KORE: Keyphrase Overlap Relatedness for Entity Disambiguation

Johannes Hoffart
Max Planck Institute for Informatics
jhoffart@mpi-inf.mpg.de

Stephan Seufert
Max Planck Institute for Informatics
sseufert@mpi-inf.mpg.de

Dat Ba Nguyen
Max Planck Institute for Informatics
datnb@mpi-inf.mpg.de

Martin Theobald
Max Planck Institute for Informatics
mtb@mpi-inf.mpg.de

Gerhard Weikum
Max Planck Institute for Informatics
weikum@mpi-inf.mpg.de

ABSTRACT

Measuring the semantic relatedness between two entities is the basis for numerous tasks in IR, NLP, and Web-based knowledge extraction. This paper focuses on disambiguating names in a Web or text document by jointly mapping all names onto semantically related entities registered in a knowledge base. To this end, we have developed a novel notion of semantic relatedness between two entities represented as sets of weighted (multi-word) keyphrases, with consideration of partially overlapping phrases. This measure improves the quality of prior link-based models, and also eliminates the need for (usually Wikipedia-centric) explicit interlinkage between entities. Thus, our method is more versatile and can cope with long-tail and newly emerging entities that have few or no links associated with them. For efficiency, we have developed approximation techniques based on min-hash sketches and locality-sensitive hashing. Our experiments on semantic relatedness and on named entity disambiguation demonstrate the superiority of our method compared to state-of-the-art baselines.

Categories and Subject Descriptors

H.3.1 [**Information Storage and Retrieval**]: Content Analysis and Indexing

Keywords

Semantic Relatedness, Entity Relatedness, Entity Disambiguation, Locality-Sensitive Hashing

1. INTRODUCTION

1.1 Motivation

Semantic relatedness measures between entities like people, places, songs, companies, etc. are a fundamental asset for many applications dealing with natural language processing (NLP), text mining, or Web analytics [4]. Its usefulness has been shown in a wide variety of tasks: named entity disambiguation [23, 19, 16], general word sense disambiguation [31, 24, 26], query expansion for information retrieval [18, 25], Web-based information extraction [1], knowledge base population [9, 30], and more.

In this paper, we focus on semantic relatedness of entities and its role in named entity disambiguation (NED). Consider the terms "Cash" and "Jackson" as an example. When trying to measure the relatedness between these surface forms, we face high ambiguity and would assess them as weakly related only. Most possible meanings of "Cash" and "Jackson", such as movies entitled "Cash" and people or cities named "Jackson", have nothing in common. At the entity level, however, once we identify the surface form "Cash" with the singer *Johnny Cash* and "Jackson" with a song he performed, *Jackson (song)*, the relatedness measure should be very high. Wikipedia and derived knowledge bases like DBpedia, YAGO, or Freebase provide us with millions of uniquely identifiable entities, and this enables computing relatedness measures at the entity level instead of ambiguous words or phrases. Conversely, entity relatedness is key for disambiguating names that occur in surface text. In our example, the entity names in a sentence like "The audience got wild when Cash performed Jackson." can be correctly disambiguated because only the singer-song combination, out of thousands of other pairs of entities named "Cash" and "Jackson", yields a semantically coherent interpretation.

State-of-the-art measures for entity relatedness that perform well in NED and other tasks are based on the extensive link structure of Wikipedia. Most notably, the *Milne-Witten measure* considers the overlap of the incoming links to two entity articles as a notion of their relatedness [22]. This is a great asset for the 3 million entities known to Wikipedia, but it is a limitation for other entities that do not (yet) have a Wikipedia page. For example, in the sentence "The audience was haunted when Cave performed Hallelujah.", the correct meaning of "Hallelujah" is neither the Händel chorus nor the Leonard Cohen composition, but a totally different song by the Australian singer Nick Cave. That song has a Web page in the music portal last.fm but there is no Wikipedia article for the song. Link-based relatedness has no chance to capture the tight connection between Nick Cave and his song. This limitation already applies to the long tail of not so prominent Wikipedia entities whose articles have only few incoming links. In these cases, the Milne-Witten measure does not properly capture the semantic relatedness between entities.

Our goal in this paper is to overcome this limitation of Wikipedia-link-based relatedness measures for named entities.

1.2 Our Approach

A relatedness measure for link-poor and out-of-Wikipedia entities needs to tap into information that is more widely available and can be gathered with reasonable effort. In this paper, we use entity-specific *keyphrases* to this end [21, 5]. Keyphrases can be mined from any text describing an entity: people not popular enough for Wikipedia have personal homepages, small companies have websites, singers and songs are discussed in online communities like last.fm or YouTube, and so on. For our difficult example, the last.fm page on Nick Cave's song contains keyphrases like "No More Shall We Part", "Australian singer", "Warren Ellis", "eerie cello", "Nick Cave and the Bad Seeds", etc. These exhibit significant overlap with phrases contained in the Wikipedia article about Nick Cave (or, alternatively, his Web page), thus giving cues for the high relatedness. We propose a new notion of entity relatedness, coined KORE, based on the overlap of two sets of keyphrases.

While the idea for this approach may seem obvious, it entails a number of technical difficulties, addressed in this paper:

Keyphrases and Weights: Multi-word keyphrases are more informative than single-token keywords and thus preferrable for characterizing entities. On the other hand, this makes it less clear which word sequences should be considered as salient keyphrases, and how we could associate them with weights that can be computed in an effective and efficient manner. If we simply used keywords, we could use precomputed *idf*-style weights; the generalization to keyphrases is not obvious.

Partial Matches: When comparing the keyphrases of two entities, the chance of finding exact matches drops with the length of multi-word phrases. For example, we may not find the phrases "Australian singer" and "Nick Cave and the Bad Seeds", associated with the song Hallelujah, in this exact form in the keyphrase-set of Nick Cave, where we may instead have "Australian male singer" and "Cave's Bad Seeds". Therefore, a viable notion of keyphrase-based overlap measure must consider partial matches between phrases as well.

Efficient Computation: In NED, we are interested in the relatedness of all entity pairs in the candidate space. For n potentially relevant entities, a straightforward approach would require $\binom{n}{2}$ computations. If we consider a short news article with say 10 mentions of ambiguous entity names, each of which has 10 possible meanings, we obtain 100 candidate entities and would need to compute the relatedness for 5000 pairs. This is too expensive for most online applications, and totally infeasible for longer input texts or mentions with higher degrees of ambiguity.

1.3 Contribution

Our approach addresses the above challenges in the following way. We extract particular kinds of noun phrases from entity-specific pages, and compute weights for both entire phrases and their constituent words. We use MI (mutual information) for phrases, contrasting the entity page with the entire Wikipedia corpus (or the Web), and idf (inverse document frequency) for single words, based on the Wikipedia corpus. We define the KORE measure in a two-stage manner, thus allowing for partial matches between keyphrases: 1) two keyphrases are paired up if there is a partial match of at least one word, with word-level weights influencing the matching score; 2) a weighted Jaccard coefficient captures the overlap between the keyphrase-sets of two entities, where the scores of the partially matching keyphrase pairs are aggregated and the phrase-level weights are considered. Finally, to solve the efficiency challenge, we use a two-level approximation technique: keyphrases, as sequences of words, are approximated by min-hash sketches, and these sketches are organized by locality-sensitive hashing (LSH). This way, for a given input set of entity pairs, we compute their relatedness only if their keyphrase representations are mapped to at least one common LSH bucket.

The novel contributions made in this paper are as follows:

- KORE improves the quality of assessing entity relatedness, compared to state-of-the-art link-based methods, while reducing the dependence on Wikipedia linkage and tapping into out-of-Wikipedia entities.
- KORE is integrated into a joint-inference NED method, outperforming the best prior methods on mentions of long-tail entities.
- KORE can be efficiently computed and avoids the quadratic complexity of all-pairs comparisons in NED-like tasks, by a judiciously devised two-stage hashing technique.
- Our improvements in the quality of semantic relatedness and NED accuracy are demonstrated by systematic experiments with different corpora. Run-time measurements show the efficiency gains of our approximation techniques.

2. RELATED WORK

2.1 Semantic Relatedness of Words

The classical approaches for quantifying the relatedness of words (common nouns, verbs, etc.) make use of the WordNet thesaurus, by measuring the token overlap of the glosses describing the concepts (Lesk) or by means of paths in the concept taxonomy or other lexical relations (Resnik, Wu & Palmer). Budanitsky and Hirst [4] give an overview of all these measures.

More recently, Gabrilovich and Markovitch [12] and Radinsky et al. [28] have exploited Wikipedia (and Zesch et al. Wiktionary [33]) to compute word relatedness. These methods represent words by vectors of the concepts they are used to describe, i. e. a word is represented by all the articles it occurs in. Then they use vector-space measures (e.g., cosine similarity) for relatedness. A powerful variant of this approach, proposed by Hassan and Mihalcea [14], is to represent words as vectors of co-occurring Wikipedia entities. However, the method crucially depends on the availability of a rich link structure like that of Wikipedia.

2.2 Semantic Relatedness of Entities

Work on semantic relatedness between entities (people, places, songs, companies, products, etc.) has started with the advent of large knowledge bases, largely derived from Wikipedia. This created huge opportunities for specifically dealing with named entities (as opposed to general words). The emerging need for lifting Web search and advertising to the entity level created a mega-application as well.

Ponzetto and Strube [27] applied previously WordNet-based measures to Wikipedia, showing that some work better on the larger though less rigorous collection of entities and classes. Milne and Witten [22] used the link structure in Wikipedia to model entity relatedness. Links within Wikipedia refer to canonical entities; thus, the source-target pairs of links can be used as the basis for a relatedness measure. Building on the co-occurrence-based information-theoretic model of [20, 6], the *Milne-Witten relatedness* measure, MW for short, for two entities e and f with sets of incoming Wikipedia links I_e and I_f, respectively, is defined as (where N is the total number of articles in Wikipedia):

$$\mathrm{MW}(e,f) = 1 - \frac{\log(\max\{|I_e|,|I_f|\}) - \log(|I_e \cap I_f|)}{\log(N) - \log(\min\{|I_e|,|I_f|\})}. \quad (1)$$

So far, the MW measure has achieved the best results on quantifying entity relatedness.

2.3 Named Entity Disambiguation (NED)

NED is the task of mapping surface names of entities (people, places, etc.) in text documents onto canonicalized entities registered in a knowledge base like DBPedia [2], Freebase.com, or YAGO [32], this way disambiguating the entity mentions. Classical approaches were based on comparing the *text similarity* of the context that surrounds a mention against textual descriptions of candidate entities. This can be cast into IR-style bag-of-words or language-model comparisons, using words, bigrams, noun phrases, and other linguistic features. The mentions in a given text are processed one at a time.

Cucerzan [8] was the first to recognize the potential of *joint inference* by mapping all named entities in a text simultaneously, aiming to enforce coherence among the chosen entities as expressed by their pair-wise semantic relatedness. Milne & Witten [23], Kulkarni et al. [19], Dredze et al. [9], Ferragina & Scaiella [10], and Hoffart et al. [16] extended and improved this line of methodology, using combinatorial or statistical-learning algorithms for joint inference. The best performing of these NED algorithms utilize the link-based MW measure as defined in Equation (1). None of the methods can robustly cope with long-tail entities that have only few or no Wikipedia links.

NED is a specific case of the more general task of word sense disambiguation (WSD) where all words in a text should be mapped onto canonicalized denotations: entities for proper nouns (i.e., names) and WordNet senses (so-called synsets) for common nouns and other words or phrases. This task can benefit from semantic relatedness as well. However, we do not consider general WSD in this paper. Navigli [24] surveys state-of-the-art WSD methods. Another task related to but different from NED is cross-document coreference resolution (e.g., [30]). Here, mentions are grouped into equivalence classes, but there is no explicit mapping onto entities in a knowledge base.

3. COMPUTING ENTITY RELATEDNESS

3.1 Definitions

All measures of semantic relatedness between entities described in this section are based on (multi-word) key*phrases* and their constituting (single-word) key*words*. When we do not need to distinguish between (multi-word) key*phrases* and (single-word) key*words* we speak of key*terms*.

Example. The keyphrase English rock guitarist is represented by the set of keywords {English, rock, guitarist}.

We gathered keyterms from the Wikipedia pages of each entity: *link anchors* of both internal and external links, *titles of citations*, and *names of categories*. The choice of Wikipedia is merely for convenience; keyterms can also be mined from other textual descriptions of entities, such as homepages of singers or scientists, or pages about songs in online communities. In these cases, we can use similar heuristics like focusing on link anchors, headings, etc., or alternatively extract all noun phrases. The latter will pick up noise, but our weighting scheme, described next, will keep the noise level low.

All keyterms are weighed by a combination of Inverse Document Frequency (idf), which is global and entity-independent, and Mutual Information (MI), which depends on the co-occurrence of the keyterms with entities.

Idf weights capture a global notion of how important a keyterm is. The weights are computed using the standard formula, $\log_2 \frac{N}{df}$, where N is the size of the collection (in this case the total number of entities in the knowledge base) and df is the number of entities that have the phrase among their keyphrases (for keyphrase idf) or have at least one keyphrase that contains the keyword (for keyword idf).

MI weights capture the notion of how important a keyterm is with respect to a given entity. It is based on the relative entropy between the events of an entity-keyterm pair occurring individually or jointly. For its computation, we define the superdocument of an entity to be the union of its keyphrases with the keyphrases of all entities linking to it. The joint occurrence of a keyterm and an entity is the occurrence of the keyterm in the superdocument. We use this definition in a normalized variant of MI, which is computed as follows:

$$\mu(E,T) = 2 \cdot \frac{H(E) + H(T) - H(E,T)}{H(E) + H(T)}, \quad (2)$$

where $H(E)$ and $H(T)$ are the marginal entropies of the entity and term, respectively, and $H(E,T)$ is the joint entropy. We compute the μ weights for all entity-keyphrase and entity-keyword pairs. Note that all weights are with respect to the keyphrase space, not the original texts they were mined from.

3.2 Keyterm Cosine Relatedness

If no links are available upon which the entity relatedness can be computed, a baseline method is as follows. We compare two entities (e,f) represented by keyterm sets $T(e)$ and $T(f)$ by casting the keyterms into a weighted vector. This works for both **keyphrases** and **keywords**. If the terms are phrasal in nature, they can either be treated as a single unit in the vector or tokenized before constructing the vector. In our experiments, the vectors are filled with the MI weights of the keyterms. The relatedness between these vectors can be calculated using cosine similarity. Let (e,f) denote a pair of entities with keyterm vectors V_e and V_f, respectively. The keyterm relatedness is:

$$\mathrm{ktr}(e,f) = \frac{V_e \cdot V_f}{\|V_e\|\,\|V_f\|} \quad (3)$$

If keywords are derived from keyphrases by tokenization, their weights should take the phrase weights into account.

We simply multiply the weights of the words with the average weight of the phrases from which the words are taken.

3.3 Keyphrase Overlap Relatedness

Concepts are often represented as phrases (at least in the English language, but to a large extent also in other languages); so relatedness measures should consider multi-word keyphrases instead of single keywords only. However, when entities are represented by keyphrases, it is crucial to consider partial matches rather than focusing solely on exact-match comparisons. For example, "English rock guitarist" should be more similar to "English guitarist" than to "German President". To solve this issue, the relatedness measure needs to match keyphrases based on overlap and needs to take into account both keyphrase and keyword weights.

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. A phrase is identified with the set of constituent terms $p_i := \{w_1, w_2, \ldots\}$. We associate with every keyword w a weight $\gamma_e(w)$ with respect to the entity e.

In the following, let $(p, q) \in P_e \times P_f$ denote a pair of phrases. This allows us to introduce a measure of *phrase overlap* (PO) for the pair (p, q), where its constituent words are weighted with respect to the entities (e, f), given by the weighted Jaccard similarity of the keywords:

$$\mathrm{PO}(p,q) = \frac{\sum_{w \in p \cap q} \min\{\gamma_e(w), \gamma_f(w)\}}{\sum_{w \in p \cup q} \max\{\gamma_e(w), \gamma_f(w)\}} \tag{4}$$

We use PO to measure the *keyphrase overlap relatedness* (KORE) of a pair of entities (e, f) based on the sets of their associated phrases P_e and P_f. 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. Keyphrases are weighted with respect to the entities; recall that these weights φ_e are different from the keyword weights γ_e. We define the keyphrase overlap relatedness measure:

$$\mathsf{KORE}(e,f) = \frac{\sum_{p \in P_e, q \in P_f} \mathrm{PO}(p,q)^2 \cdot \min\{\varphi_e(p), \varphi_f(q)\}}{\sum_{p \in P_e} \varphi_e(p) + \sum_{q \in P_f} \varphi_f(q)} \tag{5}$$

The factor $\mathrm{PO}(p, q)$ is squared to penalize phrases which do not fully overlap. Our experiments have shown that using MI weights for keyphrases (φ_e) and IDF weights for keywords (γ_e) works best. 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 normalize 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.

4. EFFICIENT COMPUTATION

4.1 Need for Speed

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 coreference resolution (record linkage). The KORE measure is relevant for all these tasks, assuming that the input data is a keyphrase set 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 NED (e.g., for news streams) or high-throughput tasks such as NED on an entire corpus (e.g., one day's social-media postings).

Precomputing the similarities is prohibitive with regard to both space and time. Today's knowledge bases contain millions of entities. The quadratic space and time complexity would lead to more than 10^{12} pairs. Even if we merely needed a single byte to store a pair's relatedness value, we would consume Terabytes of storage, and potentially much more. This is not practically viable. So the goal is to avoid the quadratic effect and devise appropriate pre-processing that facilitates fact relatedness computations on the fly. To this end, we have developed hash-based approximation techniques, using min-hash sketches [3] and the method of locality-sensitive hashing (LSH), invented by Indyk et al. [17, 13]. LSH has been used in numerous applications, including clustering in the context of NLP tasks (e.g., [29]). To the best of our knowledge, no prior work has considered such hashing techniques for entity relatedness and NED tasks.

4.2 Two-Stage Hashing

Entities are represented as sets of (weighted) keyphrases, which in turn are represented as sets of (weighted) keywords. Thus, keyphrases should not be compared atomically. "President of the United States" should be more similar to "United States President" than to "German President".

If we want to use LSH to speed up the computation, partial matches among phrases must be taken into consideration. As a solution, we propose the following two-stage hashing scheme:

1. Bucketize all highly similar keyphrases. Entities are then represented as a set of identifiers of such keyphrase buckets without losing the notion of partial phrase matches.

2. Group entities together that have sufficiently high overlap in terms of keyphrase bucket identifiers. The exact pairwise relatedness score can then be computed within a single entity-group, reducing the complexity of $O(n^2)$ to linear in the number of entities if the hashing techniques achieve near-uniform distribution of keyphrases (step 1) and entities (step 2). In principle, skewed distributions could still incur bottlenecks; however, our experiments with millions of entities did not exhibit such adverse effects in practice.

In the following, we explain the two steps in more detail. Figure 1 gives a pictorial overview of our technique.

4.2.1 *Grouping Highly Similar Keyphrases*

The goal of this step is to group keyphrases that are near duplicates in the sense that their Jaccard similarity is very high. As a compact representation of the words in a keyphrase, we first compute min-hash sketches for each phrase.

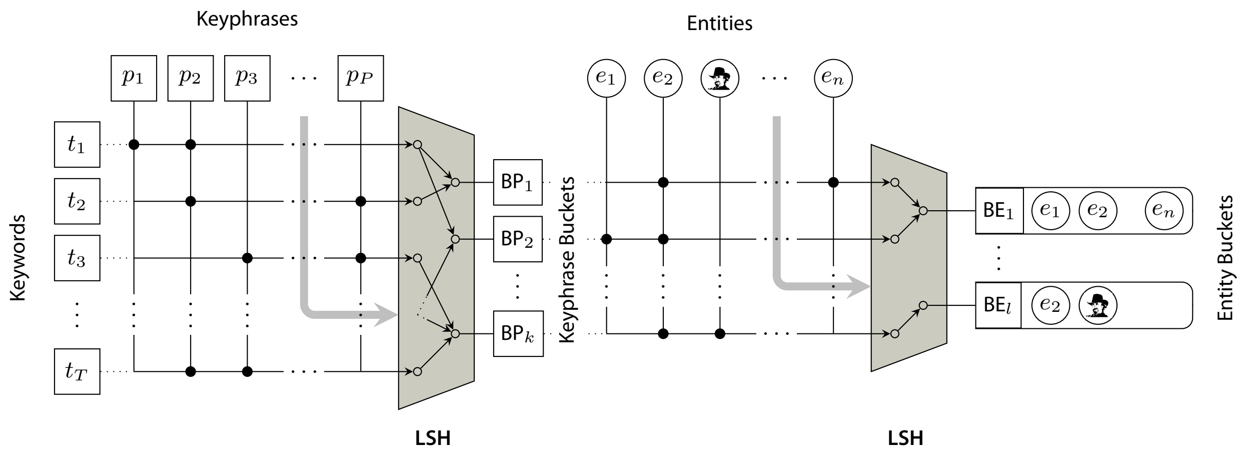


Figure 1: Overview of the two-stage hashing technique

This form of sketches allows us to approximate (as an unbiased estimator) the Jaccard similarity between sets (of words, in our case). To avoid the pair-wise comparison of all keyphrases, we apply LSH on the min-hash sketches of the phrases.

In detail, we first represent a keyphrase by a set of words and assign each word a globally unique id. For each keyphrase (the average length in our knowledge base is 2.5 words) we sample 4 times by min-hashing, thus representing each keyphrase by a vector of length 4. For LSH, we divide the vectors in two bands of length two, and combine the two ids in each band by summing up their ids, losing the order among them. Each keyphrase is finally represented by the two bucket ids (one per band) it was hashed to, combining near duplicate keyphrases. Note that we do not perform this stage-one hashing to reduce the dimensionality of the keyphrase space, but to capture the notion of partial overlapping keyphrases in KORE and to improve the second stage of grouping entities.

4.2.2 *Grouping Related Entities*

While the first stage above is precomputed for all entities and their keyphrases (with linear complexity), the second stage, described now, is executed at task run-time with a set of entities as input – for NED these are all candidate entities in a document. The algorithm first retrieves the sets of min-hash sketches for all input entities, pre-computed from the phrase-bucket ids output by the previous stage. These ids are now the input to a second stage of LSH. Algorithmically, this is fairly analogous to the first-stage technique, but the inputs are different from the first stage.

The exact KORE measure between two input entities is computed only if they share at least one hash bucket. Otherwise, we assume the entity relatedness is sufficiently low to consider the entities as unrelated. For the 3 million entities in our knowledge base we can fit the sketches for $\text{KORE}_{\text{LSH-G}}$ into main memory, merely requiring about 2 GBytes.

For $\text{KORE}_{\text{LSH-G}}$, a reasonably fast approximation of KORE which has nearly the same quality as KORE, we partition the sketched phrase-bucket id vectors into 200 bands of size 1. For $\text{KORE}_{\text{LSH-F}}$, a really fast approximation of KORE which degrades the approximation quality a bit, we use 1,000 bands of size 2, again combining the sketched ids by summing them up before hashing. $\text{KORE}_{\text{LSH-G}}$ is geared towards high recall so that the actual computation is executed between all somewhat related entities, but filters out noise. The speed-up is not so high, but the quality is close to exact KORE. Sometimes, quality is even improved as noisy candidates are removed. $\text{KORE}_{\text{LSH-F}}$ is geared towards higher precision with bands of size two, allowing LSH to prune even more entity pairs, speeding up the subsequent computation of the semantic relatedness due to fewer comparisons.

As we create the LSH hashtables dynamically during runtime for a given input set of entities, using more bands of larger size means that we need longer min-hash sketches – for $\text{KORE}_{\text{LSH-F}}$ a total of 2,000 per entity. This increases the time for constructing the hashtables. However, in our experiments the time difference in creating the candidate pairs using LSH for $\text{KORE}_{\text{LSH-G}}$ and $\text{KORE}_{\text{LSH-F}}$ is low; so the overall runtime is improved, especially for large sets of entities.

5. EXPERIMENTS ON RANKING RELATED ENTITIES

5.1 Dataset

Existing datasets to evaluate semantic relatedness quantify the relatedness between surface forms of words, not between entities registered in a knowledge base, which makes these datasets unsuitable for our task. We have created a new dataset to evaluate the quality of different measures for the relatedness between entities. A conceivable way to create such data would try to determine numeric scores for the degrees of relatedness (e.g., [11]). However, for most tasks a good ranking among the candidates is important; numeric scores are rarely needed. Moreover, semantic relatedness is difficult to judge by humans in an absolute manner. For these reasons, we settled for relative ranking judgments and use a crowdsourcing platform (Crowdflower) to average out the subjectivity of such judgments.

We selected a set of 20 entities from YAGO2 [15] from 4 different domains: IT companies, Hollywood celebrities, video games, and television series. For each of the 20 seed entities we selected 20 candidates from the set of entities linked to by the seed's Wikipedia article. Using entities from Wikipedia articles allows us to capture the candidates in the context which relates them to their seed entity. We tried to

Domain	**KWCS**	**KPCS**	**MW**	**KORE**	**$\text{KORE}_{\text{LSH-G}}$**	**$\text{KORE}_{\text{LSH-F}}$**
IT Companies	0.660	0.759	0.721	0.764	0.586	0.208
Hollywood Celebrities	0.722	0.715	0.667	0.646	0.647	0.522
Television Series	0.577	0.599	0.628	0.519	0.538	0.426
Video Games	0.604	0.760	0.431	0.780	0.722	0.499
Chuck Norris	0.481	0.498	0.571	0.585	0.585	0.653
Average (11 entities with $\leq$ 500 links)	**0.597**	**0.637**	**0.513**	**0.640**	**0.625**	**0.496**
Average (all 21 entities)	**0.633**	**0.698**	**0.610**	**0.673**	**0.621**	**0.425**

Table 1: Spearman correlation of relatedness measures with human ranking

Seed	**Related Entity (Rank)**
Apple Inc.	Steve Jobs (1), Steve Wozniak (2) ... NeXT (10), Safari (web browser) (11) ... Ford Motor Company (20)
Johnny Depp	Pirates of the Caribbean (1), Jack Sparrow (2) ... Into the Great Wide Open (10) ... Mad Love (20)
GTA IV	Niko Bellic (1), Liberty City (2) ... New York (10), Bosnian War (11) ... Mothers Against Drunk Driving (20)
The Sopranos	Tony Soprano (1), David Chase (2) ... Golden Globe Award (10), The Kinks (11) ... Big Love (20)
Chuck Norris	Chuck Norris facts (1), Aaron Norris (2), ... Northrop Corporation (10) ... Priscilla Presley (20)

Table 2: Example seed entities and gold-standard ranks of related entities

select the candidates in such a way that their semantic relatedness to the seed entity should be clearly distinguishable among each other, and included highly related as well as only remotely related entities in the set of candidates.

Example. For the entity *Chuck Norris* we have *United States Air Force* and *Chun Kuk Do* as candidates, described by the following context: "After serving in the *United States Air Force*, he began his rise to fame as a martial artist and has since founded his own school, *Chun Kuk Do*". This contextual information is given to the human judges, together with links to the Wikipedia articles for more detailed information. The crowdsourcing worker is then asked to rank *United States Air Force* versus *Chun Kuk Do* in terms of their relatedness to *Chuck Norris*. This included a "They are about the same" option.

The gold-standard ranking of the 20 candidate entities per seed is created as follows:

- All possible comparisons of the 20 candidates with respect to their seed are created (190 in total).
- 10 of the 190 comparison pairs are created as gold standard by the authors. The gold standard is used to compute the confidence in the human judges and to exclude them if they get too many wrong.
- For of the remaining 180 comparison pairs, 5 distinct judges are asked which of the given two entities is more related to the seed entity.
- All comparisons are aggregated by Crowdflower into a single confidence that one entity is more (or equally) related to the seed.
- The 20 candidate entities are then ranked by these confidence values as described by Coppersmith et al. [7], which has shown that the number of times an entity wins and the weights of the wins gives a good ranking.

The final output is a set of 20 ranked lists consisting of 20 entities each, against which we compare the rankings generated by the semantic relatedness measures. For the previous example, *Chun Kuk Do* was ranked 5 and *United States Air Force* 10, reflecting the stronger relatedness of a person to a school of martial arts initiated by him than to an organization he worked for. More examples of entity pairs with their crowdsourcing-derived ranks are given in Table 2.

All seed entities were chosen to be among the most popular individuals in their respective domain (*Apple* as IT company, *Johnny Depp* as Hollywood celebrity, *Grand Theft Auto IV* (GTA IV) as video game, and *The Sopranos* as television series). We also added *Chuck Norris* consisting only of *Chuck Norris* (a singleton set). The dataset, which contains a total of 441 entities (20 candidates for 21 seeds), is available at *http://www.mpi-inf.mpg.de/yago-naga/aida*.

5.2 Experimental Results

Experimental results for this dataset are given in Table 1. The numbers are the Spearman correlation between the gold-standard ranking and the automatically generated rankings by the relatedness measures.

The MW measure is by Milne and Witten (Equation (1)). KWCS is cosine similarity between the IDF-weighted keyword vectors derived from the (MI-weighted) keyphrases, KPCS is the cosine similarity on MI-weighted keyphrase vectors. KORE is the keyphrase overlap relatedness, with two configurations of LSH approximations. All measures are detailed in Section 3.

We see that all the measures using keyphrases – and do not depend on links – work better than the link-based MW, with KPCS performing best. The difference becomes even more obvious when we average the Spearman correlation not over all entities but only over those with less than 500 incoming links in Wikipedia (relatively "link-poor" entities) – here KORE works best. The small size of the dataset did not allow meaningful significance tests, though. We emphasize the point that even without links, the keyphrase-based measures perform at least as good as the MW measure. So we no longer depend on the rich Wikipedia link structure without losing quality, and we are well geared for dealing with truly long-tail entities that have very few or no links at all.

6. EXPERIMENTS ON NAMED ENTITY DISAMBIGUATION

For the NED experiments with different relatedness measures, we modified the AIDA framework [16] to incorporate the KORE measure. Figure 2 illustrates how AIDA represents the mentions in a text document and their entity candidates as a graph.

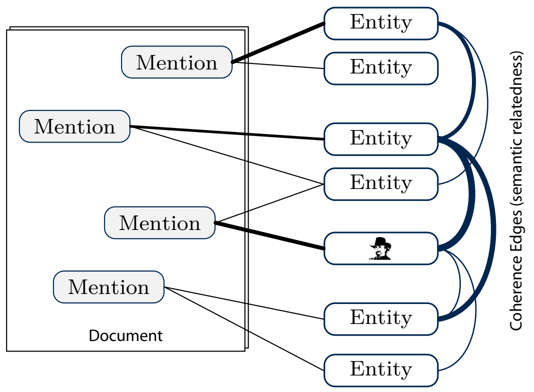


Figure 2: Disambiguation graph

Mention-entity edges have weights that combine a popularity-based prior (for a name denoting a specific entity) and the contextual similarity between a text window around the mention and a set of keyphrases for the candidate entity. Entity-entity edges have weights based on semantic relatedness measures. AIDA uses Milne and Witten's measure of semantic relatedness MW (Equation (1)). This method has outperformed all other coherence-aware alternatives in the experiments of [16]. AIDA does not consider keyphrases for entity-entity coherence.

6.1 Datasets

We conducted experiments on three datasets:

CoNLL-YAGO. The CoNLL-YAGO data set which is originally used in [16]. It is based on the CoNLL 2003 dataset, which consists of 1,393 newswire articles with an average article length of 216 words. In each article, all mentions are annotated and mapped to the correct entity in YAGO2. All numbers are from runs on the original 231 withheld test documents.

KORE50. We hand-crafted 50 difficult test sentences from five domains: celebrities, music, business, sports, and politics. The sentences were formulated according to a set of criteria:

- Short context: on average, only 14 words per sentence.
- High density of entity mentions: 148 mentions in 50 sentences, nearly 3 per sentence on average and a ratio of 20%. The mention-to-word ratio in CoNLL-YAGO is 12%.
- Highly ambiguous mentions: 631 candidate entities per mention, on average. In contrast, CoNLL-YAGO has only 27.
- Sentences contain long-tail entities with very few incoming links.

Examples are persons referred to by their first name only instead of last name or full name, or football clubs and players that are not well known outside their own home states. It is available at *http://www.mpi-inf.mpg.de/yago-naga/aida/*.

WP. This dataset is a prepared slice of Wikipedia with similar characteristics as KORE50. It contains all articles in categories ending with "heavy metal musical groups". Each article is split into sentences, and only sentences containing at least 3 named entities as link-anchor texts are kept. The link targets of these mentions provides us with ground-truth mappings. As a stress test, to make disambiguation more difficult, we replaced all occurrences of person names with the family name only (e. g. "Johnny Cash" replaced by "Cash"). In the same vein, we disabled the popularity-based prior for name-entity pairs for this dataset in all methods under comparison. We gathered 2,019 sentences this way, with an average length of 52 words. (The automatic sentence boundary detection did not always work properly, so some output segments were longer than expected). Each sentence has an average of 5 mentions with 64 candidates.

6.2 Experimental Results

We measured NED accuracy as the fraction of mentions that were correctly disambiguated, ignoring all mentions that do not have a candidate entity at all, following the evaluation methodology of [16]. The results are shown in Table 3: micro-averaged numbers aggregate over all mentions, macro-averaged numbers aggregate over all documents of a dataset. We also broke down the per-mention results by the number of Wikipedia inlinks of the mention's true entity. Link-averaged numbers are the macro-average over the groups of mentions with the same number of inlinks.

Overall, the KORE-based NED performed about as well as the original AIDA method, which uses the MW measure based on the rich link structure of Wikipedia. MW performs better on the CoNLL-YAGO dataset, KORE performs better on the KORE50 and WP datasets. A paired t-test shows that there is no significant difference between MW and KORE on CoNLL-YAGO or KORE50 for macro-average accuracy; however, KORE is significantly better on the WP dataset (p-value < 0.01 in a paired t-test). With link-averaged accuracy, KORE and $\text{KORE}_{\text{LSH-G}}$ also perform significantly better than MW on KORE50 (p-value < 0.01). On the WP dataset, Table 3 additionally gives the average accuracy for mentions whose true entities have few links, to emphasize the accuracy of KORE for long-tail entities.

Figure 3 shows a detailed comparison of MW and the KORE variations (using LSH) on the KORE50 dataset. The accuracy at each point x on the x-axis is the average accuracy of all entities with up to x links. As expected, KORE works a lot better for really link-poor entities, because it builds on keyphrases instead of links. With more links, the performance of KORE is still significantly better, but the difference between KORE and MW becomes smaller.

Performing well on link-poor entities is really important considering the fact that entities with ≤ 500 incoming links make up more than 90% of Wikipedia.

6.3 Efficiency

To evaluate the efficiency, we compared the running time of AIDA with the link-based MW measure against AIDA with the KORE measure, both in its exact form and with the LSH approximation for speed-up. We tested on the full CoNLL-YAGO collection consisting of 1,393 newswire articles (216 words on average) with a total of 27,820 mentions

Dataset	Evaluation	KWCS	KPCS	MW	KORE	KORE$_{LSH-G}$	KORE$_{LSH-F}$
CoNLL-YAGO	Micro Avg.	79.53%	82.18%	**82.31%**	80.71%	81.76%	81.18%
	Macro Avg.	79.07%	81.91%	**82.00%**	80.59%	81.22%	80.08%
	Link Avg.	79.28%	**82.31%**	81.34%	80.21%	81.80%	80.80%
WP	Micro Avg.	83.29%	85.30%	84.73%	**85.36%**	84.68%	84.50%
	Macro Avg.	82.50%	**84.59%***	83.86%	84.56%*	83.84%	83.61%
	Link Avg.	77.90%	80.49%	**82.45%**	80.12%	80.64%	80.36%
	Link Avg. $\leq$ 500 links	84.80%	86.80%	85.59%	**87.15%**	86.43%	86.29%
	Link Avg. $\leq$ 50 links	83.39%	85.19%*	83.87%	**85.52%***	84.56%	84.90%*
	Link Avg. $\leq$ 5 links	80.89%	81.10%	80.89%	**82.40%***	81.75%	81.38%
KORE50	Micro Avg.	55.56%	55.56%	57.64%	63.89%	**64.58%**	53.19%
	Macro Avg.	55.17%	54.70%	56.00%	62.17%	**62.60%**	52.07%
	Link Avg.	62.11%	61.79%	63.21%	70.75%*	**71.70%***	58.58%

Table 3: Disambiguation accuracy (best method per row is in boldface; statistically significant improvements over MW are marked with an asterisk)

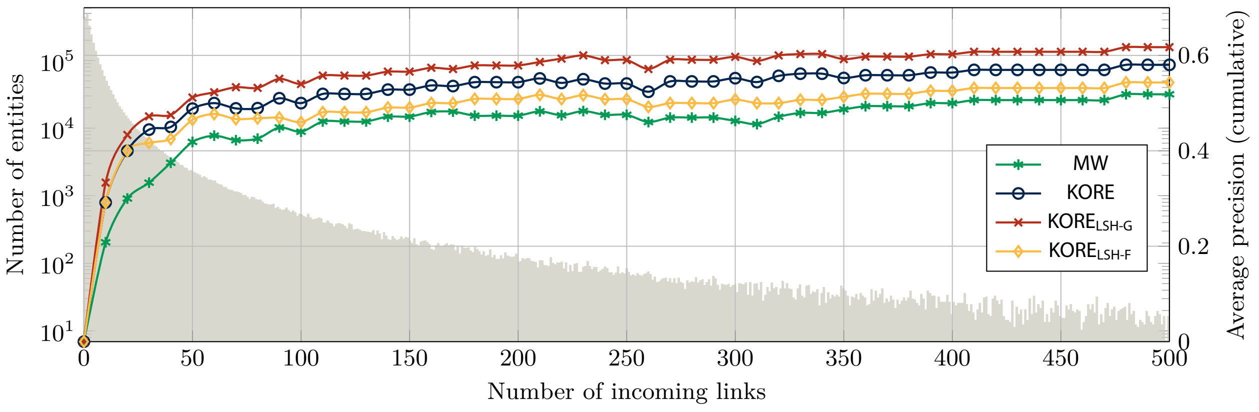


Figure 3: Accuracy of relatedness measures on entities with fewer than 500 incoming links on the KORE50 dataset

(we exclude mentions with no gold standard entity). On average, there are 25 mentions of named entities per article, and 73 entity candidates per mention. AIDA computes coherence weights only between candidate entities that are candidates for at least two different mentions. If they share only one mention, they are mutually exclusive candidates anyway. Thus, the number of comparisons (and the run-time) is still in the order of n^2, but it does not increase monotonically with the number of candidate entities for a document. Figure 4 shows the documents on the x-axis and the measured run-times for computing the NED on a document on the y-axis. The documents on the x axis are sorted in ascending order by the respective number of candidate entities ("largest" documents are thus rightmost).

We observe that the exact version of KORE is already much faster than MW. This is because MW is based on links, and some popular entities have a large number of incoming links (sometimes above 100,000, e.g., for places). In such cases, the intersection of bitvectors that AIDA uses to represent inlinks is very time-consuming. The number of keyphrases of popular entities does not nearly grow at the same rate, as it is bounded by the length of the document describing the entity. In contrast, incoming links can grow independently of the document size.

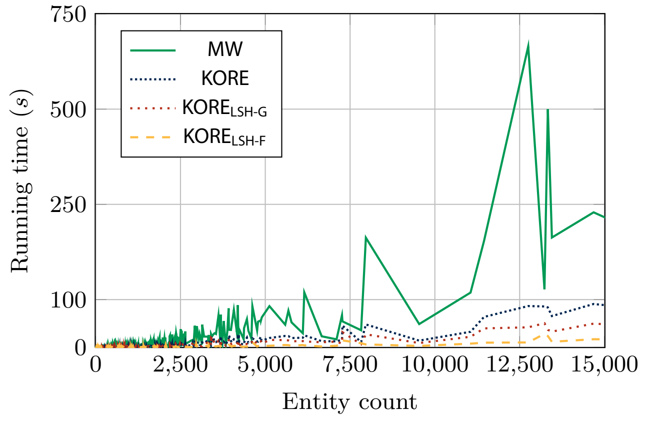
Table 4 gives detailed figures for the average run-time, its standard deviation, and the 90%-quantile of the run-time distribution (over the documents), and the same measures also for the number of entity-pair relatedness comparisons. KORE and MW perform the same number of comparisons, but the run-times differ significantly. The accelerated variants of KORE reduce the number of comparisons by a large margin and reduce the run-time further. In particular, the KORE$_{LSH-F}$ variant reduces the number of comparisons and the average run-time by an order of magnitude. For the 90%-quantile, which is relevant for being fast on the hardest cases, KORE$_{LSH-F}$ gains almost a factor of 20.

7. CONCLUSIONS

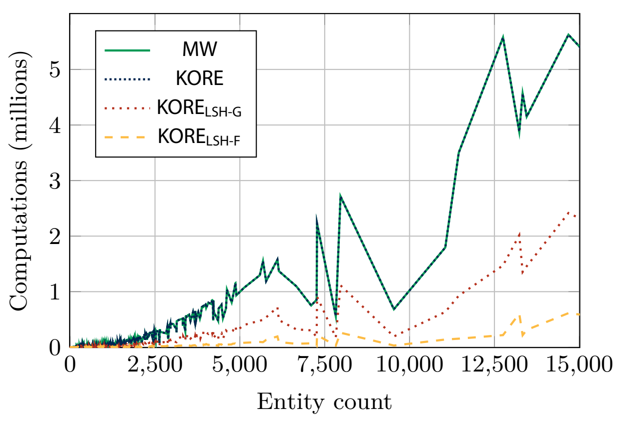
The keyphrase overlap relatedness measure KORE provides a high-quality notion of semantic relatedness of entities. In our experiments, KORE consistently performed at least as good as the best state-of-the-art methods, and often significantly better. Most importantly, KORE eliminates the reliance on explicit linkage between entities, and is thus geared for dealing with long-tail entities with few or no links. The approximate computation by the two-stage hashing method speeds up KORE and makes it suitable for online tasks that require interactive responses. These techniques also boost the throughput of KORE computations when applied in batch mode, e.g., for annotating an entire corpus (news, blogs, etc.) with NED mappings. Our future work will consider ap-

Method	Comparisons			Running time (s)		
	mean	stddev	0.9-quantile	mean	stddev	0.9-quantile
MW	898,253	3,578,524	1,594,226	5.335	27.357	9.738
KORE	898,253	3,578,524	1,594,226	1.884	6.592	3.519
KORE$_{\text{LSH-G}}$	315,302	1,306,293	548,561	1.285	3.925	2.308
KORE$_{\text{LSH-F}}$	60,580	301,712	87,240	0.413	1.557	0.586

Table 4: Results of timing experiments



(a) Running time



(b) Number of computations

Figure 4: Efficiency on CoNLL-YAGO

plications of these techniques in the growing space of Linked Open Data and for social media on the Web.

8. ACKNOWLEDGMENTS

We would like to thank Edwin Lewis-Kelham for his help with the WP dataset, and also Mohamed Amir Yosef and Marc Spaniol for their contributions to the KORE50 dataset.

9. REFERENCES

[1] E. Alfonseca, M. Pasca, and E. Robledo-Arnuncio. Acquisition of Instance Attributes via Labeled and Related Instances. In *Proceeding of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2010, Geneva, Switzerland*, pages 58–65, 2010.

[2] S. Auer, C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak, and Z. Ives. DBpedia: A Nucleus for a Web of Open Data. In *Proceedings of the 6th International Semantic Web and 2nd Asian Semantic Web Conference, ISWC 2007/ASWC2007, Busan, South Korea*, pages 722–735, 2007.

[3] A. Z. Broder, M. Charikar, A. M. Frieze, and M. Mitzenmacher. Min-Wise Independent Permutations. In *Proceedings of the 30th Annual ACM Symposium on the Theory of Computing, STOC 1998, Dallas, Texas, USA*, pages 327–336, 1998.

[4] A. Budanitsky and G. Hirst. Evaluating WordNet-based Measures of Lexical Semantic Relatedness. *Computational Linguistics*, 32(1), 2006.

[5] K. Chakrabarti, S. Chaudhuri, T. Cheng, and D. Xin. EntityTagger: Automatically Tagging Entities with Descriptive Phrases. In *Proceedings of the 20th International Conference Companion on World Wide Web, WWW 2011, Hyderabad, India*, pages 19–20, 2011.

[6] R. L. Cilibrasi and P. M. Vitanyi. The Google Similarity Distance. *IEEE Transactions on Knowledge and Data Engineering*, 19:370–383, 2007.

[7] D. Coppersmith, L. K. Fleischer, and A. Rurda. Ordering by Weighted Number of Wins Gives a Good Ranking for Weighted Tournaments. *Transactions on Algorithms*, 6(3), 2010.

[8] S. Cucerzan. Large-Scale Named Entity Disambiguation Based on Wikipedia Data. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, EMNLP-CoNLL 2007, Prague, Cech Republic*, pages 708–716, 2007.

[9] M. Dredze, P. McNamee, D. Rao, A. Gerber, and T. Finin. Entity Disambiguation for Knowledge Base Population. In *Proceedings of the 23rd International Conference on Computational Linguistics, COLING 2010, Beijing, China*, pages 277–285, 2010.

[10] P. Ferragina and U. Scaiella. TAGME: On-The-Fly Annotation of Short Text Fragments (by Wikipedia Entities). In *Proceedings of the 19th ACM Conference on Information and Knowledge Management, CIKM 2010, Toronto, Ontario, Canada*, pages 1625–1628, 2010.

[11] L. Finkelstein, E. Gabrilovich, Y. Matias, E. Rivlin, Z. Solan, G. Wolfman, and E. Ruppin. Placing Search

in Context: The Concept Revisited. *ACM Transactions on Information Systems*, 20(1):116–131, 2002.

[12] E. Gabrilovich and S. Markovitch. Computing Semantic Relatedness using Wikipedia-based Explicit Semantic Analysis. In *Proceedings of The 20th International Joint Conference for Artificial Intelligence, IJCAI 2007, Hyderabad, India*, pages 1606–1611, 2007.

[13] A. Gionis, P. Indyk, and R. Motwani. Similarity Search in High Dimensions via Hashing. In *Proceedings of 25th International Conference on Very Large Data Bases, VLDB 1999, Edinburgh, Scotland*, pages 518–529, Sept. 1999.

[14] S. Hassan and R. Mihalcea. Semantic Relatedness Using Salient Semantic Analysis. In *Proceedings of the 25th AAAI Conference on Artificial Intelligence, AAAI 2011, San Francisco, California, United States*, pages 884–889, 2011.

[15] J. Hoffart, F. M. Suchanek, K. Berberich, and G. Weikum. YAGO2: A Spatially and Temporally Enhanced Knowledge Base from Wikipedia. *Artificial Intelligence*, 2012.

[16] 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 *Conference on Empirical Methods in Natural Language Processing, EMNLP 2011, Edinburgh, Scotland, 2011*, pages 782–792, 2011.

[17] P. Indyk and R. Motwani. Approximate Nearest Neighbors: Towards Removing the Curse of Dimensionality. In *Proceedings of the 30th Annual ACM Symposium on the Theory of Computing, STOC 1998, Dallas, Texas, United States*, pages 604–613, 1998.

[18] A. Kotov and C. Zhai. Tapping into Knowledge Base for Concept Feedback: Leveraging ConceptNet to Improve Search Results for Difficult Queries. In *Proceedings of the 5th ACM International Conference on Web Search and Data Mining, WSDM 2012, Seattle, Washington, United States*, pages 403–412, 2012.

[19] S. Kulkarni, A. Singh, G. Ramakrishnan, and S. Chakrabarti. Collective Annotation of Wikipedia Entities in Web Text. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD 2009, Paris, France*, pages 457–466, 2009.

[20] M. Li, X. Chen, X. Li, B. Ma, and P. M. B. Vitányi. The Similarity Metric. *IEEE Transactions on Information Theory*, 50(12):3250–3264, 2004.

[21] R. Mihalcea and P. Tarau. TextRank: Bringing Order into Texts. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing, EMNLP 2004, Barcelona, Spain*, pages 404–411, 2004.

[22] D. Milne and I. H. Witten. An Effective, Low-Cost Measure of Semantic Relatedness Obtained from Wikipedia Links. In *Proceedings of the AAAI 2008 Workshop on Wikipedia and Artificial Intelligence, WIKIAI 2008, Chicago, Illinois, United States*, 2008.

[23] D. Milne and I. H. Witten. Learning to Link with Wikipedia. In *Proceedings of the 17th ACM Conference on Information and Knowledge Mining, CIKM 2008, Napa Valley, United States*, pages 509–518, 2008.

[24] R. Navigli. Word Sense Disambiguation: A survey. *ACM Comput. Surv.*, 41(2), 2009.

[25] P. Pantel and A. Fuxman. Jigs and Lures: Associating Web Queries with Structured Entities. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, ACL 2011, Portland, Oregon, United States*, pages 83–92, 2011.

[26] S. P. Ponzetto and R. Navigli. Knowledge-Rich Word Sense Disambiguation Rivaling Supervised Systems. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, ACL 2010, Uppsala, Sweden*, pages 1522–1531, 2010.

[27] S. P. Ponzetto and M. Strube. Knowledge Derived from Wikipedia for Computing Semantic Relatedness. *Journal of Artificial Intelligence Research*, 30(1):181–212, 2007.

[28] K. Radinsky, E. Agichtein, E. Gabrilovich, and S. Markovitch. A Word at a Time: Computing Word Relatedness using Temporal Semantic Analysis. In *Proceedings of the 20th International Conference on World Wide Web, WWW 2011, Hyderabad, India*, pages 337–346, 2011.

[29] D. Ravichandran, P. Pantel, and E. Hovy. Randomized Algorithms and NLP: Using Locality Sensitive Hash Functions for High Speed Noun Clustering. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics, ACL 2005, Ann Arbor, United States*, 2005.

[30] S. Singh, A. Subramanya, F. C. N. Pereira, and A. McCallum. Large-Scale Cross-Document Coreference Using Distributed Inference and Hierarchical Models. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, ACL 2011, Portland, Oregon, United States*, pages 793–803, 2011.

[31] R. Sinha and R. Mihalcea. Unsupervised Graph-based Word Sense Disambiguation Using Measures of Word Semantic Similarity. In *Proceedings of the 1st IEEE International Conference on Semantic Computing, ICSC 2007, Irvine, California, United States*, pages 363–369, 2007.

[32] F. M. Suchanek, G. Kasneci, and G. Weikum. YAGO: A Core of Semantic Knowledge. In *Proceedings of the 16th International Conference on World Wide Web, WWW 2007, Banff, Canada*, pages 697–706, 2007.

[33] T. Zesch, C. Müller, and I. Gurevych. Using Wiktionary for Computing Semantic Relatedness. In *Proceedings of the 23rd AAAI Conference on Artificial Intelligence, AAAI 2008, Chicago, Illinois, United States*, pages 861–867, 2008.