Automated Semantic Tagging of Textual Content

Jelena Jovanovic[1], Ebrahim Bagheri[2], John Cuzzola[2], Dragan Gasevic[3], Zoran Jeremic[2], Reza Bashash[4]

[1]University of Belgrade, Belgrade, Serbia
jeljov@fon.bg.ac.rs

[2]Ryerson University, Toronto, Canada
{bagheri, jcuzzola, zoran.jeremic}@ryerson.ca

[3]Athabasca University, Athabasca, Canada
dgasevic@acm.org

[4]SideBuy Technologies, Vancouver, Canada
bashash@gmail.com

Abstract: Motivated by a continually increasing demand for applications that depend on machine comprehension of text-based content, researchers, in both academia and industry, have developed innovative solutions for automated information extraction from text. In this article, we focus on a subset of such tools – i.e., semantic taggers – that not only extract and disambiguate entities mentioned in the text, but also identify topics that unambiguously describe the text's main themes. We offer insight into the process of semantic tagging, the capabilities and specificities of today's semantic taggers, and also indicate some of the criteria to be considered when choosing a tagger to use.

Keywords: I.2.7.h Text analysis < I.2.7 Natural Language Processing < I.2 Artificial Intelligence < I Computing Methodologies; I.2.5.a Expert and knowledge-intensive system tools and techniques < I.2.5 Programming Languages and Software < I.2 Artificial Intelligence < I Computing Methodologies

Introduction

In the last few years, there has been a constant increase in the number and variety of online applications that rely on machine comprehension of human language to offer advanced functionalities such as semantic search, question answering, and recommendation. These functionalities are often enabled by information extraction services that couple Text Analysis and Machine Learning methods and techniques, with large, general-purpose knowledge bases such as Wikipedia.

Information Extraction – an active area of Text Analysis research – has seen a steady growth in the recent years [1]. The latest developments in data storage and processing, enabled by cloud infrastructure, have synergized many advanced Information Extraction methods and techniques, so that text can be processed and relevant information be extracted almost instantaneously. The significant increase in efficiency of automated text analysis has been coupled with an increase in the overall quality of extracted information. This made Information Extraction services highly appealing for a wide variety of real-world application areas such as:

- Knowledge management: information extraction from unstructured text can produce structured, unambiguous (meta-)data, and thus enable more effective and efficient search over and management of the organization's content repositories, as exemplified by the KnowMed (http://knowmed.com/) solution for healthcare and medical research.

- Business analytics: tools such as RavenPack News Analytics (http://www.ravenpack.com/services/rpna_dj.htm) perform analysis of news articles in order to extract diverse kinds of entities and events that could be relevant for business decision making.
- Social media monitoring and reputation management: organizations and individuals can keep track of social media chatter and manage their reputation by leveraging Information Extraction techniques as showcased by Radian6 (http://radian6.com).
- Contextual advertising: solutions such as the one developed by ADmantX (http://www.admantx.com/) enable better positioning of advertisements on Web pages based on the semantics of the main content of the page, namely the entities mentioned in the text, the opinions and/or emotions expressed and the message the text aims to communicate.

In this article, we focus on one specific Information Extraction task, namely the extraction and disambiguation of entities and topics mentioned in or related to a given text. We refer to this task as *semantic tagging*, and the tools that perform this task as *semantic taggers*. After providing some insight into the task of semantic tagging and how it is typically performed, we offer a comparative overview of the state-of-the-art semantic taggers. However, this paper does not aim at providing an exhaustive and in-depth review of the semantic tagging techniques and algorithms. Instead, the objective is to inform IT practitioners about the new potentials offered through the employment of semantic taggers and to discuss practical issues related to the use of these tools such as the criteria to consider when choosing a tool.

What are semantic tagging tools?

Named Entity Recognition (NER) is a traditional Information Extraction task that consists of recognizing entities of a restricted set of types (e.g., Person, Organization, and Date) in a given text [2]. However, the richness of information that the state-of-the-art NER tools provide is rather limited in two significant ways. First, the set of supported entity types is restricted with the majority of tools only supporting up to a dozen types. Second, while NER tools can recognize that a piece of text represents a certain entity and its type, they cannot determine the ‘identity’ of the entity. For instance, in the sentence “Novak Djokovic is number one on the ATP list,” a NER tool would identify Novak Djokovic as an entity of type Person. However, it would not be able to relate it to the corresponding real-world entity. The term “ATP” would impose an even greater challenge: whereas a NER tool might be able to recognize “ATP” as an entity of type Organization, it would have difficulties relating it with the Association of Tennis Professionals since this abbreviation can have many different meanings. The task of relating a piece of text with the actual intended real-world entity is called *disambiguation* or *entity linking*. It consists of relating an entity recognized in the text with an entry in a knowledge base – e.g., Wikipedia – that uniquely identifies and provides further information about that entity.

Semantic tagging tools can be thought of as an advanced version of NER tools that do not suffer from the above stated deficiencies. Semantic tagging is a kind of formal (i.e., machine processable) information overlay that has explicitly defined semantics, typically expressed as a reference to an entity or resource defined in a knowledge base or ontology [3]. Semantic taggers overcome the limitations of NER tools; namely *i)* they are able to recognize all entity types that are defined in the underlying knowledge base or ontology, which are often in the order of thousands of types; and *ii*) their disambiguation processes are facilitated by the entities and resources explicitly defined in the underlying knowledge base or ontology. While the first generation of semantic tagging tools relied primarily on domain specific ontologies [3], today’s state-of-the-art taggers are based on large, general-purpose, Web-based knowledge bases, primarily Wikipedia and its more structured and semantics-rich ‘derivatives’ such as DBpedia (http://dbpedia.org), YAGO (http://www.mpi-inf.mpg.de/yago-naga/yago/), and Freebase (http://www.freebase.com/). The use of these large scale knowledge bases allows for overcoming the problem of ontology brittleness prevalent in the previous generation of semantic taggers [4] that caused a decline in their performance (especially recall) if the processed text spanned beyond the domain of the underlying lexicon.

Kinds of semantic tagging tools

Semantic tagging tools can be classified into *manual, automated and semi-automated* [3]. Manual tools require human participation throughout the tagging process. Automated taggers process the text in complete isolation and without any guidance from the user. Semi-automated taggers allow users to intervene in the tagging process by choosing the best option from the list of candidate tags, or by removing some of the proposed tags that they consider incorrect or irrelevant. The dependance on human active participation impacts the efficiency of manual and semi-automated tools and makes them less desirable for the kinds of application cases described earlier. Therefore, in this article, we focus on tools that allow for automated semantic tagging of textual content. For the sake of brevity, in the rest of the article we simply refer to this category of tools as *taggers*.

To further distinguish between different kinds of taggers, we use the types of problems and tasks they can deal with as the criteria for differentiation. Similar to [5], we identify the following types of tasks:

- Document topics – identifying topics (i.e., concepts from a knowledge base) that are relevant for the overall document.
- Document tagging – identifying entity mentions in the document and linking each mention to the appropriate concept(s) from the knowledge base.
- Suggestion of related topics – identifying topics that are not directly mentioned in the document, but are semantically related to the content of the document.
- Role assignment – identifying the role that a topic or a concept plays in the given context. For instance, in the sentence “Add the flavor of a banana to your recipe”, “banana” would be disambiguated as a kind of fruit that has the “ingredient” role in the given context. In contrast, in "John threw a banana to the monkey", banana still refers to a fruit, but its role in this context would be "projectile".

In each of these types of tasks, each assigned topic, concept or role is often associated with a score indicating the likelihood that it is correct.

Typical semantic tagging process

The state-of-the-art taggers primarily rely on a combined use of text processing, large-scale knowledge bases, semantic similarity measures and machine learning techniques [5][6]. Even though each tool has its own unique approach to performing semantic tagging, some commonalities in the underlying methods of different tools can be observed. In particular, in our analysis of the state-of-the-art taggers, we have identified three main phases in a *typical* tagging process: 1) detection of entity candidates, 2) disambiguation, and 3) result set pruning.

Detection (‘spotting’) of entity candidates. The objective of this phase is twofold: 1) to identify ‘mentions’ in the input text, which are the parts of text (single words or phrases) to be tagged; and 2) to identify a set of candidate entities from a knowledge base for each of the mentions. The detection (i.e., ‘spotting’) of mentions and candidate entities is typically done as a dictionary look-up task. Each tool functions over its own custom dictionary of terms that are matched against the input text. The dictionary is typically created through the extraction of entity labels and descriptions from a specific knowledge base. Some tools enrich each dictionary entry with statistical information such as the frequency of appearance in the knowledge base, link probability and co-occurrence rates.

Disambiguation. The purpose of this phase is to select, for each mention spotted in the text, the entity/entities that properly reflect(s) its/their semantics. The selection is made from among numerous candidate entities identified in the spotting phase. Numerous approaches have been proposed for accomplishing this task, which can be classified into the following generalized groups [7]:

- *Popularity-based (mention-entity) prior* consists of choosing the most prominent entity for a given piece of text, i.e., entity mention [8]. This approach is simple but could lead to erroneous results, which are often caused by the lack of proper attention to the entities' context and the main theme of the input text. For this reason, most taggers combine this approach with other approaches.
- *Context-based approaches* consider the context of the mention to be disambiguated, and compare it to the context of the candidate entities. The context of a mention is often modeled through bag-of-words, based on which distance measures – such as cosine similarity measure [9] or Wikipedia links-based measure [8] – can be used to determine the similarity between any two given contexts.
- *Collective disambiguation* consists of jointly disambiguating multiple mentions in the input text. This can be viewed as an extension to the context-based approach in the sense that apart from considering context similarity scores of each mention-entity pair, the *collective disambiguation* approach also considers the coherence (i.e., semantic relatedness) of the target entities.
- *Graph-based approach* is a specific version of the collective disambiguation approach originally proposed by Hoffart *et al.* [7] and applied in their AIDA tool. In this approach, mentions and candidate entities are represented as vertices in a graph, while weighted edges between mentions and entities and also weighted edges among entities denote contextual similarity and coherence, respectively. Disambiguation is performed by identifying a dense sub-graph that contains exactly one mention-entity edge for each mention, indicating the most likely meaning for the given mention.

Each of these approaches has its specific pros and cons and should be applied depending on the type of the input text. For instance, context-based approaches are suitable for sufficiently long and relatively clean input texts; however, they tend to produce weaker results for shorter texts such as tweets [5]. Hoffart *et al.* [7] suggested a procedure for selecting the best disambiguation approach depending on the characteristics of the input text.

Result set pruning. The objective of this phase is to remove tags that would be of no interest to the user, e.g., overly general tags or those that are just marginally related to the main subject of the text. Values of the filtering parameters can be either determined by the tool itself through experiments on different test datasets or set by the end users. However, not all tools perform result filtering as a separate task. For instance, AIDA and Lupedia [12] perform the final selection of the best mention-entity pairs in the disambiguation phase. In contrast, to meet the requirements of high generality and flexibility, DBpedia Spotlight postpones the final selection to the post-disambiguation phase, and allows its users to fine tune the selection through a number of configurable parameters [10].

State-of-the-art semantic tagging tools

No tagging tool is a priori better than any other, since its suitability depends on many factors such as who will be using it, what it will be used for, or which features are of the highest importance for the user [4]. Therefore, we have opted for a descriptive comparison of the current tagging tools. Our intention is to familiarize readers with inner workings of the available tools (Table 1) and to help them choose the tool(s) that fits their needs the best (Table 2). It should be noted that while some tools are open for public use, their implementation and algorithmic details are not necessarily available.

The first comparison framework (Table 1) is aimed at facilitating the comprehension of tools' similarities and differences in the main phases of the tagging process. Accordingly, it is focused on the 'white-box' features of 'open' tools, i.e., tools whose underlying approach to semantic tagging is made publicly available (e.g., in a scientific paper or a technical report). Among such tools, we selected those that can be either accessed directly as a Web service or downloaded and run locally. For each tool, we briefly describe the approach applied in each of the three main phases of the semantic tagging process, as well as the custom-made dictionary or dataset the tool uses in the entity-spotting and disambiguation phases.

Table 1. Comparison of 'white-box' features of open state-of-the-art semantic tagging tools.

	Detection (spotting) of entity candidates	Disambiguation*	Pruning the results set	Source of entities used in entity detection and disambiguation phases
TagMe [5]	Dictionary-based: uses Ancor Dictionary which is a custom-made, indexed dictionary comprising of anchor terms from Wikipedia articles, titles of Wikipedia pages and redirect pages	Combined use of popularity-based prior and collective disambiguation approaches	Automated pruning of results: the pruning is based on the average value of two features: the link probability of each mention, and the coherence between its candidate tag and the candidate tags of the other mentions in the input text (computed in the disambiguation phase)	Wikipedia-based Page Catalog which is an indexed collection of (titles of) all Wikipedia pages except for disambiguation pages, list pages, and redirect pages
DBpedia Spotlight [10]	Dictionary-based: uses a custom-made dictionary (called Lexicon) that associates DBpedia resources with appropriate labels; labels are created from titles of Wikipedia articles, redirect and disambiguation pages	Context-based approach	User-defined pruning criteria: this task relies on a number of parameters tunable by the user; e.g., the support parameter allows for specifying the minimum number of incoming links a DBpedia resource has to have in order to be used for tagging	DBpedia Lexicalization dataset: this dataset comprises the Lexicon dictionary data encoded using the Lexvo vocabulary (http://www.lexvo.org/); additionally, each association between a resource and a label is assigned a score (prior probability)
Wikipedia Miner [6]	Pure text processing: detection of all n-grams in the input text, and keeping those whose link probability exceeds a low threshold set to discard only nonsense phrases and stop words.	Combined use of popularity-based prior and collective disambiguation approaches	Automated pruning of results: done using a classifier that relies on a number of features of candidate entities such as prior probability, relatedness between the entity and its surrounding context, disambiguation confidence (computed in the disambiguation phase), ...	Label Vocabulary which is a kind of Wikipedia-based dictionary that consists of article titles, redirects and link anchors; each label is associated with usage statistics (i.e., prior link probability)
AIDA [7]	Pure text processing for entity spotting: a Named Entity Recognition tool is used to detect noun phrases as candidate entity mentions; YAGO2 is used for the selection of candidate entities for each mention	Combined use of popularity-based prior, context-based approach and graph-based approach	N/A	YAGO2, a general purpose knowledge base where each entity is associated with a set of labels gathered from Wikipedia disambiguation pages, redirects, and links

Illinois Wikifier [11]	Pure text processing for entity spotting: a Named Entity Recognition tool and a shallow parser are used to detect named entities and noun-phrases as candidate entity mentions. A Wikipedia-based anchor-title index is used for the selection of candidate entities for each mention.	Combined use of popularity-based prior, context-based approach and collective disambiguation approach	N/A	Wikipedia-based anchor-title index; the index, computed by crawling Wikipedia, maps each distinct anchor text (i.e., text with embedded hyperlink) to its target Wikipedia title(s).
LUpedia [12]	Dictionary-based: sequences of tokens from the input text are searched for in a custom-made dictionary (Alias Dictionary) that relates entities with their labels and types	N/A	Automated pruning of results: pruning is based on weights that reflect the following features of candidate entities: the level of matching between the entity's alias and the input text, the general "relevancy" of the predicate and the class of the entity (the more often they are used, the more relevant they are considered)	Alias Dictionary within which each alias (label) is related to: i) the corresponding entity, ii) the class/type of the entity, and iii) the predicate defining the alias-entity relationship; it is derived from DBpedia and Linked Movie Database (http://linkedmdb.org/)
Denote [13]	Multiple strategies based on text processing and statistics: Inverse Document Frequency for single words; Named Entity Recognition & sentence chunking for multiple words/phrases for deep (role-based) tagging. N-grams and nearest neighbor for shallow tagger	Context-based approach where the best mention is selected through the conditional likelihood of a mention given the context's subject category	Partially automated and partially user-defined pruning criteria: e.g., Bayesian probability filter is applied to remove non-relevant candidate results, and Genetic Algorithm to automate the selection of "best" threshold values	A subset of DBPedia datasets. Summary statistics such as frequency counts are produced on article categories, datatypes, properties, labels, and concept descriptions (abstracts). Disambiguation links and redirects supplement these statistics for improved tagging and disambiguation. Keywords are linked to concepts and categories.
* Types of disambiguation approaches are described in the section Typical semantic tagging process				

The second comparison framework (Table 2) compares ‘black-box’ features of both commercial and ‘open’ tools. We refer to those features as ‘black-box’ since they do not reveal any information about the inner functioning of the tools, but only provide insight into their capabilities, access modes, restrictions on (free) use and the like. The selection of features was driven by the criteria that could be relevant for selecting the right tool(s) to use in a specific application case. Besides open tools, we have also included a selection of commercial tools. Tools and services for semantic tagging of text are offered by an increasing number of companies, so any attempt at providing an exhaustive list of all the available tools is destined to fail. Therefore, we have opted for a set of tools that could be considered representative based on the number of users as reported by ProgrammableWeb (http://www.programmableweb.com/) and/or their citation within academic publications. We also opted for the tools that provide some level of free program-based access, so that interested users can test them. One should note that there is also an increasing number of enterprise metadata management solutions that offer some form of semantic tagging as one of their feature (e.g., SmartLogic.com, ContentAnalyst.com, BasisTech.com). However, as those systems do not meet one or more of our criteria, we do not review them here.

Table 2. Comparison of ‘black-box’ features of state-of-the-art semantic tagging tools

	URL	Supported tagging tasks*	Knowledge base(s) for entity disambiguation	Restrictions on free use	Forms of program-based access	Support for entity typing	The type of text the tool is suitable for**
TagMe	http://tagme.di.unipi.it/	Document tagging (with relevance score)	Wikipedia	No restrictions	RESTful API	Wikipedia categories	Short texts like tweets and search snippets
DBpedia Spotlight	http://spotlight.dbpedia.org/	Document tagging (with relevance scores)	DBpedia	No restrictions; open source, available under Apache License 2.0 (the Spotting component uses LingPipe with more restricted license)	RESTful API	DBpedai/ Wikipedia types/categories	Longer texts like Web pages or Web feeds
Wikipedia Miner	http://wikipedia-miner.cms.waikato.ac.nz/	Document tagging (with relevance score); Document topics	Wikipedia	No restrictions; open source, available under GNU GPLv2 license	RESTful API; Java toolkit	No direct support	General purpose; equally suitable for all kinds of text
AIDA	https://github.com/yago-naga/aida	Document tagging (with relevance scores)	YAGO2	No restrictions; open source, licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 3.0 Unported License	Java toolkit	YAGO types	Bioinformatics Documents and Workflows
Illinois Wikifier	http://cogcomp.cs.illinois.edu/page/software_view/Wikifier	Document tagging (with relevance scores)	Wikipedia	No restrictions; open source, available under Illinois Open Source license (http://cogcomp.cs.illinois.edu/download/software/39)	Java toolkit	Wikipedia categories	Short Newswire Articles
LUpedia	http://lupedia.ontotext.com/	Document tagging (with relevance scores)	DBpedia, Linked Movie Database	No restrictions	RESTful API	DBpedia types	Entertainment and TV related texts
Denote	http://inextweb.com/denote_demo	Document tagging (with relevance scores); Suggestion of related topics (with relevance scores); Role assignment	DBpedia	Commercial; up to 500 API calls per month are free	RESTful API	DBpedia/ Wikipedia types/categories	Long descriptive documents

Alchemy API	http://www.alchemyapi.com/	Document topics (with relevance scores); Document tagging (with relevance scores)	DBpedia, Freebase, YAGO, GeoNames, and a few other knowledge bases from Linked Open Data (LOD) Cloud (http://lod-cloud.net/)	Commercial; for academic use, up to 30,000 API calls per day are free	RESTful API; toolkits for all major programming languages including Java, C/C++, C#, Ruby, Python, PHP, ...	Hundreds of entity types from a custom made classification scheme	News articles, blog posts
Open Calais	http://www.opencalais.com/	Document topics; Document tagging (with relevance scores)	DBpedia, Freebase, Geonames, Wikipedia, Linked Movie Database	Commercial; up to 50,000 API calls per day are free	RESTful and SOAP Web service API	39 custom-defined entity types; out of that, 21 types are disambiguated to LOD knowledge bases	News articles, blog posts
Wikimeta	http://wikimeta.com/	Document tagging (with relevance scores)	DBpedia	Commercial; up to 100 API calls per day are free; for students, no restrictions on use	RESTful API	Traditional Named Entity types: Person, Organization, Product, and the like	General purpose; equally suitable for all kinds of text
Textwise	http://textwise.com/	Document topics (with relevance scores)	N/A	Commercial; unspecified number of API calls are free ("the query limit is determined by TextWise at its sole discretion")	RESTful API	No support for entity recognition or typing; a modified version of the Open Directory Project classification scheme is used for doc. topics	Equally suitable for all kinds of text (notable application content authoring, blogging)
TextRazor	http://www.textrazor.com/	Document topics (with relevance scores); Document tagging (with relevance scores)	DBpedia, Freebase	Commercial; up to 500 API calls per day are free	RESTful API, Python client	DBpedia and Freebase types	Legal documents, news articles, emails

* Types of tagging tasks are described in the section Kinds of semantic tagging tools

** Note that the information in this column has not been explicitly made by the developers of the reviewed tools; it has only been implied in research papers or websites of other community developers who have used them in some context. Therefore, this information is not definitive. For a more reliable conclusion, a systematic study needs to be performed.

How to choose a tool

When deciding upon a tagging tool to use, one should primarily consider the characteristics of the task the tool is intended for [4]. In particular, the following features of the tagging task are especially important:

i) The specificity of the topics covered by the documents that are processed; for instance, tagging of domain-specific medical documents would require the use of highly specialized medical terminology, whereas general news articles could be tagged with broadly used terminology such as the one offered by DBPedia.

ii) The characteristics of the documents/text to be tagged such as length, target domain, writing style, and registers.

iii) The response time requirements of the application domain, i.e., whether the task requires real-time tagging or offline (asynchronous) tagging could be an acceptable alternative.

iv) The customizability of a tagging tool, i.e., whether the tool provides sufficient means to adapt its tunable parameters or not. This is important as tools cannot reach their best performance if they are not appropriately configured for a specific application area.

These concerns are further discussed in Table 3.

Table 3. Things to be considered when deciding upon a semantic tagger to use

Feature	Description	State of the practice and potential issues
The specificity of the subject domain	The content can be domain specific (i.e., focused on one particular domain such as medicine or law), or more broad and general (i.e., covering a wide range of topics of general interest).	Due to their reliance on general knowledge bases (e.g., Wikipedia, DBpedia, and Freebase), a large majority of the taggers discussed in this paper are better suited for general content.
The characteristics of the text to be tagged	The length of the text, the style of writing and the use of jargon (e.g., scientific papers vs. news articles vs. tweets/status updates)	The majority of existing taggers do not specify the type of text they are intended for and claim that if appropriately configured, they are able to cover different forms of textual content. Still, some tools are specifically targeted at a certain category of text. For instance, TagMe is particularly suitable for short texts such as tweets, status updates and search result snippets, whereas OpenCalais is developed with primary orientation on news articles.
The response time requirements of the application domain (asynchronous vs. synchronous response)	This aspect of the tagging task is related to the efficiency issues that stem from the use of large-scale knowledge bases in the tagging process. By affecting the speed of the tagger, it (indirectly) affects users' satisfaction with the tool, and eventually their acceptance and adoption of the tool.	The processing speed may not be a factor if real-time results are not required; for instance, if the task is to develop a web crawler that would index and semantically tag visited Web pages. However, if a usage scenario requires real-time results, one should look for semantic taggers that are efficient even in real-time. One strategy, applied in TagMe, is to make a 'trade-off' between performance and speed, and achieve higher speed by 'sacrificing' slight performance gains that could stem from applying more sophisticated but computationally more intensive techniques.
The customizability	The default configuration of a tagger is often only suitable for some	The main challenge here lies in allowing users to tune the tagger with respect to the key issues, such as specificity and

of the tagging tool	undemanding tagging tasks. Accordingly, a tagging tool tends to be more usable if it allows its users to define a custom configuration matching the specificities of the task at hand.	comprehensiveness, without being concerned with the details of the tool's parameters and internal functioning. Many of today's tagging tools tend to mitigate this challenge by grouping related parameters and exposing only a couple of generalized and intuitively-named parameter (e.g., confidence). This way the tools make a trade-off between advantages obtainable from highly customized tagging engine and erroneous results that might follow from inappropriately set parameter values.

Concluding Remarks

Automated semantic tagging technology facilitates deeper analysis of the significant amount of text that is generated worldwide either in the form of user generated content such as tweets and weblogs, or enterprise specific content such as corporate documents and reports. This technology could allow for more accurate and semantics-aware organization, search and retrieval of textual information. However, we also need to highlight the importance of choosing the most suitable semantic tagger (one or a combination of them) for a target application domain by considering factors such as those outlined in Table 3. The deployment of suitable automated semantic tagging technology can significantly enhance textual processing by moving beyond syntactical information and into the realm of deeper textual semantics.

References

1. S. Feldman. The Answer Machine (Synthesis Lectures on Information Concepts, Retrieval, and Services). Morgan & Claypool Publishers, 2012.
2. G. S. Ingersoll, T. S. Morton, and A. L. Farris. Taming Text - How to Find, Organize, and Manipulate It. Manning Publications Co., 2013.
3. V. Uren, P. Cimiano, J. Iria, S. Handschuh, M. Vargas-Vera, E. Motta, S. Ciravegna, "Semantic annotation for knowledge management: Requirements and a survey of the state of the art," *Journal of Web Semantics*, vol. 4, no.1, 2005, pp. 14–28.
4. D. Maynard, "Benchmarking Textual Annotation Tools for the Semantic Web," Proceedings of 6th International Conference on Language Resources and Evaluation, Marrakech, Morocco, 2008. URL: http://gate.ac.uk/sale/lrec2008/benchmarking.pdf
5. M. Cornolti, P. Ferragina, M. Ciaramita, "A Framework for Benchmarking Entity-Annotation Systems," Proceedings of the 22nd International World Wide Web Conference, Rio de Janeiro, Brazil, 2013, pp. 249-260.
6. D. Milne and I. H. Witten, "An open-source toolkit for mining Wikipedia," *Journal of Artificial Intelligence*, vol.194 (January 2013), 2013, pp. 222-239.
7. 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," Proceedings of the Conference on Empirical Methods in Natural Language Processing, Edinburgh, UK, 2011, pp. 782-792.
8. D. Milne and I. H. Witten, "Learning to link with Wikipedia," Proceedings of the 17th ACM Conference on Information and knowledge management, Napa Valley, California, USA, 2008, pp. 509-518.
9. R. Feldman and J. Sanger. The Text Mining Handbook: Advanced Approaches in Analyzing Unstructured Data. Cambridge University Press, 2006.
10. P. N. Mendes, M. Jakob, A. García-Silva, C. Bizer, "DBpedia spotlight: shedding light on the web of documents," Proceedings of the 7th International Conference on Semantic Systems, New York, USA, 2011, pp. 1-8.

11. L. Ratinov, D. Roth, D. Downey, M. Anderson, “Local and global algorithms for disambiguation to Wikipedia,” Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, Portland, Oregon, USA, 2011, pp.1375-1384.
12. P. Mihaylov, D4.5 Integration of Advanced Modules in the Annotation Framework. Deliverable of the NoTube FP7 EU project (project no. 231761), 2011. Available at: http://notube3.files.wordpress.com/2012/01/notube_d4-5-integration-of-advanced-modules-in-annotation-framework-vm33.pdf
13. J. Cuzzola, Z. Jeremic, E. Bagheri, D. Gasevic, J. Jovanovic, R. Bashash, “Semantic Tagging with Linked Open Data,” Proceedings of the 4th Canadian Semantic Web Symposium, Montreal, Canada, 2013. URL: http://ceur-ws.org/Vol-1054/paper-13.pdf

Authors’ bios

Jelena Jovanovic is an Associate Professor at Univeristy of Belgrade. Her broad research interests include semantic technologies and technology enhanced learning. She can be reached at http://jelenajovanovic.net.

Ebrahim Bagheri is an Assistant Professor at Ryerson University where he leads the Laboratory for Systems, Software and Semantics (LS^3). His broad research interests are large-scale software reuse and semantic Web technologies.

John Cuzzola s a Research Associate at Ryerson University. His research interests are in the areas of artificial intelligence, natural language processing, and machine learning.

Dragan Gasevic is an Associate Professor and Canada Research Chair in Semantic and Learning Technologies at Athabasca University. His research interests are at the intersect of computational, socio-cognitive, and design aspects of information seeking, sensemaking, and learning. He can be reached at http://dgasevic.athabascau.ca.

Zoran Jeremic is a Post-Doctoral Fellow at Ryerson University. His broad research interests is in the domain of technology enhanced learning. He can be reached at http://zoranjeremic.org.

Reza Bashash is a serial entrepreneur and CTO & Founder of Denote Enterprises Inc. His area of interests are artificial intelligence and search technologies. Reza can be reached at bashash[at]gmail[dot]com.