# Supervised and Unsupervised Approaches to the Ontology-Based Disambiguation of JSON Documents

Chinmay Choudhary, Matthias Nickles, and Colm O'Riordan

College of Engineering and Informatics
National University of Ireland, Galway

**Abstract.** This paper proposes and evaluates certain supervised and unsupervised approaches to Named Entity Disambiguation in JSON documents, for linking of all ambiguous JSON objects to their most appropriate candidate DBpedia ontology classes. We achieve this by taking into account knowledge about the hierarchal structure of the document and two kinds of scores, namely Sibling Relatedness and Parental Relatedness, along with textual similarity between a class and the object indicated by the Textual Similarity score.

**Keywords:** JSON disambiguation,Linked Data, Ontology Mapping, Named Entity Disambiguation

## 1 Introduction

JSON (JavaScript Object Notation) is a lightweight language-independent data-interchange format which is widely used for sharing and extraction of data on internet. Syntax of JSON allows data to be presented in such a format that it is human readable, yet can be easily parsed and generated by computers, by adopting a hierarchy of objects with each consisting of a key as human-readable text. The paper presents an approach to the disambiguation of real-world JSON documents by linking of keys having ambiguous value-text referring to real-world entities to appropriate DBpedia classes to which the particular entity belongs out of all its candidate classes. This linking could eventually be used to decode a given JSON document representing information about popular entities to extract such information autonomously therefore possessing utility in the field of web data mining. Another potential use case would be the automated generation of documents in JSON-LD [7] format (which is a formal syntax to serialize Linked Data in JSON), from ambiguous normal JSON documents.
Data on the Semantic Web is often represented using a framework for indicating relationships of particular entities in a specific domain which is conceptualized as an ontology. The data itself is represented using Resource Description Framework (RDF) triples. RDF is a machine-readable format which makes the data extractable using (SPARQL) queries. Ontology mapping is the process of linking of concepts within any two given ontologies representing similar data from

two distinct heterogeneous sources, such that both concepts (being identified by unique individual identifiers within respective ontologies) categorize same type of real-world entities. One such ontology system is DBpedia which presents entire information available on Wikipedia in structured form as *Linked data* by classifying all Wikipedia articles as hierarchy of classes or concepts based on the type of entities that these articles describe with each class having a fixed set of properties. This structured information includes attributes about each Wikipedia page such as *Title, Hyperlinks, description* etc. as RDF triples accessible through DBpedia SPARQL server or as downloadable datasets.
The hierarchal structure of JSON documents can be informally described as an ontology-like system with objects having two types of relations namely *Parent-child* and *Sibling* elaborated in Section 4. RDF triples describing the structure of the JSON document presented as Example 1 are listed in Table 1.
Example 1:

```
"Country":{"Name": "Germany"
  "Capital":"Berlin"
  "Gaint-Companies":{"Auto":["BMW","Volkswagen","Mercedes"]}}
```

Thus the problem of disambiguation of objects of a JSON document with ambiguous value-texts is addressed within the paper as an ontology mapping problem, by collectively linking all the objects (including both ambiguous and non-ambiguous) with most appropriate candidate DBpedia ontology classes simultaneously utilizing a new proposed mapping approach based on the fundamentals of general Named Entity Disambiguation (NED) while taking into account hierarchal structure of JSON document. NED is the process of linking ambiguous name-mentions within a text document to appropriate real-world entities in a knowledge base. Most common Knowledge-base is Wikipedia for which each article becomes single Entity. The entire NED process comprises of three major steps including the recognition of ambiguous name-mentions within a document, identification of candidate entities for each such ambiguous name-mention and disambiguation of these name-mentions by linking each one with most appropriate respective entity out of all the candidates, each being distinct broad research area within itself. This paper describes research work applied on JSON document to implement final step of NED process through a new approach.
Section 3 outlines the research problem. Section 4 describes the proposed approach while sections 5 and 6 elaborates the testing and evaluation of it.

| **RDF triples** |
|---|
| [country parent name], [country parent capital], [country parent Giant-companies], [Giant-companies parent Auto], [Name sibling capital], [Capital sibling Giant-Companies], [Name sibling Giant-companies], [name child country], [capital child country], [Giant-companies child country], [Auto child Giant-companies], [Capital sibling Name], [Giant-Companies sibling Capital], [Giant-companies sibling capital ] |

**Table 1.** RDF triples describing the hierarchal structure of Example 1

## 2 Related Work

Named Entity Disambiguation is a significant area of research which exists since quite a long time. Early works within this field include proposals of individual-linking approaches such as [3], [18], [10], [8] which link each name-mention individually based on similarity between context of it within document and description of entity. On the other hand modern approaches belong to collective-linking approach category which includes approaches that link all name-mentions within single document simultaneously by considering mutual relationships between various entities being referred in a single document along with textual similarity between name-mentions and their respective entities. Collective-linking approaches can further be classified based on overall process adopted, as supervised approaches such as [14], [22], [9] and unsupervised approaches such as [11], [21], [19]
Various proposed ontology mapping approaches can be classified into three major categories namely Similarity-based, Reasoning-based and Learning-based approaches. Approaches such as [6], [23], [20], [15], [12] are the examples of similarity-based approaches that perform mapping based on linguistic and contextual similarity of text representing components of two ontologies, while [16], [2], [4], [17] are examples of reasoning-based approaches that address the problem as a logic-inference problem after being provided an initial set of mapping manually, with the goal of inferring final set of mapping. Finally learning-based approaches utilize machine learning to compute final mapping. Some examples of popular tools developed for ontology mapping are COMA++, CODI, FALCON-AO, PRIOR+, LILY [1].

## 3 Problem Definition

The research reported in this paper proposes approaches for collectively linking keys of JSON objects within a given document with ambiguous value-text referring to a real-world popular entity, with appropriate class of DBpedia ontology (`http://mappings.dbpedia.org/server/ontology/classes/`) to which that entity belongs, based on fundamentals of general collective NED. Thus for the particular application domain, keys of JSON objects with ambiguous value-text act as name-mentions while ontology classes as an entities within collection of all DBpedia classes forming the knowledge base (KB). Proposed approaches accomplish final task of NED which involves identification of most suitable entities to be linked to all name-mentions simultaneous out of respective candidate entities of each, thus can only be applied on real-world JSON documents with all the ambiguous value-texts being demarcated with a list of candidate classes for each being identified beforehand.
Owing to the hierarchal structure any two objects within single document can have either one of the three types of relationships namely Sibling, Parent-Child and Un-related, thus enabling entire structure to be represented as set of RDF triples. Two objects can be considered to have sibling relationship if both of

them have another distinct JSON object as a common immediate superior (can be considered as common parent) within the documents hierarchal structure. Whereas for two given objects $O_1$ and $O_2$ within a single document, $O_1$ will be considered as parent of $O_2$ if $O_1$ is immediate superior of it within overall document hierarchy (in which case $O_1$ and $O_2$ would have parent-child relationship). Pairs of objects having Sibling Relationships as well as Parent-child relationships within document given as Example 1 are listed as Table 2.

| **Pairs of objects (represented as keys) within Example 1 having Sibling Relationship** | **Pairs of objects (represented as keys) within Example 1 having Parent-child Relationship** |
|---|---|
| – Name & Capital<br>– Capital & Giant-Companies<br>– Name & Giant-Companies | – Country & Name<br>– Country & Capital<br>– Country & Giant-Companies<br>– Giant-Companies & Auto |

**Table 2.** Pairs of objects within Example 1 having both categories of relationships possible

In our scenario, we assume that an entire hierarchal JSON document structure is represented as a collection of specific RDF triples and it can also be depicted as a large connected graph called *main-graph* comprising of two types of nodes namely *Object-node* representing JSON objects and *Class-node* representing particular DBpedia ontology class, with each object-node being connected to at least one class-node. The graph would also consist of three kinds of edges described as follows.

1. *Sibling edge:* Connects two class-nodes and indicates both represented classes being candidates of two distinct JSON objects having sibling relationship within document hierarchy and is weighted with Sibling Relatedness (SR) score.
2. *Parental edge:* Connects two class-nodes and indicates both represented classes being candidates of two distinct JSON objects having Parent-child relationship and is weighted with Parental Relatedness (PR) score.
3. *Candidate edge:* Connects an object-node and class-node indicating it to be a candidate and is weighted with *Textual Compatibility* score.

The graph in figure 1 depicts hierarchal structure of JSON document given as Example 1. The purpose of research presented within this paper is to propose methods for computation of all three kinds of weighting scores (namely SR, PR and TC) as well as computation of evidence weight of a sub-graphs of desired structure extracted from main-graph, such that the sub-graph with maximum evidence weight is the appropriate collective link of the document. Problem can be defined mathematically as follows.

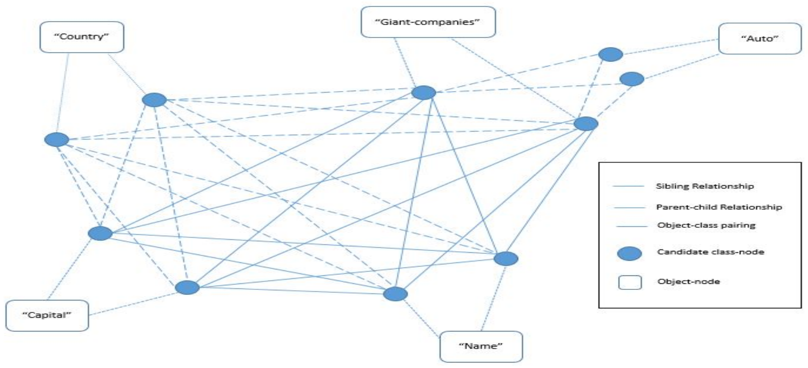

**Fig. 1.** Main-graph representing hierarchy of Example 1

A main-graph M representing hierarchy of a given JSON document consists of a set of object-nodes O and a set of class-nodes C. Each member of set O consists of a set of other sibling nodes $S_o$, children nodes $P_o$ and a set of candidate class nodes $C_o$.
For all $m_i \in M$.

$$Candidate(c_i, o) = \begin{cases} 1 \text{ if } c_i \in C_o \\ 0 \text{ otherwise} \end{cases} \tag{1}$$

$$Sibling(o_i, o_j) = \begin{cases} 1 \text{ if } o_i \in S_{o_j} and\ o_j \in S_{o_i} \\ 0 \qquad\quad \text{otherwise} \end{cases} \tag{2}$$

$$Parent(o_i, o_j) = \begin{cases} 1 \text{ if } o_j \in P_{o_i} \\ 0 \text{ otherwise} \end{cases} \tag{3}$$

For all $c_i, c_j \in C$

$$W(c_i, o_j) = \begin{cases} TC(c_i, o_j) \text{ if } Candidate(c_i, o_j) = 1 \\ \qquad 0 \qquad\qquad\quad \text{otherwise} \end{cases} \tag{4}$$

$$W(c_i, c_j) = \begin{cases} SR(c_i, c_j) \text{ if } Candidate(c_i, o_i) = 1;\ Candidate(c_i, o_i) = 1;\ Sibling(o_i, o_j) = 1 \\ PR(c_i, c_j) \text{ if } Candidate(c_i, o_i) = 1;\ Candidate(c_j, o_j) = 1;\ Parent(o_i, o_j) = 1 \\ \quad 0 \qquad\qquad\qquad\qquad\qquad \text{otherwise} \end{cases} \tag{5}$$

Here SR is *Sibling Relationship Score*, PR is *Parental Relationship Score* and TC is *Textual Compatibility Score.*

Let S be set of sub-graphs of main-graph.

$$CollectiveLink = max_s(EvidenceWeight(s)) \quad For\ all\ s \in S \tag{6}$$

The objective of research is to formulate and test methods for the computation of values of TC, PR, SR and Evidence Weight within equations 4, 5 and 6 respectively. Section 3 proposes methods for computation of SR, PR, TC and Evidence Weights.

## 4 Proposed Approaches

This section describes and proposes methods to compute three kinds of scores (SR, PR and TC) as well as methods for computation of Evidence Weight of a sub-graph of main graph utilizing these scores.

**Sibling Relatedness** Score between two ontology classes $C_1$ and $C_2$ referred to as SR($C_1$,$C_2$) or SR($C_2$,$C_1$) indicates the chances of both classes being linked to any two objects having sibling relationships within a given JSON document. In other words it indicates general chances that instances of both class would form two distinct pieces of information about a single common entity (common parent). Based on this logic SR value between two given classes can be computed by analysing the properties sharing a unique ontology class as common domain and with either one of the two classes in consideration as range, based on intuition that the instances of these classes would have higher chances of representing individual properties of a particular instance of common domain within a JSON document. For two classes A and B, the value of SR can be computed irrespective of any document, as commonness of properties indicated by Google Normalization Distance [5] through Equation 7, with $|a|$ and $|b|$ being number of properties with classes A and B as ranges respectively while $|a \cap b|$ is the number of properties having range as either A and B and a common set of distinct domains, and T represents total number of classes within entire DBpedia ontology.

$$SR(A,B) = 1 - \frac{\log(\max(|a|,|b|)) - \log(|a \cap b|)}{\log T - \log(\min(|a|,|b|))} \tag{7}$$

**Parental Relatedness score** between two ontology classes $C_1$ and $C_2$ with $C_1$ being parent, referred to as PR($C_1$,$C_2$), indicates the chances of both classes being linked to JSON objects within same document such that object linked to $C_1$ is immediate superior (parent) of object linked to $C_2$ within overall document hierarchy. In other words it indicates general chances that instances of $C_2$ would form a distinct piece of information about an instance of class $C_1$. Thus Value of PR($C_1$,$C_2$) can be computed as a factor of the number of properties having $C_1$ as domain as well as $C_2$ as range with respect to the total number of properties having either domain as $C_1$ or range as $C_2$. The Parental Relatedness Score between two ontology classes A and B is computed simply as a probability of a property having class having A as domain and B as range out of all properties having either A as domain and B as range by applying Equation 8 with $|a|$ being total number of properties with domain as A and $|b|$ being total number of properties with range as class B whereas $|a \cap b|$ representing number of properties with domain as A and range as B. It is based on intuition that higher the chances of an instance of B representing distinct piece of information about an instance of class A implies higher chances of A being parent of B within a JSON document. One is added to both numerator and denominator as Laplace's smoothing.

$$PR(A,B) = 2\frac{|a \cap b| + 1}{|a| + |b| + 1} \tag{8}$$

It is important to note that Sibling and Parental Relatedness scores between any two classes are general and are therefore applicable for any given JSON document (not document specific) whereas Textual Compatibility is between an object and ontology class and thus, is document specific.
**Textual Compatibility** score between a given JSON object and a DBpedia ontology class indicates the chances of the object to be representing information related to an instance of the particular ontology class, based on fundamentals of general NED [18]. For an object O, the key of it and its sub-objects within the entire document hierarchy form context of it (with key of O being name-mention), whereas for ontology class C labels of all its properties form its entity-description (with C being the candidate entity). Textual Compatibility Score between C and O is computed through N-Gram similarities [13] between keys within the context and labels within the entity-description.
For a given key-text (k) and a property label (p), the N-Gram Similarity ($SS_N$) for any value of N is computed by formula shown in Equation 9.

$$SS_N = 2\frac{|t_k \cap t_p|}{|t_k| + |t_p|} \tag{9}$$

Here $|t_k|$ is the total number of N-grams extracted out of string k and $|t_p|$ is the total number of N-grams extracted out of string P. ($|t_k \cap t_p|$) is the total number of common N-grams extracted out of both k and p.
Apply Equation 9 values of $SS_N$ can be computed for values of N ranging from 3 to n, where n is the length of smaller string out of k and p. The Overall Similarity Score (SS) between key-text k and property label p can be computed by applying the heuristic formula stated as Equation 10.

$$SS(k,p) = 2^n * SS_n + 2^{(n-1)} * SS_{n1} + ... + 2^3 * SS_3 \tag{10}$$

Similarity Score between key-text k and entire DBpedia ontology class C is given by formula stated as Equation 11.

$$SS(k,C) = \max_{\forall pi \in P}(SS(k,pi)) \tag{11}$$

Here P is the set of strings consisting of labels of all properties of class C with the DBpedia mapping.
Let K be the set consisting of n key-texts including keys of JSON object O and of all its sub-objects within the hierarchy given by $k_1, k_2, ...., k_n$. Textual Similarity Score between O and ontology class C is given Equation 12.

$$TC(O,C) = \frac{\sum_{i=1}^{n} SS(k_i, C)}{n} \tag{12}$$

As explained in Section 3 verall collective linking of given JSON document is computed through maximum evidence weighted sub-graph of desired structure extracted from main-graph representing entire document. The structure of sub-graph should be such that it consists of all *object-nodes* existing within main-graph with each one being linked to only single *class-node*. This paper proposes

an *unsupervised* and a *supervised* methods to compute this *Evidence Weight (EW)* through three attributes namely Sum of all Textual Compatibility Score values ($\sum TC$), Sum of all Sibling Relatedness Score values ($\sum SR$) and Sum of all Parental Relatedness Score values ($\sum PR$).
Let a given sub-graph S consists of O as a set of all object-nodes and C as a set of all class-nodes with each object-node being linked to single class-node. Two approaches (supervised and unsupervised) for computation of Evidence weight of S (EW(S)) is explained as Sections 4.1 and 4.2.

### 4.1 Unsupervised/Direct Method

This method assumes the importance of all three attribute values in determining the suitable DBpedia classes for all JSON objects collectively as equal. For the same reason the value of EW is given by simply the addition all three attributes. For the given sub-graph S the value of Evidence Weight is determined by Equation 13, for all $c_a, c_b \in C, c_p, c_c \in C$ and all $o_n \in O; c_n \in C$.

$$EW(S) = \sum_{\forall Parent(o_n,c_n)=1} TC(o_n, c_n) + \sum_{\forall Sibling(c_a,c_b)=1} SR(c_a, c_b) + \sum_{\forall Parent(c_p,c_c)=1} PR(c_p, c_c) \tag{13}$$

### 4.2 Supervised Method

This method assigns the importance of all three attribute values in determining the suitable DBpedia classes for all JSON objects collectively through parameters. These parameters can be learnt from a set of appropriate sub-graphs of the main graph identified as a positive or negative example by applying machine learning algorithms such as Logistic Regression. A sub-graph can be considered as positive example if all the nodes of it representing JSON objects with ambiguous value-text are linked to correct DBpedia ontology classes to which the Wikipedia entity link being referred by their value-text belong. For the given sub-graph S the value of evidence weight is determined by Equation 14 with parameters $\theta_1$, $\theta_2$ and $\theta_3$ being learnt from Logistic Regression.

$$EW(S) = \theta_1 * (\sum TC(o_n, c_n)) + \theta_2 * (\sum SR(c_1, c_2)) + \theta_3 * (\sum PR(c_p, c_c)) \tag{14}$$

## 5 Experiments

Testing of proposed approaches for final task of collective disambiguation of all ambiguous objects, would require a set of JSON documents with all their ambiguous objects being demarcated and at-least two candidates of each being identified. Due to unavailability of sufficient sized dataset innovative method is adopted for very close predication of legitimacy of proposed hypothesis in real-world indirectly. Both approaches are preliminarily evaluated using a single large JSON document JSON document (`http://www.carqueryapi.com/api/0.`

`3/?callback=?&cmd=getMakes`) consisting of two relevant pieces of information about 155 auto-mobile manufactures namely *Name of Company* and *Country of origin*, thus having 310 ambiguous value-texts. After demarcating all ambiguous JSON objects and identifying three candidate entities (two incorrect and one correct links) for all objects (including both ambiguous and non-ambiguous) manually, a main-graph representing entire document is created, from which all possible sub-graphs of desired structure (with each object-node being lined to only single class-node) are extracted and labelled as positive if all ambiguous objects are linked to respective correct class-nodes and negative otherwise. Details of final-datasets used for training and testing of both approaches is summarized as Table 3.

| | Examples in Dataset for testing of unsupervised approach | Examples in Dataset for training of supervised approach | Examples in Dataset for testing of supervised approach |
|---|---|---|---|
| **Total** | 1863 | 1118 | 745 |
| **Negative** | 1656 | 993 | 663 |
| **Positive** | 207 | 125 | 82 |

**Table 3.** Information about datasets

For both unsupervised and supervised methods, two distinct m x 1 and m x 3 dimensional dataset matrices along with single boolean matrix of size m x 1 representing labels, are generated respectively with m being the total number of examples, and are utilized to formulate two distinct probability matrices with respect to each. Data-matrix corresponding to unsupervised method consists of total weight of each sub-graph being a single example within dataset, with examples being re-grouped together such that ones with same non-ambiguous pairs forming one group, thereby forming 207 such groups. Probabilities of each such sub-graph is computed with respect to other sub-graphs within the same group through equation 15.

Let EW(s) be the total evidence weight of sub-graph s belonging to group of sub-graphs S then probability of s being selected is simply calculated as

$$P(s) = \frac{EW(s)}{\sum_{\forall s_i \in S} EW(s_i)} \tag{15}$$

However for supervised method probability is computed by applying sigmoid function after learning parameters within Equation 14 through *Logistic Regression*, thus no such re-grouping is required. For the purpose of training, dataset matrix corresponding to supervised approach is split in the ratio of 60% and 40% to be utilized as training and testing dataset.

Evaluation of proposed approaches involve making predictions for all examples being indicated by probability values (as probability matrix) after assuming a threshold value and eventually comparing the predicted Boolean matrix with actual one to compute values of various indicators of accuracy.

## 6 Results

Entire evaluation process involves computation of four indicators namely Accuracy, Precision, Recall and F-Score. Both unsupervised and supervised methods are evaluated for various values of threshold distributed uniformly through-out entire range from 0 to 1. Figures 2 and 3 demonstrate values of Accuracy and F-Score indicators achieved with different values of threshold whereas Table 4 compares maximum values of all four indicators achieved for both methods.

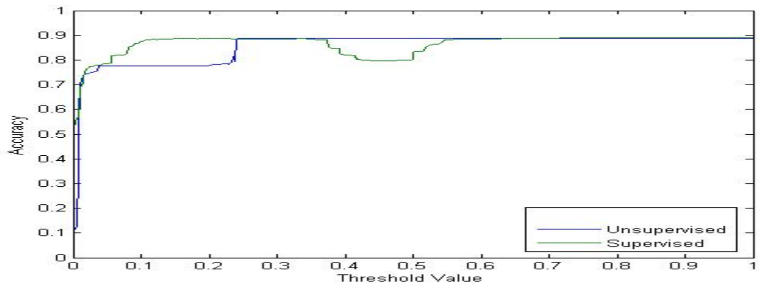

**Fig. 2.** Values of Accuracy achieved against various thresholds

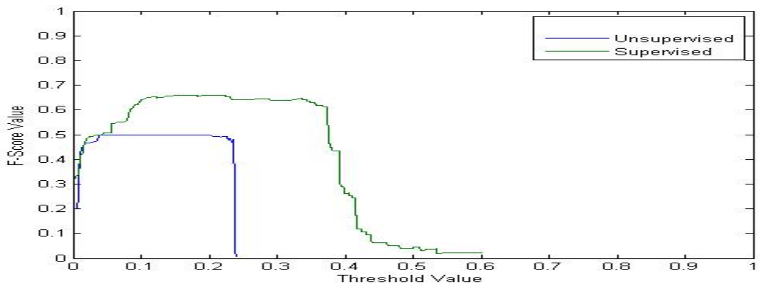

**Fig. 3.** Values of F-Score achieved against various thresholds

As explained in Section 4 supervised method requires training through Logistic Regression to learn parameters $\theta_1$, $\theta_2$ and $\theta_3$. Values of these parameters learnt through training datasets (first 60% of datasets) are enlisted as Table

| Method | Maximum Accuracy | Maximum Precision | Maximum Recall | Maximum FScore | Threshold Probability of Maximum F-Score |
|---|---|---|---|---|---|
| Unsupervised | 0.86 | 0.35 | 0.9 | 0.5 | 0.2 |
| Supervised | 0.88 | 0.49 | 1.0 | 0.66 | 0.36 |

**Table 4.** Comparison of results achieved by both unsupervised and supervised approaches

| **Parameters** | $\theta_1$ | $\theta_1$ | $\theta_1$ |
|---|---|---|---|
| **Values** | 0.0620 | -0.6083 | -4.3920 |

**Table 5.** Values of parameters obtained

5. The obtained results suggest that supervised method performed better than unsupervised method on all four indexes, as is evident from the table. Though there is no state of the art existing approach available to compare these results and draw inferences, these results can act as benchmark for future research.

## 7 Conclusion

This paper introduced *Parental Relatedness* and *Sibling Relatedness* score between two ontology classes which are candidates of two distinct JSON objects, computed through analysis of their shared properties. Utilizing these scores as well as Textual Compatibility score between an object and its candidate class, two approaches to collective disambiguation of given real-world JSON documents have been proposed (an unsupervised and a supervised approaches), based on general NED. Results obtained by testing of these approaches on limited semi-manually created dataset are used to compare the performance of both. Future research involves more exhaustive testing of both approaches on elaborate datasets.

## References

1. 50 ontology mapping and alignment tools. `http://www.mkbergman.com/1769/50-ontology-mapping-and-alignment-tools/`. Accessed: 2017-08-30.
2. Alexander Borgida and Luciano Serafini. Distributed description logics: Assimilating information from peer sources. *J. Data Semantics*, 1:153–184, 2003.
3. Razvan C Bunescu and Marius Pasca. Using encyclopedic knowledge for named entity disambiguation. In *Eacl*, volume 6, pages 9–16, 2006.
4. Silvana Castano, Alfio Ferrara, Davide Lorusso, Tobias Henrik Näth, and Ralf Möller. Mapping validation by probabilistic reasoning. In *European Semantic Web Conference*, pages 170–184. Springer, 2008.
5. Rudi L Cilibrasi and Paul MB Vitanyi. The google similarity distance. *IEEE Transactions on knowledge and data engineering*, 19(3), 2007.

6. Graham Cormode and S Muthukrishnan. The string edit distance matching problem with moves. *ACM Transactions on Algorithms (TALG)*, 3(1):2, 2007.
7. Douglas Crockford. The application/json media type for javascript object notation (json). 2006.
8. Mark Dredze, Paul McNamee, Delip Rao, Adam Gerber, and Tim Finin. Entity disambiguation for knowledge base population. In *Proceedings of the 23rd International Conference on Computational Linguistics*, pages 277–285. Association for Computational Linguistics, 2010.
9. Xianpei Han, Le Sun, and Jun Zhao. Collective entity linking in web text: a graph-based method. In *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*, pages 765–774. ACM, 2011.
10. Jian Hu, Lujun Fang, Yang Cao, Hua-Jun Zeng, Hua Li, Qiang Yang, and Zheng Chen. Enhancing text clustering by leveraging wikipedia semantics. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 179–186. ACM, 2008.
11. Hongzhao Huang, Yunbo Cao, Xiaojiang Huang, Heng Ji, and Chin-Yew Lin. Collective tweet wikification based on semi-supervised graph regularization. In *ACL (1)*, pages 380–390, 2014.
12. Glen Jeh and Jennifer Widom. Simrank: a measure of structural-context similarity. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 538–543. ACM, 2002.
13. Grzegorz Kondrak. N-gram similarity and distance. In *String processing and information retrieval*, pages 115–126. Springer, 2005.
14. Sayali Kulkarni, Amit Singh, Ganesh Ramakrishnan, and Soumen Chakrabarti. Collective annotation of wikipedia entities in web text. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 457–466. ACM, 2009.
15. Jayant Madhavan, Philip A Bernstein, and Erhard Rahm. Generic schema matching with cupid. In *vldb*, volume 1, pages 49–58, 2001.
16. Christian Meilicke, Heiner Stuckenschmidt, and Andrei Tamilin. Repairing ontology mappings. In *AAAI*, volume 3, page 6, 2007.
17. Christian Meilicke, Heiner Stuckenschmidt, and Andrei Tamilin. Reasoning support for mapping revision. *Journal of logic and computation*, 19(5):807–829, 2008.
18. Rada Mihalcea and Andras Csomai. Wikify!: linking documents to encyclopedic knowledge. In *Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*, pages 233–242. ACM, 2007.
19. Ali M Naderi. Unsupervised entity linking using graph-based semantic similarity. 2016.
20. Gonzalo Navarro, Ricardo A. Baeza-Yates, Erkki Sutinen, and Jorma Tarhio. Indexing methods for approximate string matching. *IEEE Data Eng. Bull.*, 24(4):19–27, 2001.
21. Xiaoman Pan, Taylor Cassidy, Ulf Hermjakob, Heng Ji, and Kevin Knight. Unsupervised entity linking with abstract meaning representation. In *HLT-NAACL*, pages 1130–1139, 2015.
22. Lev Ratinov, Dan Roth, Doug Downey, and Mike Anderson. Local and global algorithms for disambiguation to wikipedia. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 1375–1384. Association for Computational Linguistics, 2011.
23. Esko Ukkonen. Approximate string-matching over suffix trees. In *Combinatorial Pattern Matching*, pages 228–242. Springer, 1993.