information

MDPI

Article

Error Detection in a Large-Scale Lexical Taxonomy

Yinan An, Sifan Liu and Hongzhi Wang *

Department of Computer Science and Technology, Harbin Institute of Technology, Harbin 150000, China; 1170300428@stu.hit.edu.cn (Y.A.); sifanl@andrew.cmu.edu (S.L.)

* Correspondence: wangzh@hit.edu.cn

Received: 13 December 2019; Accepted: 3 February 2020; Published: 11 February 2020

Abstract: Knowledge base (KB) is an important aspect in artificial intelligence. One significant challenge faced by KB construction is that it contains many noises, which prevent its effective usage. Even though some KB cleansing algorithms have been proposed, they focus on the structure of the knowledge graph and neglect the relation between the concepts, which could be helpful to discover wrong relations in KB. Motived by this, we measure the relation of two concepts by the distance between their corresponding instances and detect errors within the intersection of the conflicting concept sets. For efficient and effective knowledge base cleansing, we first apply a distance-based model to determine the conflicting concept sets using two different methods. Then, we propose and analyze several algorithms on how to detect and repair the errors based on our model, where we use a hash method for an efficient way to calculate distance. Experimental results demonstrate that the proposed approaches could cleanse the knowledge bases efficiently and effectively.

Keywords: knowledge base; error detection; cleansing

1. Introduction

Knowledge bases (KB), such as YAGO [1] and DBpedia [2], which typically contain a set of concepts, instances, and relations, has become significantly important in both industry and academia [3]. With the increasing of data size on the web, KB automatic construction approaches [4] are proposed to extract knowledge from massive data efficiently. Due to the low-quality of raw data and the limitation of KB construction approaches, automatically constructed KBs are nagged by quality problems, which will cause serious accidents in applications. Thus, KB cleansing approaches are in great demand to increase the usability of KBs.

Many KB and database cleansing algorithms have been proposed in recent years such as building directed acyclic graph [5] and applying other new knowledge. The main idea of the building directed acyclic graph method is to enumerate cycles and eliminate the relation with low trust because cycles are highly likely to contain incorrect relations. However, such approach focuses on the structure and level of the KB with the perspective of a graph, and does not have the ability to detect low frequency errors that are not included in a cycle. As a contrast, our method is good at dealing with low-frequency data. As for applying other new knowledge such as another KB or internet knowledge, it would take lots of time and each KB contains its unique relations that would be very different from the one that we are dealing with. A statistical regression method is a kind of mathematical method [6]. It uses statistical technology for data integration. This method is mainly aimed at dealing with the linear relationship in the knowledge map. By dealing with the quantitative relationship between digital attributes, we can get the return method for the linear relationship. It is obvious that regression is only effective for finding linear relationships. However, this method can not get an effective solution to the nonlinear relationship. Therefore, this method is not very effective for most knowledge maps.

In summary, these algorithms focus on the structure of the knowledge graph and external knowledge without sufficient usage of the relations in the KB that need to be cleansed. When we think

about concepts, we usually consider the relation of two concepts. For example, bird and fish are two conflicting concepts while bird and animal are two related concepts. Motivated by this, we attempt to adopt the relations between instances sets corresponding to the concepts for effective KB cleansing. Such approach brings the following technical challenges.

First, in a KB, the frequency of many relations is 1. For example, in Probase, 7M relations only emerge once. Thus, we can hardly derive error from frequency and have to seek frequency-independent error detection approaches. Second, current KBs are almost in large scale, which leads to inefficient data accessing during KB cleansing. It requires sophisticated data structures and algorithms for cleansing algorithms to achieve high performance. Third, with the existence of homonyms, even though a conflicting is discovered, it should be distinguished whether it is caused by incorrect triples or homonyms. Such distinguishing is crucial for cleansing strategy determination and in great demand.

Facing these challenges, we attempt to develop efficient KB cleansing algorithms in this paper. We observe that the relation between concept sets corresponding to conflicting concepts could be adopted for KB cleansing.

Considering the example for motivation, the concepts of bird and fish are conflicting, which means *a creature* could not be both a *fish* and a *bird*. Therefore, the incorrect triple *(eagle, Isa, fish)* in Probase is detected since *eagle* is in the intersection of the concept sets corresponding to *fish* and *bird*.

Based on this observation, we develop two error detection algorithms with different kinds of distance measures between concept sets from different aspects. For effectiveness issues, these algorithms adopt Hamming distance and Jaccard distance respectively to take advantage of more trustworthy relations and take full use of the total domain size. These two algorithms could allow us to take full use of the information provided in the Knowledge base itself. All of these algorithms are based on set distance computation. For efficiency issues, the optimization of set distance computation is easier than that of a large graph since complex structural information is not required to be collected and existing similarity join method for sets could be applied. To accelerate processing, we apply Simhash and Minhash LSH (Locality Sensitive Hashing for these algorithms, respectively.

For detecting incorrect triples, we develop a two-level repair mechanism for a more accurate result. For the easy cases, we repair the triples according to frequencies. When the frequency-based approach does not work, we develop the crowdsourcing-based repair approach to distinguish homonyms.

Furthermore, in the past, many different algorithms have been proposed for homonym extraction such as the simple lexical patterns [7], statistical, and machine learning techniques [8]. The usage of Isa relations are often used in induction of taxonomies. However, there are lots of errors in the automatically constructed KBs with low frequency findings in the corpus and web, and it is a big challenge to detect and repair them.

About the hash method, several hash methods have been largely used on detecting duplicate web pages and eliminating them from search results (AltaVista search engine) [9], and they have also been applied in large-scale clustering problems, such as clustering documents by the similarity of their sets of words [10]. These hash methods also give us a technique for quickly estimating how similar the two sets are. Therefore, we could apply these methods in our research.

Then, we talked about error detection. Many attempts have been proposed to detect and resolve conflicts in knowledge bases. Li et al. [11] proposed the OWL-based method to detect several conflict types and resolve them in RDF knowledge base integration. Liang et al.proposed to enumerate cycles and eliminate the relation with low trustworthy score [5]. However, these methods do not pay attention to the properties of concept. In our paper, we propose to use the set views according to the Knowledge base.

The contributions of this paper are summarized as follows:

First, we detect errors in KBs according to the relation between the concept sets. Such approach makes full usage of the Isa relationship between concepts and is easy to accelerate for set operations. Second, we develop three efficient KB error detection algorithms for various distances with acceleration strategies as well as the two-level repair algorithm. Such KB error detection and repairing algorithms

achieve universal by diversification. Third, experimental results demonstrate that the proposed algorithms outperform existing approaches and could cleanse the KB efficiently and effectively.

This paper is organized as follows. In Section 2, we discuss our distance-based model, containing corresponding concepts and calculation method of distance. In Section 3, we show how we do error detection with our distance-based model. In Section 4, we introduce our algorithm used for repairing. In Section 5, we conduct experiments. Sections 6 and 7 will be related work, discussion, and conclusions.

2. Distance-Based Models

Even though some KB error detection methods have been proposed, they fail to fully make use of the relations in KB. Major KB error detection approaches could be classified into two kinds.

One is the frequency-based approach [12]. The low frequency means that the relation is seldom noticed in the corpus or the web. Therefore, some approaches have been proposed to use frequency to determine whether a relation is correct or not. Clearly, many relations with low frequencies are correct. In Table (a), we list the percentage of frequencies in different range. We randomly selected 100 relations in the Probase [5] to check the correctness, and the answer is given by human judgment. In addition, we list the correctness rate of these data with different frequencies in Table (b). From this experiment, just using frequency to determine whether a relation is wrong could be an unwise way to find errors and would achieve a low accuracy. We give specific example to show its limits.

Example 1: *The frequencies of triples (*`snake`*, Isa,* `herb`*) and (*`fruit fly`*, Isa,* `creature`*) are both 1 in Probase. However, the latter is correct while the former is wrong. It shows that just using frequency fails to detect many errors.*

The other is structure-based approaches [5,13,14]. It has been observed that most cycles in a knowledge graph contain wrong Isa relations since a prefect knowledge base should be acyclic. Therefore, eliminating cycles could correct wrong Isa relationships. However, many wrong Isa relations are not contained in any cycle since the cycle is just a specific structure and many wrong errors do not happen to be in such specific structure.

Therefore, we attempt to find a more general approach for error detection. At first, we discuss the motivation of our distance-based approaches. Then, we discuss the definition of distances as well as their combinations.

2.1. Motivations

We observe that many errors are caused by misclassification of instances to concepts. Using the example stated before, *fish* and *bird* are two obvious conflicting concept sets because there is no such *creature* in the world is both a *bird* and a *fish*. If the same *creature* belongs to the concept of both *fish* and *bird*, at least one of these two Isa relations are wrong. According to this intuition, we detect errors using the intersection of conflicting sets.

We first define some related concepts.

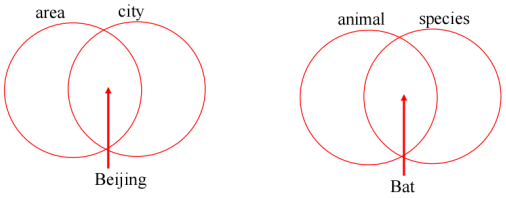
Definition 1 (Concept Sets) *Given a Knowledge base with Isa Relation, the weighted concept sets constructed based on this KB contains all the instances which have Isa relation with the concepts, and each instance is associated with a weight w(p).*

For example, we use the instance stated before, *fish* is a concept in Probase, and we build a concept set called *fish*, which contains all the instances such as *tuna*, *salmon* and *catfish*. In addition, if we use the frequency as the weight w(p), the weight of these three are 1892, 3733, and 562.

Definition 2 (Conflicting Concept Sets) *Given all the concept sets in a Knowledge base with Isa Relation. Consider two of these concept sets each, if these two concepts are incompatible in semantics, their corresponding concept sets are called conflicting concept sets.*

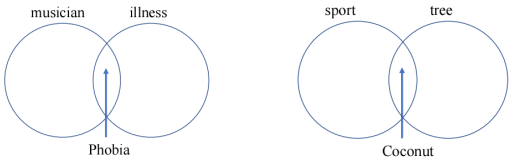
For example, we could construct concept sets *bird* and *fish* from the knowledge base according to Definition 1. It is obvious that *bird* and *fish* are conflicting concept sets since they are incompatible in semantics.

We now show how to use the intersection of two Concept Sets to detect error. In the figure showed above, we present two kinds of relations of concept sets in the knowledge base. Figure 1 shows two similar concept sets, which means each set has some similar properties with the other, while Figure 2 shows two conflicting concept in semantics, which means the Conflicting Concept Sets stated before. It is obvious that the intersection of two Conflicting Sets contains errors of misclassification of instances. Therefore, computing the intersection of conflicting concept sets allows us to detect errors of low frequency with high accuracy. Clearly, the ideal intersection of two conflicting concept sets should be empty. On the other hand, if their intersection is not empty, errors have a highly probability to occur on the instance in their intersection. In this way, we perform error detection with set operations, which are pretty efficient. Additionally, this approach is not affected by the frequencies.



(a) Similar Sets 1 (b) Similar Sets 2

Figure 1. Similar concept sets.



(a) Conflicting Sets 1 (b) Conflicting Sets 2

Figure 2. Conflicting concept sets.

For a human, it is easy to distinguish conflicting concepts. However, the challenge is to find them automatically in massive knowledge bases. Inspired by the similarity measure for documents [15], we attempt to use the distance between concept sets to determine conflicting sets. Intuitively, with the assumption that a KB is large enough to contain all instances of each concept, if two concept sets are very different in their instances, they are different in semantics correspondingly. Based on this intuition, we determine the conflicting concept sets according to the distance. Thus, the definition of distance is crucial to conflicting concept sets determination, and we will discuss it in the next subsection.

2.2. Distances

Many distances or similarities have been proposed to measure the difference of sets. In order to fully use the information that was provided by the KB, the more trustworthy relations should be considered importantly according to each concept sets, and the Hamming distance could take consideration of the importance of each relation. In addition, as we stated before, an ideal intersection of two conflicting concept sets should be empty. Therefore, we also need to consider the number of relations in the intersection that is the reason to use Jaccard distance. In conclusion, we select two different distances for conflicting concept detection, Hamming distance [16], and Jaccard distance [17].

Their definitions are listed as follows.

Hamming distance: For two sets A and B, we define a hash function $h(.)$ to map a set to a string. Thus, the Hamming distance between A and B $H(A, B) = h(A)XORh(B)$.

Jaccard distance: For two sets A and B, their Jaccard distance $J(A, B) = 1 - \frac{|A \bigcap B|}{|A \bigcup B|}$.

The main purpose to use these two distances is that it could serve as different criteria for us to determine the conflicting concept sets. The Hamming distance could calculate the distance between two hash signatures, and it could be easy for us to embed frequency of instance to the distance measure. The Jaccard distance uses the percentage of the number of intersection of two concept sets to determine the conflicting between two concept sets. In addition, this is similar to the people cognition strategy since we know that the two conflicting concept sets have no instances in their intersection. At the same time, Jaccard could take use of the full domain size of these two concept sizes. Therefore, these two distances could work well to fully consider the information provided by the knowledge base and to detect errors. Moreover, we consider some other distance measures. Measures for vectors, like Chebyshev distance, are not fit because it is not worth converting concept sets to vectors only for a distance. Compared with it, the chosen distances are more convenient. We also consider measures for vector sets like Hausdorff distance, and we give up for similar reasons.

2.3. The Combination of Distances

Therefore, as listed before, each of these distances has its own pros and cons. To achieve high accuracy, a feasible approach is to combine them to avoid the limitations. According to above discussions, we have the following combination strategies:

1. Using Hamming distance to generate the set S_H of conflicting concept sets.
2. Using Jaccard distance to generate the set S_J of conflicting concept sets.
3. Combining the result S_H and result S_J to get final result S.

3. Detecting Algorithm

In the previous section, we show the approach of distance-based error detection. Due to the large scale of KBs, efficient and scalable algorithms are in demand. Thus, in this section, we develop efficient distance-based error detection algorithms.

Definition 3 (Signature) *Transform the concept set into a string S_c with some kind of hash function. Regard the string S_c as the signature of the concept set.*

For efficiency issues, we adopt the hash method for the acceleration. That is, for each concept set S, we generate a signature S_c. Thus, a set S is generated for all signatures. Then, a distance join operation, which is similar to the similarity join, is performed on S to generate the results.

For these three distances, we apply different hash functions and distance join strategies for efficiency issues.

Simhash Method for Hamming distance: Simhash is a fast algorithm for us to calculate Hamming distance in large scale data. However, one significant characteristic of this method is that it makes use of the weights of every instance, so the hash signature of each concept set is mainly depending on the larger weighted instance, and the smaller instance relation only acts as a very small influence; therefore, it is useful to distinguish the conflicting when the two concept sets are very different with each other, while Hamming distance could not distinguish when two concept sets have very different domain sizes.

Minhash LSH Method for Jaccard distance: Minhash LSH is also a very useful method in dealing with the document duplicate removal or web page comparing, and it is based on the Jaccard distance; the main advantages of Minhash LSH are that it could produce the Jaccard distance in a very short time; therefore, it is suitable for gigantic data size. However, as stated before, Jaccard distance faces two problems: the subset inclusion relations problem and the big set domain size influence.

Combination of Simhash Method and Minhash LSH Method: In order to find more conflicting concept sets with high accuracy, we propose to combine the Simhash Method and Minhash LSH Method stated above. The Simhash Method could make full use of the high frequency information according to each relation because higher frequency represents higher trustworthiness. Therefore, the Simhash Method could find conflicting concept sets which have very little of the same high frequency relations in both of the concept sets, but it does not consider the differences between domain size and

the influence of low frequency relations. However, the Minhash LSH Method makes full use of domain size and regards each relation with the same importance without considering frequency. Therefore, both the Simhash Method and Minhash LSH Method could find conflicting concept sets with high accuracy, but each of them may ignore some kind of conflict while the other could detect just as stated above. In addition, we will discuss our two-level repair method in the next section.

Therefore, to make a better detection, we propose to use the Combination of Simhash Method and Minhash LSH Method for general Knowledge base. In addition, if we are detecting a Knowledge base without frequency, we propose to use the Minhash LSH Method.

4. Repairing Algorithm

After computing the conflicting concept sets, we could achieve our main purpose of KB cleansing by computing the intersection of conflicting concept sets. Our main purpose is to repair errors, and there are another two kinds of relations that can be detected in the intersection of conflicting concept sets. The following variables, L and B, are thresholds specially chosen. We'll use them to judge which situation the word in intersection is in and then we decide what to do in the situation.

Definition 4 (Errors) *Given all the relations in an intersection of two conflicting concept sets, if there is an instance P with large differences between two weights regarding two different concept sets, the larger weight is bigger than B, and the low weight is smaller than L, then the relationship R with the smaller weight according to instance P is the error.*

For Examples, consider the intersection of *bird* and *fish* concept sets as we discussed before, and the frequency of *turkey* Isa *bird* in Probase is 211, which means that there are 211 sentences containing this Isa relation. In addition, the frequency of *turkey* Isa *fish* in Probase is 1. It is obvious that turkey belongs to bird not fish. Therefore, we could say that the relationship turkey Isa fish is an error.

Since the concept sets we are dealing with are conflicting with each other, for the second level of repairing, we could use the frequency in the KB as the weight to decide which relations should be deleted. At the same time, we could also apply different weights for cleansing according to different situations.

Definition 5 (Homonyms) *Given all the relations in the intersection of two conflicting concept sets, if there is an instance P with both large weights, where the weights are both bigger than B, then the relations in both concept sets can be correct, and P is a homonym instance in both concept sets.*

As we stated before, the relationships in the intersection of two conflicting concept sets can be both correct because one instance could have multiple meanings. However, in the experiment, we find that conflicting concept sets could have very little homonyms because the similarity degree of these two sets is significantly low. Therefore, we propose the idea to give these instances sub-attributions to identify them in the different relations and concepts.

Definition 6 (Suspicious relations) *Given all the relations in the intersection of two conflicting concept sets, if there is an instance P with both very low weights regarding the both concept sets, where the weights are both lower than L, then the relations in both concept sets are suspicious.*

We still consider the intersection of *bird* and *fish* conflicting concept sets, the frequency of *maple* Isa *fish* is 1, and the frequency of *maple* Isa *bird* is also 1. As humans, we could successfully judge these two relations to be both errors, but, if we look at the intersection of *fish* and *herb*, the frequency of *health supplement* Isa *fish* is 1, and the frequency of *health supplement* Isa *herb* is also 1. As humans, we view these two relations differently according to different people; therefore, it is hard to automatically distinguish these suspicious relations.

We notice that the suspicious instances are usually both seldom using relations or even wrong relations in the conflicting concept sets. In addition, it is even hard for human to raise up an agreement of right or wrong of these relations. Our cognition could provide very different conclusions. Therefore, there is no efficient automatically way to efficient classify these suspicious relations. In the experiment, we propose to build a suspicious knowledge base(SUSKB) and put these suspicious relations into SUSKB, and people could manually remove these relations in the SUSKB as they want.

5. Experiment

To verify the effectiveness and efficiency of the proposed approaches, we evaluate our approach in this section. We first analyze the influence of the parameters based on 100 concept sets. Next, we will apply our method on Probase and compare it with the latest error detection algorithms in precision. In addition, we finally show our method could also be used in other knowledge bases.

Definition 7 (Bucket number) *A threshold used in Minhash LSH algorithm, in which we devide big data into buckets and then we only need to find the same set of buckets to calculate the similarity. As the name suggests, buckets number means the number of buckets we divide data into.*

We can tell from the result that using the Combination distance could perform the best result. Because we apply hash method to conducting our results, our method largely depends on the parameters that we select. In the Minhash LSH method that we use involves two parameters, one is the buckets number and the other is the threshold to determine whether two sets are conflicting with each other. In the next subsection, we are going to analyze the parameters carefully using 100 concept sets randomly selected from the total concept sets.

5.1. Exp1: Analyzing for Parameters

From the total results above, we know that Jaccard distance could bring a wonderful result to cleanse the knowledge bases. Since the Minhash LSH method that we use involves two parameters, one is the buckets number and the other is the threshold to determine whether two sets are conflicting with each other. The number of bucket decides the possibility that two concept sets will be mapped into the same bucket. In addition, the smaller bucket number means a looser criteria to determine two concept sets are conflicting. The more possible two concept sets are judged conflicting, the more errors we can find. The second parameter influences the result in the same way. We use 100 concept sets to analyze the influence of these parameters.

Figure 3 shows the number of errors that can be found when we set the bucket number as 64, 128, and 256. It is easy to find that, when the bucket number is 64, the errors of these 100 concept sets are the highest, while the bucket number being 256 gets the lowest error number. This is in line with the algorithm of Minhash LSH. Since the number of bucket decides the possibility that the pair will be mapped into the same bucket, the smaller bucket number means a looser criteria to determine two concept sets that are conflicting. Let us use a more intuitive way to analyze. Consider five identical concept sets: animal, bird, fish, fruit, and herb. In addition, the results of the human judgement and the bucket number influence show below.

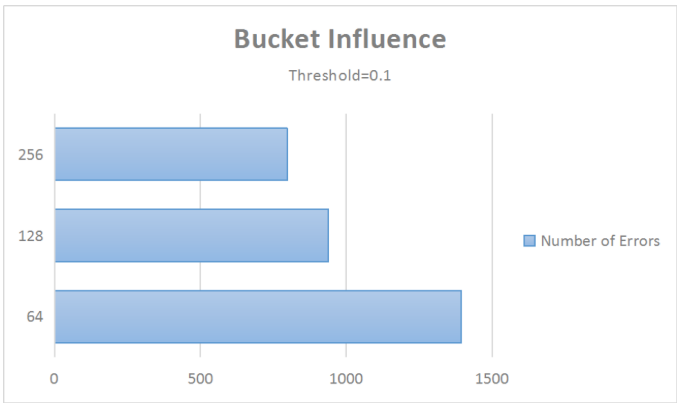


Figure 3. Bucket influence.

We can see Table 1 clearly that when the bucket number = 64, there are six concept sets being determined to be conflicting while when it comes to 128 and 256, there is only four concept sets are conflicting with each other. In addition, there is one more important thing is that, as we can tell from the table, no matter how many conflicting concept sets are found, the judgment by the computer is right

according to the human judgment. Therefore, if there need to find more errors from the knowledge base, we suggest setting a smaller bucket number, while if the criteria need to be strict, then the bucket number should be higher. In the following analysis of parameters, we choose to use 128.

Table 1. Case study

Intersection	Human Judgement	64	128	256
animal_fruit	yes	no	no	no
animal_herb	yes	yes	yes	yes
animal_bird	no	no	no	no
animal_fish	no	no	no	no
fruit_herb	yes	no	no	no
fruit_bird	yes	yes	yes	yes
bird_fish	yes	yes	yes	yes
fish_fruit	yes	yes	no	no
bird_herb	yes	yes	no	no
fish_herb	yes	yes	no	no

Next, we are going to show the Figure 4 threshold influence. We know that the Minhash LSH is Locality Sensitive Hashing, which is often used to find the similarity in large scale data with high dimensions. Using LSH is faster than linear search. Hamming distance and Jaccard distance in our method are all locality sensitive. Below is the result when we change the threshold of the Minhash.

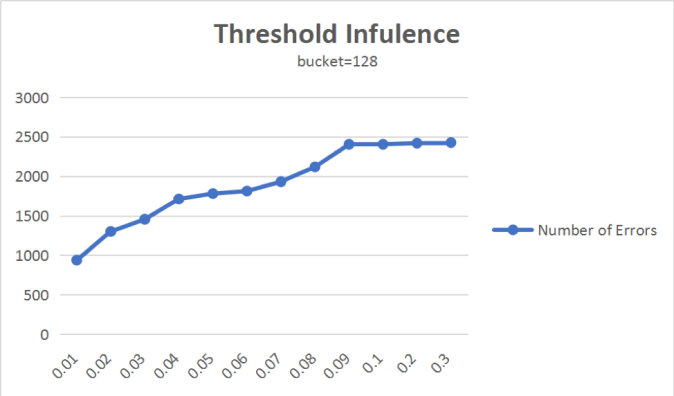


Figure 4. Threshold influence.

We can see clearly that, when the threshold is larger than 0.1, the number of the errors we find from the 100 concept sets doesn't change much, which means that the conflicting concept sets selected from the 100 concept sets are almost the same.

One of the important aspects of our method is that we use a two-step verification to ensure that the errors are more trustworthy. In the next step after selecting the conflicting concept sets using Hamming distance and Jaccard distance, we use three weight differentials to determine whether the relations in the intersection of two conflicting concept sets are errors, Homonyms, or Suspicious relations as we stated before. In the following, we pay attention to the weight differential to find errors, which means the differential needs to be large enough. We first show Figure 5 the error distribution when using different weight differentials.

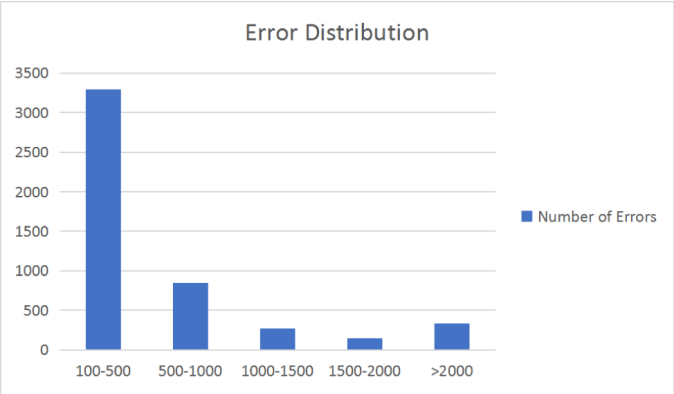


Figure 5. Error distribution.

From the error distribution, we can easily tell that the number of errors are decreasing as the differentials are setting stricter, which is obvious since the percentage of large frequency in Probase is small. Then, we list all the results when selecting the different weight differential and different minimum weight. The results contain numbers of errors that can be found and the precision deciding by human volunteers. In our experiment, we make the bucket as 128 and the threshold as 0.01.

We set the minimum weights to be 1 to 10 and above 10 in different section of weights differential. Then, we measure the number of errors that can be found in 100 concept sets according to each situation, and we also show the errors truly wrong by human judgment. From the above result, we see that, when we set the weights differential as 100 to 500 and the minimum weights to be 1, the number of errors is the largest. In addition, the minimum weights as 1 takes the largest proportion in each weights differential. Since our purpose is to find and remove errors, we need to obtain the highest precision rate. From the human judgment, we find that, when the weights differential becomes higher, the precision rate is higher too. In addition, when the minimum weight is less than 5, the precision rate remains high. However, when the minimum weight is larger or equal to 5, the precision rate drops to nearly 50.

Figure 6 shows the errors detected. The left pictures show distribution of the lower weight (mentioned in Section 4) with different distances between concept sets. The right picture shows the number of errors detected and the numbers of correct judgement. From the pictures above, obviously, the more different the two concept sets are, the higher the accuracy rate we get. In addition, we can see that the lower weight 1 really accounts for a large proportion.

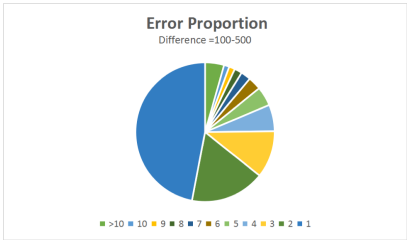
Table 2. Detection results

Model	Errors	Concepts Set Precision (%)	Errors Precision (%)
baseline	74.2K	-	91.3
Hamming Distance	100K	83.3	89.2
Jaccard Distance	90K	86.0	91.4
Combination	120K	92.7	92.3

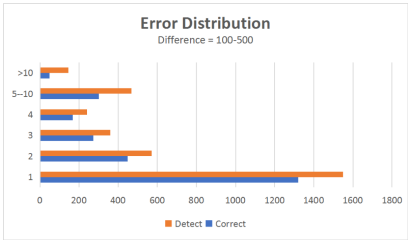
5.2. Exp2: Applying on Probase

In this experiment, we apply our methods on the core version of IsA data mined from billions of web pages, which contains 5,376,526 unique concepts, 12,501,527 unique instances, and 85,101,174 IsA relations, and most of them have small frequencies as stated before.

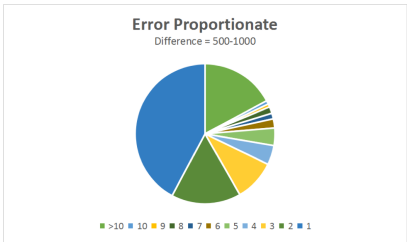
Because we do not know the number of errors that are contained in Probase, we then use the precision rate to evaluate our models, and we use the highest precision rate of the latest error detection method as a comparison. Precision rate means the proportion of the truly wrong IsA relations in all relations detected by our methods. We randomly pick 500 wrong IsA relations to determine whether it



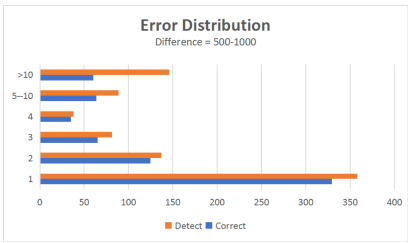
(a) difference = 100 - 500



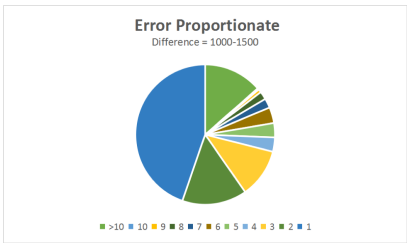
(b) difference = 100 - 500



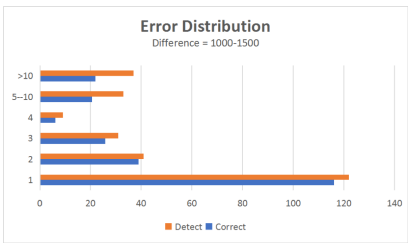
(c) difference = 500 -1000



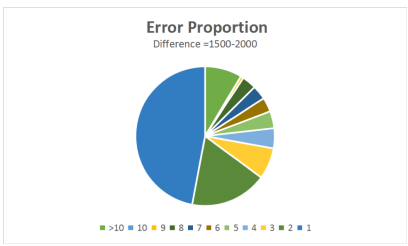
(d) difference = 500 -1000



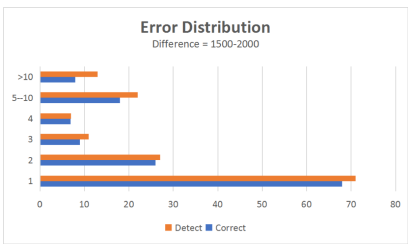
(e) difference = 1000 -1500



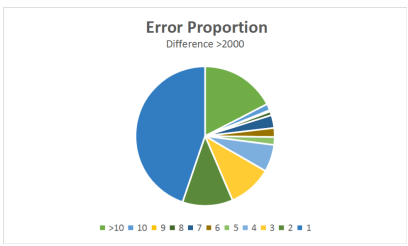
(f) difference = 1000 -1500



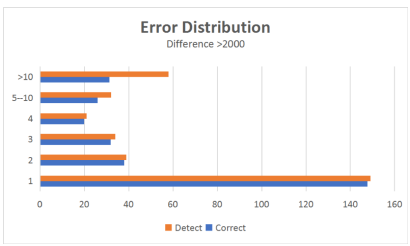
(g) difference = 1500 - 2000



(h) difference = 1500 - 2000



(i) difference > 2000



(j) difference > 2000

Figure 6. Weights Influence

is right or wrong. Since this work could only be done by people, we ask 50 volunteers to judge our results. In addition, the final results are in Table 2.

6. Related Work and Comparison

In this section, we list related work and some of the recent KB and database cleansing algorithms and compare our algorithm with them.

Many taxonomies have been constructed manually or automatically. We can extract relations from existing data bases, or we can train structured HTML tables for specific domains. However, no matter which kind of method we use, low frequency errors tend to be a challenge. In addition, our algorithm can partly deal with it. In addition, if we build a knowledge base manually, it will be important to find and correct the errors in data sets. Existing methods like the statistical regression method [6] and OWL-based method [11] cannot be applied to noisy automatically generated taxonomies.

One famous algorithm is a building directed acyclic graph [5]. The main idea of it is to enumerate cycles and eliminate the relation with low trustworthiness because cycles are highly likely to contain wrong relations. However, such approach focuses on the structure and level of the KB with the perspective of a graph, and does not have the ability to detect low frequency errors that are not included in a cycle. As a contrast, our method is good at dealing with low frequency data. Another method is applying other new knowledge. As for applying other new knowledge such as another KB or internet knowledge, it would take lots of time, and each KB contains its unique relations that would be very different from the one that we are dealing with. A statistical regression method is a kind of mathematical method [6]. It uses statistical technology for data integration. This method is mainly aimed at dealing with the linear relationship in the knowledge map. By dealing with the quantitative relationship between digital attributes, we can get the return method for the linear relationship. It is obvious that regression is only effective for finding linear relationships. However, this method can not get an effective solution to the nonlinear relationship. Therefore, this method is not very effective for most knowledge maps.

7. Discussion and Conclusions

In our work, we try to solve the problem of identifying errors of the IsA relationships from the automatically constructed Knowledge bases. Our key contribution is that we find that the intersection of two conflicting concept sets are highly likely to contain errors. We thus propose to use two different distance based models to efficiently and effectively compute the conflicting concept sets. In addition, we analyze many influences of the error detection. In addition, we suggest to give sub-attributes for homonyms and build a suspicious knowledge base for suspicious relations. We evaluate all our models with experiments and show that our model could produce a higher accuracy in error detecting and repairing.

There is some further work to do in this issue:

- To further improve the quality of knowledge bases, we are going to find better weights to classify the relations in the intersection of two conflicting concept sets.
- Homonyms is a hard problem in cleansing, and we need to find more systematic ways to deal with it.
- We will further expand our method to other types of knowledge bases, and provide a suggesting order on how to apply the cleansing method.

Author Contributions: Conceptualization, H. W.; methodology, S. L.; software, Y. A. and S. L.; validation, Y. A. and S. L.; formal analysis, S. L and H. W.; investigation, Y. A. and S. L.; resources, Hongzhi Wang; data curation, Y. A. and S. L.; writing–original draft preparation, Y. A. and S. L.; writing–review and editing, H. W.; visualization, Y. A. and S. L.; supervision, Hongzhi Wang; project administration, H. W.; funding acquisition, H. W. All authors have read and agreed to the published version of the manuscript.

Funding: This paper was funded the National Key R&D Program of China under Grant No. 2018YFB1004700, and NSFC Grant Nos. U1866602, 61602129, 61772157.

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

References

1. Weikum, G. Yago: a core of semantic knowledge. *International Conference on World Wide Web*, Association for Computing Machinery: New York, NY, USA, 2007, pp. 697–706.
2. Yu, L. DBpedia. *A Developer's Guide to the Semantic Web*, Springer Science & Business Media: Atlanta, GA, USA, 2014, pp. 383–414.
3. Nakai, K.; Kanehisa, M. A knowledge base for predicting protein localization sites in eukaryotic cells. *Genomics* **1992**, *14*, 897.
4. Murray, K.J.B. Knowledge-based model construction: an automatic programming approach to simulation modeling, PHD Thesis, Texas A&M University, College Station, TX, USA, **1986**.
5. Liang, J.; Xiao, Y.; Zhang, Y.; Hwang, S.; Wang, H. Graph-Based Wrong IsA Relation Detection in a Large-Scale Lexical Taxonomy. In Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, 4–9 February 2017, San Francisco, CA, USA, 2017, pp. 1178–1184.
6. Lu, H.; Fan, W.; Goh, C.H.; Madnick, S.E.; Cheung, D.W. Discovering and reconciling semantic conflicts: a data mining perspective. *Data Mining and Reverse Engineering*; Springer: Boston, MA, USA, 1998; pp. 409–427.
7. Hearst, M.A. Automatic acquisition of hyponyms from large text corpora. *Conference on Computational Linguistics*, Association for Computational Linguistics: Nantes, France. 1992, pp. 539–545.
8. Espinosa-Anke, L.; Ronzano, F.; Saggion, H. *Hypernym Extraction: Combining Machine-Learning and Dependency Grammar*. CICLing: Cairo, Egypt, 2015, 372–383.
9. Broder, A. *On the Resemblance and Containment of Documents*; IEEE: PaloAlto, CA, USA. 1997; pp.21–29.
10. Broder, A.Z.; Charikar, M.; Frieze, A.M.; Mitzenmacher, M. Min-Wise Independent Permutations. *J. Comput. Syst. Sci.* **2000**, *60*, 630–659.
11. Li, C.; Ling, T.W. OWL-Based Semantic Conflicts Detection and Resolution for Data Interoperability. *Lec. Notes Comput. Sci.* **2004**, *3289*, 266–277.
12. Van, d.B.J.; Cunningham, S.A.; Eeckels, R.; Herbst, K. Data cleaning: detecting, diagnosing, and editing data abnormalities. *Plos Medicine* **2005**, *2*, e267.
13. Clauset, A.; Moore, C.; Newman, M.E. Hierarchical structure and the prediction of missing links in networks. *Nature* **2008**, *453*, 98.
14. Gupte, M.; Shankar, P.; Li, J.; Muthukrishnan, S.; Iftode, L. Finding hierarchy in directed online social networks. In proceedings of the International Conference on World Wide Web, Hyderabad, India, March 28–April 1 2011, pp. 557–566.
15. Tong, S. Document similarity detection, US Patent No. 8650199, 6 March 2014.
16. Hamming, R.W. Error Detecting and Error Correcting Codes. *Bell Syst. Tech. J.* **2014**, *29*, 147–160.
17. Jaccard, P. The Disbution of the flora in the alpine zone.1. *New Phytolog.* **2010**, *11*, 37–50.