Discovering and Disambiguating Named Entities in Text

Johannes Hoffart
Max Planck Institute for Informatics
Saarbrücken, Germany
jhoffart@mpi-inf.mpg.de
Supervisor: Gerhard Weikum

ABSTRACT

Disambiguating named entities in natural language texts maps ambiguous names to canonical entities registered in a knowledge base such as DBpedia, Freebase, or YAGO. Knowing the specific entity is an important asset for several other tasks, e.g. entity-based information retrieval or higher-level information extraction. Our approach to named entity disambiguation makes use of several ingredients: the prior probability of an entity being mentioned, the similarity between the context of the mention in the text and an entity, as well as the coherence among the entities. Extending this method, we present a novel and highly efficient measure to compute the semantic coherence between entities. This measure is especially powerful for long-tail entities or such entities that are not yet present in the knowledge base. Reliably identifying names in the input text that are not part of the knowledge base is the current focus of our work.

Categories and Subject Descriptors

H.3.1 [**Information Storage and Retrieval**]: Content Analysis and Indexing; I.2.7 [**Natural Language Processing**]: Text Analysis

Keywords

Entity Disambiguation, Entity Discovery, Knowledge Bases, Semantic Relatedness, Entity Relatedness

1. INTRODUCTION

Texts on the Web like news articles or blog posts contain mentions of named entities such as people, organizations, and places. These names are often ambiguous. Take for example the sentence "Paris stole Helen from her husband, the king of Sparta." Finding the correct meaning for each name is the goal of named entity disambiguation. In the example sentence, this means identifying "Paris", "Helen", and "Sparta" as names, and assigning *Paris* and *Helen from Troy*, both characters in Greek mythology, as well as *Sparta*, as the entities.

It is immediately clear that this requires a repository of known entities to disambiguate to, a collection of all potential meanings of names. Wikipedia is the most often used entity catalogue for general world knowledge about entities. In recent years, knowledge bases have been derived from Wikipedia, making the data in Wikipedia available in a structured and clean format. Academic endeavors include `dbpedia.org` and `yago-knowledge.org`. The most prominent commercial one is `freebase.com`, which is part of Google. These knowledge bases are the most frequently used ones in research on named entity disambiguation. All of them contain millions of persons, organizations, songs, products, etc.

The knowledge about which entities are mentioned in a text is useful for many tasks. Higher-level information extraction tasks, for example fact extraction [10], rely on the correct disambiguation of the entity arguments. More recently, entity disambiguation has been used as part of answering natural language questions over structured knowledge bases [15, 14]. Entities are now also an important building block in Google's current Web search. They are shown in the search auto completion, and the results are only about the specific entity selected. Their slogan for this new approach, "Things, not Strings", succinctly describes the recent trend towards entities.

It should be noted that although our research focuses on disambiguating named entities in text, the method is also applicable to the task of entity resolution in more structured data, e. g. to resolve entities that are part of the linked data cloud [3] but not yet linked. If the entities are accompanied by textual descriptions, the method can be used out of the box. Yet even if the entities are only part of the linked data graph, the neighborhood in the graph can be transformed into a reasonably good context. This is an open research problem that we would like to address in the future.

2. DISAMBIGUATING ENTITY NAMES

The most common scenario for disambiguating named entities has a natural language text as input. The method should identify all mentions of named entities with their canonical meaning in the entity repository. In most approaches, the tasks of recognizing the names and disambiguating the correct meaning are tackled one after the other. First, names are recognized based on surface or syntax features. Then, all the names are disambiguated. This is the approach taken by our method, called AIDA [6].

SIGMOD'13, June 22–27, 2013, New York, New York, USA.

	Our Methods					Competitors	
	sim-k	prior sim-k	prior sim-k coh	KORE	KORE_{LSH}	prior	Kul
Macro Acc.	76.53	80.71	**82.00**	80.59	81.22	71.24	76.74
Micro Acc.	76.09	79.57	**82.31**	80.71	81.76	65.84	72.87

Table 1: Accuracy on 216 test documents of the CoNLL-YAGO dataset in %

Name Recognition There are several clues for identifying a name in a text. In English, all names start with a capital letter. This by itself is a very good indicator that a word (or a sequence of words) is actually a name. However, it is not sufficient, as the heuristic for example fails at the beginning of sentences, which always start with upper case. This is the case for "Paris" in our Greek example sentence. To correctly recognize all names, state of the art methods employ conditional random fields. The most prominent recognizer is the Stanford NER [2], which we are using in our AIDA system. We focus our research on the task of disambiguating named entities.

Entity Disambiguation The most important insights for a correct disambiguation are the following. The name probably refers to the most prominent entity. This probability can be estimated using the Wikipedia link structure, counting how often a certain anchor text refers to a given entity in Wikipedia. Thus we know that when the name "Paris" is found in a text, it generally refers to the French capital. There has to be really good contextual evidence that suggests otherwise. Take again the example sentence "Paris had to steal Helen from her husband, the king of Sparta." The words surrounding "Paris" in this sentence suggest that "Paris" should refer to a person from the Greek mythology. This kind of contextual evidence is the second feature for our disambiguation mechanism. Every entity in our knowledge base is associated with a textual description in the form of keyphrases that is compared to the surrounding context of the name. The more the context and the description overlap, the better the indication for the entity. The keyphrases are harvested from Wikipedia, comprising all link anchors, category names, and titles of incoming pages. By this, *Paris (mythology)* is associated with the keyphrase "Sparta's king". In the example, "king of Sparta" would be a good indicator given this keyphrase, however it is only present partially and in wrong order.

Our measure still captures this by matching the phrases' individual words and rewarding their proximity in an appropriate score. To this end we compute, for each keyphrase, the shortest window of words that contains a maximal number of words of the keyphrase. We refer to this window as the phrase's *cover*. By this rationale, the score of partially matching phrase q in a text is set to:

$$score(q) = z \left(\frac{\sum_{w \in cover} weight(w)}{\sum_{w \in q} weight(w)} \right)^2$$

where $z = \frac{\#\ matching\ words}{length\ of\ cover(q)}$ and $weight(w)$ is either the mutual information weight between keyphrase word w and the entity or the collection-wide IDF weight of the keyphrase word w. Note that the second factor is squared, so that there is a superlinear reduction of the score for each word that is missing in the cover.

For the similarity of a mention m to candidate entity e, this score is aggregated over all keyphrases of e and all their partial matches in the text, leading to the *similarity score*

$$simscore(m, e) = \sum_{q \in KP(e)} score(q)$$

To evaluate the accuracy of our method, we annotated a set of nearly 1,400 Reuters newswire articles. The CoNLL 2003 shared task [12] annotated 35,000 names in these articles. We disambiguated all of them with the correct entity registered in the YAGO2 knowledge base [5], creating the biggest gold standard dataset available for entity disambiguation research. We evaluated our methods with respect to two measures: Micro accuracy, dividing the number of correctly disambiguated mentions by the total number of mentions. The macro accuracy averages this fraction over all documents. The evaluation shows that the prior probability (*prior*) and the keyphrase based similarity (*sim-k*) already achieve good results, especially when combined (*prior sim-k*) see Table 1.

3. ENTITY COHERENCE

In some cases, matching keyphrases to the context is not enough, especially when the contextual evidence is very limited. To deal with these cases our methodology enforces coherence among the disambiguated entities, preferring candidates that go well together. In the example sentence "Paris met Helen.", Paris and Helen from Troy are a better fit than Paris Hilton and Helen, Georgia, a small U.S. city. The idea that this example demonstrates is that commonly, entities occurring in the same text are at least somewhat related. To capture this idea, we introduce a notion of coherence among all entities of a single input text in our method. Of course, this kind of information can only be used to its full extend when all names are disambiguated at the same time. Our method does a joint disambiguation over all names at once using a greedy graph algorithm. The graph representation this algorithm runs on is an undirected graph with two kinds of nodes: mentions and entities. The edges between mention-entity nodes are weighted using the combined prior popularity and keyphrase based similarity method introduced above. The entity-entity edges are weighted using one of several possible coherence measures, detailed below. The entity disambiguation graph for an example sentence is shown in Figure 1.

The goal of the graph algorithm is to compute a dense subgraph, ideally containing exactly one entity per mention node, in effect disambiguating the mentions. We face two main challenges here. The first is how to specify a notion of density that is best suited for capturing the coherence of the resulting entity nodes. The seemingly most natural approach would be to measure the density of a subgraph in terms of its total edge weight. Unfortunately, this will not work robustly for the disambiguation problem. A few entity nodes with very high weights of incident edges could dominate the solution, so the approach could work for prominent

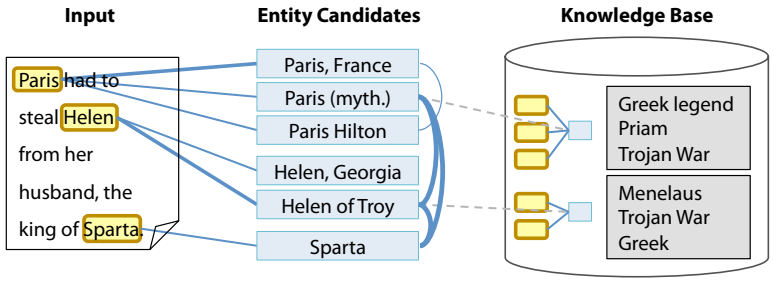

Figure 1: Entity disambiguation graph example

targets, but it would not achieve high accuracy for the long tail of less prominent and more sparsely connected entities. We need to capture the weak links in the collective entity set of the desired subgraph. For this purpose, we define the *weighted degree* of a node in the graph to be the total weight of its incident edges. We then define the density of a subgraph to be equal to the minimum weighted degree among its nodes. Our goal is to compute a subgraph with maximum density. The second critical challenge that we need to face is the computational complexity. Dense-subgraph problems are almost inevitably NP-hard as they generalize the Steiner-tree problem. Hence, exact algorithms on large input graphs are infeasible. To address this problem, we adopt and extend an approximation algorithm of [13] for the problem of finding strongly interconnected, size-limited groups in social networks. The algorithm starts from the full mention-entity graph and iteratively removes the entity node with the smallest weighted degree. Among the subgraphs obtained in the various steps, the one maximizing the minimum weighted degree will be returned as output. To guarantee that we arrive at a coherent mention-entity mapping for all mentions, we enforce each mention node to remain connected to at least one entity.

3.1 Wikipedia-Based Entity Relatedness

The most commonly used measure of entity relatedness for coherence is based on the Wikipedia link graph. The idea is that entities occurring together often in a Wikipedia article are related. A measure built on this assumption, introduced by Milne and Witten [9], works very well. In our original work, we used this measure successfully for coherence, improving the accuracy of the AIDA disambiguation significantly, see the (*prior sim-k coh*) results in Table 1. The same measure was also used by the best competitor system against which we compared, *Kul*, published by Kulkarni et al. [8]. Their approach is similar to ours, but uses a rounded linear programming method to solve the weighted graph. Due to AIDA's more powerful keyphrase similarity measure and the higher robustness of our graph algorithm we were able to outperform them significantly. More recent related work by Ratinov and Roth [11] has also used this measure.

3.2 Keyphrase-Based Entity Relatedness

There are two major downsides to computing the semantic relatedness of entities using links. First, it only works for entities that are present in Wikipedia. This is not an immediate problem when using knowledge bases derived from Wikipedia, but this is of course not always the case. Wikipedia is just one example use case for disambiguation, and there exist many special domain knowledge bases that would also benefit from entity disambiguation. The second problem appears even for entities present in Wikipedia. More than a third of Wikipedia articles describing named entities have at most 5 incoming links, which results in a crude semantic relatedness measure for these long-tail entities. To counter this problem of non-existent or sparse links, other data is needed as a basis for computing the semantic relatedness. Keyphrases are a good choice, as they can be gathered more easily than links. For Wikipedia entities AIDA already uses them for the mention-entity similarity. But also for entities that do not exist in Wikipedia because they are only of interest in a special domain, they can be extracted. Take for example this sentence in the music domain: "The performance of Nothing to You in the last Two Gallants concert was their best yet". To capture the semantic relatedness in the sentence, the keyphrases for the entity *Two Gallants* harvested from Wikipedia can be used. However, Wikipedia does not know the song *Nothing to You*. Yet on the social Website Last.fm, where fans tag and discuss their favorite bands and songs, this song's page contains keyphrases like "driving music", "folk rock", and "full of energy" among its tags and in the user comments. These overlap significantly with the keyphrases for the *Two Gallants*.

To make the most use out of every entity keyphrase, even partial matches between keyphrases should be considered. The rationale is similar to the one behind matching keyphrases partially against the context of a mention, as motivated in Section 2, to counter issues of data sparsity. Our approach to compute the semantic relatedness between entities based on keyphrases does this, and is thus dubbed KORE, for Keyphrase Overlap RElatedness [4]. The measure captures the spirit of weighted Jaccard similarity while at the same time allowing keyphrases to contribute to the measure based on their overlap with *all* keyphrases of the other entity. Let (e, f) denote a pair of entities with keyphrase sets $P_e = \{p_1, p_2, \ldots\}$ and $P_f = \{q_1, q_2, \ldots\}$, respectively. We associate with every keyphrase p a weight $\varphi_e(p)$ with respect to the entity e. We define the keyphrase overlap relatedness measure:

$$\mathsf{KORE}(e,f) = \frac{\sum_{p\in P_e, q\in P_f} \mathrm{PO}(p,q)^2 \cdot \min\{\varphi_e(p), \varphi_f(q)\}}{\sum_{p\in P_e} \varphi_e(p) + \sum_{q\in P_f} \varphi_f(q)}$$

$\mathrm{PO}(p,q)$, the partial overlap of the keyphrases p and q is the standard weighted Jaccard measure, omitted for brevity. It is squared to penalize phrases that do not fully overlap. The numerator re-weights the phrase overlap PO with the lesser weight of the two phrases that match. The denominator sums up all the keyphrase weights of each entity to normal-

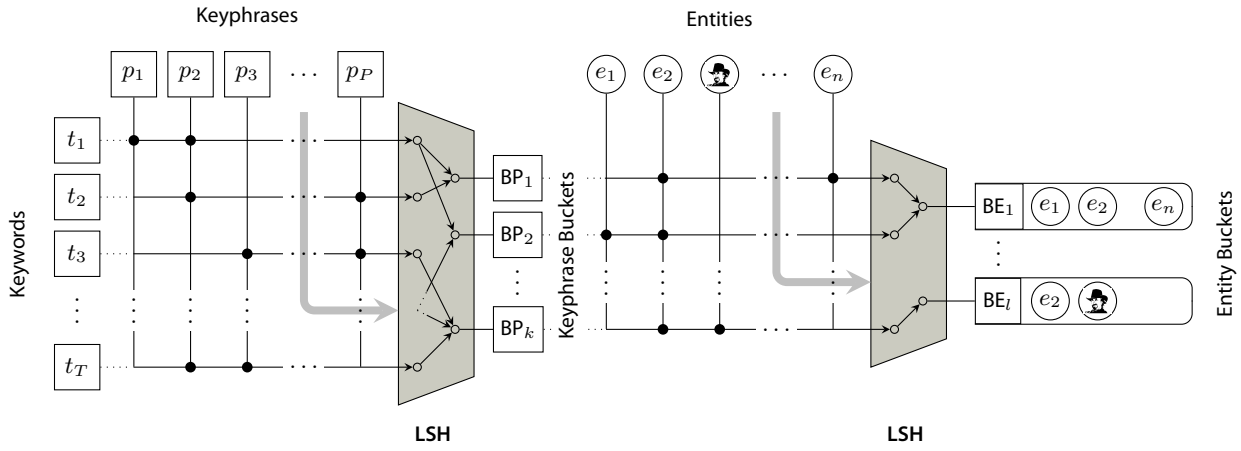

Figure 2: Overview of the two-stage hashing scheme

ize the numerator. Notice that we do not normalize by maximum possible intersection, which would be the sum over the full cartesian product $\sum_{p \in P_e, q \in P_f} \max\{\varphi_e(p), \varphi_f(q)\}$. Using this for normalization would unduly penalize popular entities with a larger keyphrase set, as the Cartesian product grows much faster than the intersection.

The experimental evaluation of this measure showed no significant difference in accuracy on the CoNLL-YAGO dataset, but in absolute numbers it performed a little worse. See the *KORE* results in Table 1. Other experiments in the KORE publication [4] however show that it performs better on long-tail entities in Wikipedia with fewer links. The key contribution is to achieve the quality of the semantic relatedness measure based on Wikipedia links using only keyphrases, which are both easier to harvest and more flexible to use.

3.3 Efficient Relatedness Computation

Computing similarities between a set of n objects is an important step in many applications, not only in entity disambiguation, but also in clustering or record linkage. The KORE measure is relevant for all these tasks, assuming that the input data is represented by keyphrase sets and partial matching improves the quality of the similarity measure.

The naive approach is to compute all $\binom{n}{2}$ pairwise similarities. This quickly becomes a bottleneck for large n. Even if the task is parallelizable, overcoming the $O(n^2)$ complexity is necessary to achieve good scalability, especially for interactive tasks such as on-the-fly entity disambiguation (e.g., for news streams) or high-throughput tasks such as entity disambiguation on an entire corpus (e.g., one day's social-media postings). To avoid the quadratic effect, we have developed hash-based approximation techniques, using min-hash sketches and locality-sensitive hashing (LSH), invented by Indyk et al. [7]. All candidate entities of a given input text are first preprocessed using this scheme, and the full KORE similarity is only computed between those that are at least somewhat related.

The scheme itself works in two stages to capture the notion of partial overlap present in KORE. First, keyphrases are clustered if they are similar enough with respect to their tokens. For example, "U.S. folk singer" will be unified with "folk singer from the U.S", but not with "U.S. president". This step is done once for the full knowledge base in a single pre-processing step. Entities are associated with the keyphrase clusters corresponding to their keyphrases. The entities are clustered in a similar manner as the keyphrases. Entities fall in the same cluster if they share enough keyphrase clusters. A visual depiction of this scheme is shown in Figure 2.

An additional benefit is that LSH can be parameterized to trade off running time against quality. In our experiments with different configurations of the two-stage scheme, we achieved an accuracy even higher than with the original KORE measure. The LSH-based preprocessing thus not only speeds up the computation, but can also clean noise that degrades the performance of KORE. See the results for $KORE_{LSH}$ in Table 1. Running times on our test documents could be improved by a factor of 4 in the mean and a factor of 7 in the 0.9-quantile with respect to KORE, without a significant loss in accuracy.

4. DISCOVERING NEW ENTITIES

The current AIDA method cannot robustly deal with mentions that have no correct entity candidate in the knowledge base. As soon as there is a candidate for a named entity in the text, the algorithm is destined to choose one, possibly disregarding it later based on a confidence threshold. Resolving this issue and identifying mentions that have no good match in the knowledge base is the current and last research goal of the Ph. D. work.

The restriction of having to disambiguate every named entity is especially problematic as the world is constantly changing. People are born, new organizations are created, and even new countries are formed and dissolved. A knowledge base must keep up with this change, but of course it can never capture everything. In addition, there is a long tail of entities that are not captured in Wikipedia-based knowledge bases because they lack the importance. Methods that try to identify canonical entities in natural language text must cope with this incompleteness. The key problem to solve then is determining when an existing name refers to a new entity, e. g. when a new person or band or song or product with given name "Paris" makes the news. Ideally, the new entity should be added to the knowledge base with an appropriate representation.

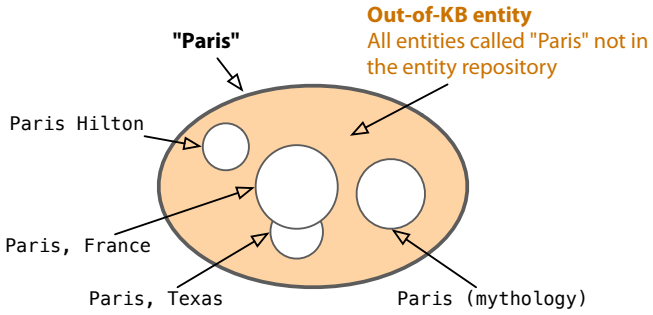

Figure 3: Determining keyphrases for an out-of-knowledge base entity for "Paris".

Until now, most related work bases the decision when a mention referes to a new entity on a threshold on some score, e.g. when the mention-entity similarity is too low, or train a classifier on a multitude of features (mostly different kinds of similarity scores). These approaches exhibit a problem that is hard to overcome. They judge if a mention refers to a new entity based on the absence of indication for an existing entity, which shows in a low score for all entity candidates. There is however no direct indication for a new entity.

Our idea to introduce a notion of direct positive indication for a new entity is to define a representation of unseen entities. This cannot be done directly, as one cannot model the unknown. However, the representation can be derived from a global representation of all entities referred to by a given name. By subtracting the representation of the existing entities in the knowledge base from the global model, what remains represents the out-of-knowledge-base entity for a given name. Figure 3 shows an example of this approach for "Paris". A suitable corpus is mined for keyphrases for "Paris", for example the Web (for all entities) or, in a restricted setting, a collection of news (for entities just gaining popularity). The keyphrases for "Paris" are all phrases that occur in the name's textual vicinity in the corpus. The keyphrases for the entities in our knowledge base we already have. To get the out-of-kb entity keyphrases (the filled part in the figure), the keyphrases for *Paris Hilton*, *Paris, France*, etc. need to be removed from the keyphrases for "Paris", e. g. by subtracting the co-occurrence counts.

From an algorithmic point of view, the out-of-kb entity is then just one more candidate for a given mention in the entity disambiguation graph, and if the graph algorithm leaves the out-of-kb entity as last/best candidate, this means that the knowledge base does not contain a correct entity for this mention. Preliminary experiments show that the approach is viable. However, the method still needs to be evaluated and compared against the state of the art baselines.

5. CONCLUSIONS AND OUTLOOK

We proposed a robust system addressing the problem of named entity disambiguation, AIDA, based on three different features: the prior probability, the keyphrase based similarity, and the coherence among the entities. To evaluate it, we created a large corpus of newswire articles with all named entities annotated. On this dataset, we outperformed our competitors, and are cited by the researchers behind the IBM Watson system as state of the art in academic entity disambiguation [1]. We extended AIDA with a more flexible measure to compute the coherence among the entities. Using only data that is easier to gather, it achieves the same performance as the state of the art in entity relatedness measures. In addition, we introduced a method to get rid of the quadratic complexity when computing the semantic relatedness between a set of entities based on hashing techniques. AIDA is used by multiple groups in academia and industry via our web service and the recently released source code.

The remaining work for the thesis is the reliable discovery of mentions that refer to an entity not registered in the knowledge base, detailed in Section 4.

6. REFERENCES

[1] D. A. Ferrucci. Introduction to "This is Watson". *IBM Journal of Research and Development*, 2012.

[2] J. R. Finkel, T. Grenager, and C. D. Manning. Incorporating Non-local Information into Information Extraction Systems by Gibbs Sampling. In *ACL*, 2005.

[3] T. Heath and C. Bizer. *Linked Data*. Morgan & Claypool Publishers, 2011.

[4] J. Hoffart, S. Seufert, D. B. Nguyen, M. Theobald, and G. Weikum. KORE: Keyphrase Overlap Relatedness for Entity Disambiguation. In *CIKM*, 2012.

[5] J. Hoffart, F. M. Suchanek, K. Berberich, and G. Weikum. YAGO2: A spatially and temporally enhanced knowledge base from Wikipedia. *Artificial Intelligence*, 2013.

[6] J. Hoffart, M. A. Yosef, I. Bordino, H. Fürstenau, M. Pinkal, M. Spaniol, B. Taneva, S. Thater, and G. Weikum. Robust Disambiguation of Named Entities in Text. In *EMNLP*, 2011.

[7] P. Indyk and R. Motwani. Approximate nearest neighbors: towards removing the curse of dimensionality. In *STOC*, 1998.

[8] S. Kulkarni, A. Singh, G. Ramakrishnan, and S. Chakrabarti. Collective Annotation of Wikipedia Entities in Web Text. In *KDD*, 2009.

[9] D. Milne and I. H. Witten. An Effective, Low-Cost Measure of Semantic Relatedness Obtained from Wikipedia Links. In *WIKIAI*, 2008.

[10] N. Nakashole, M. Theobald, and G. Weikum. Scalable knowledge harvesting with high precision and high recall. In *WSDM*, 2011.

[11] L.-A. Ratinov, D. Roth, D. Downey, and M. Anderson. Local and Global Algorithms for Disambiguation to Wikipedia. In *ACL*, 2011.

[12] E. F. T. K. Sang and F. De Meulder. Introduction to the CoNLL-2003 shared task: language-independent named entity recognition. In *CONLL*, 2003.

[13] M. Sozio and A. Gionis. The community-search problem and how to plan a successful cocktail party. In *KDD*, 2010.

[14] C. Unger, L. Bühmann, J. Lehmann, A.-C. N. Ngomo, D. Gerber, and P. Cimiano. Template-based question answering over RDF data. In *WWW*, 2012.

[15] M. Yahya, K. Berberich, S. Elbassuoni, M. Ramanath, V. Tresp, and G. Weikum. Natural Language Questions for the Web of Data. In *EMNLP-CoNLL*, 2012.