1N2-OS-39a-1

Knowledge Discovery from Linked Data

Lihua Zhao[*1] Natthawut Kertkeidkachorn[*2] Ryutaro Ichise[*2*1]

[*1] National Institute of Advanced Industrial Science and Technology, Tokyo, Japan
[*2] National Institute of Informatics, Tokyo, Japan

Linked Data has been increasing rapidly by publishing machine readable structured data. DBpedia and YAGO are cross-domain data sets, which provide semantic knowledge of things. Although both data sets contain millions of entities, there are still missing knowledge exist in each data set. In this paper, we analyze graph patterns of Linked Data entities to discover missing knowledge in the data sets. We apply word embedding method with traditional ontology matching method to integrate heterogeneous ontologies. By querying Linked Data with integrated ontology, we can discover missing knowledge in the data sets so that we can automatically extend the Linked Data.

1. Introduction

Linked Data contains various machine-readable interlinked data resources and provides useful information among the entities [Bizer et al. 2009]. An entity is represented as a collection of Resource Description Framework (RDF) triples in the form of <subject, predicate, object>, where the former two are Uniform Resource Identifiers (URIs) and the latter is either an URI or a value.

Although many linked data sets have been published with millions of entities, there are still missing or incorrect knowledge exist. DBpedia and YAGO are two large data sets derived from multilingual Wikipedia articles and there are millions of interlinked same entities. These interlinked same entities are essential resources to discover similar properties (also called predicates) between two data sets. There are two principal types of properties: object properties that describe the relationship between two entities and datatype properties that link entities with data values.

In this paper, we will introduce how to match similar properties between DBpedia and YAGO. We apply graph-based method for mapping similar object properties and apply string-based similarity measures and GRU based Recurrent Neural Network (RNN) method to map similar datatype properties.

2. Similar Property Matching

Linked Data consist of RDF triples that describe relations between entities and attributes of entities. The relation between entities is defined as object properties (owl:ObjectProperty) and the attribute is defined as datatype property (owl:DatatypeProperty). In this paper, we use OP to represent object properties (excluding *owl:sameAs*) and use DP to represent datatype properties. The same entities in different data sets are interlinked by *owl:sameAs*, which are important resources to explore matching OPs and DPs.

2.1 Object Property Matching

The main steps of discovering similar OPs are as follows:

Step 1: Randomly retrieve some interlinked entities.

Step 2: Retrieve preliminary similar OP pairs by checking the graph patterns of interlinked entities and memorize the co-occurrence of OP pairs.

Step 3: Retrieve frequent pairs that co-occur more than a predefined threshold.

Step 4: Group similar OPs based on the frequent OP pairs.

In Step 1, we randomly chose some interlinked entities. In Step 2, we retrieve preliminary similar OP pairs by analyzing graph patterns. If there are four RDF triples $<Ent_1$, *owl:sameAs*, $Ent_2>$, $<Ent_1$, OP_1, $Ent_3>$, $<Ent_2$, OP_2, $Ent_4>$, and $<Ent_3$, *owl:sameAs*, $Ent_4>$, we can infer that $OP_1 = OP_2$. In Step 3, we need to filter out OP pairs that occur occasionally and then group similar OPs in the last step.

2.2 Datatype Property Matching

Since the same values of subjects do not have *owl:sameAs* links, we need a different approach to discover similar datatype properties from interlinked entities. Three aspects are considered to match similar DP pairs: 1) co-occurrence of DP pairs, 2) similarity of values, and 3) similarity of the terms in DP pairs. Therefore, the similarity between DP pairs is calculated as follows:

$$Sim(dp_1, dp_2) = conf\_co(dp_1, dp_2) * \Big(\alpha * strSim(val_1, val_2) + \beta * weSim(dp_1, dp_2)\Big) \quad (1)$$

where the weights $\alpha + \beta = 1$, $conf\_co(dp_1, dp_2)$ is the co-occurrence confidence of DP pairs dp_1 and dp_2, $strSim(val_1, val_2)$ is string-based similarity of val_1 and val_2 that correspond to dp_1 and dp_2, and $weSim(dp_1, dp_2)$ is word embedding based similarity for the terms of DP pairs.

2.2.1 Co-Occurrence Confidence of Datatype Property Pairs

Simply using the co-occurrence of DPs cannot guarantee they have high confidence values. Therefore, we also consider the occurrence of each single DP in the data sets. The co-occurrence confidence of two DPs ($conf\_co(dp_1, dp_2)$) is defined as follows:

$$conf\_co(dp_1, dp_2) = \frac{1}{2} * \Big(\frac{occur(dp_1, dp_2)}{occur(dp_1)} + \frac{occur(dp_1, dp_2)}{occur(dp_2)}\Big)$$

where $occur(dp_1, dp_2)$ is the co-occurrence of dp_1 and dp_2, $occur(dp_1)$ and $occur(dp_2)$ represent the occurrence of dp_1 and dp_2 in the linked data sets, respectively.

Contact: 2-4-7, Aomi, Koto-ku, Tokyo, Japan, 135-0064, lihua.zhao@aist.go.jp,
2-1-2 Hitotsubashi, Chiyoda-ku, Tokyo, Japan, 101-8430, {natthawut, ichise}@nii.ac.jp

2.2.2 String-based Similarity of DP Values

The values are mainly categorized into three types: number, date, and string. For the DP pairs with the values of number and date, we use exact matching method, which returns 1 if they are the same and returns 0 otherwise. For the string values, we use three string-based similarity measures, namely, JaroWinkler distance, EditDistance (Levenshtein), and n-gram.

$$strSim(val_1, val_2) = \lambda * JaroWinker(val_1, val_2) + \mu * EditDistance(val_1, val_2) + \nu * ngram(val_1, val_2) \quad (2)$$

where $\lambda + \mu + \nu = 1$.

2.2.3 Word Embedding based Similarity for DP Terms

The terms of properties are mostly meaningful individual words or compound words, e.g., "birthDate" or "dateOfBirth". Continuous Bag-of-Words (CBOW) model is one of the most popular neural network models for learning distributed representations of words, in other words, embedding words in a vector space (word2vec) [Mikolovet al. 2013].

Since distributed representation of a compound word cannot be directly represented as a sum of individual vectors, we apply Recurrent Neural Networks based approach, specifically, GRU-based approach to represent compound words. For a given sequence of input $x_1, x_2, ..., x_n$, RNN model learns the current latent state with the input data at time t and the previous latent state at time $t-1$. Then the current latent state is used to predict the output. However, RNN cannot capture long-term dependencies due to the gradients tend to either vanish or explode. The gated recurrent unit (GRU) is able to handle long-term dependencies and perform better than using traditional tanh unit [Chung et al. 2014]. Therefore, we use GRU-based RNN approach for training compound words.

3. Experiments

We used DBpedia (2015-10) and YAGO 3 dump datasets, which are cross-domain data derived from Wikipedia articles. There are 1,099 object properties and 1,734 datatype properties of DBpedia in our Virtuoso RDF triple store. We used 75 properties from YAGO 3, which have confidence higher than 0.9, among which 31 are object properties and 44 are datatype properties.

3.1 Discover Object Property Triples

We randomly selected 10% of interlinked entities for graph-based approach and got 18 groups of similar object properties from DBpedia and YAGO. For example, *yago:isMarriedTo*, *db-onto:spouse*, and *db-prop:spouse* are grouped together and *yago:isCitizenOf*, *db-onto:nationality*, and *db-prop:nationality* are grouped together.

We used the 18 groups of similar OPs to discover missing Object Property Triples (OPTs). Table 1 shows the number of missing OPTs we retrieved for each group of similar object properties in DBpedia and YAGO, respectively. The quality of discovered OPTs depends on the accuracy of existing knowledge in the Linked Data and the confidence of matching similar OPs.

3.2 Discover Datatype Property Triples

The similarity of datatype properties are calculated according to the co-occurrence of DP pairs, string-based similarity of values in DPTs, and word embedding based similarity of the terms in DP pairs. In this experiment, we filtered out DP pairs with co-occurrence less than 100 times. In Equation 1, we set both α and β as 0.5, and set λ, μ, and ν as $\frac{1}{3}$ in Equation 2. We used GRU-based RNN approach to model the terms of DPs and calculated word-embedding based cosine similarity between the terms.

Property Group	DBpedia	YAGO
isMarriedTo	25,799	6,568
hasCapital	6,538	2,739
isCitizenOf	33,196	77,178
wasBornIn	254,574	841,270
diedIn	94,246	231,353

Table 1: The number of discovered missing OPTs.

DBpedia		YAGO	
db-onto:height	11,947	yago:hasHeight	117,239
db-onto:deathDate	31,024	yago:diedOnDate	18,940
db-onto:deathYear	30,530	yago:diedOnDate	
db-prop:dateOfDeath	34,443	yago:diedOnDate	
db-prop:deathDate	251,682	yago:diedOnDate	
db-prop:dateOfBirth	78,298	yago:wasBornOnDate	26,956
db-onto:birthDate	58,555	yago:wasBornOnDate	
db-onto:birthYear	70,833	yago:wasBornOnDate	

Table 2: Number of DPTs discovered in DBpedia and YAGO.

In total, we discovered 112 DP pairs with word2vec and 145 DP pairs with GRU-based approach. Since there are too many discovered similar DP pairs including infobox properties, we mainly focus on non-infobox DP pairs with similarity higher than 0.5 to discover missing Datatype Property Triples (DPTs) in the data sets, Table 2 shows some of the non-infobox datatype properties and the number of discovered DPTs from DBpedia and YAGO. Some of the YAGO datatype properties match to several DBpedia DPs because of the ontology heterogeneity problem.

4. Conclusion

In this paper, we proposed graph-based method to match similar object property pairs and GRU-based approach to match similar datatype properties from DBpedia and YAGO. With the matching property pairs, we discovered missing knowledge from both data sets. By using both Semantic Web technology and NLP methods, we could successfully discover missing knowledge from the interlinked data sets that can be used for extending Linked Data.

Acknowledgements

This work was partially supported by NEDO (New Energy and Industrial Technology Development Organization).

References

[Bizer et al. 2009] Christian Bizer, Tom Heath and Tim Berners-Lee. Linked Data - The Story So Far. *International Journal on Semantic Web and Information Systems*, 5(3):1–22, 2009.

[Chung et al. 2014] Junyoung Chung, Çaglar Gülçehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *Computing Research Repository (CoRR)*, abs/1412.3555, 2014.

[Mikolovet al. 2013] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *CoRR*, abs/1301.3781, 2013.