Detecting New and Arbitrary Relations among Linked Data Entities using Pattern Extraction

Subhashree S and P Sreenivasa Kumar

Department of Computer Science and Engineering,
Indian Institute of Technology - Madras,
Chennai, India
{ssshree,psk}@cse.iitm.ac.in

Abstract. Although several RDF knowledge bases are available through the LOD initiative, often these data sources remain isolated, lacking schemata and links to other datasets. While there are numerous works that focus on establishing that two resources are identical and on adding more instances of an already existing relation, the problem of finding new relations between any two given datasets has not been investigated in detail. In this paper, given two entity sets, we present an unsupervised approach to enrich the LOD cloud with new relations between them by exploiting the web corpus. During the first phase we gather prospective relations from the corpus through pattern extraction and paraphrase detection. In the second phase, we perform actual enrichment by extracting instances of these relations. We have empirically evaluated our approach on several dataset pairs and found that the system can indeed be used for enriching the existing datasets with new relations.

Keywords: Linked Data, LOD enrichment, pattern extraction, relation detection, arbitrary relations, RDF knowledge base

1 Introduction

The Linked Data initiative aims to provide a set of guidelines and best practices for publishing structured data and associating it with other resources. The Linking Open Data community project\(^1\) works with the main objective of publishing open data sets as RDF triples and establish RDF links between objects from different data sets and has the potential of complementing the world wide web with a data space of entities connected to one another with labelled edges. Many organizations have built systems to exploit the power of Linked Data for specific purposes. For example, the British Broadcasting Corporation (BBC) uses linked datasets such as DBpedia and Musicbrainz to enable cross-domain navigation and enhanced search\(^2\) in their websites. IBM has been using Linked Data as an integration technology for several years and their new cognitive system, Watson\(^3\) has DBpedia and YAGO as part of its major data sources [1]. Though Linked Data can prove to be an influential technology in such scenarios, providing

\(^1\)http://www.w3.org/wiki/SweoIG/TaskForces/CommunityProjects/LinkingOpenData

\(^2\)https://www.w3.org/2001/sw/sweo/public/UseCases/BBC/

\(^3\)http://watson.kmi.open.ac.uk/WatsonWUI/
enormous benefits, its scope will be limited to a large extent unless it is fully grown and updated. For example, in the case mentioned above, Watson may not be able to answer questions about Greek mythology if it does not have information about personalities and facts from the Greek mythology represented as Linked Data. Hence, there arises a need to keep enriching the Linked Open Data cloud\(^4\) in all aspects, i.e., more domains need to be covered, and more entities, concepts and links between them are required.

Due to the rapid growth of the Open Data and Linked Data movements, a large amount of data is being made available as Linked Data and this has helped fulfil the domain coverage part of the Linked Data growth, stated above. However, Linked Data is heterogeneous and sparsely interlinked. Most often, before publishing data, publishers do not look for existing data and establish links between the existing data and their own data. Instead, they define their own vocabularies and publish them as a new isolated dataset giving rise to a very sparsely interlinked cloud. There are numerous works in the literature which focus upon asserting that two entities (or two properties or two concepts) are actually the same by means of establishing owl:sameAs or similar kind of links [2]. However, the other dimension to this linking process, namely, establishing new relations between the two entities, is comparatively less explored. More specifically, to the best of our knowledge, given two datasets there are not many techniques available in the literature to identify the possible relations between them. For example, given the names of vegetables and non-contagious diseases, one would like to know the possible relations existing between them. Though one can intuitively guess a couple of relations between the two datasets to be “cures”(vegetable x cures disease y), and “prevents”(vegetable x prevents disease y), one does not know which among these two relations exactly occurs between a given vegetable and a disease. It is this problem that we intend to solve in this paper by using pattern extraction on unstructured text.

In this paper, we address the problem of discovering new and arbitrary relations between two Linked Data entity sets. We adopt an unsupervised approach that is based on extraction of patterns from large text corpora. We conducted experiments on datasets belonging to various domains to prove that our approach is versatile. For example, we are able to extract triples of the form (broccoli\(^5\), prevention of, cancer\(^6\)), and thus enrich LOD. The rest of the paper is organized as follows: Section 2 describes the related works in the literature. Section 3 gives an overview of the approach with each phase of the approach explained in detail. The experiments conducted by us as a part of evaluating this approach are presented in Section 4 along with the comparison of our system with one of the existing related works. Section 5 concludes the paper and gives directions for future work.

2 Related Work

Extracting patterns for expressing relations has been a topic of research in depth in the past few decades [3]. Supervised approaches for automatic relation extraction from text require a large amount of labeled training data which is expensive and hence a number

\(^4\)http://lod-cloud.net/
\(^5\)http://rdf.freebase.com/ns/m.0hkxq
\(^6\)http://rdf.freebase.com/ns/m.0qcr0
of semi-supervised approaches such as Snowball [4], KnowItAll [5], TextRunner [6], NELL [7] have been proposed recently for extracting relations using a few seed examples. However, not many of them have been used for the enrichment of Linked Data. Some of the semi-supervised relation extraction systems that have been attempted for the enrichment of the linked datasets are [8], SOFIE ([9]) and LEILA ([10]). A new class of approaches called the distantly supervised approaches ([11], [12], [13], [14], [15]) have been proposed in the recent past for extracting instances of existing relations. Distant supervision is the technique of utilizing a large number of known facts (a huge linked dataset like Freebase) for automatically labeling mentions of these facts in an unannotated text corpus, hence generating training data. A classifier is learnt based on this weakly labeled training data in order to classify unseen instances [11].

One common characteristic of all the above systems is that, existing relations in the dataset play a major role in the detection of new instances of the relations. There are a few more relation extraction systems proposed in the context of Linked Data enrichment, which work when the relation to be extracted does not exist in the dataset, but is specified in advance. For example, the authors of [16] have proposed a method exclusively to identify the “part-of” relation between the linked dataset entities.

All the above systems either concentrate on extracting more instances of existing relations or the system aims to find new relations, provided the system is given what the desired new relations are. In this paper, our goal is to find new relations between entities when the system does not know what the possible relations are. The authors of ([17]) also focus upon a very similar goal of determining if there is any possible connectivity between the entities of two datasets. However they detect the relation in the form of a path.

The work carried out by the authors of [18], in which they make use of the Open Information Extraction paradigm for the purpose of assessing and modifying existing relations’ values and also enriching the datasets with new properties, is the closest to our goal. They have collected 10000 web pages related to their dataset entities to form a corpus and then applied ReVerb [19] system on the corpus to extract relations and entity pairs. ReVerb is a state-of-the-art relation extraction system that works on a text corpus which is Part-Of-Speech tagged and noun phrase chunked. It returns a set of triples through two steps. First, given a sentence, for each verb in the sentence, ReVerb finds the longest sequence of words such that the sequence starts at that verb and satisfies the syntactic and lexical constraints to become a relation phrase. Syntactic constraint is checked by matching the obtained phrase against a pre-defined part-of-speech-based regular expression. Lexical constraint is checked by determining whether the obtained phrase is one among the relation phrases present in a large dictionary consisting of 1.7 million normalized valid relation phrases. Then, for each relation phrase identified in the previous step, the system finds the nearest noun phrase to the left of the phrase in the sentence such that it is not a relative pronoun, WH-term (interrogative words such as what, when, etc.), or existential “there”. Then it finds the nearest noun phrase to the right of phrase in the sentence and finally the left noun phrase, the relation phrase and the right noun phrase are returned as an extracted triple. Comparison of the proposed system with ReVerb has been described in Section 4.
3 Overview of the proposed approach

Given two datasets, the following two steps - Pattern-Discovery and Triple-Finding, are followed in order to extract relations between their entities. It is to be noted that the system is not aware of the possible relations between the concerned datasets. Hence such relations, which we intend to detect using our system are called new and arbitrary relations (as defined in [19]).

3.1 Dataset Preparation

The LOD cloud consists of several datasets across various domains which have been published in RDF format. A dataset in the LOD cloud typically comprises many classes, their entities and relations between the classes. In our work, any two classes present in the LOD cloud can be considered as the two input datasets (the words datasets and classes will be used interchangeably in this paper). The possible relations between the two datasets are not known a priori. Hence the system starts off by taking a sample of the cross-product of the two datasets. Different sampling methods such as simple random sampling\(^7\) and systematic random sampling\(^8\) were experimented during the implementation stage of our work in order to choose this fraction in an unbiased and unsupervised manner. Arranging the cross product pairs in a predefined manner with human supervision will threaten the randomness and the unbiased nature of the sampling technique and hence, the pairs were sorted in alphabetical order before sampling. Systematic random sampling with a sample size of 25% of the cross product was chosen empirically.

3.2 Pattern-Discovery phase

The Pattern-Discovery phase, given in Algorithm 1, takes the sample pairs and the class names of the input datasets D1 and D2 (class1 and class2 respectively) as input and outputs a set of prospective relations. The sample pairs are considered one by one and the labels (obtained from the rdfs:label property\(^9\)) of the entities in the pair are fed as a query to a popular web search engine. The top 100 web results are obtained. In each of these 100 results, sentences which have the labels of both the entities of the pair are identified and the words between the two entity labels form what is called a “pattern”. Identical patterns are considered as a single pattern with the frequencies added accordingly. In this manner, the full set of patterns along with their frequencies are obtained (Lines 2-13). These patterns are then clustered and clusters of frequency 1 are eliminated (Lines 14, 15).

\(^7\)http://study.com/academy/lesson/simple-random-samples-definition-examples.html

\(^8\)http://study.com/academy/lesson/systematic-random-samples-definition-formula-advantages.html

\(^9\)https://www.w3.org/TR/rdf-schema/#ch_label
Algorithm 1: PATTERN-DISCOVERY($sample, class_1, class_2$)

**Input:** $sample$, the set of sample pairs

$class_1$, the class name of 1st dataset; $class_2$, the class name of 2nd dataset

**Output:** $pRels$, the set of prospective relations discovered

1. $allPatterns \leftarrow \emptyset$
2. for each pair $(e_1, e_2) \in sample$
do
3. Feed (label of $e_1$ followed by label of $e_2$) to web search engine
4. for each result $res \in$ top 100 search results do
5. for each sentence $s_1 \in res$ do
6. if $s_1$ contains $e_1$ and $e_2$ then
7. $pattern \leftarrow$ words between $e_1$ and $e_2$ in $s_1$
8. update frequency of $pattern$
9. add $pattern$ to the set of $allPatterns$
end
end
end
end
12. $clusts \leftarrow (allPatterns$ grouped based on similarity$)$
13. $clusts \leftarrow clusts$ - (clusters of frequency equal to 1)
14. $repPatterns \leftarrow \emptyset$
15. for each cluster $cl \in clusts$ do
16. $repPattern \leftarrow repPatternSelection(cl)$
17. add $repPattern$ to the set of $repPatterns$
end
19. $pRels \leftarrow$ prospective relations obtained from $repPatterns$
20. return $pRels$

The clustering strategy adopted is as follows. The similarity value of each pattern $p$ with every other pattern is computed and the cluster which has the pattern having maximum similarity with $p$ is noted, provided this maximum similarity value is above the similarity threshold ($=0.5$ in our work). Then $p$ is added to the above noted cluster. If there is no cluster with patterns similar to $p$, then $p$ is added to a new cluster. The paraphrase detection method used to compute the similarity between two patterns is explained in Subsection 3.3.

Instead of passing the obtained clusters directly for further processing, we intend to pass a *representative* pattern from each of these clusters, as all the patterns in a single cluster are anyway similar to each other semantically. A pattern is defined as a representative pattern for a cluster if it has the maximum similarity with all the other patterns in the same cluster. Also, the pattern should have occurred many times in the corpus when compared to the other patterns, i.e its normalized frequency (its frequency divided by the total frequency of the cluster) should be high.
Detecting New Relations among Linked Datasets using Pattern Extraction

**Procedure** `repPatternSelection(cl)`

*Input*: `cl`, a cluster of patterns along with the pattern frequencies

*Output*: `repPattern`, the most representative pattern of the cluster `cl`

1. `sum ← sum of frequencies of all patterns in the cluster cl`
2. `maxPattern ← null`
3. `maxVal ← 0`
4. **for each** pattern `p1 ∈ cl` **do**
   5. `repVal ← 0`
   6. `asso ← 1`
   7. `normFreq ← (frequency of p1)/sum`
   8. **for each** pattern `p2 ∈ cl` **do**
      9. `asso ← asso × (similarity between patterns p1 and p2)`
   **end**
   11. `repVal ← asso × normFreq`
   12. **if** `(repVal >= maxVal)` **then**
      13. `maxVal ← repVal`
      14. `maxPattern ← p1`
   **end**
16. **end**
17. `repPattern ← maxPattern`
18. `return repPattern`

Hence in the procedure `repPatternSelection`, the value of the product of the pattern’s similarity with all the other patterns, multiplied by the normalized frequency of the pattern is calculated. The pattern which has the maximum value for this product becomes the representative pattern for that cluster.

The representative patterns for all the clusters are stored (Lines 17-20 of Algorithm 1). A set of prospective relations are formed from these representative patterns and passed as input to the Triple-Finding phase (Lines 21, 22). Prospective relation formation is explained in Subsection 3.4.

In case the initial set of patterns obtained (in Lines 2-13 of Algorithm 1) has a size > 500, in order to reduce the computation time involved in clustering such a huge set of patterns, we filter the patterns and clusters in the following manner: Patterns with frequency less than 3 are eliminated, and among the remaining patterns, only the top 80% of them are considered for clustering. After the clusters are obtained, again only the top 80% of the clusters, ordered on the total cluster frequency, are considered for representative pattern selection.

### 3.3 Paraphrase Detection

The method used to find if two patterns are similar is a modified form of the similarity detection technique proposed by Mihalcea et al in [20]. The mathematical formula underlying our paraphrase detection technique is given in equation (1).
Detecting New Relations among Linked Datasets using Pattern Extraction

\[
sim(T_1, T_2) = \frac{1}{2} \left( \frac{\sum_{w \in T_1} \maxSim(w, T_2)}{\text{len}(T_1)} + \frac{\sum_{w \in T_2} \maxSim(w, T_1)}{\text{len}(T_2)} \right)
\]

(1)

where, \(T_1\) and \(T_2\) represent the input text segments, \(\maxSim(w, T_2)\) refers to the similarity value of the word in \(T_2\) which is most similar to the word \(w\) in \(T_1\), \(\maxSim(w, T_1)\) is defined in the same way with respect to \(T_1\) and \(\text{len}(T_i)\) refers to the number of words in \(T_i\). In our implementation, the threshold value chosen to consider two segments \(T_1\) and \(T_2\) to be similar is 0.5.

Among the various word-to-word similarity measures used in [20], the measure “LESK” [21] which works for all combinations of parts of speech was adopted in our work while computing the values of \(\maxSim(w, T_2)\) and \(\maxSim(w, T_1)\) in equation (1).

In [20], the individual word-to-word similarity values were weighted using a word specificity measure. This word specificity had been incorporated in their system so that higher importance can be given to a semantic matching identified between two specific words such as “collie” and “sheepdog” when compared to a matching identified between words such as “get” and “become”. In our system, words such as “get” and “become” (any verb in general) have a good chance of occurring in the input text segments as we intend to extract relations between entities. Hence giving a low weightage to such words (as done in [20]) might give undesired results. Therefore, we have removed the word specificity weights.

3.4 Keyword Extraction and Prospective Relation Formation

As we are considering all words between the two given entities in the web search results as a “pattern”, the set of patterns extracted from the web corpus might contain many irrelevant patterns. Also, a pattern might contain random characters in addition to meaningful words. Hence, there is a need for a way to spot only the keywords of these patterns, that are meaningful to represent the relation between the two given datasets. The approach adopted to recognize keywords is to consider each word of the pattern, that is not a function word\(^{10}\) and to determine whether that word is associated with either of the class names. By associated, we mean one of the following: the word can be an adjective of the class name (or) the word can be a verb or a noun which appears in the context of the class name. The Datamuse API\(^{11}\) has been used for this purpose.

Given a word, the API has provisions to determine the word’s synonyms, related verbs and nouns, adjectives, adjacent words and homophones. The API uses Google Books Ngrams dataset\(^{12}\) to build the language model at its back-end.

If a word in the pattern is not a function word and is associated with either of the two class names, then we consider it as a keyword in our work. If the pattern consists of a single word, which happens to be a function word, then it is considered as a keyword.

\(^{10}\) [http://www.sequencepublishing.com/academic.html]

\(^{11}\) [http://www.datamuse.com/api/]

\(^{12}\) [http://storage.googleapis.com/books/ngrams/books/datasetsv2.html]
In this manner, all keywords are extracted from the given patterns. In order to form the prospective relations from these keywords, we append the keyword with the adjacent preposition (if present in the pattern) or the class name (again, if present adjacent to the keyword in the pattern). Also, if the keyword happens to be an adjective, then the noun which is qualified by that adjective (determined using the Stanford CoreNLP toolkit [22]) is appended to it. For example, while intending to find the relation between the datasets dance-forms and states, if the pattern obtained from the web corpus is “classical indian dance practice at shakti school of dance in” then the keywords obtained using the Datamuse API are: classical; indian; practice; school. Following our algorithm, by appending the prepositions, class names and qualified nouns to these keywords, the prospective relations are: classical practice; indian practice; dance practice at; school of.

3.5 Triple-Finding phase

Algorithm 2: \textsc{Triplet-Finding}(p\textit{Rels}, D_1, D_2, \textit{class} _1, \textit{class} _2)

\textbf{Input:} \textit{pRels}, the set of prospective relations  
\hspace{1em} \textit{D}_1 \text{ the 1st dataset; } \textit{D}_2 \text{ the 2nd dataset}  
\hspace{1em} \textit{class} _1 \text{ class name of } \textit{D}_1 \text{; } \textit{class} _2 \text{ class name of } \textit{D}_2  

\textbf{Output:} \textit{assertions}, the set of final assertions

\begin{algorithmic}
\State \textit{assertions} \leftarrow \emptyset
\For {each entity $e_1 \in \textit{D}_1$}
\For {each relation $\textit{rel} \in \textit{pRels}$}
\State form \textit{query} as $e_1$ followed by $\textit{rel}$
\State Feed \textit{query} to web search engine
\For {each sentence $s_1 \in$ the text of the top 60 search results}
\If {($s_1$ contains no other entity apart from $e_1$) and ($s_1$ contains only a single entity $e_2 \in \textit{D}_2$)}
\State \textit{phr} \leftarrow \text{words in } s_1 \text{ between } e_1 \text{ and } e_2
\If {number of words in \textit{phr} $\leq 10$}
\State \textit{otherWords} \leftarrow \text{all the words in } \textit{phr} \text{ apart from } \textit{rel}
\If {($\textit{phr}$ contains $\textit{rel}$) and (each word in \textit{otherWords} is a function word or $\textit{class}_1$ or $\textit{class}_2$ or adjacent word of either class names)}
\State add the triple $(e_1, \textit{rel}, e_2)$ to \textit{assertions}
\EndIf
\ElseIf {similarity between \textit{phr} and \textit{rel} $\geq 0.5$}
\State add the triple $(e_1, \textit{rel}, e_2)$ to \textit{assertions}
\EndIf
\EndIf
\EndIf
\EndFor
\EndFor
\EndFor
\State \textbf{return} \textit{assertions}
\end{algorithmic}
The Triple-Finding phase (given in Algorithm 2) takes the set of prospective relations and the two datasets in order to determine which among the prospective relations are actually present between the entities of the two datasets, i.e., we extract the final triples (instances of the prospective relations) through this phase thus performing enrichment of the linked datasets.

The entities in the dataset D1 are considered one by one and each of them is concatenated with each relation \( rel \) in the prospective relations set. This concatenated string is given as a query to a web search engine (Lines 2-5). In the top 60 search results obtained, sentences which contain the entity under consideration from dataset D1 (let us call this entity, \( e_1 \)) are identified. Such sentences are checked to find if any entity from dataset D2 (let us call this entity, \( e_2 \)) is present in them (Lines 6,7). While looking for entity \( e_2 \) from D2 to be present in the sentence, it should be noted that multiple entities from dataset D2 are not allowed to be present in the same sentence. Hence, the word “single” has been included in Line 7 of Algorithm 2. Also, whether another entity apart from \( e_1 \) from the dataset D1 is present in the sentence, is checked in Line 7 of Algorithm 2. The intuition behind these two additional checks is to avoid the wrong pairing of entities. For example, consider the sentence, “Tamil Nadu’s classical dance Bharata Natyam is less difficult than the classical dance of Andhra Pradesh, Kuchipudi”. If the algorithm doesn’t have these two additional checks, we might get the assertion (Bharata Natyam, classical dance, Andhra Pradesh) which is actually wrong. In order to avoid such situations which reduce the precision value, we check if multiple entities of the same class are present in the sentence under consideration. If so, we discard such a sentence.

After determining the entity \( e_2 \), the words between \( e_1 \) and \( e_2 \) in the sentence are extracted to form the relation phrase (Line 8). Let us call this relation phrase \( phr \). \( phr \) intuitively represents the relation between the entities \( e_1 \) and \( e_2 \) in that sentence. As a long phrase would mean that the relation is too specific to those two entities alone, this phrase \( phr \) needs to be within a certain limit of length and hence this condition is included in Line 9 of Algorithm 2, with length limit as 10 (fixed empirically). Then the phrase \( phr \) needs to be checked with the prospective relation \( rel \) and if both have some resemblance to each other, we can assert \( e_1 \) and \( e_2 \) to be related by that prospective relation. In order to perform this resemblance checking (Lines 10-16), first we check if \( phr \) contains that prospective relation. If yes, then we check if every other word of \( phr \) is either a function word or one of the class names or one of the class names’ adjacent words (determined using the Datamuse API). If this condition is true, then we add the triple \( (e_1, rel, e_2) \) to the set of assertions (Lines 11-13). If this condition is not true, then the similarity between \( phr \) and the prospective relation is checked, again using the paraphrase detection technique explained in Subsection 3.3. If the similarity value crosses the threshold of 0.5, \( (e_1, rel, e_2) \) is added to the set of assertions (Lines 14-16).

The same procedure is repeated with the prospective relations and entities of the dataset D2. In this case, the entities of D1 (instead of D2) are searched for in the web results and assertions are obtained in the same manner. These two sets of assertions form the final output triples of our system. Note that the outer-join kind of approach we adopted for finding triples is computationally more efficient than checking if a prospective relation exists between every pair of entities from the two datasets.
4 Evaluation

The proposed system has been implemented in Java 1.7 and the Linked Data entity sets have been obtained by accessing the corresponding SPARQL endpoints. We conducted experiments for the evaluation of our approach on two types of dataset pairs:

- Datasets between which properties already exist in the LOD cloud, and we try to rediscover the properties through our system. These experiments have been performed mainly to verify the effectiveness of our approach. The dataset pairs (rivers and states), (billionaires and companies) fall under this category.
- Datasets between which properties do not exist in the LOD cloud and our system adds to the LOD cloud by discovering new relations between them. The dataset pairs (dance-forms and states), (vegetables and diseases) belong to this type.

Recall that the Pattern-Discovery phase generates a set of prospective relation names through pattern extraction from the web corpus and the Triple-Finding phase is used to extract the instances of these relations. Ground truth is required for the purpose of evaluating the relation instances obtained. However, there might be a few prospective relations where the ground truth is not found in the LOD cloud and so it is manually generated. There might also be cases where ground truth can neither be found in the LOD cloud nor be generated manually. Hence for evaluation purposes, we have divided the prospective relations obtained at the end of Pattern-Discovery phase into two types: “Relations retrieved for which ground truth is available” and “Additional relations retrieved”. The two measures used for the evaluation of the system, namely, precision and recall, have been calculated only for the relations for which ground truth is available. Precision and recall measures have been defined in the following manner: let $n_1$ be the number of pairs related by one of the prospective relations, as per the available ground truth; let $m$ of these $n_1$ pairs be found in the triples produced by our system and $n_2$ be the total number of triples containing the prospective relations generated by the system, then recall = $m / n_1$ and precision = $m / n_2$.

4.1 Dance-forms and States

We found that between the YAGO classes DancesOfIndia\(^\text{13}\) and StatesAndTerritoriesOfIndia\(^\text{14}\), there are no property links in the LOD cloud. Hence we intended to find the relations between these two classes. After eliminating few entities which actually do not indicate a dance-form or state of India, such as “Seven Sister States”, the dataset D1 of dance-forms comprised of 88 entities and the dataset D2 of Indian states comprised of 32 entities. The patterns and prospective relations generated by the system at the end of the Pattern-Discovery phase are given in Table 1.

As mentioned in the beginning of this section, the relations given in the prospective relations column of Table 1 can be split into two types, namely, “Relations retrieved for which ground truth is available” and “Additional relations retrieved” as given in Table 2. For relations under the first type, since the ground truth is not available in the

\(^{13}\)http://dbpedia.org/class/yago/DancesOfIndia

\(^{14}\)http://dbpedia.org/class/yago/StatesAndTerritoriesOfIndia
 LOD cloud, and since Wikipedia articles form the main source for DBpedia dataset, the precision and recall values have been computed with respect to the ground truth prepared manually using the corresponding Wikipedia pages of each dance-form. We obtained a precision value of 0.84 and a recall of 0.72. Table 2 also shows the relations between the datasets obtained by ReVerb. We give a detailed account of the comparison of our system with ReVerb in Subsection 4.5.

| Patterns extracted | Prospective relations identified |
|-------------------|---------------------------------|
| classical indian dance practice at shakti school of dance in | classical practice; indian practice; dance practice at; school of; |
| legislative assembly election, 2011 teatr en | |
| is like theyyam in north | |
| is unique to central travancore, comprising the pathanamthitta-alappuzha-kottayam belt of | |
| is the most popular folk dance of | popular dance; folk dance |
| is very popular in | popular in |
| arts of | arts of |
| (malayalam: ) is a traditional martial dance of | traditional dance |

Table 1: Patterns and Relations extracted for Dance-forms and States datasets

| Relations retrieved for which ground truth is available | Additional relations retrieved | Relations from ReVerb |
|---------------------------------------------------------|-------------------------------|-----------------------|
| popular dance; popular in; folk dance; arts of; traditional dance | classical practice; indian practice; dance practice at; school of; | is a must-see in; takes over; traces its roots back in; is very popular in; is an art form very popular in; is a colourful ancient art form in; is a masked dance of |

Table 2: Results for Dance-forms and States datasets

4.2  Rivers and States

Though the DBpedia dataset has an object property “state” which represents the states through which a river flows, by manual inspection we found that this property is not present for all the rivers. There seemed to be some scope for enrichment and hence these two classes were taken into consideration to know the kind of results produced by
our system. Few entities from the concept RiversOfIndia of YAGO\(^{15}\) such as “Interstate River Water Disputes Act”, which do not indicate an actual river of India have been removed manually in order to form the dataset D1 (of size 31). Similarly, few entities from the concept StatesAndTerritoriesOfIndia of YAGO such as “Seven Sister States” have been removed to form the dataset D2 (of size 32). The cross product is of size 992 and hence the set of sample pairs is of size 248 (1/4th of 992 pairs obtained through systematic random sampling). Due to space constraints, the Patterns and Relations Table for rivers and states datasets, corresponding to Table 1 has not been presented. The prospective relations produced by our system have been divided into two types and are given in Table 3. Recall has been computed with respect to the ground truth given by the DBpedia dataset, considering the “state” property of the rivers. Recall obtained is 0.89. Precision value calculated with respect to the DBpedia dataset’s ground truth is only 0.54. However, when examined manually, one could see that there were many correct triples which were not present in the DBpedia dataset. Hence we decided to calculate the precision value by gathering ground truth from Wikipedia articles. The final precision value is 0.86.

| Relations retrieved for which ground truth is available | Additional relations retrieved | Relations from ReVerb |
|--------------------------------------------------------|-------------------------------|-----------------------|
| flows through; in                                       | waters to; coastal rivers; village; district of; headed by; basin; sea; southwest through; south; cruise; upper catchment area; falls in; beautiful temples; post | ***no relations were extracted*** |

Table 3: Results for Rivers and States datasets

4.3 Billionaires and Companies

In the case of the above two experiments, the dataset pairs were such that, one can intuitively be sure that a relation will definitely exist between the two datasets, i.e an Indian dance-form would have definitely originated from one of the Indian states and a river of India would definitely flow through one or few of the Indian states. We wanted to check the effectiveness of our system on a dataset pair where the certainty of existence of a relation is less intuitive and less guaranteed. Hence the concepts ConglomerateCompaniesOfIndia\(^{16}\) and IndianBillionaires\(^{17}\) of the YAGO dataset were chosen as the input datasets D1 and D2 for this experiment. D1 is of size 50 and D2 is of size 58 giving a cross product of 2900 entity pairs. Sample pairs set is of size 725=(2900/4). Results are given in Table 4.

\(^{15}\)http://dbpedia.org/class/yago/RiversOfIndia  
\(^{16}\)http://dbpedia.org/class/yago/ConglomerateCompaniesOfIndia  
\(^{17}\)http://dbpedia.org/class/yago/IndianBillionaires
Relations retrieved for which ground truth is available | Additional relations retrieved | Relations from ReVerb
--- | --- | ---
founded; founder; ceo; director at; chairman; owned | bank; associate of; market; private industry; conglomerate; business; member of; china; magnate; party | ***no relations were extracted***

Table 4: Results for Billionaires and Companies datasets

In DBpedia, 4 relations (namely, foundedBy, founder, owner and keyPerson) existed between these 2 concepts. Since our system generated slightly different relations from those that are present in DBpedia, it was not possible to compare our output with DBpedia’s triples. Hence ground truth was manually generated using Wikipedia pages. Precision and recall have been calculated only for the relations under the first column in Table 4 as ground truth could not be determined for the additional relations. A precision of 0.77 and a recall of 0.88 have been obtained. Hence it can be concluded that the proposed approach is largely effective that it can determine relations even for two datasets between which the existence of relations cannot be assured a priori.

### 4.4 Vegetables and Diseases

As mentioned in Section 1, since it would be interesting to determine the possible relations between a given set of vegetables and a set of non-contagious diseases, this experiment was conducted. The vegetables class from Freebase dataset\(^\text{18}\) formed the input dataset D1. As the disease class of Freebase\(^\text{19}\) is too huge (12646 instances), and also, since we wanted to consider only non-contagious diseases, we chose a set of aging-associated diseases\(^\text{20}\) as the input dataset D2. D1 is of size 165 and D2 is of size 37. Results are given in Table 5.

| Significant relations retrieved | Additional relations retrieved | Relations from ReVerb |
| --- | --- | --- |
| cured; prevent; prevention of | against colon; juice; patients with; bone; consumption; cancer; breast; products; dark green leafy vegetables; garlic; human; is very effective for; slows; slow the onset of; could help prevent; can help slow progression of; reverse; stop; has been used to treat; is to eat; causes; was beneficial to; can help fight |

Table 5: Results for Vegetables and Diseases datasets

As already discussed in Section 1, few relations that one might intuitively guess to be present between a set of vegetables and a set of diseases are “cures” and “prevents”. From Table 5, it can be seen that both these relations have been captured through the...

---

\(^\text{18}\) https://www.freebase.com/user/jamie/food/vegetable

\(^\text{19}\) https://www.freebase.com/medicine/disease

\(^\text{20}\) http://dbpedia.org/resource/Category:Aging-associated_diseases
proposed system. Since the ground truth for this dataset pair is not available in the LOD cloud, and the manual calculation of ground truth is also infeasible (most of this topic is under research i.e there are no standard results published yet as to which vegetable can definitely prevent or cure which disease), precision and recall values could not be reported for this experiment. However, at the end of the Triple-Finding phase, 95 triples were extracted through this experiment.

The performance of all the experiments have been consolidated and presented in Table 6.

| Datasets                  | Precision | Recall |
|---------------------------|-----------|--------|
| Dance-forms and states    | 0.84      | 0.72   |
| Rivers and states         | 0.86      | 0.89   |
| Billionaires and Companies| 0.77      | 0.88   |
| Vegetables and Diseases   | N/A       | N/A    |

Table 6: Consolidated Results

4.5 Comparison with ReVerb

The authors of [18] have applied ReVerb, on a web corpus of 10000 pages and extracted three types of relations: Relations which had both the arguments in the LOD dataset (17219 relations), relations which had a single argument in the LOD dataset (198487 relations) and relations that did not associate objects from the LOD dataset (290714 relations). Among these three types, those relations falling under the first type are similar to the relations we extract through our method. Hence it would be valuable to make a comparison between the number of correct relations obtained through our method and the number of correct relations (out of the 17219 relations) of their method. However, the authors of [18] do not give any measure regarding the number of correct relations out of these 17219 relations, thus making such a kind of direct comparison infeasible. Hence we decided to compare both the systems with respect to their corresponding pattern extraction phases alone, i.e we compared the results of the Pattern-Discovery phase of our approach with the results of applying ReVerb on the corpus used by us. The relations extracted through ReVerb have been given in Tables 2 - 5. Among the four dataset pairs that we have used for our evaluation, ReVerb did not produce any relations for two of them (rivers and states, billionaires and companies). A possible reason for not obtaining any relation through ReVerb might be that, ReVerb enforces syntactic and lexical constraints on the relation obtained. For example, in the case of the dataset pair rivers and states, the proposed approach obtained the pattern “which flows through the sunderbans” from which the prospective relation “flows through” was extracted during the Pattern-Discovery phase. But when ReVerb was applied to the same web corpus, the phrase “which flows through the sunderbans” would not have satisfied the syntactic and lexical constraints and hence would have been eliminated from being produced in the final output. On the other hand, for the dataset pair of vegetables and diseases, ReVerb is able to produce a few more meaningful relations than the proposed system. When we analysed the reasons for this behaviour, we found that, though our system could initially extract patterns such as “slow the onset of”, “causes”, “has been used to
Detecting New Relations among Linked Datasets using Pattern Extraction

5 Conclusion and Future Work

The central idea behind this paper is to propose a completely automated and unsupervised technique to identify possible arbitrary relations between two entity sets of Linked Data. For this purpose, we have built a system, whose first phase connects the paradigms of pattern-based information extraction and paraphrase detection into a unified framework for discovering prospective relations between the datasets. These prospective relations then pave way for the actual enrichment of the linked datasets in the second phase of the system. The results gathered reveal the potential of the system to unearth many interesting relations between a given pair of datasets thus leading to the growth of a relationship-rich LOD. The proposed system has also produced several additional relations for which ground truth could not be collected and hence could not be evaluated. On close manual inspection of these additional relations, we could find that they contained a few meaningful relations and a few completely irrelevant relations. Elimination of these irrelevant relations forms a major part of our future work.

References

1. The AI Behind WATSON - The Technical Article, http://www.aaai.org/Magazine/Watson/watson.php
2. Gunaratna, K., Lalithsena, S., Sheth, A.: Alignment and dataset identification of linked data in semantic web. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery 4(2), 139–151 (2014)
3. Bach, N., Badaskar, S.: A review of relation extraction. Tech. rep., Language Technologies Institute, Carnegie Mellon University (2007)
4. Agichtein, E., Gravano, L.: Snowball: Extracting relations from large plain-text collections. In: Proceedings of the Fifth ACM Conference on Digital Libraries. pp. 85–94. DL ’00, ACM, New York, NY, USA (2000)
5. Etzioni, O., Cafarella, M., Downey, D., Popescu, A.M., Shaked, T., Soderland, S., Weld, D.S., Yates, A.: Unsupervised named-entity extraction from the web: An experimental study. Artif. Intell. 165(1), 91–134 (Jun 2005)
6. Etzioni, O., Banko, M., Soderland, S., Weld, D.S.: Open information extraction from the web. Commun. ACM 51(12), 68–74 (Dec 2008)
7. Carlson, A., Betteridge, J., Kisiel, B., Settles, B., Jr., E.R.H., Mitchell, T.M.: Toward an architecture for never-ending language learning. In: Fox, M., Poole, D. (eds.) AAAI. AAAI Press (2010)
8. Welty, C., Fan, J., Gondek, D., Schlaikjer, A.: Large scale relation detection. In: Proceedings of the NAACL HLT 2010 First International Workshop on Formalisms and Methodology for Learning by Reading, pp. 24–33. FAM-LbR ’10, Association for Computational Linguistics, Stroudsburg, PA, USA (2010)
Detecting New Relations among Linked Datasets using Pattern Extraction

9. Suchanek, F.M., Sozio, M., Weikum, G.: Sofie: A self-organizing framework for information extraction. In: Proceedings of the 18th International Conference on World Wide Web. pp. 631–640. WWW ’09, ACM, New York, NY, USA (2009)
10. Kasneci, G., Ramanath, M., Suchanek, F., Weikum, G.: The yago-naga approach to knowledge discovery. SIGMOD Rec. 37(4), 41–47 (Mar 2009)
11. Krause, S., Li, H., Uszkoreit, H., Xu, F.: Large-scale learning of relation-extraction rules with distant supervision from the web. In: The Semantic Web - ISWC 2012. Lecture Notes in Computer Science, vol. 7649, pp. 263–278. Springer Berlin Heidelberg (2012)
12. Mintz, M., Bills, S., Snow, R., Jurafsky, D.: Distant supervision for relation extraction without labeled data. In: Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2 - Volume 2. pp. 1003–1011. ACL ’09, Association for Computational Linguistics, Stroudsburg, PA, USA (2009)
13. Aprosio, A.P., Giuliano, C., Lavelli, A.: Extending the coverage of dbpedia properties using distant supervision over wikipedia. In: Hellmann, S., Filipowska, A., Barriere, C., Mendes, P.N., Kontokostas, D. (eds.) NLP-DBPEDIA@ISWC. CEUR Workshop Proceedings, vol. 1064. CEUR-WS.org (2013)
14. Assis, P., Casanova, M.: Distant supervision for relation extraction using ontology class hierarchy-based features. In: Presutti, V., Blomqvist, E., Troncy, R., Sack, H., Papadakis, I., Tordai, A. (eds.) The Semantic Web: ESWC 2014 Satellite Events, Lecture Notes in Computer Science, vol. 8798, pp. 467–471. Springer International Publishing (2014)
15. Nguyen, T.V.T., Moschitti, A.: End-to-end relation extraction with distant supervision from external semantic repositories. In: Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: Short Papers - Volume 2. pp. 277–282. HLT ’11, Association for Computational Linguistics, Stroudsburg, PA, USA (2011)
16. Jain, P., Hitzler, P., Verma, K., Yeh, P.Z., Sheth, A.P.: Moving beyond sameas with plato: Partonomy detection for linked data. In: Proceedings of the 23rd ACM Conference on Hypertext and Social Media. pp. 33–42. HT ’12, ACM, New York, NY, USA (2012)
17. Pereira Nunes, B., Dietze, S., Casanova, M.A., Kawase, R., Fetahu, B., Nejdl, W.: The Semantic Web: Semantics and Big Data: 10th International Conference, ESWC 2013, Proceedings, chap. Combining a Co-occurrence-Based and a Semantic Measure for Entity Linking, pp. 548–562. Springer, Berlin, Heidelberg (2013)
18. Antonis Koukourikos, V.K., Vouros, G.A.: Towards enriching linked open data via open information extraction. In: In Workshop on Knowledge Discovery and Data Mining meets Linked Open Data (KnowLOD). pp. 37–42 (2012)
19. Etzioni, O., Fader, A., Christensen, J., Soderland, S., Mausam, M.: Open information extraction: The second generation. In: Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence - Volume One. pp. 3–10. IJCAI’11, AAAI Press (2011)
20. Mihalcea, R., Corley, C., Strapparava, C.: Corpus-based and knowledge-based measures of text semantic similarity. In: Proceedings of the 21st National Conference on Artificial Intelligence - Volume 1. pp. 775–780. AAAI’06, AAAI Press (2006)
21. Banerjee, S., Pedersen, T.: An adapted lesk algorithm for word sense disambiguation using wordnet. In: Proceedings of the Third International Conference on Computational Linguistics and Intelligent Text Processing. pp. 136–145. CICLing ’02, Springer-Verlag, London, UK, UK (2002)
22. Manning, C.D., Surdeanu, M., Bauer, J., Finkel, J., Bethard, S.J., McClosky, D.: The Stanford CoreNLP natural language processing toolkit. In: Association for Computational Linguistics (ACL) System Demonstrations. pp. 55–60 (2014)