Semantic Annotation of Documents Based on Wikipedia Concepts

Janez Brank, Gregor Leban and Marko Grobelnik
Jožef Stefan Institute, Jamova 39, Ljubljana, Slovenia
E-mail: janez.brank@ijs.si, gregor.leban@ijs.si, marko.grobelnik@ijs.si

Keywords: semantic annotation, wikification, disambiguation, text mining

Received: January 30, 2017

Semantic annotation is the task of augmenting an unstructured textual document with semantic information, such as concepts from an ontology. In wikification, the Wikipedia is used as an ontology and its pages (articles) are regarded as (representations of) concepts. We describe an efficient approach for annotating a document with relevant concepts from the Wikipedia. A global disambiguation method based on constructing a mention-concept graph and computing pagerank over it is used to identify a coherent set of relevant concepts considering the input document as a whole. The presented approach is suitable for parallel processing and can support any language for which a sufficiently large Wikipedia is available. Several heuristics involved in the disambiguation of candidate annotations are discussed and an experimental evaluation of their influence is presented.

Povzetek: Semantično anotiranje je postopek, s katerim bi radi nestrukturirano besedilo dopolnili s semantičnimi informacijami, na primer s koncepti iz neke ontologije. Pri wikifikaciji se kot ontologijo uporablja Wikipedijo, pri čemer strani oz. članke v njej obravnavamo kot predstavitve konceptov. Opisujemo učinkovit pristop za anotiranje besedila z relevantnimi koncepti iz Wikipedije. Pri tem uporabljamo globalen pristop k razdvoumljanju, ki temelji na izgradnji grafa omemb in konceptov ter računanju pageranka na tem grafu, kar je nato podlaga za določitev nabora konceptov, ki se lepo skladajo med seboj in so relevantni glede na vhodno besedilo kot celoto. Opisani pristop je primeren za paralelno procesiranje in deluje za poljuben jezik, v katerem je na voljo dovolj velika Wikipedija. V članku predstavljamo in eksperimentalno ovrednotimo tudi več hevristik, ki se jih lahko uporabi pri razdvoumljanju kandidatov za anotacije.

1 Introduction

Recent years have seen a growth in the use of semantic technologies. However, in many contexts we still deal with largely unstructured textual documents that lack explicit semantic information which might be required for further processing with semantic technologies. This leads to the problem of semantic annotation or semantic enrichment as an important preparatory step before further processing of a document. Given a document and an ontology covering the domain of interest, the challenge is to identify concepts from that ontology that are relevant to the document or that are referred to by it, as well as to identify specific passages in the document where the concepts in question are mentioned.

A specific type of semantic annotation, known as *wikification*, involves using the Wikipedia as a source of possible semantic annotations [1][2]. In this setting, the Wikipedia is treated as a large and fairly general-purpose ontology: each page is thought of as representing a concept, while the relations between concepts are represented by internal hyperlinks between different Wikipedia pages, as well as by Wikipedia's category memberships and cross-language links.

The advantage of this approach is that the Wikipedia is a freely available source of information, it covers a wide range of topics, has a rich internal structure, and each concept is associated with a semi-structured textual document (i.e. the content of the corresponding Wikipedia article) which can be used to aid in the process of semantic annotation. Furthermore, the Wikipedia is available in a number of languages, with cross-language links being available to identify pages that refer to the same concept in different languages, thus making it easier to support multilingual and cross-lingual annotation.

The remainder of this paper is structured as follows. In Section 2, we present the pagerank-based approach to wikification used in our wikifier. In Section 3, we describe our implementation and some experimental evaluaton. Section 4 contains conclusions and a discussion of possible future work.

2 Pagerank-based Wikification

The task of wikifying an input document can be broken down into several closely interrelated subtasks: (1) identify phrases (or words) in the input document that refer to a Wikipedia concept; (2) determine which concept exactly a phrase refers to; (3) determine which concepts are relevant enough to the document as a whole that they should be included in the output of the system (i.e. presented to the user).

We follow the approach described by Zhang and Rettinger [1]. This approach makes use of the rich internal structure of hyperlinks between Wikipedia pages. A hyperlink can be thought of as consisting of a source page, a target page, and the link text (also known as the anchor text). If a source page contains a link with the anchor text a and the target page *t*, this is an indication that the phrase a might be a reference to (or representation of) the concept that corresponds to page *t*. Thus, if the input document that we're trying to wikify contains the phrase *a*, it might be the case that this occurrence of *a* in the input document also constitutes a mention of the concept *t*, and the concept *t* is a candidate annotation for this particular phrase.

2.1 Disambiguation

In the Wikipedia, there may be many different links with the same anchor text *a*, and they might not all be pointing to the same target page. For example, in the English-language Wikipedia, there are links with *a* = "Tesla" that point to pages about the inventor, the car manufacturer, the unit in physics, a band, a film, and several other concepts.

Thus, when such a phrase a occurs in an input document, there are several concepts that can be regarded as candidate annotations for that particular mention, and we have to determine which of them is actually relevant. This is the problem of disambiguation, similar to that of word sense disambiguation in natural language processing.

There are broadly two approaches to disambiguation, local and global. In the local approach, each mention is disambiguated independently of the others, while the global approach aims to treat the document as a whole and disambiguate all the mentions in it as a group. The intuition behind the global approach is that the document that we're annotating is about some topic, and the concepts that we use as annotation should be about that topic as well. If the document contains many mentions that include, as some of their candidate annotations, some car-related concepts, this makes it more likely that we should treat the mention of "Tesla" as a reference to Tesla the car manufacturer as opposed to e.g. a reference to Nikola Tesla or to Tesla the rock band.

2.2 The mention-concept graph

To implement the global disambiguation approach, our Wikifier begins by constructing a *mention-concept graph* for the input document. (Some authors, e.g. [2], refer to this as a *mention-entity* graph, but we prefer to use the term "mention-concept graph" as some of the Wikipedia pages do not necessarily correspond to concepts that we usually think of as entities, and our wikifier does not by default try to exclude them.) This can be thought of as a bipartite graph in which the left set of vertices corresponds to mentions and the right set of vertices corresponds to concepts. A directed edge $a \to c$ exists if and only if the concept *c* is one of the candidate annotations for the mention *a* (i.e. if, in the Wikipedia, there exists a hyperlink with the anchor text *a* and the target *c*). A transition probability is also assigned to each such edge, $P(a \to c)$, defined as the ratio [number of hyperlinks, in the Wikipedia, having the anchor text *a* and the target *c*] / [number of hyperlinks, in the Wikipedia, having the anchor text *a*].

This graph is then augmented by edges between concepts, the idea being that an edge $c \to c'$ should be used to indicate that the concepts c and c' are "semantically related", in the sense that if one of them is relevant to a given input document, the other one is also more likely to be relevant to that document. (For example, the semantic relationship between the concepts "Electric vehicle" and "Tesla Inc." should be much stronger than between the concepts "Electric vehicle" and "Tesla (rock band)". This measure of semantic relatedness will be used subsequently to encourage the formation of a group of annotations that are semantically related in the sense that they refer to the same topic, which is hopefully also the topic of the document. This would encourage mentions of "Tesla" in a document about electric cars to be annotated with the concept "Tesla Inc." rather than "Tesla (rock band)".

Following [1], the internal link structure of the Wikipedia is used to calculate a measure of semantic relatedness. Informally, the idea is that if c and c' are closely related, then other Wikipedia pages that point to c are likely to also point to c' and vice versa. Let L_c be the set of Wikipedia pages that contain a hyperlink to *c*, and let N be the total number of concepts in the Wikipedia; then the semantic relatedness of c and c' can be defined as

$$SR(c, c') = 1 - \frac{\log \max\{|L_c|, |L_{c'}|\} - \log|L_c \cap L_{c'}|}{\log N - \log \min\{|L_c|, |L_{c'}|\}}.$$

Intuitively, this formula considers two concepts to be semantically related if pages that link to one of them typically also link to the other one (and vice versa). More specifically, SR will be higher if the overlap (i.e. the intersection) of L_c and $L_{c'}$ is large (relative to the size of L_c and $L_{c'}$), and the formula also rewards situations where the sets L_c and $L_{c'}$ are themselves large (relative to the overall number of documents N), as this means that the dataset includes more evidence of a semantic relationship between c and c'.

In the graph, we add an edge of the form $c \to c'$ wherever the semantic relatedness $SR(c, c')$ is > 0. The transition probability of this edge is defined as proportional to the semantic relatedness: $P(c \to c') = SR(c, c') / \Sigma_{c''} SR(c, c'')$.

This graph is then used as the basis for calculating a vector of pagerank scores [3], one for each vertex. This is done using the usual iterative approach where in each iteration, each vertex distributes its pagerank score to its immediate successors in the graph, in proportion to the transition probabilities on its outgoing edges:

$$PR_{new}(u) = \tau\, PR_0(u) + (1 - \tau)\, \Sigma_v\, PR_{old}(v)\, P(v \to u).$$

The baseline distribution of pagerank, PR_0, is used both to help the process converge and also to counterbalance the fact that in our graph there are no edges pointing into the mention vertices. In our case, $PR_0(u)$ is defined as 0 if u is a concept vertex; if u is a mention vertex, we use $PR_0(u) = z \cdot$ [number of

Wikipedia pages containing the phrase u as the anchor-text of a hyperlink] / [number of Wikipedia pages containing the phrase u], where z is a normalization constant to ensure that $\Sigma_u PR_0(u) = 1$. We used $\tau = 0.1$ as the stabilization parameter.

The intuition behind this approach is that in each iteration of the pagerank calculation process, the pagerank flows into a concept vertex c from mentions that are closely associated with the concept c and from other concepts that are semantically related to c. Thus after a few iterations, pagerank should tend to accumulate in a set of concepts that are closely semantically related to each other and that are strongly associated with words and phrases that appear in the input document, which is exactly what we want in the context of global disambiguation.

2.3 Using pagerank for disambiguation

Once the pagerank values of all the vertices in the graph have been calculated, we use the pagerank values of concepts to disambiguate the mentions. If there are edges from a mention a to several concepts c, we choose the concept with the highest pagerank as the one that is relevant to this particular mention a. We say that this concept is *supported* by the mention a. At the end of this process, concepts that are not supported by any mention are discarded as not being relevant to the input document.

The remaining concepts are then sorted in decreasing order of their pagerank. Let the i'th concept in this order be c_i and let its pagerank be PR_i, for $i = 1, \ldots, n$. Concepts with a very low pagerank value are less likely to be relevant, so it makes sense to apply a further filtering step at this point and discard concepts whose pagerank is below a user-specified threshold. However, where exactly this threshold should be depends on whether the user wants to prioritize precision or recall. Furthermore, the absolute values of pagerank can vary a lot from one document to another, e.g. depending on the length of the documents, the number of mentions and candidate concepts, etc. Thus we apply the user-specified threshold in the following manner: given the user-specified threshold value $\theta \in [0, 1]$, we output the concepts $c_1, \ldots, c_m$, where m is the least integer such that $\Sigma_{i=1..m} PR_i^2 \geq \theta \ \Sigma_{i=1..n} PR_i^2$. In other words, we report as many top-ranking concepts as are needed to cover θ of the total sum of squared pageranks of all the concepts. We use $\theta = 0.8$ as a broadly reasonable default value, though the user can require a different threshold depending on their requirements.

The motivation for using squares of pageranks instead of the pageranks themselves is to put a greater emphasis on the annotations with the highest values of pagerank, while culling the lower-scoring annotations more thoroughly. In our preliminary experiments, this led to a small improvement in performance compared to using the sums of pageranks without squaring them.

For each reported concept, we also output a list of the mentions that support it.

2.4 Treatment of highly ambiguous mentions

Our wikifier supports various minor heuristics and refinements in an effort to improve the performance of the baseline approach described in the preceding sections.

As described above, anchor text of hyperlinks in the Wikipedia is used to identify mentions in an input document (i.e. words or phrases that may support an annotation). One downside of this approach is that some words or phrases occur as the anchor text of a very large number of hyperlinks in the Wikipedia and these links point to a large number of different Wikipedia pages. In other words, such a phrase is highly ambiguous; it is not only unlikely to be disambiguated correctly, but also introduces noise into the mention-concept graph by introducing a large number of concept vertices, the vast majority of which will be completely irrelevant to the input document. This also slows down the annotation process by increasing the time to calculate the semantic relatedness between all pairs of candidate concepts. (As an example of such a highly ambiguous mention, consider the word "Country". Most of the time, when it appears as the anchor-text of a link, it's a link to the concepts "Country" or "Country music", but it also occurs in links to more than a hundred other concepts, mostly individual countries.)

We use several heuristics to deal with this problem. Suppose that a given mention a occurs, in the Wikipedia, as the anchor text of n hyperlinks pointing to k different target pages, and suppose that n_i of these links point to page c_i (for $i = 1, \ldots, k$). We can now define the entropy of the mention a as the amount of uncertainty regarding the link target given the fact that its anchor text is a: $H(a) = - \Sigma_{i=1..k} (n_i/n) \log(n_i/n)$. If this entropy is above a user-specified threshold (e.g. 3 bits), we completely ignore the mention as being too ambiguous to be of any use. For mentions that pass this heuristic, we sort the target pages in decreasing order of n_i and use only the top few of them (e.g. top 20) as candidates in our mention-concept graph. A third heuristic is to ignore candidates for which n_i itself is below a certain threshold (e.g. $n_i < 2$), the idea being that if such a phrase occurs only once as the anchor text of a link pointing to that candidate, this may well turn out to be noise and is best disregarded.

Optionally, the Wikifier can also be configured to ignore certain types of concepts based on their Wikidata class membership. This can be useful to exclude from consideration Wikipedia pages that do not really correspond to what is usually thought of as entities (e.g. "List of…" pages).

Another heuristic that we have found useful in reducing the noise in the output annotations is to ignore any mention that consists entirely of stopwords and/or very common words (top 200 most frequent words in the Wikipedia for that particular language). For this as well as for other purposes the text processing is done in a case-sensitive fashion, which e.g. allows us to ignore spurious links with the link text "the" while processing those that refer to the band "The The".

2.5 Miscellaneous heuristics

Semantic relatedness. As mentioned above, the definition of semantic relatedness of two concepts, $SR(c, c')$, is based on the overlap between the sets L_c, $L_{c'}$ of immediate predecessors of these two concepts in the Wikipedia link graph. Optionally, our Wikifier can compute semantic relatedness using immediate successors or immediate neighbours (i.e. both predecessors and successors) instead of immediate predecessors. However, our preliminary experiments indicated that these changes do not lead to improvements in performance, so they are disabled by default.

Extensions to disambiguation. Our Wikifier also supports some optional extensions of the disambiguation process. As described above, the default behavior when disambiguating a mention is to simply choose the candidate annotation with the highest pagerank value. Alternatively, after any heuristics from section 2.4 have been applied, the remaining candidate concepts can be re-ranked using a different scoring function that takes other criteria besides pagerank into account. This is an opportunity to combine the global disambiguation approach with some local techniques. In general, a scoring function of the following type is supported:

$$score(c|a) = w_1\, f(P(c|a))\, PR(c) + w_2\, S(c, d) + w_3\, LS(c, a) \quad (1)$$

Here, a is the mention that we're trying to disambiguate, and c is the candidate concept that we're evaluating. $P(c|a)$ is the probability that a hyperlink in the Wikipedia has c as its target conditioned on the fact that it has a as its anchor text. $f(x)$ can be either 1 (the default), x, or $\log(x)$. $PR(c)$ is the pagerank of c's vertex in the mention-concept graph. $S(c, d)$ is the cosine similarity between the text of the input document d and of the Wikipedia page for the concept c. $LS(c, a)$ is the cosine similarity between the context (e.g. previous and next 3 words) in which a appears in the input document d, and the contexts in which hyperlinks with the target c appear in the Wikipedia. Finally, w_1, w_2, w_3 are weight constants. However, our preliminary experiments haven't shown sufficient improvements from the addition of these heuristics, so they are disabled by default ($f(x) = 1$, $w_2 = w_3 = 0$) to save computational time and memory (storing the link contexts needed for the efficient computation of LS has turned out to be particularly memory intensive).

3 Implementation and evaluation

3.1 Implementation

Our implementation of the approach described in the preceding section is running as a web service and can be accessed at http://wikifier.org. The approach is suitable for parallel processing as annotating one document is independent of annotating other documents, and any shared data used by the annotation process (e.g. the Wikipedia link graph, and a trie-based data structure that indexes the anchor text of all the hyperlinks) need to be accessed only for reading and can thus easily be shared by an arbitrary number of worker threads. This allows for a highly efficient processing of a large number of documents.

The only need to modify shared data structures arises when a new dump of the Wikipedia becomes available (the Wikipedia publishes new dumps of its content twice per month). We use a separate process to periodically check the Wikipedia web site for new dumps, download them, parse them, and build indexes in a form that can be used by our wikifier. Once the wikifier web service is notified of the availability of a new index, it loads its contents into memory, temporarily stops issuing new requests to worker threads, waits for them all to finish processing their current requests, and then updates the shared data structures to include the new index and discard the old one. In this way, new indices can be brought online without shutting down the service and with a minimal interruption to its availability.

Our implementation currently processes on average more than 500,000 requests per day (the total length of input documents averages about 1.2 GB per day), including all the documents from the JSI Newsfeed service [4]. The output is used among other things as a preprocessing step by the Event Registry system [5]. The wikifier currently supports all languages in which a Wikipedia with at least 1000 pages is available, amounting to a total of 134 languages. Admittedly, 1000 pages is much too small to achieve an adequate coverage; however, about 60 languages have a Wikipedia with at least 100,000 pages, which is enough for many practical applications.

Annotations are returned in JSON format and can optionally include detailed information about support (which mentions support each annotation), alternative candidate annotations (concepts that were considered as candidates during the disambiguation process but were rejected in favour of some other higher scored concept), and WikiData/DbPedia class membership of the proposed annotations. Thus, the caller can easily implement any desired class-based post-processing.

Our wikifier also allows the user to define custom vocabularies that can be used to generate annotations in addition to the Wikipedia-based annotations described so far. A custom vocabulary is a set of concepts where each concept consists of an ID and a set of one or more words of phrases which, if they appear in the input document, trigger the inclusion of this concept among the annotations. This allows the user to extend the system with custom sets of annotations, but the downside is that such custom annotations are not part of the Wikipedia and thus cannot be included in the usual wikification process, especially not in the pagerank-based disambiguation algorithm.

As a preprocessing step, the user may specify one or more sets of "alternative labels", which are really rewriting rules of the form "$w_1\ w_2\ \dots\ w_n \rightarrow x_1\ x_2\ \dots\ x_m$" indicating that the sequence of the words $w_1\ w_2\ \dots\ w_n$ may, if it occurs in the input document, be replaced by the sequence $x_1\ x_2\ \dots\ x_m$ prior to the main part of the wikification algorithm. (The word "may" in the preceding sentence means that the original sequence of words from the left-hand side of the rule is also kept in the document. Thus, the document is no longer a simple sequence of words, but may gradually turn into an arbitrary directed acyclic graph, the various paths through which indicate different alternatives into which the text of the document may be brought through the application of the rewriting rules.) Owing to such transformations, certain candidate mentions might appear in the document that did not appear in the original document. Several such rules may be applied one after another and may affect the same part(s) of a document. Theoretically, such rewriting rules form a Turing-complete formalism, and to keep the problem tractable our wikifier makes only three passes through the document to look for occurrences of left-side word sequences and replace them with the corresponding right-side word sequences. Currently the main use of this mechanism in our wikifier is to provide additional alternative spellings of some proper names in cases where these are not adequately covered in the Wikipedia. This has been found to be particularly useful in case of names transliterated from languages that use a different script and where several different transliteration schemes are in use.

3.2 Evaluation

One way to evaluate wikification is to compare the set of annotations with a manually annotated gold standard for the same document(s). Performance can then be measured using metrics from information retrieval, such as precision, recall, and the F_1-measure, which is defined as the harmonic mean of precision and recall. We used two manually annotated datasets:

(Dataset 1.) A set of 1393 news articles that was made available from the authors of the AIDA system and was originally used in their experiments [2]. This manually annotated dataset excludes, by design, any annotations that do not correspond to named entities. Since our wikifier does not by default distinguish between named entities and other Wikipedia concepts, we have explicitly excluded concepts that are not named entities (based on their class membership in the WikiData ontology) from the output of our Wikifier for the purposes of this experiment.

(Dataset 2.) A set of 491 news articles taken randomly from the JSI Newsfeed [4] on 29 June 2016 and annotated manually with relevant Wikipedia concepts. Unlike the first dataset, the annotations here included concepts that were not named entities.

In addition to our wikifier, we obtained annotations from the following systems: AIDA [2], Waikato Wikipedia Miner [7], Babelfy [8], Illinois [9], and DbPedia Spotlight [10]. The Waikato system is not included in experiments involving dataset 2 as their web service was no longer available at the time.

Tables 1(*a*) and 1(*b*) show the agreement not only between each of the wikifiers and the gold standard, but also between each pair of wikifiers (the lower left triangle of the matrix is left empty as it would be just a copy of the upper right triangle, since the F_1-measure is symmetric). As this experiment indicates, on the first dataset (the AIDA dataset) our wikifier ("JSI" in the table) performs slightly worse than AIDA but significantly better than the other wikifiers. On the second dataset (the JSI dataset), the best performance was achieved by the Babelnet wikifier, ours is slightly worse while AIDA is significantly worse on this dataset. Thus, overall we can conclude that our wikifier has solid performance over a pair of two considerably different dataset. Furthermore, experiments on both datasets show that there is relatively little agreement between different wikifiers, which indicates that wikification itself is in some sense a vaguely defined task where different people can have very different ideas about whether a particular Wikipedia concept is relevant to a particular input document (and should therefore be included as an annotation) or not, which types of Wikipedia concepts can be considered as annotations (e.g. only named entities or all concepts), etc. Possibly the level of agreement could be improved by fine-tuning the settings of the various wikifiers; in the experiment described above, default settings were used.

	Gold	JSI	AIDA	Waikato	Babelfy	Illinois	Spotlight
Gold	1.000	0.593	0.723	0.372	0.323	0.476	0.279
JSI		1.000	0.625	0.527	0.431	0.489	0.363
AIDA			1.000	0.372	0.352	0.434	0.356
Waikato				1.000	0.481	0.564	0.474
Babelfy					1.000	0.434	0.356
Illinois						1.000	0.376
Spotlight							1.000

Table 1(*a*): F_1 measure of agreement between the various wikifiers and the gold standard on dataset 1.

	Gold	JSI	AIDA	Babelfy	Illinois	Spotlight
Gold	1.000	0.378	0.197	0.417	0.372	0.282
JSI		1.000	0.278	0.360	0.413	0.397
AIDA			1.000	0.206	0.283	0.383
Babelfy				1.000	0.380	0.282
Illinois					1.000	0.367
Spotlight						1.000

Table 1(*b*): F_1 measure of agreement between the various wikifiers and the gold standard on dataset 2.

We also conducted a small experiment on dataset 2 to compare two forms of the thresholding criterion: one is based on the sums of squares of pageranks (as currently described in Section 2.3) and one based on the sums of the pageranks themselves. The F_1-measure between our annotations and the gold standard drops from 0.378 when using squared pageranks to 0.344 when using the pageranks directly. We used squared pageranks for thresholding in all other experiments in this section.

Evaluation of disambiguation heuristics. In the following experiment, we evaluate some of the additional disambiguation heuristics described in Section 2.5. The purpose of the experiment was to find the best-performing combination of the following heuristics and parameters from that section:

(*i*) Logarithmic link counts: in Section 2.2, we defined the transition probability $a \rightarrow c$ in the mention-concept graph as being proportional to the number of links, in the Wikipedia, with the anchor text a and the target c (the "link count" of c given a). Alternatively, it can be defined as being proportional to the *logarithm* of this link count. The purpose of this heuristic is to provide a kind of smoothing and discourage too much of the pagerank score from flowing into just one candidate c for that particular mention a. The Wikipedia is known for having various biases in terms of how frequently certain topics are covered, so this sort of smoothing may soften the more extreme differences in the frequency of coverage while still preserving some information about which concepts c are associated more often with a phrase a.

(*ii*) Set of links used in the computation of sematic relatedness (*SR*) between two concepts in the Wikipedia link graph: this can be the in-links (the default setting), out-links, or all neighbours.

(*iii*) Threshold for re-ranking: in this scenario, the candidates c for a given mention a are first sorted by pagerank, the top few candidates are kept and are then re-ranked using the more detailed (and computationally-intensive) scoring function denoted by eq. (1). The question then is what counts as "top few candidates" to be included in the re-ranking. We define this by introducing a parameter $\vartheta \in [0, 1]$ such that a candidate c proceeds to re-ranking if its pagerank is $PR(c) \geq \vartheta \max_{c'} PR(c')$, where c' goes over all the candidates for the current mention a.

(*iv*) Linearization of pagerank in the scoring function denoted by eq. (1) into a linear rank: instead of using the pagerank directly, all the candidate concepts c for a given mention a are sorted by pagerank and a linear rank is assigned to each. If there are k candidates, the i'th of them in this order gets a linear rank of i/k. This is then used instead of $PR(c)$ in eq. (1), as well as in the ϑ-based thresholding criterion in the previous paragraph (where ϑ then simply becomes the proportion of candidates that proceeds to the re-ranking phase). The purpose is to make sure that the range [0, 1] is covered evenly, instead of the pagerank values possibly being clustered in a small part of that range.

(*v*) Weight w_2 of the cosine similarity between the input document and the Wikipedia page of a candidate concept, in the scoring function of eq. (1). (The weight w_1 of the candidate concept's pagerank value in the scoring function was then set to $1 - w_2$. The weight w_3 of the link context similarity was kept to 0 throughout these experiments, because of the considerable additional memory and time consumption required for the link context computation and because preliminary experiments indicated that the results were not promising.)

The possible values of these five parameters that were investigated in this experiment can be summarized as follows:

(*i*) linkCounts $\in$ {normal, log}
(*ii*) SR $\in$ {in, out, all}
(*iii*) $\vartheta \in \{0, 0.2, 0.4, 0.6, 0.8, 1\}$
(*iv*) PR $\in$ {normal, linearized}
(*v*) $w_2 \in \{0, 0.25, 0.5, 0.75, 1\}$

The default settings are: normal link counts, SR = in, $\vartheta = 1$ (no second-stage re-ranking), PR = normal, $w_2 = 0$.

Table 2 shows, for each possible value of each parameter, the best and the average performance (in terms of F_1-measure relative to the gold standard) that can be achieved by fixing that parameter to that value and allowing the other parameters to range over all the possible values indicated above.

For comparison, the last two rows show the performance with no parameters fixed (allowing us to tune the best possible combination of all parameters) and the performance with all parameters fixed at their default values.

This experiment was done on Dataset 1 and used 10-fold cross-validation. Nine folds (the training set) were used to tune any parameters that were not held fixed and the best resulting combination of parameters was then evaluated on the tenth fold (the test set). This was repeated for all ten choices of the test fold. Table 2 shows the average and the standard deviation of the F_1 performance on the test fold over all 10 choices of the test fold.

As we can see from this experiment, it is indeed possible to achieve a small improvement in performance by employing some of the heuristics described here. The best-performing combination of parameters was {normal link counts, SR = in, $\vartheta = 0.6$, PR = normal, $w_2 = 1$}, which resulted in an F_1 score of 0.6152, up from the score of 0.5917 achieved by the default parameter values. A paired t-test showed this difference to be significant at a p-value of 0.0005. However, we can also see that, in practical terms, this improvement is small and might not be noticed by the user. Furthermore, it is clear that several of the heuristics employed were in fact counterproductive: using logarithms of link counts to define the mention-concept transition probabilities; using out-links or all neighbors (instead of just in-links) in the definition of semantic relatedness; and using linearized pageranks. Shifting these parameters away from their default settings in fact led to a deterioration of F_1 (in all these cases, the deterioration is statistically significant with a p-value of 0.001 or less.) Improvements in performance mostly came from re-ranking the most promising candidates ($\vartheta = 0.6$) based on the cosine similarity between the input document and the Wikipedia pages of the candidate concepts.

The "Average F_1" column of Table 2 shows that no parameter by itself can ensure good performance unless the other parameters are also chosen suitably, as the average performance over all the combinations of other parameters is poor regardless of which parameter has been fixed and at what value.

4 Use in a real-life application

Semantically annotating documents can be of high importance in several real-life applications. An example of such an application is Event Registry [5]. Event Registry is a system that collects and analyzes news content generated globally and identifies the world

Parameter	Avg. F_1	Max. F_1
linkCounts = normal	0.5883 ± 0.0253	0.6152 ± 0.0239
linkCounts = log	0.5368 ± 0.0268	0.5833 ± 0.0221
SR = in	0.5800 ± 0.0232	0.6152 ± 0.0239
SR = out	0.5626 ± 0.0265	0.5945 ± 0.0253
SR = all	0.5451 ± 0.0283	0.5927 ± 0.0268
PR = normal	0.5644 ± 0.0261	0.6152 ± 0.0239
PR = linearized	0.5607 ± 0.0259	0.5978 ± 0.0223
$\vartheta = 0$	0.5604 ± 0.0256	0.5974 ± 0.0223
$\vartheta = 0.2$	0.5614 ± 0.0256	0.5982 ± 0.0221
$\vartheta = 0.4$	0.5634 ± 0.0258	0.6054 ± 0.0225
$\vartheta = 0.6$	0.5649 ± 0.0259	0.6152 ± 0.0239
$\vartheta = 0.8$	0.5642 ± 0.0260	0.6004 ± 0.0216
$\vartheta = 1$	0.5610 ± 0.0273	0.5939 ± 0.0280
$w_2 = 0$	0.5610 ± 0.0273	0.5939 ± 0.0280
$w_2 = 0.25$	0.5646 ± 0.0266	0.5979 ± 0.0209
$w_2 = 0.5$	0.5655 ± 0.0265	0.6060 ± 0.0228
$w_2 = 0.75$	0.5654 ± 0.0253	0.6128 ± 0.0248
$w_2 = 1$	0.5563 ± 0.0245	0.6152 ± 0.0239
Nothing fixed	0.5626 ± 0.0260	**0.6152 ± 0.0239**
All fixed to default values	0.5917 ± 0.0226	0.5917 ± 0.0226

Table 2: F_1 measure of agreement between our wikifier and the gold standard while keeping one parameter fixed and tuning the others. "Avg. F_1" shows average performance over all possible combinations of non-fixed parameters; "Max. F_1" shows the best performance achieved by tuning the non-fixed parameters on the training folds. Both columns show the F_1 performance on the test fold. Since cross-validation was used, the performances are shown in the form "average ± standard deviation" over all 10 possible splits of the data into 9 training folds and 1 test fold.

events mentioned in the news. As the application aims to extract knowledge in structured form from the unstructured text, we will now describe how the Wikifier's semantic annotations provide the critical input required by the system.

For each news article, Event Registry stores the list of identified semantic annotations. Among other things, this allows the users to search for news content using the semantic tags and not keywords, as we are used to in the search engines. The main advantages of using tags versus keywords are that one can e.g. (*a*) specifically ask for articles about apple, the fruit, versus Apple, the company, (*b*) find articles about IBM regardless of how it's mentioned in the news articles ("IBM", "I.B.M.", "International Business Machines", etc.), and (*c*) find articles about White House in any language. The last use case is available because the Wikipedia also maintains information on which Wikipedia pages in different languages represent the same concept. Consequently, the tag for "White House" in an English article will be the same, as the tag for "Casa Blanca" in a Spanish article.

When Event Registry identifies a group of news articles that represent the same event, it uses the semantic information in the news articles to determine the core event information. First, it analyzes all news articles in the event and calculates how frequently individual concepts appear in these articles. A ranked list of commonly mentioned concepts is then used as a semantic summary or a "fingerprint" of an event.

Another critical piece of information about the event is its geographical location. In order to determine the location, Event Registry again analyzes concepts mentioned in the news articles about the event, and considers as possible candidates only those that refer to a geographical location. For each candidate location, a set of learning features is extracted. The learning features that we extract are as follows:

- Mentions of the location in the articles about the event. The value is simply the ratio of the number of news articles about the event that mention the location somewhere in the text and the total number of articles about the event.
- Mentions of the location in the dateline (beginning of the article). This feature is computed as the ratio of the number of articles about the event in which the location is mentioned in the dateline and the total number of articles about the event.
- Normalized versions of the previous two features. In this case we compute a variation of the previous two features, where we don't compute a simple ratio, but weight the contribution of an individual article by the cosine similarity of the article to the centroid of the event. Articles closer to the centroid (more relevant articles) therefore contribute more to the final feature value.
- Commonality of the location — how frequently is the location generally present in the news articles. The value is computed as the ratio of the number of articles in Event Registry that mention this location and the total number of articles.

Based on these features, a logistic regression model computes a probability for each of the candidate locations to be the location of the event. If the location with the highest probability is above the predetermined threshold, the location is chosen as the location of the event. The logistic model was trained on 1239 manually labeled events and has 96.2% classification accuracy.

Semantic annotations are also of high importance when a search is performed and a large number of results need to be summarized. An example of such a summary is displayed in Figure 1, where we searched for events about hurricanes. The resulting list that contained over 23 000 events was summarized as shown in the figure to illustrate what are the top concepts mentioned in these events.

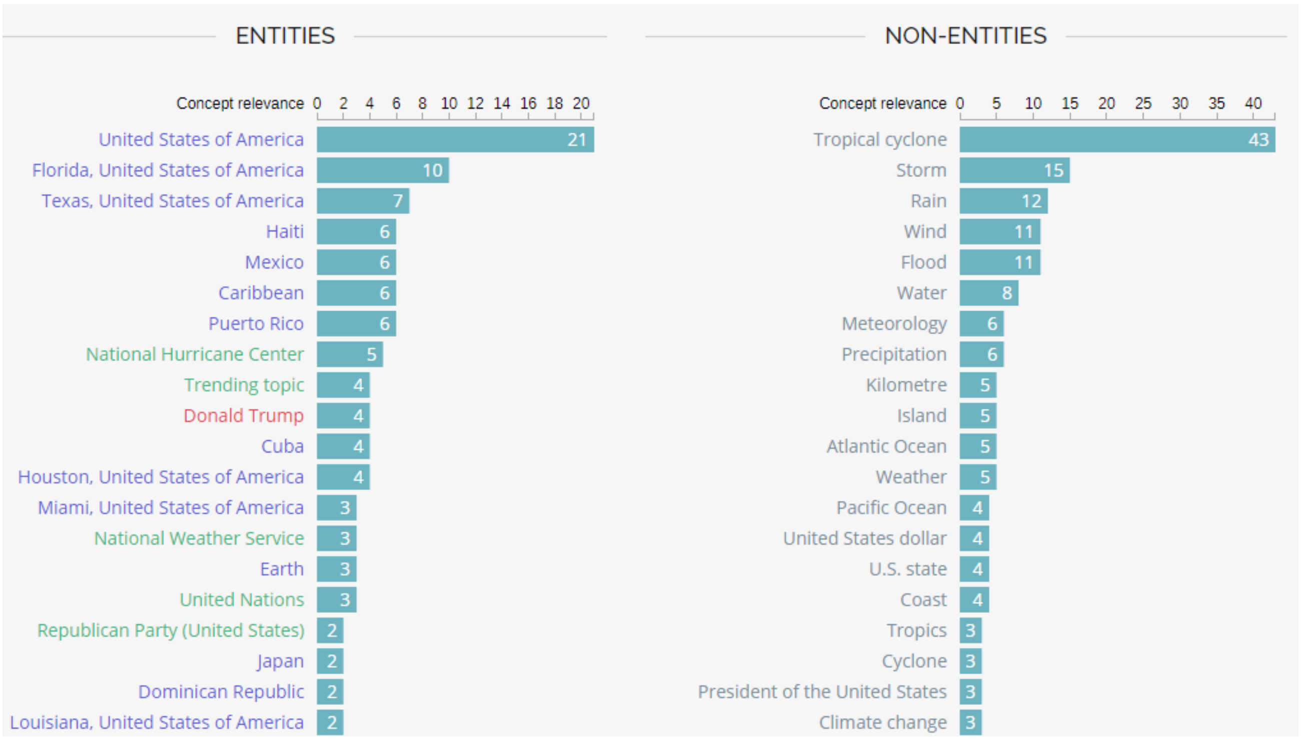

Figure 1: Summary of top concepts in events about hurricanes.

5 Conclusions and future work

We have presented a practical and efficient approach to wikification that requires no external data except the Wikipedia itself, can deal with documents in any language for which the Wikipedia is available, and is suitable for a high-performance, parallelized implementation.

The approach presented here could be improved along several directions. One significant weakness of the current approach concerns the treatment of minority languages. When dealing with a document in a certain language, we need hyperlinks whose anchor text is in the same language if we are to identify mentions in this input document. Thus, if the document is in a language for which the Wikipedia is not available, it cannot be wikified using this approach; and similarly, if the Wikipedia is available in this language but is small, with a small amount of text, low number of pages, and generally poor coverage, the performance of wikification will be low. One idea to alleviate this problem is to optionally allow a second stage of processing, in which Wikipedias in languages other than the language of the input document would also be used to identify mentions and provide candidate annotations. This might particularly improve the coverage of concepts that are referred to by the same words or phrases across multiple languages, as is the case with some types of named entities. For the purposes of pagerank-based disambiguation in this second stage, a large common link-graph would have to be constructed by merging the link-graphs of the Wikipedias for different languages. This can be done by using the cross-language links which are available in the WikiData ontology, providing information about when different pages in different languages refer to the same concept.

Another interesting direction for further work would be to incorporate local disambiguation techniques as a way to augment the current global disambiguation approach. When evaluating whether a mention *a* in the input document refers to a particular concept *c*, the local approach would focus on comparing the context of *a* to either the text of the Wikipedia page for *c*, or to the context in which hyperlinks to *c* occur within the Wikipedia. Preliminary steps taken in this direction in Sec. 2.5 did not lead to improvements in performance, but this subject is worth exploring further. Instead of the bag-of-words representation of contexts, other vector representations of words could be used, e.g. word2vec [6].

6 Acknowledgement

This work was supported by the Slovenian Research Agency as well as the euBusinessGraph (ICT-732003-IA), EW-Shopp (ICT-732590-IA) and RENOIR (H2020-MSCA-RISE-691152) projects.

7 References

[1] L. Zhang, A. Rettinger. *Final ontological word-sense-disambiguation prototype.* Deliverable D3.2.3, xLike Project, October 2014.

[2] J. Hoffart, M. A. Yosef, I. Bordino, *et al.* Robust disambiguation of named entities in text. *Proc. of the 2011 Conf. on Empirical Methods in Natural Language Processing*, Edinburgh, Scotland, 2011, pp. 782–792.

[3] L. Page, S. Brin, R. Motwani, T. Winograd. *The PageRank citation ranking: Bringing order to the web*. Digital Libraries Project Report SIDL-WP-1999-0120, Stanford University, 1998.

[4] M. Trampuš, B. Novak. Internals of an aggregated web news feed. *Proc. SiKDD 2012*.

[5] G. Leban, B. Fortuna, J. Brank, M. Grobelnik. Event registry: Learning about world events from news. *Proc. of the 23rd Int. Conf. on the World Wide Web* (WWW 2014), pp 107–110.

[6] T. Mikolov, K. Chen, G. Corrado, J. Dean. *Efficient estimation of word representations in vector space*. Arxiv.org, 1301.3781 [cs.CL], 2013.

[7] D. Milne, I. H. Witten. An open-source toolkit for mining Wikipedia. *Artificial Intelligence*, 194:222–239 (January 2013).

[8] A. Moro, A. Raganato, R. Navigli. Entity linking meets word sense disambiguation: A unified approach. *Trans. of the Assoc. for Comp. Linguistics*, 2:231–234 (2014).

[9] L. Ratinov, D. Roth, D. Downey, M. Anderson. Local and global algorithms for disambiguation to Wikipedia. *Proc. of the 49th Annual Meeting of the Assoc. for Comp Linguistics: Human Language Technologies* (2011), pp. 1375–84.

[10] J. Daiber, M. Jakob, C. Hokamp, P. N. Mendes. Improving efficiency and accuracy in multilingual entity extraction. *Proc. of the 9th Int. Conf. on Semantic Systems*, 2013.