Fast Text Annotation with Linked Data

Viktor, Ivannikov
ISPRAS
Moscow, Russia
e-mail: ivan@ispras.ru

Denis, Turdakov
ISPRAS
Moscow, Russia
e-mail: turdakov@ispras.ru

Yaroslav, Nedumov
ISPRAS
Moscow, Russia
e-mail: yaroslav.nedumov@ispras.ru

ABSTRACT

The paper describes a system for text annotation with links to ontologies extracted from linked data. We show that there are no principle differences between the well-known wikification task and the task of text annotation with linked data. In this paper, we discuss performance issues and critical parts of annotators.

Keywords

Text Processing, Semantic Similarity, Word Sense Disambiguation, Semantic Annotation

1. INTRODUCTION

Ontologies represent knowledge as a set of concepts and relationships between them. Automatic ontology extraction from user-generated content and usage of extracted information in natural language processing (NLP) applications became extremely popular in recent years.

There are two general ways for ontology extraction different in a way of deal with relationships between concepts. Complex approach tries to build ontology with defined types of relations between concepts, such as *"Part of"* or *"is a"* relation. This way requires complex analysis of the content.

Another approach represents knowledge as linked data where link only shows that concepts are related without defining type of relation. This way is much more simple but yet powerful for semantic analysis of text. In this paper we show how to apply linked data for such analysis.

Wikipedia is a first and most popular user-generated resource for building ontologies. There are number of attempts to build systems for automatic annotation of textual documents with links to Wikipedia or ontologies extracted from Wikipedia [7, 9, 12, 13, 14, 17]. Process of such annotation is called *wikification* and mentioned system are known as *wikifiers.*

Documents marked in this way could be useful for various types of applications [5] such as semantic search systems, recommender systems, classification and clustering tasks, etc. In addition, it is popular to annotate complex terms with links to Wikipedia articles with descriptions of terms meaning. This wikification helps to organize complex collections of documents.

Wikipedia has well-organized structure and this is the reason of its popularity. Each regular article describes one concept of real word. Articles are interconnected, and link between articles means that corresponding concepts are somehow related between each other. Utilization of Wikipedia as a corpus allows gathering of statistics extremely useful for many NLP tasks.

Growing number of open linked data [2] (IMDB, Gene Ontology, DBLP, etc.) allows to create new domain-specific ontologies. Therefore the task of wikification transforms to the task of annotation of natural language text with links to domain-specific ontologies [11, 15].

In the rest of the paper, we compare different approaches for building fast annotators of this kind and show that there are no principle differences between wikification and linking to domain-specific ontology.

2. TASK DEFINITION

Assume the existence of an ontology of interrelated concepts, $c \in O$. Relation between concepts is represented by a link $\lambda(c_1, c_2)$. Each concept c is associated with descriptive information such as a main textual representation (name), list of synonymous representations, description of the concept, and statistical properties.

In general, the task is defined as follows. For a given document d composed of tokens (words or punctuation mark):

- **Term Annotation:** find a set of non-overlapping non-partitioning sequences of tokens referred to as terms, $t \in d$, and establish unambiguous connection between meaning m of term and concept $c \in O$ or mark term with special symbol ζ that means that the meaning of the term is not presented in the ontology.
- **Key Terms Detection:** from set of detected terms select most representative one for a given document subset and mark each term in this subset as key term.

3. TERM ANNOTATION

In this section, we describe general architecture of systems that annotate text with links based on analysis of existing systems. Most of annotators consist of three main parts: knowledge base (KB), knowledge base management system (KBMS), and natural language processing (NLP) part. Knowledge base contains ontology and statistical features of concepts. KBMS provides the mechanism for compact storing of KB and the interfaces for effective working with it. NLP part contains algorithms for processing of textual documents.

KBMS functionality usually can be represented as two-level model. The first level deals with concepts identified by their IDs. Each concept has some attributes (title, synonyms) and is linked to other concepts. Thus, the first function of KBMS is to provide this information about the concept (concepts' attributes and results of simple depth-first graph traversal). Another function of KBMS is to compute semantic relatedness between concepts with the aid of information about link structure of ontology.

The second level of KBMS model deals with text representations (terms). For each term it allows to get corresponded concepts and statistical properties (see section 6 for details).

NLP part is intended for text processing. It detects terms, disambiguates their senses, and detects key terms for processed text. In addition, it usually contains common NLP algorithms such as sentence detector and POS tagger that help to improve precision of main function. We describe text processing in details below.

4. SEMANTIC RELATEDNESS

Semantic relatedness is a cornerstone of link annotators. All approaches for semantic relatedness computation based on linked data can be divided into two groups: "local" and "global". Relatedness computation over normalized number of common neighbours is widely used. Most known measures are cosine, Dice, and Jaccard measures. They are defined as

$$rel(a,b) = \frac{1}{\Theta}\left|N(a) \cap N(b)\right| \quad ,$$

where Θ is a normalization coefficient that varies for different measures.

Another measure appeared in publications is Google Distance:

$$rel_{GD}(a,b) = \frac{\log(\max(|N(a)|,|N(b)|)) - \log(|N(a) \cap N(b)|)}{\log(|W|) - \log(\min(|N(a)|,|N(b)|))} \quad ,$$

where W is a number of nodes in the graph.

These measures use only local information about nearest neighbours. This means that only second neighbours can have non-zero relatedness.

Global approach for computation of the semantic relatedness uses recursive definition of relatedness. Most interesting method is SimRank [6]. SimRank is based on following preposition: two objects are similar if they are referenced by similar objects. Base for recursion is that object is similar to itself with maximal weight. For the graph $G(V,E)$, where V is a set of nodes and E is a set of edges, for each node v, $I(v)$ denotes a set of nodes referenced to v. Then SimRank is defined as:

$$s(a,a) = 1,$$

$$s(a,b) = \frac{C}{|I(a)|\,|I(b)|}\sum_{i=1}^{|I(a)|}\sum_{j=1}^{|I(b)|} s(I_i(a), I_j(b)) \quad ,$$

where C is damping factor, $0 < C < 1$.

In spite of recursive methods seem very interesting they are not applicable to huge graphs in practice due to their high computation complexity. For example, complexity of direct way for SimRank computation requires $O(N^4)$ operation. In the recent publications several methods with less computation complexity were proposed, but they still can not be applied to huge graphs. We use measures based on normalized number of nearest neighbours.

Calculation of relatedness requires computation of the size of nearest neighbours intersection. Thus, the best representation of a node in the ontology's graph is a sorted array of all nearest neighbours. For such representation computation complexity of one relatedness function is $O(N(a) + N(b))$.

Calculation of relatedness is the most critical operation for text processing. Pre-computation of relatedness can significantly increase overall performance. However, due to scale-free structure of user-generated linked data, the number of non-zero similarities would be almost W^2, where W is a number of nodes in graph.

However, we found that pre-computation of relatedness for nodes that share common link can improve performance without significant decrease of precision. Results of comparing are discussed in section 8.

5. TERM DETECTION

Most common way for term detection is to find terms presented both in text for processing and in KBMS. These terms can consist of several words but they must not overlap with each other. Fastest way to detect such terms is to apply greedy algorithm. However, for some cases this algorithm can provide wrong result. For instance, there are two terms in Wikipedia's dictionary: *"French army"* and *"Army officer"*. And it depends on context what term should be detected in text *"... French army officer ... "*. Thus, another way is to detect both cases and then choose correct one at the disambiguation step. We implemented both approaches in the Texterra system and found that naive greedy approach performs comparably to second one, while being more computationally efficient.

Dictionary based approach produces good coverage for large ontologies such as Wikipedia-based one. However, for small domain-specific ontologies it can not be sufficient. Authors of [11] propose to use supervised term detection algorithm and then bind detected terms with concepts of ontology with the aid of several heuristics. However, this approach is much more slower than previous two.

6. TERM SENSE DISAMBIGUATION

Several algorithms were proposed for choosing proper meaning of term depending on the context. All of these algorithms are based on semantic relatedness between meanings of terms. Semantic relatedness shows how related are two concepts of ontology and is computed with the aid of ontology (see section 4).

Other features that are commonly used are following:

- Prior probability of a meaning for a given term $P(c|t)$.

- Probability of a synonym of the concept $P(t|c)$.
- Prior probability that the term is a key term $P(\text{t is a keyterm})$.

These features are pre-computed and are stored in the knowledge base.

Structure of links allows easily estimate these probabilities. Links of linked data usually consist of two parts: caption and concept. Caption is a term that is shown to user. Concept is article with description of concept that link points to. For example, Wikipedia's link *"[[Platform (computing) | Platform]]"* means that user sees word *"Platform"* in text and link from this term points to article *"Platform (computing)"*. Thus, required probabilities can be estimated in following way.

$P(c|t) = \frac{count(c,t)}{count(t)}$, where $count(c,t)$ - number of links with caption t and destination c. Denominator is a number of links with caption t.

$P(t|c) = \frac{count(c,t)}{count(c)}$, where $count(c)$ is a number of links pointing to c.

Probability that the term is a key term is usually estimated through probability that the term is a caption of link.

$P(\text{t is a keyterm}) = P(t \in \Lambda) = \frac{count_{doc}(t \in \Lambda)}{count_{doc}(t)}$,
that is fraction of documents where term was presented as caption of link.

Algorithm that chooses most common sense (MCS) for all terms is usually used as a baseline. This algorithm doesn't use any information about context. An opposite approach is to choose meaning of term that is most similar to context [17], where context consists of meanings of nearest unambiguous terms. We will refer to this approach as conformity disambiguation. Authors of [13] combined Lesk algorithm and naive bayes classifier trained on Wikipedia. Finally, assuming orthogonality of these methods, the authors used discrepancies in the results as a sign of a potential error and ignored such results. Milne and Witten [14] got better results by using machine learning algorithms for choosing best meaning between most common and most conformal. Part of Wikipedia articles was employed as a training corpus where meanings of terms were extracted from links. Authors compared several algorithms and showed that bagged C4.5 produces better results. Usage of sequence classifiers for this tasks was discussed in [16].

HMM classifier didn't produce any sufficient improvement in compare to baseline. Authors of the paper [16] proposed generalization of HMM to the set of independent markov chains. This algorithm produces significantly better results. However, modification of Viterbi algorithm that is used for decoding works much slower than non-sequence classifiers.

For comparing purposes we implemented algorithm that is based on maximum entropy classifier and ideas proposed in [14]. Description of evaluation methods and results are presented below.

7. KEY TERMS DETECTION

There are a lot of works about keyphrases detection. In works [8, 3, 10] keyphrases are treated as simple parts of text. Several features like length of phrase in words, position of phrase in the text, frequency of phrase in the text, and other more complex ones are combined by hand [3] or by a machine learning classifier [8, 10] in order to get the answer.

We propose another approach for key terms detection. After disambiguation each term t in the text is linked with unambiguous concept c. We propose to detect key concepts and then use their representation in text as key terms.

As we know the value of semantic relatedness between concepts, we can use more precise methods together with mentioned above. First of all we group all concepts into semantically related clusters in order to determine the main topic of the document [4]. Then for each concept from the main topic, we calculate geometrical mean of following features: concept frequency (a number of concept representations in the document), average number of words in all concept occurrences in the document and then we take maximal link probability for all concept occurrences in the document.

Then we sort concepts list and mark several concepts from the top as key concepts for the document. We do not use machine learning here because we want to be domain independent and have not large enough domain independent training set.

8. EVALUATION

In this section we discuss the evaluation of sense disambiguation and keyword extraction. As a testing corpus we use collection of documents manually annotated within WikifyMe project [1]. Users were inspired to mark terms in plain text and select appropriate Wikipedia article for description of meaning of each term. After marking of all terms users selected several concepts as key concepts. Thus testing set for evaluation of both tasks was created. It contains 132 documents with 7145 marked terms; whole size is 191 KB of plain text.

In the table 1 results of comparing of F_1 measure and processing speed are presented for different combinations of disambiguator, on-line and off-line computation of relatedness. Each cell contains F_1 measure and time of processing of whole collection in milliseconds. As we see GHMM-based disambiguator produces best results, however it is very slow. Maximum Entropy based disambiguator showed the best ratio of precision and performance. The reason is that for many cases, most common sense is a correct answer and algorithm doesn't need to compute relatedness.

	on-line relatedness computation	pre-computed relatedness
MCS	66.08 / 30280	66.08 / 26899
Conformity	67.66 / 39991	66.98 / 36270
GHMM	73.79 / 104848	72.74 / 76732
MaxEnt	73.17 / 32735	70.49 / 27985

Table 1. Comparing of disambiguators

Results for key term detection are presented in table 2. As described in section 7 we take top-n concepts and select their representations as key terms. Selection of top 5 concepts produces best results.

	Precision	Recall	F_1
Top 3	40.71	29.68	34.33
Top 5	31.29	38.03	34.34
Top 7	26.61	45.27	33.52

Table 2. Key term detection

9. FUTURE WORK

Despite of growing number of open linked data some specific domains will not be covered anyway, for instance, internal terminology of a company. Creation of ontology that describes business of particular company is expensive task.

However in the model of linked data we can propose automatic acquiring of knowledge base from collection of documents that contains description of terminology. As we showed, for creation of knowledge base we need only concepts with descriptions that are interlinked between each other. However, it is possible to automatically create linked data from textual description of concept by using described annotators and automatically extend existing knowledge base with new concepts.

REFERENCES

[1] Sergey Bartunov, Alexander Boldakov, and Denis Turdakov. Wikifyme: Creating testbed for wikifiers. In *Proceedings of the Spring Researchers Colloquium on Database and Information Systems, Moscow, Russia, 2011*, 2009.

[2] Christian Bizer, Tom Heath, and Tim Berners-Lee. Linked data - the story so far. *Int. J. Semantic Web Inf. Syst*, 5(3):1–22, 2009.

[3] Samhaa R. El-Beltagy and Ahmed Rafea. Kp-miner: Participation in semeval-2. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, SemEval '10, pages 190–193, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics.

[4] Maria Grineva, Maxim Grinev, and Dmitry Lizorkin. Extracting key terms from noisy and multitheme documents. In *Proceedings of the 18th international conference on World wide web*, WWW '09, pages 661–670, New York, NY, USA, 2009. ACM.

[5] Maria Grineva, Maxim Grinev, Dmitry Lizorkin, Alexander Boldakov, Denis Turdakov, Andrey Sysoev, and Alexander Kiyko. Blognoon: exploring a topic in the blogosphere. In *Proceedings of the 20th international conference companion on World wide web*, WWW '11, pages 213–216, New York, NY, USA, 2011. ACM.

[6] Glen Jeh and Jennifer Widom. Simrank: a measure of structural-context similarity. In *KDD*, pages 538–543. ACM, 2002.

[7] Sayali Kulkarni, Amit Singh, Ganesh Ramakrishnan, and Soumen Chakrabarti. Collective annotation of wikipedia entities in web text. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '09, pages 457–466, New York, NY, USA, 2009. ACM.

[8] Patrice Lopez and Laurent Romary. Humb: Automatic key term extraction from scientific articles in grobid. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, SemEval '10, pages 248–251, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics.

[9] O. Medelyan, I. H. Witten, and D. Milne. Topic indexing with Wikipedia. In *1st AAAI Workshop on Wikipedia and Artificial Intelligence*, 2008.

[10] Olena Medelyan, Eibe Frank, and Ian H. Witten. Human-competitive tagging using automatic keyphrase extraction. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 3 - Volume 3*, EMNLP '09, pages 1318–1327, Stroudsburg, PA, USA, 2009. Association for Computational Linguistics.

[11] Gabor Melli and Martin Ester. Supervised identification and linking of concept mentions to a domain-specific ontology. In *Proceedings of the 19th ACM international conference on Information and knowledge management*, CIKM '10, pages 1717–1720, New York, NY, USA, 2010. ACM.

[12] Rada Mihalcea. Using wikipedia for automatic word sense disambiguation. In *North American Chapter of the Association for Computational Linguistics (NAACL 2007)*, 2007.

[13] Rada Mihalcea and Andras Csomai. Wikify!: linking documents to encyclopedic knowledge. In *Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*, CIKM '07, pages 233–242, New York, NY, USA, 2007. ACM.

[14] David Milne and Ian H. Witten. Learning to link with wikipedia. In *Proceeding of the 17th ACM conference on Information and knowledge management*, CIKM '08, pages 509–518, New York, NY, USA, 2008. ACM.

[15] Delia Rusu, Blaz Fortuna, and Dunja Mladenic. Automatically annotating text with linked open data. In Christian Bizer, Tom Heath, Tim Berners-Lee, and Michael Hausenblas, editors, *4th Linked Data on the Web Workshop (LDOW 2011), 20th World Wide Web Conference (WWW 2011).*, Hyderabad, India, 2011.

[16] Denis Turdakov and Dmitry Lizorkin. Hmm expanded to multiple interleaved chains as a model for word sense disambiguation. In *Proceedings of the 23rd Pacific Asia Conference on Language, Information and Computation*, pages 549–558, Hong Kong, December 2009. City University of Hong Kong.

[17] Denis Turdakov and Pavel Velikhov. Semantic relatedness metric for wikipedia concepts based on link analysis and its application to word sense disambiguation. In *Proceedings of the SYRCODIS 2008 Colloquium on Databases and Information Systems*, 2008.