (or `s'`) not at random, but at the entities that are directly adjacent to `s` (or `o`). By doing that, the E-LCWA-based method is able to generate more probable false fact triples between semantically related entities that are connected by more meaningful paths. Ortona et al. [18] states that E-LCWA always true for functional predicates (e.g., `capital`), but it might not hold for non-functional predicates (e.g., `child`).

In this paper, our purpose is to verify all the functional and non-functional predicates that have one or more instances in a knowledge graph. According to our analysis, the number of non-functional predicates in a knowledge graph is larger than the functional predicates, which indicates the importance of the negative sampling method working on both functional and non-functional predicates. For example, 67.67% (161/498) of the object type predicates that have one or more instances in the knowledge graph, K-Box [4], are non-functional (i.e., they have more than one object values on average for a given subject entity), and in the case of English DBpedia [1], 85% (93/620) of the object type predicates are non-functional.

To make a negative sampling method that works on both functional and non-functional predicates, we reimplemented Ortona's approach as a baseline and applied it on non-functional predicates. According to our analysis, using directly adjacent entities as the replacement `o'` (or `s'`) works in functional predicates as presented by Ortona et al. [18], but is probable to make true fact triples, not false ones for non-functional predicates. For example, among the false fact triples about

the predicate relative generated by using directly adjacent entities as the replacement, at least 47.54% of them are indeed true, not false; The reason is that the two entities directly connected by the predicates, child, parent, spouse, and so on, are used to generate false fact triples by connecting the predicate relative between them, and these lead to true fact triples, not false ones.

To resolve this issue, we present a novel negative sampling method for functional and non-functional predicates using the entities that are less adjacent, but not too far between them to keep semantic relatedness as proposed in Ortona's study. Specifically, we present a negative sampling method based on *distant local closed world assumption* (D-LCWA). D-LCWA states that if a knowledge graph contains one or more object values for a given subject and predicate, then there may be other possible values directly adjacent to the given subject and no other possible values less adjacent to the given subject. For example, if a knowledge graph contains one or more children of Clint Eastwood, then it may contain other children directly adjacent to Clint Eastwood in a knowledge graph. The D-LCWA-based negative sampling method generates false fact triples as follows: given a true fact triple (s, p, o), a false fact triple (s, p, o′) is generated by replacing o with o′ that has the same type with o, is not directly adjacent to s, and is located at a distance maxPathLen or less from s in a knowledge graph. For example, given the true fact triple (Clint Eastwood, child, Alison Eastwood), the D-LCWA-based negative sampling method generates the false fact triple (Clint Eastwood, child, Laura Linney) if Laura Linney has the same type with Alison Eastwood, is not directly adjacent to Clint Eastwood, and is located at a distance 3 or less from Clint Eastwood in a knowledge graph. Likewise, given a true fact triple (s, p, o), a false triple (s′, p, o) is generated by replacing s with s′ that has the same type with s, is not directly adjacent to o, and is located at a distance maxPathLen or less from o in a knowledge graph.

LCWA vs. E-LCWA vs. D-LCWA. The difference among the LCWA-based, E-LCWA-based, and D-LCWA-based negative sampling methods are shown in Fig. 3. Given the true fact triple (Hee-yeol You, almaMater, Seoul National University), the E-LCWA-based negative sampling method generates (Tae-hee Kim, almaMater, Seoul National University) as a false fact triple by replacing Hee-yeol You with Tae-hee Kim that is directly adjacent to Seoul National University. The proposed D-LCWA-based negative sampling method generates (Jong-shin Yoon, almaMater, Seoul National University) as a false fact triple by replacing Hee-yeol You with Jong-shin Yoon that is less adjacent to Seoul National University. The LCWA-based negative sampling uses directly adjacent entities, less adjacent entities, and far distant entities as the replacement, and is probable to generate (Steve Wozniak, almaMater, Seoul National University) as a false fact triple by replacing Hee-yeol You with Steve Wozniak that has no meaningful path to Seoul National University.

3.4. Rule Generation

We generate a set of positive rules using true fact triples in a knowledge graph. In contrast, negative rules are generated using false fact triples that are generated by a negative sampling method. The rule generation process is almost the same for positive and negative rules, and the only difference is whether an input fact triple is true or false. The rule generation process consists of three steps, as follows:

- Before generating a rule, we first construct a local graph for each input fact triple. A local graph for a fact triple (s, p, o) is a subgraph in a knowledge graph, which comprises a set of paths between s and o, which have a length maxPathLen or less. To enhance the expressive power of generated rules, we add the predicate $\neq$ to a local graph by comparing all combinations of object type entities, such as person, location, organization, and so on, in a local graph. Since comparing all the entities in a knowledge graph is inefficient, we compare only the entities of the same type in a local graph. To prevent from connecting the different entity URIs (Unique Resource Identifiers) pointing to the same entity by the predicate $\neq$, we unified the different entity URIs pointing to the same entity in a knowledge graph into a representative entity URI by using the redirect information provided in a knowledge graph. For example, dbr:Einstein, dbr:Einsteinian, dbr:Einstein, Albert, and so on are unified into the representative entity URI, dbr:Albert Einstein, according to the redirect information provided by the knowledge graph, DBpedia [1]. To enhance the expressiveness of rules

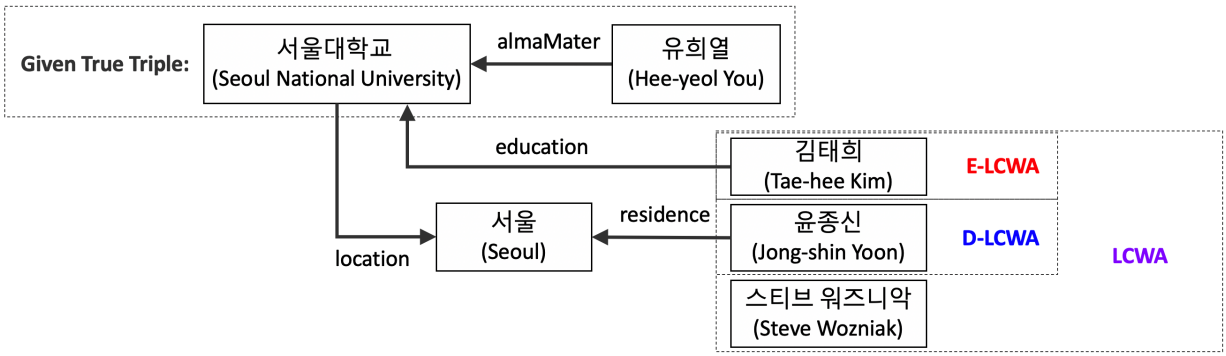


Fig. 3. The Difference among The Negative Sampling Methods.

further, we add the predicates, $>$ and $<$, to a local graph by comparing all combinations of datatype entities contained in a local graph, such as integer, real number, and datetime. Similar to the predicate $\neq$, we compare only the datatype entities of the same type in a local graph.

– After constructing a local graph for each input fact triple, we generate a rule instance using a constructed local graph. Specifically, given an input fact triple `(s, p, o)`, we generate a rule instance whose head part is the given input triple `(s, p, o)`, and body part is a path between `s` and `o` in a local graph, which has a length `maxPathLen` or less. A path in the body part of a rule instance is started from a node `s`, and expanded by 1-hop using the breadth-first search until that a length of a path becomes `maxPathLen` or less. For example, if `maxPathLen` is three, we generate all the paths that have a length from one to three. The expansion of paths proceeds on an undirected local graph that is a modified local graph by removing all the directions of the edges. After finding all the paths by the expansion, we discard the paths such that a node `o` is not located at the end of a path, or there is a cycle in a path, or the predicates, $\neq$, $>$, and $<$, are contained more than once in a path to remove redundant paths. When generating an instance of positive rules, we discard the paths that contain the predicates, $\neq$, $>$, and $<$, since positive rules containing such operators are less meaningful according to our observation on the data. After removing the redundant paths, we restore the directions of the edges in the remaining paths, and generate them as the body part of a rule instance.
– After generating rule instances, we generate a generalized rule by replacing all the entities contained in a rule instance with corresponding variables.

It is notable that if true fact triples are provided as input fact triples for the afore-mentioned three steps, then positive rules are generated. In contrast, if false fact triples are provided as input fact triples, then negative rules are generated.

3.5. Rule Weighting by Logical Validity

A rule weighting task is to calculate a weight score for a given rule based on how much the given rule is logically valid. A weight score is a real value ranging from 0.0 (valid) to 1.0 (invalid) to express the logical validity of the given rule. For example, the positive rule `(x, child, y)` $\leftarrow$ `(y, parent, x)` is valid since the given rule covers all correct examples and does not cover any counter examples. In contrast, the positive rule `(x, child, y)` $\leftarrow$ `(x, residence, z)` $\wedge$ `(y, residence, z)` is less valid since all people lived with someone are not children of him or her. A weight score gets close to 0.0 if a given rule is valid, and has the ability to find strong evidential paths in a knowledge graph.

To properly weight according to the logical validity of a given rule, we should check whether a given rule covers correct examples and whether a given rule does not cover counter examples. In the case of a positive rule, a set of true fact triples in a knowledge graph are the correct examples to cover, and a set of false fact triples generated by negative sampling methods are the counter examples not to cover. Conversely, in the case of a negative rule, a set of false fact triples are the cor-

rect examples to cover, and a set of true fact triples are the counter examples not to cover.

Most rule-based approaches, KStream, KLinker, COPPAL, and PredPath, only use correct examples to calculate a weight score for a given rule. In our observation, the rule weighting without using counter examples often makes an inappropriate situation in which less valid rules are strongly weighted. For example, the less valid positive rule `(x, child, y)` ← `(x, nationality, z)` ∧ `(y, nationality, z)` that covers 76% of correct examples (i.e., a set of true fact triples) is strongly weighted even though the given rule covers 46% of counter examples (i.e., a set of false fact triples).

To prevent this issue, we reimplemented the rule weighting measure proposed by Ortona et al. [18] to use both correct and counter examples to calculate a weight score of a given rule. Specifically, we calculate a weight score of a given rule r by the weight measure $w_2(r)$, which uses a set of correct examples E_{corrt}, and a set of counter examples E_{count} in calculation, as follows:

$$w_2(r) = \alpha \frac{1 - |C_r(E_{corrt}))|}{|U_r(E_{corrt})|} + \beta \frac{|C_r(E_{count})|}{|U_r(E_{count})|}$$

where $C_r(E)$ is the number of the fact triples in E covered by r, $U_r(E)$ is the number of the fact triples in E covered by unbounded r, and α and β are the real constants that become 1.0 when added. An unbounded rule is obtained by replacing variables paired with `x` and `y` in the body part of a rule to new unique variables. For example, given the rule `(x, child, y)` ← `(x, residence, z)` ∧ `(y, residence, z)`, the unbounded version of the given rule is `(x, child, y)` ← `(x, residence, a)` ∧ `(y, residence, b)` obtained by replacing the variable `z` to the new unique variables, `a` and `b`.

This weight measure, w_2, is designed to produce the stronger weight score if a given rule covers more correct examples and less counter examples. According to our analysis, this weight measure works in most cases, but not in the case that a rule covers the small number of correct and counter examples. For example, given the less valid positive rule `(x, spouse, y)` ← `(x, occupation, z)` ∧ `(y, occupation, z)` which covers 5% of correct examples (i.e., a set of true fact triples) and 3% of counter examples (i.e., a set of false fact triples), the weight score, w_2, of the given rule becomes 0.12, which is the fairly strong weight score close to 0.0. It is certain that a weight score should not be overestimated over the logical validity of a given rule. To handle this issue, we present the improved weight measure, $w_c(r)$, as follows:

$$w_c(r) = 1 - \frac{C_r(E_{corrt}) - \frac{1}{\gamma} C_r(E_{count})}{C_r(E_{corrt})} (1 - w_2(r))$$

where γ is a real constant ranging from 0.0 to 1.0, which is to adjust the effect of $C_r(E_{count})$ on the overall weight score. The parameter γ is required since there are the different between the qualities of true and false fact triples. We use true fact triples in a knowledge graph, which are mostly correct. In contrast, we use false fact triples automatically generated by negative sampling methods, which are probable to contain noise labels that lead to the reduced effect of the false fact triples on the overall weight score.

The improved weight measure, w_c, has the constraint that the overall weight score is lowered as much as $C_r(E_{count})$ is greater than or similar with $C_r(E_{corrt})$, which prevents less valid rules from being strongly weighted. According to our analysis, the proposed rule weighting using both correct and counter examples is more effective for a truth scoring task by an average of 0.0746 AUC-ROC than the rule weighting using only correct examples.

3.6. Truth Scoring by an Ensemble of Rules

To calculate a truth score for a given fact triple, we find the most valid positive and negative rules that cover (imply) the given fact triple.

If a rule covers a fact triple, it indicates that there is an evidential path corresponding to the body part of the rule in a knowledge graph. For example, if the fact triple `(Barack Obama, child, Sasha Obama)` is covered by the rule `(x, child, y)` ← `(y, parent, x)`, then it indicates that there is the evidential path `(Sasha Obama, parent, Barack Obama)` in a knowledge graph.

After finding the most valid positive and negative rules, which cover a given fact triple, `(s, p, o)`, we calculate a truth score for the given fact triple, $\mathcal{S}(s, p, o)$, by an unsupervised ensemble of the found positive and negative rules, as follows:

$$\mathcal{S}(s, p, o) = \frac{((1 - w_c(r_p)) - (1 - w_c(r_n)) + 1)}{2}$$

where r_p is the most valid positive rule that covers the given fact triple `(s, p, o)` and r_n is the most valid negative rule that covers the given fact triple `(s, p, o)`.

If we find the most valid rule, r_p, which covers the given fact triple `(s, p, o)`, then we assign $w_c(r_n)$ as 0.0, and calculate a truth score; i.e., if there is a positive evidential path found, we do not use a negative evidential path found in a truth scoring task since it shows the better performance by an average of about 1%.

Our truth scoring approach by an ensemble of the most valid rules is totally unsupervised, and does not require any hand-labeled data in training while outperforming the state-of-the-art unsupervised and supervised approaches in our dataset as well as publicly available two datasets.

4. Evaluation Dataset

4.1. Publicly Available Datasets

For evaluation, we used our dataset constructed in this paper, and also used two different publicly available datasets. First of all, we describe the characteristics and issues of the publicly available datasets, and then present our novel dataset constructed in the direction of solving the issues of the publicly available datasets.

- The *Synthetic* dataset constructed by Shiralkar et al. [16], which mainly comprises true-labeled fact triples manually extracted from Wikipedia tables, and false-labeled fact triples automatically generated by the LCWA-based negative sampling method.
- The *Real-World* dataset derived from Google Relation Extraction Corpora[7] and WSDM Cup Triple Scoring Challenge[8], which mainly comprises true-labeled fact triples manually extracted from Wikipedia texts by crowdsourcing, and false-labeled fact triples automatically generated by the LCWA-based negative sampling method.

[7] https://ai.googleblog.com/2013/04/50000-lessons-on-how-to-read-relation.html/

[8] https://www.wsdm-cup-2017.org/triple-scoring.html/

4.2. Characteristics and Issues on The Publicly Available Datasets

There are two common issues in the afore-mentioned publicly available datasets, as follows:

- False-labeled fact triples in the datasets are automatically generated by the LCWA-based negative sampling method that is probable to generate true fact triples, not false, as shown in this paper. Therefore, these datasets contain false-labeled true fact triples, which are the false negatives that is probable to make our evaluation inaccurate. According to our analysis, at least 4% of false-labeled fact triples in these datasets are already contained in the knowledge graph, DBpedia, which means that they are indeed false-labeled true fact triples. For accurate evaluation, we corrected such false-labeled fact triples by relabeling them as true, and we used the corrected datasets in our evaluation.
- Some true-labeled fact triples in these datasets are already contained in the existing knowledge graph, DBpedia. This means the test cases in the datasets can be easily solved by checking whether a given fact triple is contained in a knowledge graph or not. According to our analysis, 77.26% of true-labeled fact triples in the Synthetic dataset are already contained in DBpedia. In the case of the Real-World dataset, 8.58% of true-labeled fact triples are already contained in DBpedia, which seems small, but for some predicates, e.g., nationality and profession, 100% of true-labeled fact triples are contained in DBpedia. Therefore, the Synthetic dataset is a fairly easy dataset since we are able to solve 77.26% of true-labeled test cases easily by just checking whether they are contained in DBpedia. In contrast, the Real-World dataset is relatively difficult, but still we are able to solve 8.58% of true-labeled test cases easily by just checking whether they are contained in DBpedia.

4.3. Our Dataset Construction

In this paper, we constructed a novel dataset in the direction of solving the afore-mentioned issues. Specifically, our dataset is constructed by manually labeling true or false labels on fact triples extracted from Wikipedia texts by the state-of-the-art BERT-based relation extractor [10]. Our dataset is constructed to

solve the afore-mentioned issues in the two ways, as follows:

- All the false-labeled fact triples are manually checked to prevent from inaccurate evaluation by false-labeled true fact triples. Specifically, we labeled a given fact triple as true if there is a Wikipedia article that describes an evidential sentence that supports the given fact triple is true. Otherwise, we labeled the given fact triple as false.
- The true-labeled fact triples are included in our dataset only if a given fact triple is not already contained in the knowledge graph, K-Box [4].

These makes our dataset more reasonable and challenging to evaluate the fact checking capability on newly found fact triples missing in a knowledge graph than the other publicly available datasets. The statistics of our dataset and the publicly available two datasets are shown in Table 3. The detailed process for constructing our dataset is described in the following subsections.

4.3.1. Collecting A Raw Dataset

The main purpose of constructing our dataset is to evaluate the fact checking capability on the fact triples newly found by relation extraction. To collect a raw dataset, we extracted the K-Box [4] style fact triples from Korean Wikipedia abstracts by the BERT-based relation extractor [10]. Among the extracted fact triples, only those triples that satisfy all the following conditions were included in the raw dataset.

- A triple that is not contained in the knowledge graph, K-Box.
- A triple whose subject and object entities have at least one ontological type in K-Box.
- A triple that satisfies the domain and range constraints specified in the ontology schema of K-Box.

To ensure that only newly found triples are included in our dataset, we did not contain those triples already contained in K-Box. In order for our dataset to contain only triples related to a fact checking problem, we discarded those triples that are easily identified with errors by the domain and range constraints specified in an ontology schema. For example, the triple `(Incheon Station, owner, Songdo Station)` is not included in our dataset since the range constraint of the predicate `owner` is specified as the type `Agent` in the ontology schema of K-Box, and the object entity `Songdo Station` has the type `Place`, not `Agent`.

For the predicates whose domain or range constraint is not specified in the ontology schema of K-Box, we used the instances of the predicates contained in K-Box. For example, the triple `(Romani people, occupation, Musician)` is not included in our dataset since `Romani people` has the type `EthnicGroup`, and the instances of the predicate `occupation` in K-Box do not contain entities of the type `EthnicGroup` as their subject entities.

Among all the extracted fact triples that satisfy the afore-mentioned conditions, we sampled 2% for each predicate, and used them as a raw dataset. The statistics of the collected raw dataset are shown in Fig. 4.

4.3.2. True and False Labeling on The Raw Dataset

For each of the fact triples in the collected raw data, we manually assigned true or false label by checking whether there is a sentence in a Korean Wikipedia abstract, which supports that a given fact triple is true. Specifically, for a given triple `(s, p, o)`, if there is a sentence in Korean Wikipedia abstracts for entities `s` and `o`, which supports that the given triple is true, we label the given triple as true. Otherwise, we label the given triple as false. For example, the triple `(SG Wannabe, bandMember, Chae Dongha)` is labeled as true since there is the sentence "In 2008, original member Chae Dong-ha left the group" in the Wikipedia abstract of `SG Wannabe` as shown in Fig. 5, which supports that the given triple is true.

If there is no supporting sentence in a Wikipedia abstract, all the texts in a Wikipedia article are checked. In addition to a sentence in a Wikipedia text, we used the information in infoboxes to decide true or false label for the given triple. For example, the triple `(Dicronocephalus adamsi, family, Dicronocephalus)` is labeled as false since `Dicronocephalus` is a genus of `Dicronocephalus adamsi`, not a family according to the infobox specified in the Wikipedia article of `Dicronocephalus adamsi`, as shown in Fig. 6.

4.3.3. Dataset Construction Results

The statistics of the constructed dataset are shown in Fig. 7. Overall, 1,759 triples are labeled as true or false. 16.49% triples are labeled as true, and 83.51% triples are labeled as false. The false ratio is different from each predicate as shown in Fig. 7, and is dependent on the performance of a relation extractor.

Table 3
Statistics of The Synthetic, Real-World, and Our Datasets

Dataset	Predicate #	Fact Triples			True-False Ratio
		True #	False #	Total #	
Synthetic	8	839	7,253	8,092	10 : 90
Real-World	6	7,565	22,688	30,253	25 : 75
Ours	35	290	1,469	1,759	16 : 84

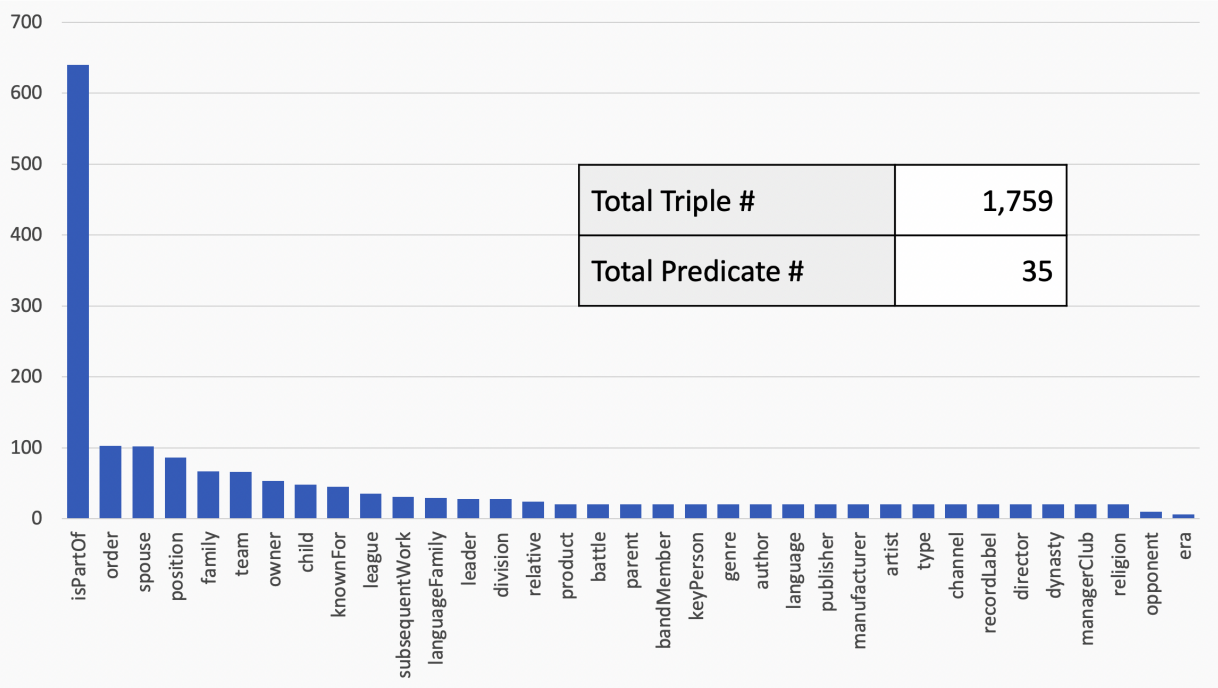


Fig. 4. Statistics of The Collected Raw Dataset.

SG Wannabe

From Wikipedia, the free encyclopedia

SG Wannabe (Korean: SG 워너비) is a South Korean vocal group consisting of members Kim Yong-jun, Kim Jin-ho and Lee Seok-hoon.[1] The group debuted in 2004 with the single, "Timeless", from the album *SG Wanna Be+*, and won Best New Artist at the Golden Disc Awards, Seoul Music Awards, and SBS Gayo Daejeon.[2] The group's second album, *Saldaga*, was the best-selling album of 2005 in South Korea and won Album of the Year at the Golden Disc Awards.[3][4] Their next two albums, *The 3rd Masterpiece* (2006) and *The Sentimental Chord* (2007) were also commercial and critical successes.[5]

In 2008, original member Chae Dong-ha left the group and Lee Seok-hoon joined as a new member.[6][7] The group released the albums *My Friend* (2008), *Gift from SG Wannabe* (2009), *SG Wannabe by SG Wannabe 7 Part.I* (2010) and *SG Wannabe by SG Wannabe 7 Part.II* (2011) before going on hiatus while the members served their mandatory military service and focused on their solo careers.

In 2015, the group released their first album in four years with the extended play *The Voice*, which they followed in 2016 with *Our Days*.

Fig. 5. An Example of Supporting Sentence in A Wikipedia Abstract.

Fig. 6. An Example of Supporting Information in A Wikipedia Infobox.

5. Experimental Analysis

5.1. Experimental Settings

Background Knowledge graphs. For evaluation on the Synthetic and Real-World datasets, we used English DBpedia as a background knowledge graph, and for evaluation on our dataset, we used K-Box as a background knowledge graph for our and competing approaches.

Evaluation metric. We used the area under the Receiver Operating Characteristic curve (AUC-ROC) as an evaluation metric, which is widely used in the recent fact checking studies [16, 17, 20]. The AUC-ROC expresses the probability that a true fact triple receives higher truth score than a false one.

Parameter settings. We set `maxPathLen`=3, α=0.1, β=0.9, and γ=0.25 as the parameters in our proposed approach for all the evaluations in this paper. The parameter values are chosen based on several experiments, as shown in Fig. 8 and 9 in which x-axis represents different parameter values and y-axis represents AUC-ROC performance scores. In Fig 8, the settings, α=0.1, β=0.9, show the stable performance in the Synthetic and Real-World datasets. In Fig. 9, the setting, γ=0.25, shows the best performance in both the datasets.

Competing approaches. We compare our approach proposed in this paper to the state-of-the-art rule-based approaches: (1) KStream, (2) KLinker, (3) COPPAL, (4) RUDIK, and (5) PredPath. The differences between the approaches are summarized in Table 4 in which U denotes unsupervised learning, S denotes supervised learning, P and N denote positive and negative rules each, W1 denotes a weight measure using only correct examples, W2 denotes a weight measure using both correct and counter examples, W2-M is the marginal weight proposed in RUDIK, and W2-C is the counter-weight proposed in this paper.

KStream, KLinker, and COPPAL are unsupervised approaches that use only positive rules in a truth scoring task, and weight their positive rules using only correct examples (i.e., a set of true fact triples). RUDIK is an unsupervised approach that uses only negative rules in a truth scoring task, and weights its negative rules using both correct and counter examples (i.e., a set of true and false fact triples). PredPath is a supervised approach that uses both positive and negative rules in a truth scoring task, and weights its positive and negative rules using only correct examples. For example, in the case of weighting positive rules, PredPath uses only true fact triples as correct examples. In the case of weighting negative rules, PredPath uses only false fact triples as correct examples.

For KStream, KLinker, and PredPath, we used the implementation published by their authors through the GitHub repository[9]. We re-implemented RUDIK based on the paper published by Ortona et al. [18]. For all the competing approaches, we used the default parameter settings provided by their authors.

5.2. Experimental Results

Table 5 and 6 show the comparison results in the Synthetic and Real-World datasets each.

– The positive rule-based approaches, KStream, KLinker, and COPPAL, show the better performance by up to 32.01% in the Real-World Dataset than the negative rule-based approach, RUDIK. This result indicates that positive rules are more effective than the negative rules in a truth scoring task. The reason is that positive rules are generated and weighted using true fact triples in a

[9] https://github.com/shiralkarprashant/knowledgestream/

Statistics

Label	Count	Ratio
True	290	16.49%
False	1,469	83.51%
Total	1,759	100%

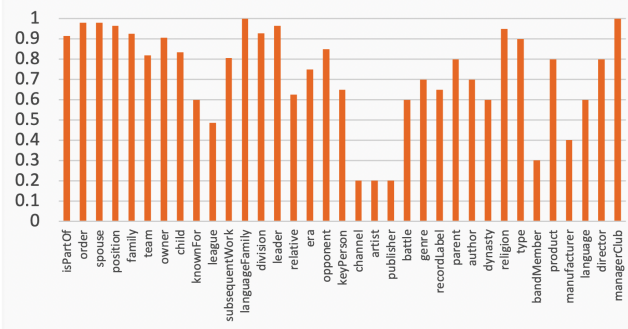


Fig. 7. Statistics of The Constructed Dataset.

knowledge graph, which are relatively more accurate than false fact triples automatically generated by negative sampling methods.

- The supervised positive and negative rules-based approach, PredPath, shows better performance by up to 17.34% in the Synthetic dataset than the unsupervised positive rule-based approaches. Obviously, the supervised approach is better than the unsupervised approach, but the point is that using both positive and negative rules in a truth scoring task is more effective than using only single type of rules.
- Our unsupervised approach outperforms the supervised approach, PredPath, by up to 5.57% in the Real-World dataset. In contrast, in the Synthetic dataset, the difference in performance (1.47%) is not as large in the Real-World dataset (5.57%). The reason is that the Synthetic dataset is easier to verify test cases than the Real-World dataset since 77.26% of the true-labeled fact triples in the Synthetic dataset are already contained in the background knowledge graph. In an easy dataset, less performing approaches can get a certain score without much effort, the difference in performance between the approaches is not huge than a difficult dataset.

Table 7 and Fig. 10 show the comparison results in our dataset. The order of performance in our dataset is similar with the order in the Synthetic and Real-World datasets.

- The positive rule-based approaches show the better performance by up to 11.15% than the negative rule-based approach.
- The supervised positive and negative rules-based approach shows the better performance by up to 13.71% than the unsupervised positive rule-based approaches.
- Our unsupervised approach outperforms the supervised approach by up to 3.72%.

The performance difference between our unsupervised approach and the supervised approach, PredPath, is shown separately in Fig. 11. The main difference between our approach and PredPath is a rule weighting measure. PredPath weights a given rule using only correct examples (i.e., statistical measure-based rule weighting), while our approach weights a given rule using both correct and counter examples (i.e., logical validity-based rule weighting). According to our ablation study, using both correct and counter examples in a rule weighting task is more effective by an average of 7.46% in a truth scoring task than using only correct examples, which supports the huge difference in the truth scoring performance between our approach and PredPath.

The performance difference between our approach and RUDIK is shown separately in Fig. 12. In this paper, we reimplemented and improved RUDIK to check logical consistency since RUDIK is able to learn weighted logical rules that are required to check logical consistency. The main difference of our approach from RUDIK is that our approach uses an ensemble of positive and negative rules in a truth scoring task, while RUDIK uses only negative rules, which makes the improvements in a truth scoring task by 30.53% on average, as shown in Fig. 12. According to our ablation study, using an ensemble of positive and negative rules in a truth scoring task is more effective by up

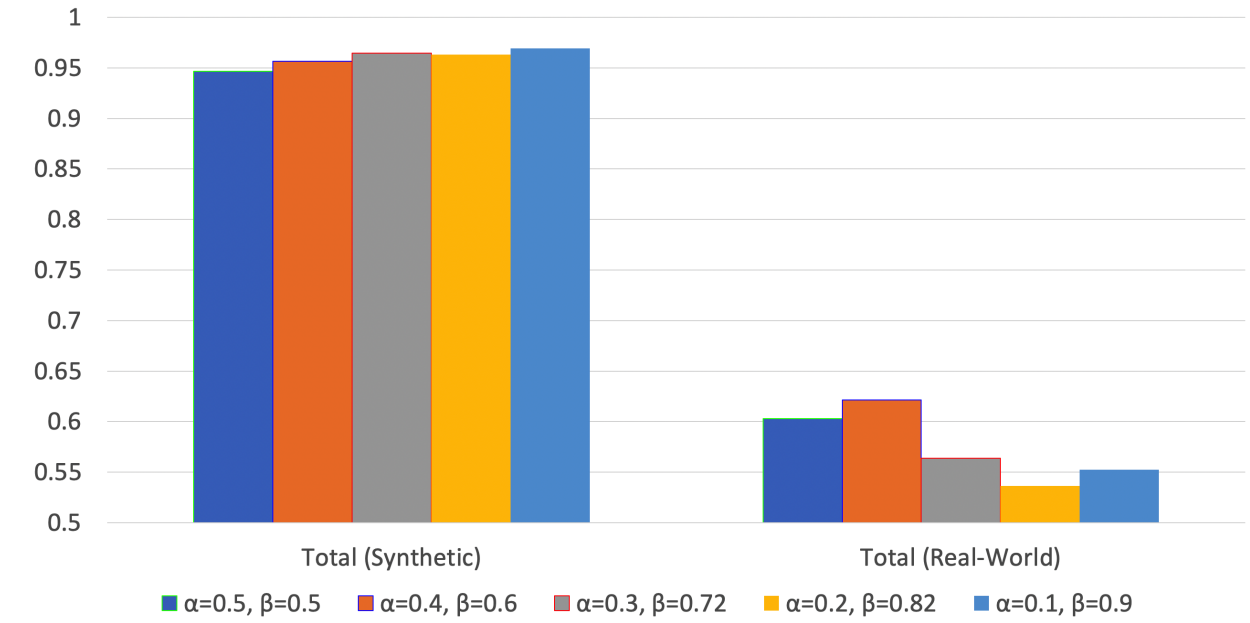


Fig. 8. Performance Change from Different α and β.

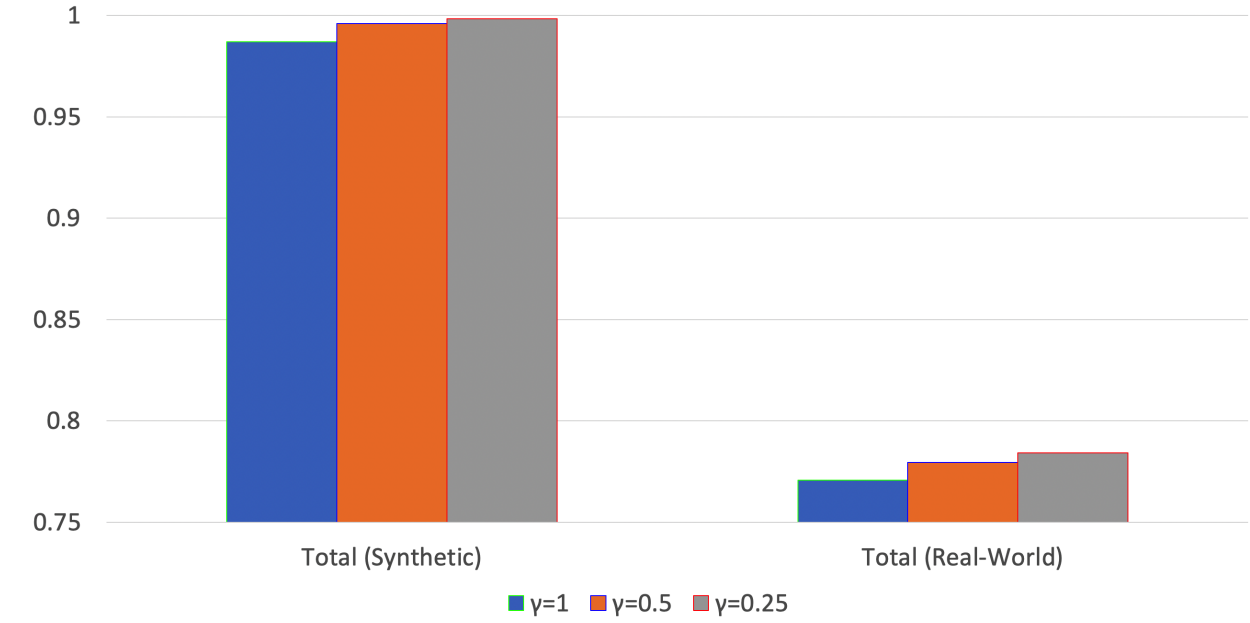


Fig. 9. Performance Change from Different γ.

to 25.12% than using only single type of rules, which supports the huge difference in the truth scoring performance between our approach and RUDIK.

5.3. Ablation Study

To see how much each of the proposed approaches in each sub-task in this paper contributes to the overall performance in a truth scoring task, we conducted an ablation study, as shown in Table 8.

The D-LCWA-based negative sampling method is more effective by up to 5.09% on average than the LCWA- and E-LCWA-based negative sampling methods in an environment where both functional and non-functional predicates are present. The reason of the outperformance is because of the fact that the D-LCWA-based negative sampling method does not use directly adjacent entities that are probable to generate false-labeled true fact triples for non-functional predicates. This result indicates that reducing the false-labeled true fact triples wrongly generated is important to improve the overall performance for non-functional predicates in a truth scoring task.

In a rule weighting task, the weight measure, W2-C, using both correct and counter examples is more effective by up to 7.46% on average than the weight measure, W1, using only correct examples. This result indicates that using counter examples in a rule weighting task has a huge effect on the performance of a truth scoring task, and explains that the reason for the outperformance of our unsupervised approach over the supervised approach, PredPath. W2 and W2-M is the weight measures proposed by Ortona et al. [18], and our weight measure, W2-C, made by improving W2 shows better results by an average of 0.66% and 7.85% each.

In a truth scoring task, using positive and negative rules (P & N) is slightly improved by an average of 0.37% compared to using only positive rules (P), and is hugely improved by an average of 25.12% compared to using only negative rules (N). The point is that using both positive and negative rules shows the best performance on all the three datasets, and this indicates that positive and negative rules are complement each other to solve other types of test cases in a fact checking problem.

5.4. Data Examples

An truth score of a given true triple calculated by our approach, and the most valid positive rule used to calculate a truth score is shown in Table 9. A truth score of the given true triple (Into the New World, artist, Girl's Generation) is calculated as the high score 0.9982 since it is logically consistent with the triple (Girl's Generation, musicalBand, Into the New World) in a knowledge graph, and the consistency is checked by the positive rule, (x, artist, y) ← (y, musicalBand, x). In contrast, a truth score of the given true triple (Annapurna Massif, isPartOf, Himalayas) is calculated as the relatively low score 0.5 since there is no path less than or equal to the length maxPathLen between Annapurna Massif and Himalayas in a knowledge graph, which is a limitation of the proposed approach.

An truth score of a given false triple calculated by our approach, and the most valid negative rule used to calculate a truth score is shown in Table 10. A truth score of the given false triple (Shin Kyungsook, spouse, Tereza Boučková) is calculated as the low score 0.316 since it is logically contradictory to the path (Shin Kyungsook, spouse, Nam Jinwoo) ∧ (Nam Jinwoo, ≠, Tereza Boučková) in a knowledge graph, and the contradiction is checked by the negative rule, ¬(x, spouse, y) ← (x, spouse, z) ∧ (z, ≠, y).

According to our analysis, we found that the negative rules containing the not equal operator, ≠, tend to be assigned a strong weight score. For example, averagely 44.64% of Top-100 negative rules ranked by the strength of their weight scores contain the not equal operator, as shown in Table 11. In addition to the negative rules with the not equal operator, there is the negative rules containing the size comparison operators; e.g., ¬(x, spouse, y) ← (x, birthDate, z) ∧ (z, >, w) ∧ (y, deathDate, w) which means that x and y could not be a spouse relationship if y dies before x is born. As an example of the negative rules not containing both not equal and size comparison operators, there is the negative rule ¬(x, spouse, y) ← (x, country, z) ∧ (w, country, z) ∧ (y, spouse, w) which means that x and y cannot be a spouse relationship since w, not x, is already a spouse of y; This rule does not contain the not equal operator, but has a similar meaning to it. As a result of examining the data examples, the negative rules containing the not equal operator or having a similar meaning to the not equal operator tend to be assigned a strong weight score.

Table 4
Differences between Ours and Competing Approaches

	Competing Approaches					Ours
	KStream	KLinker	COPPAL	RUDIK	PredPath	
Learning Type	U	U	U	U	S	U
Rule Type	P	P	P	N	P & N	P & N
Negative Sampling	-	-	-	E-LCWA	Human	D-LCWA
Rule Weighting	W1	W1	W1	W2-M	W1	W2-C

Table 5
Evaluation Results in The Synthetic Dataset

Approach	Predicates in the Synthetic dataset								Total
	battle	birthPlace	capital	director	keyPerson	spouse	team	vicePresident	
KStream	0.9765	0.0469	0.9979	0.9796	0.7975	0.9953	0.9639	0.9954	0.7717
KLinker	0.9684	0.0269	0.9936	0.9683	0.7956	0.9759	0.9706	0.9868	0.8002
COPPAL	0.927	0.1768	0.9906	0.7594	0.8312	0.9805	0.8826	0.8307	0.833
RUDIK	0.411	0.7108	0.3958	0.5542	0.5404	0.9203	0.4992	0.6006	0.5464
PredPath	0.9401	0.9545	**1.0**	0.9808	0.8714	**1.0**	0.9535	**1.0**	0.9451
Ours	**1.0**	**1.0**	0.9807	**0.9977**	**0.9215**	**1.0**	**0.9746**	0.9985	**0.9598**

Table 6
Evaluation Results in The Real-World Dataset

Approach	Predicates in the Real-World dataset						Total
	almaMater	birthPlace	deathPlace	education	nationality	profession	
KStream	0.7885	0.7414	0.7859	0.7454	0.938	0.9474	0.769
KLinker	0.8064	0.8315	0.8135	0.7734	0.9673	0.9281	0.785
COPPAL	0.6502	0.7292	0.7088	0.5011	0.9716	0.8553	0.6494
RUDIK	0.459	0.5167	0.5358	0.4891	0.4686	0.5	0.4649
PredPath	0.7242	0.7943	0.7632	**0.8335**	**1.0**	0.8227	0.7374
Ours	**0.8071**	**0.8715**	**0.8332**	0.7532	**1.0**	**1.0**	**0.7931**

Table 7
Total Performance Scores in Our Dataset

	KStream	KLinker	COPPAL	RUDIK	PredPath	Ours-dben	Ours-kbox
Total	0.6372	0.6166	0.5864	0.5257	0.6628	0.6884	**0.7**

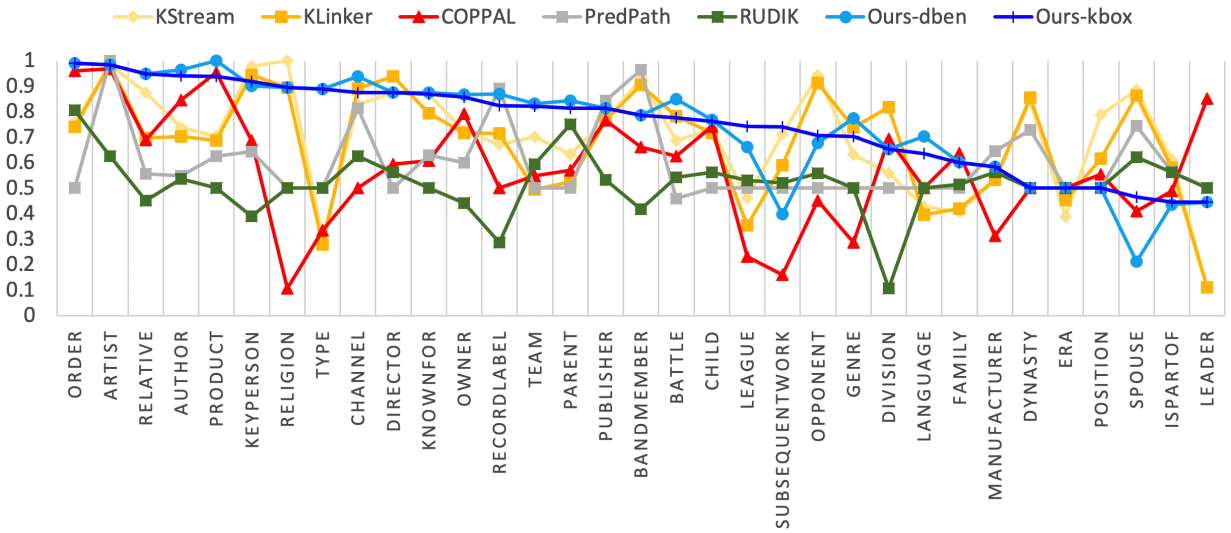
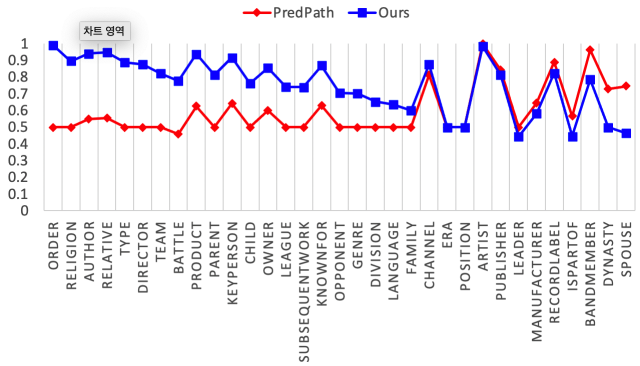
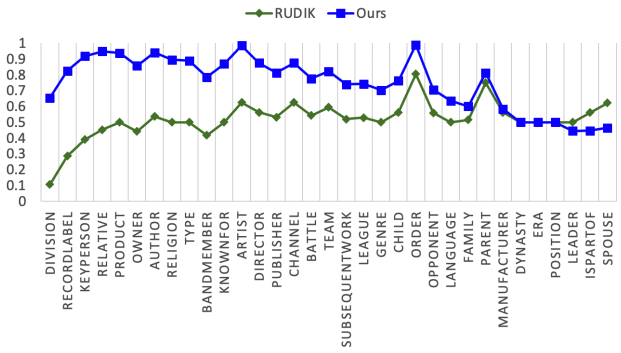


Fig. 10. Detailed Performance Scores in Our Dataset.



Dataset	Supervised Statistical	Unsupervised Logical
	PredPath	Ours
Ours	0.6628	**0.7** (↑3.76%)
Synthetic	0.9451	**0.9598** (↑1.47%)
Real-World	0.7374	**0.7931** (↑5.57%)

Fig. 11. Comparison with The Supervised Approach, PredPath.



Dataset	Negative	Positive & Negative
	RUDIK	Ours
Ours	0.5257	**0.7** (↑17.43%)
Synthetic	0.5464	**0.9598** (↑41.34%)
Real-World	0.4649	**0.7931** (↑32.82%)

Fig. 12. Comparison with The Weighted Logical Rule-based Approach, RUDIK.

Table 8
Ablation Study Results

Sub-Task	Approach	Truth Scoring Performance			Average
		Synthetic	Real-World	Ours	
Negative Sampling	LCWA	0.9131	**0.8214**	0.6831	0.8059
	E-LCWA	0.8988	0.7891	0.6123	0.7667
	D-LCWA	**0.9598**	0.7931	**0.7**	**0.8176**
Rule Weighting	W1	0.9115	0.7408	0.5766	0.743
	W2	0.9464	**0.7958**	0.6908	0.811
	W2-M	0.9423	0.5897	0.6853	0.7391
	W2-C	**0.9598**	0.7931	**0.7**	**0.8176**
Truth Scoring	P	0.9597	0.7838	0.6983	0.8139
	N	0.8042	0.3665	0.5284	0.5664
	P & N	**0.9598**	**0.7931**	**0.7**	**0.8176**

6. Conclusion

In this paper, we presented a weighted logical positive and negative rules-based approach to check logical consistency between fact triples to solve a truth scoring task in a knowledge graph.

Our approach is based on 1) an unsupervised ensemble of the most valid positive and negative rules that 2) are generated using true fact triples in a knowledge graph and false fact triples generated by the proposed D-LCWA-based negative sampling method, and 3) the generated rules are weighted according to their logical validity by the proposed rule weighting measure improved over the approach proposed by Ortona et al. [18].

To generate false fact triples for both functional and non-functional predicates, the proposed D-LCWA-based negative sampling method uses less adjacent entities, and it shows the better performance by an average of 5.09% in an environment that both functional and non-functional predicates are present than the E-LCWA-based negative sampling method [18]. The experimental result indicates that using less adjacent entities in a negative sampling task for non-functional predicates is more beneficial than using directly adjacent entities.

To weight the generated positive and negative rules according to their logical validity, we improved the rule weighting measure proposed by Ortona et al. [18], and the experimental result shows that our rule weighting measure is more effective by up to 7.85% than Ortona's rule weighting measure. The proposed rule weighting measure uses both correct and counter examples, and it shows the better performance by an average of 7.46% than the rule weighting measure using only correct examples. The experimental result indicates that using both correct and counter examples are important to evaluate the logical validity of rules than using only correct examples.

We evaluated our logical consistency-based truth scoring approach and the state-of-the-art competing approaches on our dataset constructed in this paper as well as publicly available two datasets. The experimental results showed that the proposed logical consistency-based approach significantly outperforms the state-of-the-art unsupervised approaches by up to 12.68%, and even outperforms the supervised approach by up to 5.57% in all the datasets. The difference between our approach and the competing approaches is that our approach is based on logical consistency to solve a truth scoring task whereas the competing approaches are not based on logical consistency, but more based on statistical methods. Putting all these together, the experimental results emphasize the importance of logical consistency in a truth scoring task.

References

[1] S. Auer, C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak and Z. Ives, Dbpedia: A nucleus for a web of open data, in: *The semantic web*, Springer, 2007, pp. 722–735.

[2] T. Rebele, F. Suchanek, J. Hoffart, J. Biega, E. Kuzey and G. Weikum, YAGO: A multilingual knowledge base from wikipedia, wordnet, and geonames, in: *International semantic web conference*, Springer, 2016, pp. 177–185.

[3] K. Bollacker, C. Evans, P. Paritosh, T. Sturge and J. Taylor, Freebase: a collaboratively created graph database for structuring human knowledge, in: *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, 2008, pp. 1247–1250.

[4] S. Nam, E.-k. Kim, J. Kim, Y. Jung, K. Han and K.-S. Choi, A korean knowledge extraction system for enriching a kbox, in: *Proceedings of the 27th International Conference on Computational Linguistics: System Demonstrations*, 2018, pp. 20–24.

[5] K.-S. Choi, C. Unger, P. Vossen, J.-D. Kim, N. Kando and A.-C.N. Ngomo, Proceedings of the Open Knowledge Base and Question Answering Workshop (OKBQA 2016), in: *Proceedings of the Open Knowledge Base and Question Answering Workshop (OKBQA 2016)*, 2016.

[6] K.-S. Choi, T. Mitamura, P. Vossen, J.-D. Kim and A.-C.N. Ngomo, SIGIR 2017 workshop on open knowledge base and question answering (OKBQA2017), in: *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2017, pp. 1433–1434.

[7] J.-D. Kim, C. Unger, A.-C.N. Ngomo, A. Freitas, Y.G. Hahm, J. Kim, G.-H. Choi, J.-U. Kim, R. Usbeck, M.-G. Kang et al., OKBQA: an open collaboration framework for development of natural language question-answering over knowledge bases, in: *2017 ISWC Posters and Demonstrations and Industry Tracks, ISWC-P and D-Industry 2017*, Semantic Web Science Association, 2017.

[8] J. Kim, G. Choi and K.-S. Choi, Dedicated Workflow Management for OKBQA Framework, in: *Proceedings of the Open Knowledge Base and Question Answering Workshop (OKBQA 2016)*, 2016, pp. 97–101.

[9] J. Kim, G. Choi, J.-U. Kim, E.-K. Kim and K.-S. Choi, The open framework for developing knowledge base and question answering system, in: *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: System Demonstrations*, 2016, pp. 161–165.

[10] S. Nam, M. Lee, D. Kim, K. Han, K. Kim, S. Yoon, E.-k. Kim and K.-S. Choi, Effective Crowdsourcing of Multiple Tasks for Comprehensive Knowledge Extraction, in: *Proceedings of The 12th Language Resources and Evaluation Conference*, 2020, pp. 212–219.

[11] B. Min, M. Freedman and T. Meltzer, Probabilistic inference for cold start knowledge base population with prior world knowledge, in: *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, 2017, pp. 601–612.

[12] B. Distiawan, G. Weikum, J. Qi and R. Zhang, Neural relation extraction for knowledge base enrichment, in: *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2019, pp. 229–240.

[13] J. Kim, K. Han and K.-S. Choi, KBCNN: A Knowledge Base Completion Model Based On Convolutional Neural Networks, in: *Annual Conference on Human and Language Technology*, Human and Language Technology, 2018, pp. 465–469.

[14] T.D. Nguyen, D.Q. Nguyen, D. Phung et al., A Novel Embedding Model for Knowledge Base Completion Based on Convolutional Neural Network, in: *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, 2018, pp. 327–333.

[15] C. Meilicke, M. Fink, Y. Wang, D. Ruffinelli, R. Gemulla and H. Stuckenschmidt, Fine-grained evaluation of rule-and embedding-based systems for knowledge graph completion, in: *International Semantic Web Conference*, Springer, 2018, pp. 3–20.

[16] P. Shiralkar, A. Flammini, F. Menczer and G.L. Ciampaglia, Finding streams in knowledge graphs to support fact checking, in: *2017 IEEE International Conference on Data Mining (ICDM)*, IEEE, 2017, pp. 859–864.

[17] Z.H. Syed, M. Röder and A.-C.N. Ngomo, Unsupervised discovery of corroborative paths for fact validation, in: *International Semantic Web Conference*, Springer, 2019, pp. 630–646.

[18] S. Ortona, V.V. Meduri and P. Papotti, Robust discovery of positive and negative rules in knowledge bases, in: *2018 IEEE 34th International Conference on Data Engineering (ICDE)*, IEEE, 2018, pp. 1168–1179.

[19] S. Ortona, V.V. Meduri and P. Papotti, Rudik: Rule discovery in knowledge bases, *Proceedings of the VLDB Endowment* **11**(12) (2018), 1946–1949.

[20] B. Shi and T. Weninger, Discriminative predicate path mining for fact checking in knowledge graphs, *Knowledge-based systems* **104** (2016), 123–133.

[21] J. Kim, Y. Jeong and K.-S. Choi, Knowledge Validation by Positive and Negative Rules, in: *Proceedings of the Korea Software Congress (KSC)*, Korea Software Congress, 2019, pp. 452–454.

[22] D. Gerber, D. Esteves, J. Lehmann, L. Bühmann, R. Usbeck, A.-C.N. Ngomo and R. Speck, Defacto—temporal and multilingual deep fact validation, *Journal of Web Semantics* **35** (2015), 85–101.

[23] Z.H. Syed, M. Röder and A.-C. Ngonga Ngomo, Factcheck: Validating rdf triples using textual evidence, in: *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, 2018, pp. 1599–1602.

[24] J. Thorne and A. Vlachos, Automated Fact Checking: Task Formulations, Methods and Future Directions, in: *Proceedings of the 27th International Conference on Computational Linguistics*, 2018, pp. 3346–3359.

[25] A. Bordes, N. Usunier, A. Garcia-Duran, J. Weston and O. Yakhnenko, Translating embeddings for modeling multi-relational data, in: *Neural Information Processing Systems (NIPS)*, 2013, pp. 1–9.

[26] L. Dehaspe and H. Toivonen, Discovery of frequent datalog patterns, *Data Mining and knowledge discovery* **3**(1) (1999), 7–36.

[27] S. Muggleton and L. De Raedt, Inductive logic programming: Theory and methods, *The Journal of Logic Programming* **19** (1994), 629–679.

[28] L.A. Galárraga, C. Teflioudi, K. Hose and F. Suchanek, AMIE: association rule mining under incomplete evidence in ontological knowledge bases, in: *Proceedings of the 22nd international conference on World Wide Web*, 2013, pp. 413–422.

[29] L. Galárraga, C. Teflioudi, K. Hose and F.M. Suchanek, Fast rule mining in ontological knowledge bases with AMIE+, *The VLDB Journal* **24**(6) (2015), 707–730.

[30] X. Dong, E. Gabrilovich, G. Heitz, W. Horn, N. Lao, K. Murphy, T. Strohmann, S. Sun and W. Zhang, Knowledge vault: A web-scale approach to probabilistic knowledge fusion, in: *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2014, pp. 601–610.

Table 9
An Example of True Triple and Its Truth Score calculated by Our Approach

Given True Triple		Truth Score	Used Positive Rule
S	다시 만난 세계 (Into the New World)	0. 9982	(1, artist, 0) ← (1, musicalBand, 0)
P	artist		
O	소녀시대 (Girl's Generation)		
S	원더걸스 (Wonder Girls)	0. 9765	(1, bandMember, 0) ← (1, pastMember, 0)
P	bandMember		
O	안소희 (Ahn So-hee)		
S	견훤(Gyeon Hwon)	0.9679	(1, battle, 0) ← (0, commander, 1)
P	battle		
O	공산 동수 전투 (Gongsan Dongsu Jeontu)		
S	황기영 (Wong Kei-ying)	0.9996	(1, child, 0) ← (0, parent, 1)
P	child		
O	황비홍 (Wong Fei-hung)		
S	후트그투 칸 쿠살라 (Khutughtu Khan Kusala)	0.9848	(1, child, 0) ← (1, spouse, 2) ∧ (0, parent, 2)
P	child		
O	원 영종 (Rinchinbal Khan)		
S	안나푸르나 산 (Annapurna Massif)	0.5	-
P	isPartOf		
O	히말라야 산맥 (Himalayas)		
S	당리역 (Dangni Station)	0.5	-
P	isPartOf		
O	부산 도시철도 (Busan Metro)		

Table 10
An Example of False Triple and Its Truth Score calculated by Our Approach

Given False Triple		Truth Score	Used Negative Rule
S	신경숙 (Shin Kyung-sook)	0.316	$\neg$(1, spouse, 0) $\leftarrow$ (1, spouse, 2) $\wedge$ (2, $\neq$, 0)
P	spouse		
O	테레자 보우치코바 (Tereza Boučková)		
S	대한민국 국군 (Republic of Korea Armed Forces)	0.4671	$\neg$(1, type, 0) $\leftarrow$ (1, $\neq$, 2) $\wedge$ (2, type, 0)
P	type		
O	주한 미군 (United States Forces Korea)		
S	아메리카 (Americas)	0.2755	$\neg$(1, isPartOf, 0) $\leftarrow$ (1, populationTotal, 2) $\wedge$ (2, >, 3) $\wedge$ (0, populationTotal, 3)
P	isPartOf		
O	버지니아 주 (Virginia)		
S	메리 2세 (Mary II)	0.0535	$\neg$(1, spouse, 0) $\leftarrow$ (2, spouse, 1) $\wedge$ (2, $\neq$, 3) $\wedge$ (3, spouse, 0)
P	spouse		
O	조선 선조 (Seonjo of Joseon)		
S	마리아 안토니아 (Marie Antoinette)	0.0364	$\neg$(1, spouse, 0) $\leftarrow$ (1, deathDate, 2) $\wedge$ (3, >, 2) $\wedge$ (0, birthDate, 3)
P	spouse		
O	로베르토 1세 (Robert I, Duke of Parma)		

Table 11
Ratio of Negative Rules with Not Equal Operator

Predicate in Head Part of Negative Rule	Ratio
bandMember	0.85
child	0.84
family	0.8
order	0.77
leader	0.72
relative	0.7
opponent	0.65
recordLabel	0.63
language	0.63
division	0.6
product	0.57
knownFor	0.52
isPartOf	0.49
director	0.47
publisher	0.44
artist	0.4
keyPerson	0.39
languageFamily	0.39
spouse	0.38
manufacturer	0.37
parent	0.37
managerClub	0.36
type	0.34
channel	0.33
team	0.29
author	0.25
genre	0.23
religion	0.2
subsequentWork	0.2
owner	0.17
league	0.14
battle	0.13
position	0.11
Average	0.4464