Article

Enriching Knowledge Base by Parse Tree Pattern and Semantic Filter

Hee-Geun Yoon ¹, Seyoung Park ¹ and Seong-Bae Park ^{2,*}

¹ School of Computer Science and Engineering, Kyungpook National University, 80 Daehak-ro, Bukgu, Daegu 41566, Korea; hgyoon@knu.ac.kr (H.-G.Y.); seyoung@knu.ac.kr (S.P.)

² Department of Computer Science and Engineering, Kyung Hee University, 1732 Deogyong-daero, Yongin-si 17104, Gyeonggi-do, Korea

* Correspondence: sbpark71@khu.ac.kr

Received: 4 August 2020; Accepted: 4 September 2020; Published: 7 September 2020

Abstract: This paper proposes a simple knowledge base enrichment based on parse tree patterns with a semantic filter. Parse tree patterns are superior to lexical patterns used commonly in many previous studies in that they can manage long distance dependencies among words. In addition, the proposed semantic filter, which is a combination of WordNet-based similarity and word embedding similarity, removes parse tree patterns that are semantically irrelevant to the meaning of a target relation. According to our experiments using the DBpedia ontology and Wikipedia corpus, the average accuracy of the top 100 parse tree patterns for ten relations is 68%, which is 16% higher than that of lexical patterns, and the average accuracy of the newly extracted triples is 60.1%. These results prove that the proposed method produces more relevant patterns for the relations of seed knowledge, and thus more accurate triples are generated by the patterns.

Keywords: knowledge enriching; parse tree pattern; semantic filter; word embedding; semantic relevance

1. Introduction

The World Wide Web contains abundant knowledge by virtue of the contributions of its great number of users and the knowledge is being utilized in diverse fields. Since ordinary users of the Web use, in general, a natural language as their major representation for generating and acquiring knowledge, unstructured texts constitute a huge proportion of the Web. Although human beings regard unstructured texts naturally, such texts do not allow machines to process or understand the knowledge contained within them. Therefore, these unstructured texts should be transformed into a structural representation to allow them to be machine-processed.

The objective of knowledge base enrichment is to bootstrap a small seed knowledge base to a large one. In general, a knowledge base consists of triples of a subject, an object and their relation. Existing knowledge bases are not perfect in two respects of relations and triples (instances). Note that even a massive knowledge base like DBpedia, freebase or YAGO is not perfect for describing all relations between entities in the real world. However, this problem is often solved by restricting target applications or knowledge domains [1,2]. Another problem is lack of triples. Although existing knowledge bases contain a massive amount of triples, they are still far from perfect compared to infinite real-world facts. This problem can be solved only by creating triples infinitely. Especially, according to the work by Paulheim [3], the cost of making a triple manually is 15 to 250 times more expensive than that of automatic method. Thus, it is very important to generate triples automatically.

As mentioned above, a knowledge base uses a triple representation for expressing facts, but new knowledge usually comes from unstructured texts written in a natural language. Thus, knowledge enrichment aims at extracting as many entity pairs for a specific relation from

unstructured texts as possible. From this point of view, pattern-based knowledge enrichment is one of the most popular methods among various implementations of knowledge enrichment. Its popularity comes from the fact that it can manage diverse types of relations and the patterns can be interpreted with ease. In pattern-based knowledge enrichment where a relation and an entity pair connected by the relation are given as a seed knowledge, it is assumed that a sentence that mentions the seed entity pair contains a lexical expression for the relation and this expression becomes a pattern for extracting new knowledge for the relation. Since the quality of the newly extracted knowledge is strongly influenced by that of the patterns, it is important to generate high-quality patterns.

The quality of patterns depends primarily on the method used to extract the tokens in a sentence and to measure the confidence of pattern candidates. Many previous studies such as NELL [4], ReVerb [5], and BOA [6] adopt lexical sequence information to generate patterns [7,8]. That is, when a seed knowledge is expressed as a triple of two entities and their relation, an intervening lexical sequence between the two entities in a sentence becomes a pattern candidate. Such lexical patterns have been reported to show reasonable performances in many knowledge enriching systems [4,6]. However, they have obvious limitations that (i) they fail in detecting long distance dependencies among words in a sentence, and (ii) a lexical sequence does not always deliver the correct meaning of a relation.

Assume that a sentence “*Eve is a daughter of Selene and Michael.*” is given. A simple lexical pattern generator like BOA harvests from this sentence the patterns shown in Table 1 by extracting a lexical sequence between two entities. The first pattern is for a relation *childOf*, and is adequate for expressing the meaning of a parent-child relation. Thus, it can be used to extract new triples for *childOf* from other sentences. However, the second pattern, “{arg1} and {arg2}” fails in delivering the meaning of a relation *spouseOf*. In order to deliver the correct meaning of *spouseOf*, a pattern “a daughter of {arg1} and {arg2}” should be made. Since the phrase “a daughter of” is located out of ‘Selene’ and ‘Michael’, such a pattern can not be generated from the sentence. Therefore, a more effective representation for patterns is needed to express dependencies of the words that are not located within entities.

Table 1. Example patterns generated by a lexical pattern generator.

Relation	Pattern
<i>childOf</i>	{arg1} is a daughter of {arg2}
<i>spouseOf</i>	{arg1} and {arg2}

An entity pair can have more than two relations in general. Thus, a sentence that mentions both entities in a seed triple can express other relations that are different from the relation of the seed triple. Then, the patterns extracted from such sentences become useless in gathering new knowledge for the relation of the seed triple. For instance, assume that a triple $\langle \textit{Eve}, \textit{workFor}, \textit{Selene} \rangle$ is given as a seed knowledge. Since only entities are considered in generating patterns, “{arg1} is a daughter of {arg2}” from the sentence “*Eve is a daughter of Selene and Michael.*” becomes a pattern for a relation *workFor*, while the pattern does not deliver the meaning of *workFor* at all. Therefore, it is important for generating high-quality patterns to filter out the pattern candidates that do not deliver the meaning of the relation in a seed triple.

One possible solution to this problem is to define a confidence of a pattern candidate according to the relatedness between a pattern candidate and a target relation. Statistical information such as frequency of pattern candidates or co-occurrence information of pattern candidates and some predefined features has been commonly used as a pattern confidence in the previous work [6,9]. However, such statistics-based confidence does not reflect semantic relatedness between a pattern and a target relation directly. That is, even when two entities co-occur very frequently to express the meaning of a relation, there could be also many cases in which the entities have other relations.

Therefore, in order to determine if a pattern is expressing the meaning of a relation semantically correctly, the semantic relatedness between a pattern and a relation should be investigated.

In this paper, we propose a novel but simple system for bootstrapping a knowledge base expressed in triples from a large volume of unstructured documents. Especially, we show that the dependencies between entities and semantic information can improve the performance over the previous approaches without much effort. For overcoming the limitations of lexical sequence patterns, the system expresses a pattern as a parse tree, not as a lexical sequence. Since a parse tree of a sentence presents deep linguistic analysis of the sentence and expresses long distance dependencies easily, the use of parse tree patterns results in higher performance in knowledge enrichment than lexical sequences. In addition, the deployment of a semantic confidence for parse tree patterns allows irrelevant pattern candidates to be filtered out.

The semantic confidence between a pattern and a relation in a seed knowledge is defined as an average similarity between the words of the pattern and those of the relation. Among various similarity measurements, we adopt two common semantic similarity measurements: a WordNet-based similarity and a word embedding similarity. Generally, WordNet similarity shows plausible results but sometimes it suffers from out-of-vocabulary (OOV) problem [10]. Since patterns can contain many words that are not listed in WordNet, the similarity is supplemented by word similarity in a word embedding space. Thus, the final word similarity is the combination of similarity by WordNet and that in a word embedding space. The final the semantic confidence between a pattern and a relation in a seed knowledge is defined as an average similarity between the words of the pattern and those of the relation.

2. Related Work

2.1. Knowledge Base Enrichment

A number of previous studies addressed the enrichment of knowledge bases [11–14]. Among the methods proposed in these studies, relation extraction is one of the most popular methods [4,6], because natural language is a major representation of knowledge on the Web. A lexical sequence of words as a pattern and the pattern frequency as a filtering method are commonly used in relation extraction [7]. Carlson et al. proposed a system called NELL (Never-Ending Language Learner) which performs learning by reading [4]. NELL enriches its own seed knowledge base with various knowledge extraction modules. Among the modules, CPL (Coupled Pattern Learner) learns contextual patterns from unstructured texts. When a sentence is given in which two entities of a seed knowledge exist, CPL treats an intervening sequence of words between the entities as a pattern candidate. In order to choose confident patterns from pattern candidates, it utilizes some statistical information and coupling constraints between relations or other knowledge extraction modules. BOA (BOotstrapping linked datA) proposed by Gerber et al. [6] extracts pattern candidates in a similar way to NELL. BOA measures the confidence of a pattern candidate using three statistical features of support, typicality, and specificity. Both NELL and BOA reported plausible performance, but both suffer from the weakness of lexical patterns as in Table 1.

Some researchers tried to improve the performance of relation extraction with deep NLP techniques [9]. Fei et al. presented WOE (Wikipedia-based Open Extractor), which uses dependency relation of words in a sentence to improve precision of patterns from long sentences [15]. They defined the shortest dependency path of a parse tree as a pattern. Then, they filtered out irrelevant patterns with a classification function based on pattern frequency. As a result, they achieved improved performance over lexical patterns. However, even the studies that adopt deep NLP analyses miss some valuable information needed to determine the correctness of patterns since they depend solely on frequency information of patterns. In fact, the correctness of a pattern should be determined by semantic relatedness between the pattern and the target relation, not by the frequency of the pattern. The frequency reflects just a little of the semantic relatedness indirectly.

Recently, a neural network became a notable approach to enriching knowledge bases. A neural network provides a uniform framework for end-to-end modeling or joint modeling. This helps reduce the error propagation caused by the combination of several subtasks. Xu et al. presented a Chinese encyclopedia knowledge base framework for extracting and verifying new facts [16]. In this work, relation extraction is formulated as a slot filling task that determines the values for entities and their attributes from a text. They adopted a BiLSTM [17] and an attention layer to construct their relation extractor. Trisedya et al. proposed an end-to-end model for extracting and canonicalizing triples [18]. They reduced error propagation by jointly learning two tasks of extracting triples from sentences and mapping them into an existing knowledge base. Li et al. tried to extract biomedical entities and their relations [14]. Their model was trained to extract biomedical entities as well as their relations simultaneously. Lin et al. [19], Wu et al. [20], Liu et al. [21] and Ye et al. [22] have used an extended version of CNN, so-called PCNN (piecewise CNN), for learning features automatically. Cao et al. [23], Ji et al. [24] and Xinsong et al. [25] also adopted a neural network-based approach. Neural network models show, in general, superior performance to other methods in knowledge base enrichment. However, it is difficult to interpret the rationale behind new triples from neural network models. Thus, while pattern-based methods can be adjusted to resolve their errors, revision of neural network models is usually very difficult. In addition, huge amounts of data are required for training a massive number of parameters of neural networks.

2.2. Semantic Similarity of Words

Measuring semantic relatedness between words has been studied for a long time [26]. The studies can be divided into two types: knowledge-based methods and corpus-based methods. Knowledge-based methods utilize well-organized external knowledge like WordNet [27]. However, they all suffer from the out-of-vocabulary (OOV) problem. In order to overcome this problem, corpus-based methods were introduced which solve the OOV problem by measuring the relatedness through a large volume of documents like World Wide Web [28,29]. Huang et al. proposed a neural network language model that represents every word as a vector [30]. The neural network model is trained to capture the semantics of words with both local and global word contexts. On the other hand, Mikolov et al. proposed a word embedding method that maps words into high-dimensional vectors [31]. They showed that semantic word relation can be identified well with the embedded vectors.

In general, corpus-based methods show plausible performance in finding synonym relations [32]. However, the different kinds of contexts produce noticeably different embeddings, and induce different word similarities [33]. On the other hand, knowledge-based methods can provide consistency for all kinds of relations, since the relations in WordNet are defined clearly. Thus, in order to obtain more reliable semantic relatedness between words, both knowledge-based and corpus-based methods should be used together.

3. Overall Structure of Knowledge Enrichment

Figure 1 depicts the overall structure of the proposed knowledge enriching system. For each relation r in a seed knowledge base, we first generate a set of patterns $P(r)$ for the relation r . When a seed knowledge is given as a triple $\langle e_1, r, e_2 \rangle$ with two entities (e_1 and e_2) and a relation (r), the pattern for the seed knowledge is defined as a subtree of the parse tree of a sentence that contains both e_1 and e_2 . In order to obtain $P(r)$, the sentences that mention e_1 and e_2 at the same time are first chosen. Since our pattern is a parse tree, the chosen sentences are parsed by a natural language parser, and then transformed into parse tree patterns. Then, we exclude the parse tree patterns that do not deliver the meaning of the relation r . After filtering out such irrelevant tree patterns, the remaining become $P(r)$.

Figure 1. Overall structure of the proposed knowledge enrichment.

Once $P(r)$ is prepared, it is used to generate new triples for r from the set of documents. If a sentence in the document set matches a parse tree pattern in $P(r)$, a new triple extracted from the sentence is added into the original seed knowledge base. Since a pattern has a tree structure, all sentences in the document set are also parsed by a natural language parser in advance. A new triple is extracted from a parse tree, when a pattern matches the parse tree exactly. Finally, the new triples are added into the knowledge base.

4. Pattern Generation

4.1. Generation of Pattern Candidates

The first step of knowledge base enrichment is to harvest pattern candidates for each relation in the seed knowledge base. As an expression of patterns, current successful systems such as BOA and NELL utilize the lexical information in a sentence. This lexical pattern consists of intervening words between two entities within a seed triple and is assumed to deliver the meaning of the relation of the seed triple. However, this assumption is often untrue. There could be very many sentences on the web that produce irrelevant patterns.

In order to overcome the problem of lexical patterns, we adopt dependency information within a sentence in harvesting pattern candidates. Dependency information provides the relations between predicates and their arguments in a sentence. As a way to utilize dependency information of a sentence as a pattern, our pattern is represented as a subtree of the parse tree of the sentence.

A natural language parser is required to transform a sentence into a parse tree. We adopt the Stanford dependency parser [34] as a parser. Since prepositions and conjunctions are functional words, it is better to represent them as dependencies rather than as tree nodes. The Stanford dependency parser provides a collapsed representation of parse trees that represents functional words as dependencies. Thus, we use the collapsed representation for parse trees.

Parse tree pattern candidates are generated by Algorithm 1. Assume that a seed triple $\langle e_1, r, e_2 \rangle$ and a sentence s which contains e_1 and e_2 are given as an input. The sentence s is parsed into a parse tree t by a dependency parser. Then, the nodes in the parse tree that correspond to e_1 and e_2 are identified first, and let n_1 and n_2 be the corresponding nodes. Once n_1 and n_2 are identified, the pattern candidate p for the relation r is generated by a function *subtree_extract*. This function returns, from t , the smallest subtree subsumed by n_1, n_2 , and their lowest-common-ancestor, where the lowest-common-ancestor becomes the root node of the subtree. For instance, let us consider the dependency parse tree of a sentence “Eve is a daughter of Selene and Michael.” Figure 2a is the parse tree of the sentence. When a seed triple is $\langle \text{Eve}, \text{childOf}, \text{Selene} \rangle$, n_1 and n_2 are the node of ‘Eve’ and ‘Selene’, respectively. In the parse tree, the lowest-common-ancestor node of n_1 and n_2 is the node of ‘daughter’. Thus, the pattern for the relation *childOf* from this sentence is the smallest subtree

composed by n_1 , n_2 , and their lowest-common-ancestor given in Figure 2b. As another example, assume that a seed triple $\langle \text{Selene}, \text{spouseOf}, \text{Michael} \rangle$ is given. A parse tree in Figure 2c is generated as a pattern from the same sentence. Note that the word ‘daughter’ appears in the pattern, even though it is not located between ‘Selene’ and ‘Michael’ in the sentence.

Algorithm 1: Pattern Candidate Generator

Input :an seed triple $\langle e_1, r, e_2 \rangle$,
a sentence s

Output: a pattern candidate p

$t \leftarrow \text{parser}(s)$

$n_1 \leftarrow t(e_1)$

$n_2 \leftarrow t(e_2)$

$p \leftarrow \text{subtree_extract}(t, n_1, n_2)$

Return p

Figure 2. Example pattern candidates generated from a parse tree.

The pattern candidates generated are not always correct. This is because the pattern candidates are generated by matching only entities. For instance, assume a new sentence “*Eve works for Selene’s company, SCINet.*” The parse tree of this sentence is drawn in Figure 3a. Since both entities of a seed triple $\langle \text{Eve}, \text{childOf}, \text{Selene} \rangle$ match this sentence, a new pattern candidate is generated from the parse tree as shown in Figure 3b. However, this candidate does not express the meaning of *childOf* at all. Therefore, such irrelevant pattern candidates should be removed from the set of patterns for a target relation.

Figure 3. A parse tree of a sentence “Eve works for Selene’s company, SCINet.” and the derived pattern candidate from the parse tree.

4.2. Semantic Similarity as a Semantic Filter

When a pattern candidate p does not match a relation r semantically, this candidate should be discarded. To determine whether p matches r , the semantic similarity between p and r is used. If the similarity of p and r is lower than a predefined threshold, p will be discarded. That is, a pattern candidate is discarded if

$$\text{sim}(p, r) \leq \theta_r, \quad (1)$$

where $\text{sim}(p, r)$ is a semantic similarity and θ_r is a threshold for r .

The similarity $\text{sim}(p, r)$ can be computed easily with WordNet. However, there is a number of words that are not listed in WordNet. The similarity between such words can not be computed using WordNet. Therefore, WordNet-based similarity is supplemented with the similarity in word-embedding space constructed with a great number of independent documents. Then, the similarity $\text{sim}(p, r)$ becomes

$$\text{sim}(p, r) = \frac{\text{sim}_{\text{WN}}(p, r) + \text{sim}_{\text{WE}}(p, r)}{2},$$

where sim_{WN} is a WordNet-based similarity and sim_{WE} is a similarity at a word-embedding space.

4.2.1. WordNet-Based Similarity

Assume that a relation r is expressed with m words of $\mathbf{r} = \{rw_1, \dots, rw_m\}$ and the words in the pattern p except entity words are $\mathbf{p} = \{pw_1, \dots, pw_n\}$. Then, the WordNet-based similarity $\text{sim}_{\text{WN}}(p, r)$ can be computed as an average similarity of all possible combination word pairs of $\mathbf{p}$ and $\mathbf{r}$. That is,

$$\text{sim}_{\text{WN}}(p, r) = \frac{1}{n \cdot m} \sum_{rw \in \mathbf{r}} \sum_{pw \in \mathbf{p}} \text{sim}_{\text{wn}}(pw, rw),$$

where $\text{sim}_{\text{wn}}(pw, rw)$ is a word similarity between two words pw and rw by WordNet.

Among various WordNet-based word similarities, Jiang–Conrath similarity [35] is reported to be best which is defined as

$$\text{sim}_{\text{jc}}(pw, rw) = \frac{1}{2 \cdot \log P(\text{lcs}(pw, rw)) - (\log P(pw) + \log P(rw))},$$

where $lcs(pw, rw)$ is the lowest-common-subsumer of pw and rw on WordNet, and $P(w)$ is the information content of w computed with an independent corpus. Since sim_{jc} is not bounded, we use a bounded Jiang–Conrath similarity as WordNet-based similarity sim_{wn} , where the bounded Jiang–Conrath similarity is given as

$$sim_{wn}(pw, rw) = \frac{sim_{jc}(pw, rw)}{T}.$$

Here, T is the maximum value of Jiang–Conrath similarity.

4.2.2. Similarity at Word Embedding Space

The semantic relatedness between words can be measured by comparing their contexts. In general, the context of a word is defined as its neighbor words, and a distributed word representation represents words as high-dimensional real-valued vectors by encoding the context information. Many previous studies proposed diverse methods for distributed word representation. Among them, neural-network-based word embeddings achieve state-of-the-art performance in word semantics tasks such as word analogy and word compositionality [31]. They use a simple but efficient model so that they can process a huge number of documents to encode context information. Therefore, we apply the word embedding proposed by Mikolov et al. [31] to computing word similarity.

Let a pattern p consist of n words of $\mathbf{p} = \{pw_1, \dots, pw_n\}$ except entity words. Since each word pw_i can be expressed as a vector $\mathbf{pw}_i$ at a word embedding space, the pattern p is expressed as the mean vector of $\mathbf{pw}_i$'s. That is, $\mathbf{pw}$, the average word vector of p is $\mathbf{pw} = \frac{1}{n} \sum_{i=1}^n \mathbf{pw}_i$. In a similar way, when a relation r is expressed with m words of $\mathbf{r} = \{rw_1, \dots, rw_m\}$, each word rw_i is expressed as a vector $\mathbf{rw}_i$ at the word embedding space. Then, the relation r is also expressed as the mean vector of $\mathbf{rw}_i$'s. That is, the relation r is represented by a vector $\mathbf{rw}$ defined as $\mathbf{rw} = \frac{1}{m} \sum_{i=1}^m \mathbf{rw}_i$. Since both p and r are expressed as vectors at the same space, their similarity can be computed using cosine similarity. That is,

$$sim_{WE}(p, r) = \frac{\mathbf{pw} \cdot \mathbf{rw}}{\|\mathbf{pw}\| \|\mathbf{rw}\|}.$$

5. New Knowledge Extraction

Once $P(r)$, a set of patterns for a relation r , is prepared, new triples are extracted from a large set of documents using $P(r)$. When the parse tree of a sentence matches a pattern for r completely, a new triple for r is made from the sentence. Algorithm 2 explains how new triples are made. The algorithm takes, as its input, a sentence s from a document set, a target relation r , and a pattern $p \in P(r)$. For a simple match of trees, a pattern p is transformed into a string representation Str_p by a function *ConvertToString*. This function converts a tree into a long single string by traversing the tree in order. The labels of edges are regarded as nodes since they play an important role in delivering the meaning of a relation. Let us consider the patterns in Figures 2b and 3b for instance. The pattern in Figure 2b is expressed as a string

[Subject] $\leftarrow$ nsubj $\leftarrow$ [daughter] $\rightarrow$ prep-of $\rightarrow$ [Object], while that in Figure 3b becomes [Subject] $\leftarrow$ nsubj $\leftarrow$ [works] $\rightarrow$ prep-for $\rightarrow$ [company] $\rightarrow$ poss $\rightarrow$ [Object].

The sentence s is changed into a parse tree t by a natural language parser, and all entities in s are extracted into E . For each combination (e_s, e_o) of entity pairs in E , the subtree p' of t that subsumes the entity pair is matched with the pattern p . If p' matches p , p' is regarded as a parse tree that delivers the same meaning with p .

For matching p' and p , the nodes corresponding to e_s and e_o in t are first identified as n_1 and n_2 . Then, the subtree p' that subsumes n_1 and n_2 is extracted by the function *subtree_extract* used in Algorithm 1. After that, p' is also transformed into a string representation $Str_{p'}$ by *ConvertToString*.

If Str_p and $Str_{p'}$ are same, a triple $\langle e_s, r, e_o \rangle$ is believed to follow the meaning of the pattern p . Thus, it is added into a knowledge set K as a new triple for the relation r .

Algorithm 2: New Knowledge Extraction

Input : a sentence s , a target relation r , a pattern p

Output: a set of newly added triples K

```

 $Str_p \leftarrow ConvertToStr(p)$ 
 $t \leftarrow parser(s)$ 
 $E \leftarrow getEntities(s)$ 
 $K \leftarrow \{\}$ 
foreach  $e_s \in E$  do
    foreach  $e_o \in E$  do
         $n_1 \leftarrow t(e_s)$ 
         $n_2 \leftarrow t(e_o)$ 
        if  $n_1 \neq nil$  and  $n_2 \neq nil$  then
             $p' \leftarrow subtree\_extract(t, n_1, n_2)$ 
             $Str_{p'} \leftarrow ConvertToStr(p')$ 
            if  $Str_p = Str_{p'}$  then
                 $k \leftarrow \langle e_s, r, e_o \rangle$ 
                 $K = K \cup k$ 
            end
        end
    end
end
Return  $K$ 

```

6. Experiments

To evaluate the proposed method, we perform the experiments with two datasets. The first dataset consists of Wikipedia and DBpedia. The DBpedia ontology is used as a knowledge base and the Wikipedia corpus is used as a corpus for generating patterns and extracting new knowledge triples. For quantitative evaluation, QALD-3 benchmark dataset (Ontology lexicalization task) is adopted, where the dataset consists of 30 predicates which is a subset of DBpedia. The second dataset is NYT (New York Times Corpus) benchmark dataset which is adopted in many previous studies [36].

At the experiment with Wikipedia and DBpedia, recall of patterns and new triples cannot be calculated because there are no gold standard answers on the patterns and new triples in the corpus. Thus, only the accuracy (precision) of patterns and triples is measured. However, in order to show the relatedness between recall and precision indirectly, the accuracy (precision) at K with respect to the ranked triple lists is used. All evaluations are performed manually with two assessors. In every judgment, only the predictions which both assessors determined as true are considered to be correct. On the other hand, at the experiment with NYT, we also present top- K precision, which is evaluated automatically with test data.

The proposed method is evaluated with four experiments. The objective of the first two experiments is to show the effectiveness of our pattern generation step. In the first experiment, the proposed parse tree pattern is compared with the lexical sequence pattern, and the effectiveness of the proposed semantic filter is investigated in the second experiment. New triples extracted by our parse tree patterns are evaluated in the third experiment. In the final experiment, the proposed method is compared with previous studies using another benchmark dataset, NYT.

6.1. Evaluation of Parse Tree Patterns

We show the superiority of a parse tree representation of patterns by comparing it with a lexical representation. For the evaluation of patterns, ten most-frequent relations are selected among 30 relations. The ten relations used are *artist*, *board*, *crosses*, *deathPlace*, *field*, *location*, *publisher*, *religion*, *spouse*, and *team*. Although only one third of DBpedia relations are used, the ten relations can cover most of the pattern candidates. That is, 63,704 unique pattern candidates are generated from 30 relations, but 75% of them are covered by the ten relations.

All the triples of the DBpedia ontology that correspond to the ten predicates are employed as seed triples. In order to generate both kinds of patterns, 100 sentences are randomly sampled from the Wikipedia corpus for each relation. Since one pattern is generated from a sentence, every relation has 100 patterns for parse tree representation and lexical representation, respectively. In order to obtain lexical sequence patterns which are used for previous work such as BOA or OLLIE, we follow only the pattern search stage of BOA. The correctness of each pattern is measured by two human assessors. For each pattern of a relation, the assessors determine if the words in the pattern deliver the meaning of the relation accurately. Finally, only the patterns which both assessors agree with as true are considered to be correct.

Figure 4 shows the comparison result of parse tree and lexical sequence patterns. The X-axis of this figure represents relations, and the Y-axis is the accuracy of patterns. The proposed parse tree patterns show consistently higher accuracy than the lexical sequence patterns for all relations. The average accuracy of parse tree patterns is 68%, while that of lexical sequence patterns is just 52%. The maximum accuracy difference between the two pattern representations is 35% for the relation *publisher*. Since parse trees represent dependency relations between words and thus it can reveal long distance dependencies of non-intervening words, higher accurate patterns are generated by parse trees.

Figure 4. The accuracy comparison of parse tree patterns and lexical sequence patterns.

After investigating all 1000 (=100 patterns · 10 relations) parse tree patterns, it is found that around 34% of words appearing at the patterns are non-intervening words and about 45% patterns contain at least one non-intervening word. The fact that many patterns contain non-intervening words implies that the proposed parse tree pattern represents long distance dependencies between words effectively. For example, consider the following sentence and a triple *(Floating into the Night, artist, Julee Cruise)*.

Most notably he produced and wrote lyrics for Julee Cruise's first two albums, Floating into the Night (1989) and The Voice of Love (1993).

From this sentence, the lexical pattern extracts $\langle first\ two\ album(s) \rangle$ as a pattern, and the pattern contains meaningless words such as *first* and *two*. However, the following parse tree pattern excludes such non-intervening words.

$$[Subject] \leftarrow \text{appos} \leftarrow [album] \rightarrow \text{poss} \rightarrow [Object].$$

6.2. Performance of Semantic Filter

The proposed semantic filter is based on a composite similarity of WordNet-based similarity and word-embedding similarity. Thus, we compare the composite similarity with each base similarity to see the superiority of the semantic filter. In addition, many lexical sequence representations of patterns remove irrelevant patterns based on a pattern frequency. Thus, a frequency-based filter is also compared with the proposed semantic filter.

For each relation, parse tree patterns are generated using all seed triples and the Wikipedia corpus. As a result, 47,390 parse tree patterns are generated. Thus, one relation has 4739 patterns on average. Then, four filters were applied to sort the patterns according to their similarity or frequency. Since it is impractical to investigate the correctness of 47,390 patterns manually, the correctness of top 100 patterns by each filter is checked. Figure 5 shows the average top-K accuracies of the four filters. In this figure, ‘WordNet + Word Embedding’ is the proposed semantic filter, ‘WordNet Only’ and ‘Word Embedding Only’ are two base filters, and ‘Frequency-Based’ is the frequency-based filter used in OLLIE [9]. ‘WordNet + Word Embedding’ outperforms all other filters for all k ’s. In addition, the difference between ‘WordNet + Word Embedding’ and other filters gets larger as k increases. These results imply that the proposed semantic filter preserves high-quality patterns and removes irrelevant patterns effectively.

Among the ten relations, the results for *deathPlace* show the lowest accuracy. As shown in Figure 6a, the accuracies of *deathPlace* are below 50% for all filters. In the knowledge base, the concepts *Person* and *Location* are usually used as the domain and the range of *deathPlace*, respectively. However, they are often used for many other relations such as *birthPlace* and *Nationality*. Thus, even if a number of patterns are generated from sentences with *Person* as a subject and *Location* as an object, many of them are not related at all to *deathPlace*. For instance, a parse tree pattern

$$[Subject] \leftarrow \text{nsubj} \leftarrow [\text{live}] \rightarrow \text{prep-in} \rightarrow [Object].$$

is generated from a sentence “*Kaspar Hauser lived in Ansbach from 1830 to 1833.*” with a seed triple $\langle Kaspar\ Hauser, deathPlace, Ansbach \rangle$. This pattern is highly ranked in our system, but its meaning is “*Subject lives in Object*”. Thus, it does not deliver the meaning of a death location.

When word embedding similarity is compared with WordNet-based similarity, it is more accurate than WordNet-based similarity. As seen in Figure 5, its accuracy is always higher than that of WordNet-based similarity for all k ’s. However, its accuracy is extremely low with the relation *spouse* as seen in Figure 6b. Such extremely low accuracy happens when the similar words of a relation at the word embedding space are not synonyms of the relation. The similar words of *spouse* in WordNet are its synonyms like ‘*wife*’ and ‘*husband*’, but those at the word embedding space are ‘*child*’ and ‘*grandparent*’. Even if ‘*child*’ and ‘*grandparent*’ imply a family relation, they do not correspond to *spouse*. Since the proposed semantic filter uses a combination of WordNet-based similarity and word embedding similarity, the problem of word embedding space is compensated by WordNet-based similarity. Figures 7 and 8 show Top-K accuracies of all relations except *deathPlace* and *spouse*. For most relations, the semantic-based scoring achieves higher performance than the frequency-based scoring.

Figure 5. Average top-K accuracies of semantic and frequency-based filters.

(a) *deathPlace*

(b) *spouse*

Figure 6. Average top-K accuracies for *deathPlace* and *spouse*.

Figure 7. Average top-K accuracies for *artist*, *board*, *crosses* and *field*.

Figure 8. Average top-K accuracies for *location*, *publisher*, *religion* and *team*.

6.3. Evaluation of Newly Extracted Knowledge

In order to investigate whether parse tree patterns and semantic filters produce accurate new triples, the triples extracted by the “parse tree + semantic filter” patterns are compared with those extracted by “lexical + frequency filter,” “lexical + semantic filter,” and “parse tree + frequency filter” patterns. Since the Wikipedia corpus is excessively large, 15 million sentences are randomly sampled from the corpus and new triples are extracted from the sentences.

Table 2 shows the detailed statistics of the number of matched patterns and triples extracted with the patterns. According to this table, the number of matched lexical sequence patterns is 255 and that of parse tree patterns is 713. As a result, the numbers of new triples extracted by lexical sequence patterns and parse tree patterns are 32,113 and 104,311 respectively. Though lexical sequence patterns and parse tree patterns are generated from and applied to an identical dataset, the parse tree patterns extract 72,198 more triples than the lexical sequence patterns, which implies that the coverage of parse tree patterns is much wider than that of lexical sequence patterns.

Table 2. The statistics of the matched patterns and newly extracted triples.

Relation	# Matched Lexical Sequence Patterns (LP)	# Triples from LP	# Matched Parse Tree Patterns (PP)	# Triples from PP
<i>artist</i>	17	1292	54	1489
<i>board</i>	15	376	73	6668
<i>crosses</i>	6	104	28	659
<i>deathPlace</i>	11	2560	124	16,529
<i>field</i>	28	12,532	80	5475
<i>location</i>	31	6074	112	41,903
<i>publisher</i>	83	4881	56	2178
<i>religion</i>	8	382	21	1429
<i>spouse</i>	42	1937	139	21,714
<i>team</i>	14	1975	26	6267
Sum	255	32,113	713	104,311

In the evaluation of new triples, top-100 triples are selected for each relation according to the ranks, and the correctness of the 4000 (=100 triples · 10 relations · 4 pattern types) triples is checked manually by two assessors. As done in previous experiments, only the triples marked as true by both assessors are considered correct. Table 3 summarizes the accuracy of the triples extracted by “parse tree + semantic filter” patterns and those by “lexical + frequency filter,” “lexical + semantic filter,” and “parse tree + frequency filter” patterns. The triples extracted by “parse tree + semantic filter” patterns achieve 60.1% accuracy, while those by “parse tree + frequency filter,” “lexical + semantic filter,” and “lexical + frequency filter” achieve 53.9%, 38.2%, and 32.4% accuracy, respectively. The triples extracted by “parse tree + semantic filter” patterns outperform those by “lexical + frequency filter” patterns by 27.7%. They also outperform triples extracted by the “lexical + semantic filter” and “parse tree + frequency filter” patterns by 21.9% and 6.2%, respectively. These results prove that knowledge enrichment is much improved by using parse tree patterns and the proposed semantic filter.

Most incorrect triples by parse tree patterns come from three relations of *deathPlace*, *field*, and *religion*. The accuracy of new triples goes up to 74.0% without the relations. The reason why *deathPlace* produces many incorrect triples is explained above. For relations of *field* and *religion*, it is found out that a few incorrect patterns which are highly ranked by the semantic filter produce most new triples. Therefore, it is our future work to solve the problems.

After generating all possible pattern tree candidates, the irrelevant candidates are removed using Equation (1). θ_r 's of each relation r used for filtering irrelevant candidates are given in Table 4. On average, 71 patterns of each relation are matched with Wikipedia sentences, but only 37 patterns remain after semantic filtering. And then, among 104,311 triples, triples which are extracted from eliminated patterns are excluded from results. As a result, 12,522 new triples are extracted and added into the seed knowledge.

Table 3. Accuracies of newly extracted triples by parse tree pattern and lexical sequence pattern.

Relation	Lexical + Frequency Filter	Lexical + Semantic Filter	Parse Tree + Frequency Filter	Parse Tree + Semantic Filter
<i>artist</i>	43%	52%	92%	89%
<i>board</i>	60%	58%	89%	80%
<i>crosses</i>	16%	31%	46%	68%
<i>deathPlace</i>	0%	19%	0%	21%
<i>field</i>	42%	30%	8%	39%
<i>location</i>	38%	53%	80%	75%
<i>publisher</i>	46%	49%	57%	75%
<i>religion</i>	4%	10%	38%	23%
<i>spouse</i>	40%	37%	91%	87%
<i>team</i>	35%	43%	38%	44%
Average	32.4%	38.2%	53.9%	60.1%

Table 4. The number of newly extracted triples by parse tree patterns and semantic filter.

Relation	θ_r	#Patterns	#New Triples with Filtering
<i>artist</i>	0.20	36	961
<i>board</i>	0.13	48	3163
<i>crosses</i>	0.14	23	234
<i>deathPlace</i>	0.11	58	3448
<i>field</i>	0.12	20	43
<i>location</i>	0.21	35	731
<i>publisher</i>	0.24	58	1169
<i>religion</i>	0.15	19	382
<i>spouse</i>	0.30	66	1970
<i>team</i>	0.19	9	421
Sum	-	372	12,522

6.4. Comparison with Previous Work

In order to show the plausibility of the proposed method, we perform an additional experiment with a new benchmark dataset, NYT, which is generated with Freebase relations and New York Times corpus [36]. The Freebase entities and relations aligned with the sentences of the corpus in the years 2005–2006. The triples generated by this alignment are regarded as training data, and those aligned with the sentences from 2007 are regarded as test data. The training data contain 570,088 instances with 63,428 unique entities and 53 relations with a special relation ‘NA’ which indicates there is no relation between subject and object entities. The test data contain 172,448 instances with 16,705 entities and 32 relations including ‘NA’. Note that ‘NA’ is adopted to represent negative instances. Thus, the triples with ‘NA’ relation do not deliver any information actually. Without the triples with ‘NA’ relation, there remain 156,664 and 6444 triples in training and test data. Table 5 shows simple statistics on the NYT dataset.

Table 5. Simple Statistics on the New York Times Corpus (NYT) dataset.

	# Instances	# Unique Sentences	# Relations	# Unique Entities	# Triples	# Unique Triples
<i>Train</i>	570,088	368,099	53	63,428	156,664	19,601
<i>Test</i>	172,448	61,707	32	16,705	6444	1950

The proposed method is compared with four variations of PCNN (piecewise convolutional neural network) which have used the NYT dataset for their evaluation [19–22]. These models are listed in Table 6 in which ATT implies the attention method proposed by Lin et al. [19], nc and cond_opt denote the noise converter and the conditional optimal selector by Wu et al. [20], soft-label means the soft-label method by Liu et al. [21], and ATT_RA and BAG_ATT are the relation-aware intra-bag

attention method and inter-bag attention method proposed by Ye et al. [22]. We measure the top- K precision (Precision@ K) where K is 100, 200, and 300.

Table 6. Performance comparison at Precision@ K .

Methods	100	200	300	Mean
PCNN+ATT (Lin et al. [19])	76.2%	73.1%	67.4%	72.2%
PCNN+nc+cond_opt (Wu et al. [20])	85.0%	82.0%	77.0%	81.3%
PCNN-ATT+soft-label (Liu et al. [21])	87.0%	84.5%	77.0%	82.8%
PCNN+ATT_RA+BAG_ATT (Ye et al. [22])	91.8%	84.0%	78.7%	84.8%
Proposed Method	82.3%	78.5%	76.9%	79.2%

Table 6 summarizes the performance comparison on the NYT dataset. According to this table, the proposed method achieves comparable performance to the neural network-based methods. PCNN+ATT_RA+BAG_ATT shows the highest mean precision of 84.8% while the proposed method achieves 79.2%. Thus, the difference between them is just 5.6%. The proposed method, however, is consistent against the change of K . All neural network-based methods show about a 10% difference between $K = 100$ and $K = 300$. On the other hand, the difference of the proposed method is just 5.4%, which implies that the proposed pattern generation and scoring approach is plausible for this task. In addition, the patterns generated by the proposed method can be interpreted with ease, and thus the pattern errors can be fixed without huge effort.

7. Conclusions and Future Work

The generation of accurate patterns is a key factor of pattern-based knowledge enrichment. In this paper, a parse tree pattern and semantic filter to remove irrelevant pattern candidates were proposed. The benefit of using parse tree representation for patterns is that long distance dependencies of words are expressed well by a parse tree. Thus, parse tree patterns contain many words that are not located between two entity words. In addition, the benefit of the semantic filter is that it finds irrelevant patterns more accurately than does a frequency-based filter because it reflects the meaning of relations directly.

The benefits of our system were empirically verified through experiments using the DBpedia ontology and Wikipedia corpus. The proposed system achieved 68% accuracy for pattern generation, which is 16% higher than that of lexical patterns. In addition, the knowledge extracted newly by parse tree patterns showed 60.1% accuracy, which is 27.7% higher than the accuracy of that extracted by lexical patterns and statistical scoring. Although in comparison with previous neural network based methods the proposed method could not achieve state-of-the-art performance, it showed excellent performance considering the simplicity of the model. Especially, it proves that our proposed approach is suitable for the knowledge enrichment task with robustness. These results imply that the proposed knowledge enrichment method populates new knowledge effectively.

As our future work, we will figure out a more suitable similarity metric between a pattern and a relation. We have shown through several experiments that WordNet and word embedding are appropriate for this task without further huge efforts. Nevertheless, there is still some room for performance improvement. Thus, we will explore a new semantic similarity to capture well the relatedness between a relation and a pattern in the future. Another weakness of the proposed method is that it cannot handle the unseen relations. It is critical to discover unseen relations in order to make a knowledge base as perfect as possible. Recently, translation-based knowledge base embeddings showed some potential for finding absent relations [37,38]. Therefore, in the future, we will investigate a way to discover absent relations and enrich a knowledge base by applying them to the knowledge base.

Author Contributions: Conceptualization, H.-G.Y.; Data curation, H.-G.Y.; Funding acquisition, S.P.; Methodology, H.-G.Y.; Project administration, S.P. and S.-B.P.; Supervision, S.-B.P.; Validation, S.P. and S.-B.P.; Visualization, H.-G.Y.; Writing—original draft, H.-G.Y.; Writing—review & editing, S.P. and S.-B.P. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Acknowledgments: This work was supported by Institute for Information & communications Technology Promotion (IITP) grant funded by the Korea government (MSIT) (No. 2016-0-0145, Smart Summary Report Generation from Big Data Related to a Topic).

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

References

1. Gong, F.; Chen, Y.; Wang, H.; Lu, H. On building a diabetes centric knowledge base via mining the web. *BMC Med. Inform. Decis. Mak.* **2019**, *19*, 49. [[CrossRef](#)] [[PubMed](#)]
2. Bada, M.; Stevens, R.; Goble, C.; Gil, Y.; Ashburner, M.; Blake, J.; Cherry, J.; Harris, M.; Lewis, S. A Short Study on the Success of the Gene Ontology. *Web Semant. Sci. Serv. Agents World Wide Web* **2004**, *1*, 235–240. [[CrossRef](#)]
3. Paulheim, H. How much is a Triple? Estimating the Cost of Knowledge Graph Creation. In Proceedings of the International Semantic Web Conference, Monterey, CA, USA, 8–12 October 2018.
4. Carlson, A.; Betteridge, J.; Kisiel, B.; Settles, B.; Hruschka, E.R., Jr.; Mitchell, T.M. Toward an Architecture for Never-Ending Language Learning. In Proceedings of the Conference on Artificial Intelligence, Atlanta, GA, USA, 11–15 July 2010; pp. 1306–1313.
5. Fader, A.; Soderland, S.; Etzioni, O. Identifying relations for open information extraction. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*; Association for Computational Linguistics; Edinburgh, UK, 2011; pp. 1535–1545.
6. Gerber, D.; Ngonga Ngomo, A.C. Bootstrapping the Linked Data Web. In Proceedings of the 1st Workshop on Web Scale Knowledge Extraction, Bonn, Germany, 23–27 October 2011.
7. Bhattarai, A.; Rus, V. Towards a Structured Representation of Generic Concepts and Relations in Large Text Corpora. In Proceedings of the International Conference Recent Advances in Natural Language Processing, Hissar, Bulgaria, 9–11 September 2013; pp. 65–73.
8. Ortega-Mendoza, R.M.; Villaseñor-Pineda, L.; Montes-y Gómez, M. Using lexical patterns for extracting hyponyms from the web. In Proceedings of the 6th Mexican International Conference on Advances in Artificial Intelligence, Aguascalientes, Mexico, 4–10 November 2007; pp. 904–911.
9. Mausam.; Schmitz, M.; Bart, R.; Soderland, S.; Etzioni, O. Open language learning for information extraction. In Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, Jeju Island, Korea, 12–14 July 2012; pp. 523–534.
10. Rada, R.; Mili, H.; Bicknell, E.; Blettner, M. Development and application of a metric on semantic nets. *IEEE Trans. Syst. Man Cybern.* **1989**, *19*, 17–30. [[CrossRef](#)]
11. Bühmann, L.; Lehmann, J. Pattern Based Knowledge Base Enrichment. In Proceedings of the 12th International Semantic Web Conference, Sydney, NSW, Australia, 21–25 October 2013; pp. 33–48.
12. Gavankar, C.; Kulkarni, A. Enriching an Academic Knowledge base using Linked Open Data. In Proceedings of the Workshop on Speech and Language Processing Tools in Education, Mumbai, India, 8–15 December 2012; pp. 51–60.
13. Mirza, P.; Razniewski, S.; Darari, F.; Weikum, G. Enriching Knowledge Bases with Counting Quantifiers. In Proceedings of the Semantic Web—ISWC 2018, Monterey, CA, USA, 8–12 October 2018; pp. 179–197.
14. Li, F.; Zhang, M.; Fu, G.; Ji, D. A neural joint model for entity and relation extraction from biomedical text. *BMC Bioinform.* **2017**, *18*, 1–11. [[CrossRef](#)] [[PubMed](#)]
15. Wu, F.; Weld, D.S. Open information extraction using Wikipedia. In Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, Uppsala, Sweden, 11–16 July 2010; pp. 118–127.
16. Xu, B.; Liang, J.; Xie, C.; Liang, B.; Chen, L.; Xiao, Y. CN-DBpedia2: An Extraction and Verification Framework for Enriching Chinese Encyclopedia Knowledge Base. *Data Intell.* **2019**, *1*, 271–288. [[CrossRef](#)]
17. Hochreiter, S.; Schmidhuber, J. Long Short-Term Memory. *Neural Comput.* **1997**, *9*, 1735–1780. [[CrossRef](#)] [[PubMed](#)]

18. Trisedya, B.D.; Weikum, G.; Qi, J.; Zhang, R. Neural Relation Extraction for Knowledge Base Enrichment. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, Florence, Italy, 28 July–2 August 2019; pp. 229–240.
19. Lin, Y.; Shen, S.; Liu, Z.; Luan, H.; Sun, M. Neural Relation Extraction with Selective Attention over Instances. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*; Association for Computational Linguistics: Copenhagen, Denmark, 2016; pp. 2124–2133.
20. Wu, S.; Fan, K.; Zhang, Q. Improving Distantly Supervised Relation Extraction with Neural Noise Converter and Conditional Optimal Selector. In Proceedings of the AAAI Conference on Artificial Intelligence, Honolulu, HI, USA, 27 January–1 February 2019; Volume 33, pp. 7273–7280.
21. Liu, T.; Wang, K.; Chang, B.; Sui, Z. A Soft-label Method for Noise-tolerant Distantly Supervised Relation Extraction. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*; Association for Computational Linguistics: Copenhagen, Denmark, 2017; pp. 1790–1795.
22. Ye, Z.X.; Ling, Z.H. Distant Supervision Relation Extraction with Intra-Bag and Inter-Bag Attentions. 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)*; Association for Computational Linguistics: Copenhagen, Denmark, 2019; pp. 2810–2819.
23. Cao, E.; Wang, D.; Huang, J.; Hu, W. Open Knowledge Enrichment for Long-Tail Entities. In Proceedings of the Web Conference 2020, Taipei, Taiwan, 20–24 April 2020; pp. 384–394.
24. Ji, G.; Liu, K.; He, S.; Zhao, J. Distant Supervision for Relation Extraction with Sentence-Level Attention and Entity Descriptions. In Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence (AAAI-17), San Francisco, CA, USA, 4–9 February 2017.
25. Zhang, X.; Li, P.; Jia, W.; Zhao, H. Multi-Labeled Relation Extraction with Attentive Capsule Network. In Proceedings of the AAAI Conference on Artificial Intelligence, Honolulu, HI, USA, 27 January–1 February 2019; Volume 33, pp. 7484–7491.
26. Resnik, P. Semantic similarity in a taxonomy: An information-based measure and its application to problems of ambiguity in natural language. *Artif. Intell. Res.* **1999**, *11*, 95–130. [[CrossRef](#)]
27. Miller, G.A. WordNet: A lexical database for English. *Commun. ACM* **1995**, *38*, 39–41. [[CrossRef](#)]
28. Bollegala, D.; Matsuo, Y.; Ishizuka, M. Measuring semantic similarity between words using web search engines. In Proceedings of the 16th International Conference on World Wide Web, Banff, AB, Canada, 8–12 May 2007; pp. 757–766.
29. Cilibrasi, R.L.; Vitanyi, P.M. The google similarity distance. *IEEE Trans. Knowl. Data Eng.* **2007**, *19*, 370–383. [[CrossRef](#)]
30. Huang, E.H.; Socher, R.; Manning, C.D.; Ng, A.Y. Improving word representations via global context and multiple word prototypes. In Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, Jeju, Korea, 8–14 July 2012; pp. 873–882.
31. Mikolov, T.; Sutskever, I.; Chen, K.; Corrado, G.; Dean, J. Distributed Representations of Words and Phrases and their Compositionality. In Proceedings of the Conference on Advances in Neural Information Processing Systems, Lake Tahoe, NV, USA, 5–8 December 2013; pp. 3111–3119.
32. Chen, Y.; Perozzi, B.; Al-Rfou, R.; Skiena, S. The expressive power of word embeddings. In Proceedings of the ICML 2013 Workshop on Deep Learning for Audio, Speech, and Language Processing Atlanta, GA, USA, 16–21 June 2013.
33. Levy, O.; Goldberg, Y. Dependency-Based word embeddings. In Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, Baltimore, MD, USA, 22–27 June 2014; pp. 302–308.
34. Chen, D.; Manning, C.D. A fast and accurate dependency parser using neural networks. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, Doha, Qatar, 25–29 October 2014; pp. 740–750.
35. Jiang, J.J.; Conrath, D.W. Semantic similarity based on corpus statistics and lexical taxonomy. In Proceedings of International Conference Research on Computational Linguistics, Taipei, Taiwan, 22–24 August 1997; pp. 19–33.
36. Riedel, S.; Yao, L.; McCallum, A. Modeling Relations and Their Mentions without Labeled Text. In *Machine Learning and Knowledge Discovery in Databases*; Springer, Berlin/Heidelberg, Germany, 2010; pp. 148–163.

37. Nathani, D.; Chauhan, J.; Sharma, C.; Kaul, M. Learning Attention-based Embeddings for Relation Prediction in Knowledge Graphs. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, Florence, Italy, 28 July–2 August 2019; pp. 4710–4723.
38. Sun, Z.; Huang, J.; Hu, W.; Chen, M.; Guo, L.; Qu, Y. TransEdge: Translating Relation-Contextualized Embeddings for Knowledge Graphs. In Proceedings of the ISWC, Auckland, New Zealand, 26–30 October 2019; pp. 612–629.

© 2020 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<http://creativecommons.org/licenses/by/4.0/>).