AUTOMATIC ENRICHMENT OF INFORMAL ONTOLOGY BY ANALYZING A DOMAIN-SPECIFIC TEXT COLLECTION[1]

Astrakhantsev N. A. (astrakhantsev@ispras.ru),
Fedorenko D. G. (fedorenko@ispras.ru),
Turdakov D. Y. (turdakov@ispras.ru)

Institute for System Programming of the Russian Academy of Sciences, Moscow, Russia

The core part of an entity linking system, in particular one oriented to wikification, is ontology, which is often informal and supports semantic relatedness as the only type of relation. Most of these systems suffer from the problem of ontology incompleteness. It is especially important for specific domains, since often the only source of extractable knowledge is plain text. This paper formulates the incompleteness problem as a task of ontology enrichment from domain-specific texts and presents a novel approach that combines state-of-the-art methods for terminology enrichment, our own ML-based method for homonymy detection, and methods adopted from the related field for relations extraction. Experimental evaluation shows that the bottleneck is terminology enrichment step: its average precision is about 35%, which is inapplicable for automatic usage, especially taking into account the strict requirements for ontology correctness; however, recall is high enough to help semi-automatic terminology enrichment. We also show that the best features for terminology enrichment differ from those for classic terminology recognition task.

Key words: ontology enrichment, terminology recognition, terminology enrichment, knowledge base construction, entity linking, wikification

1. Introduction

Transition from words to their meanings is essential for many natural language processing applications [2]. An important and extensively researched example is *wikification*—"the task of identifying concepts and entities in text and disambiguating them into their corresponding Wikipedia page" [4]. Some authors call this task word sense disambiguation (WSD), others prefer *entity linking* and specify concepts as "meaningful entities that have properties, semantic types, and relationships with each other" [19]. From the last definition it is obvious that in order to perform such entity linking, one should have a set of concepts with relations between them, what constitutes ontology, or knowledge base[2]. Worth noting, entity linking is not the only

[1] The reported study was supported by RFBR, research project No. 14-07-00692

[2] Some authors consider knowledge base to be a set of concepts, while ontology is "a schema for knowledge base" [26]. However, we have found term *ontology* to be commonly used in both meanings, especially in terms 'ontology learning' or 'ontology enrichment'

application of ontologies, they are widely used in Question Answering [36], Information Retrieval [14, 16], and so on.

All ontology based systems share the problem of incompleteness: even for a narrow domain with available hand-crafted ontology, there are missing concepts due to domain evolution. Moreover, domain knowledge is usually encoded in collections of plain texts only. Entity linkers can approach this problem two ways: (a) during the document processing, detect words that currently have no concept in the ontology (e.g. [19]); (b) extract new concepts with relations from domain-specific texts as a separate activity and enrich the ontology by this data.

The presented paper follows the second way, or ontology enrichment (OE), for specific domains. We preferred it to the first one for the following reasons:

- We can extract more knowledge about concepts, since we have more data about their occurrences;
- Entity linking algorithms keep simpler and faster, thus OE can be more resource consuming.

Our task differs from other OE approaches in several aspects. First, our ontology is informal: it contains concepts, their textual representations as terms, and semantic relatedness[3] as the only relation between concepts. To the best of our knowledge, we do not aware of approaches enriching ontology without at least taxonomic relations.

Second, we do not have domain-specific ontology that should be enriched explicitly; instead, we take general ontology that already has some domain-specific concepts (but we do not know explicitly which are) and add other domain-specific concepts to the whole ontology by analyzing collection of plain texts from the domain to be enriched.

This paper is organized as follows. Section 2 surveys related work in fields of ontology construction and enrichment; section 3 describes our approach in detail; in section 4 we present experimental results; the last section discusses the future work.

2. Related work

Ontology enrichment commonly means extracting new semantic relations [6, 32] or finding the appropriate place for domain-specific concepts in the existed taxonomy of the same domain [5, 25]. In this sense, our work is more related to general ontology learning, which has been widely surveyed [2, 3, 37], including our paper devoted to informal ontologies [1].

Drumond and Girardi [9] indentify so-called Ontology learning cake, see Figure 1.

---

[3] Function taking values from 0 (concepts have nothing in common) to 1 (concepts have the same semantics). Sometimes it is called ‘semantic similarity’ [21], but strictly, semantic similarity limits by ‘is-a’ relation [33].

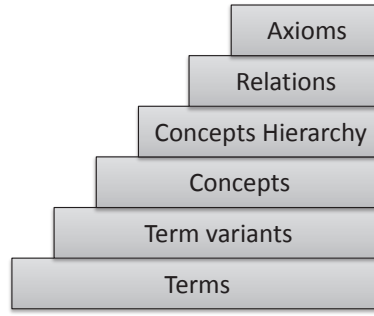

Figure 1. Ontology learning cake

The first layer, automatic terminology recognition (ATR), is the most researched one [8, 23, 27, 28, 29, 30]. In particular, we experimentally evaluated state-of-the-art approaches [11].

Term variants recognition has been studied a lot, too [7, 24, 27, 29], but most ATR works include only the simplest methods like stemming.

Regarding concepts formation, common approach is creating a concept for each term; it is based on observation that domain-specific term tends to have a single meaning [31].

Other layers correspond to formal ontologies only.

3. Ontology enrichment approach

This section describes our approach. Briefly, we take domain-specific texts and general ontology as input, then perform several sequential steps that are in keeping with Ontology learning cake, as a result we obtain domain-specific concepts (with terms and relations) that are missed in the input ontology. Each step is discussed below.

3.1. Preprocessing

At this step we apply the following methods to each input text:

1. Sentence detection
2. Tokenization
3. Part of speech tagging
4. Lemmatization
5. Word Sense Disambiguation

We used implementations from Texterra—our framework for text processing [35]. It uses, in turn, OpenNLP library[4] for the first 3 methods, heuristic algorithm based on morphologic properties of nouns for lemmatization, and Milne's algorithm [21] with another function of semantic relatedness [34] for WSD.

[4] http://opennlp.apache.org

3.2. Terminology recognition

This step takes all preprocessed texts and returns a set of domain-specific terms that are not contained in the input ontology. We adhere to the standard split of terminology extraction task [29]: collecting term candidates, computing features, and classifying term candidates into terms and not terms.

As term candidates we extract all uni-, bi-, and trigrams that occur at least 2 times and satisfy the following part of speech patterns: (N), (N_N), (Adj_N), (N_N_N), (Adj_N_N), (N_Adj_N), where N is noun and Adj is adjective.

We implemented most of state-of-the-art features, namely: CValue [13], MCValue [22], Lexical cohesion [28], Domain consensus [23], Domain relevance [27], Relevance [29], Weirdness [30], Frequency, Normalized frequency, TFIDF, Words count. As term classifier we use two approaches: Voting algorithm and supervised machine learning (ML) algorithm. The former combines features as follows:

$$V(t) = \sum_{i=1}^{n} \frac{1}{R(F_i(t))}$$

where t is a term candidate, n is a number of considered features, $R(F_i(t))$ is a rank of t among values of other term candidates considering feature F_i. Having ordered list of term candidates, one can take a top as most probable terms.

The second approach combines features in natural ML-way with Logistic regression as a particular algorithm.

Refer to our previous work [11] for details.

Since our aim is to enrich ontology, but not to construct it from scratch, we filter out terms already presented in the input ontology. For example, we enrich board game domain; term *board game* has an appropriate meaning in Wikipedia and thus should not be included into the final set of domain-specific concepts. However, one can suggest a counterexample: term *hand* in board games usually means *set of currently holding cards*, while Wikipedia has such term, but not such meaning. We describe the solution for this problem in the next subsection.

3.3. Concepts formation

There are two possible problems in transition from terms to concepts:

- synonymy—several terms have the same concept
- homonymy—several concepts have the same term

Currently we do not approach the former, partly because its effect is not so harmful, on conditions that relations for synonymic concepts are created correctly.

As for homonymy, we assume that domain-specific terminology is consistent [31] and does not contain homonyms inside the domain, i.e. we form a new concept for each newly extracted term. But we consider the case when term has a concept in general ontology and a domain-specific concept at once, see above for example with term

hand. To detect such terms, we use our approach [12]. Briefly, this method utilizes a binary logistic regression classifier based on the following features of the term:

- Relatedness to domain key concepts
- Domain relevance [27]
- Quotient of disambiguated concepts to the number of all occurrences of the term
- Quotient of disambiguated concepts to the number of possible existing concepts for the term.

3.4. Relations extraction

This section describes the way we extract relations for newly formed concepts, but firstly we discuss our ontology's organization [35]. As other systems based on Wikipedia knowledge [21], Texterra considers each article to be a concept and stores all incoming and outgoing links for an article. These links, or neighbor concepts, are used for semantic relatedness computing; in particular, Texterra uses Dice measure, that is a normalized number of common neighbors, but other measures are possible [34].

Thus, we seek at this step to extract concepts that are likely to be neighbors for each newly formed concept. If we look at distributional methods for synonyms detection [17], we can see that they are similar to the approaches for semantic relatedness computing. Indeed, the common algorithm is to collect contexts for input words, measure how similar they are, e.g. by Cosine or Dice, and extend the obtain value to the similarity between input words, on the assumption that "words that occur in the same contexts tend to have similar meanings" [15]. In this sense, neighbors represent context for concepts, therefore we take as neighbors for a concept those ones, which co-occur with the considered one not by chance. In order to find such concepts we adopt classical distributional methods for synonyms detection, particularly—measuring association with context [17].

More formally, for a newly formed concept we perform the following steps:

1. Collect neighbor candidates: for each term occurrence of each term of the input concept, find all term occurrences inside the specified window (15 occurrences to the left and to the right), and store their disambiguated concepts. As a result, we obtain a vector with concepts and their co-occurrence counts.
2. Transform each co-occurrence count into the more reliable value that shows randomness of co-occurrence: we use t-test measure with approximation of variance by sample mean [20].
3. Cut-off neighbor candidates by the predefined threshold, that is 2.0.

4. Evaluation

Approaches to evaluate ontology learning algorithm vary widely [37], because, first, ontology is not an end product; second, source corpora are usually huge and cannot be fully processed by human experts; and third, ontology construction process has

a complex structure. Indirect ontology evaluation by testing of applications that use the ontology (in-vivo testing) is not indicative in the case of bad results. It still requires direct ontology evaluation (in-vitro testing) in order to find bottleneck, if any. For these reasons we start from evaluation of each step separately.

As regards huge size of corpora, we propose the following: to markup manually only the small part of the corpus; to use the whole corpus in the prototype; to evaluate obtained results only for the marked up part. It allows having all available statistics in prototype, e.g. nonsparse counts of term occurrences, and, in the same time, obtaining reliable precision, recall, or other evaluation results.

4.1. Dataset overview

Evaluation based on a small proportion of marked up part imposes more severe requirements to the quality of the markup. At the same time, marking up each text by several people in order to average out mistakes is costly. Therefore we prepare Manifest[5]—detailed instructions for marking up text by the following annotations: terms, concepts (of the existing ontology), and domain-specificity (is the term of target domain, or another domain, or not domain-specific at all).

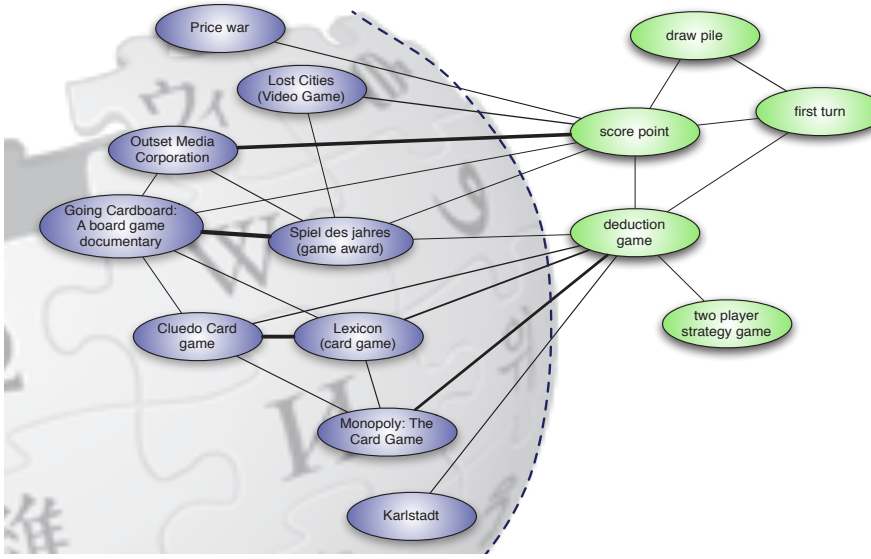

Figure 2. Examples of domain-specific concepts from marked up texts: left concepts are already presented in Wikipedia, right ones are new. Lines show semantic relatedness computed over links extracted by our tool. Thicker lines mean closer related concepts.

[5] Available on Russian at http://modis.ispras.ru/FTPContent/astrakhantsev/Manifest.pdf

We have chosen board game domain, since it is rather specific to be not fully presented in Wikipedia and is rather common to not require domain experts. We downloaded 1300 texts[6]—mostly user reviews and game specifications. 35 texts have been marked up by 9 humans (without overlapping). There have been found 1244 terms total, including 527 domain-specific ones, 246 domain-specific terms are missed in Wikipedia. Examples are shown in Figure 2.

4.2. Terminology recognition

We evaluated 2 scenarios:

1. Term recognition—compare terms without regard to their presence in Wikipedia
2. Term enrichment—compare only terms not presented in Wikipedia

We used 3 standard metrics here: average precision, precision, and recall (note that it is actually recall on candidates, i.e. fraction of term candidates that are correctly classified as terms; recall of our method for candidates collection is 71% for term recognition and 60% for term enrichment). We took top 500 of ranked candidates as terms, since it is the approximate count of domain-specific terms in marked up texts. For ML algorithm we used 2-fold cross-validation.

For each scenario we performed feature selection by exhaustive search. In case of several equally evaluated feature sets, we kept the smallest one. Results are shown in Table 1.

Table 1. Best feature subsets found by exhaustive search for Terminology recognition

		Best features subset	
		Precision, Recall	**Average Precision**
Term recognition	Voting	CValue, Relevance	MCValue, Relevance, TFIDF
	ML	Lexical cohesion, Words count, Cvalue, Relevance	Words count, CValue, Relevance
Term enrichment	Voting	CValue, Words count, TFIDF	MCValue, TFIDF
	ML	Lexical cohesion, TFIDF, Domain relevance, Relevance	Relevance, MCValue, TFIDF

Surprisingly, feature sets for Precision and Recall are the same, and this rule remains valid for all testing setups, that is why there is just one column for 2 metrics in Table 1. Another counter-intuitive observation is that ML algorithm does not select 'Words count' feature for Term enrichment.

Results for the best feature sets are presented in Table 2.

[6] From http://boardgamegeek.com

Table 2. Evaluation results for terminology recognition; the best feature subset was taken for each metric

		Precision	Recall	Average Precision
Term recognition	Voting	31.4	41.8	43.3
	ML	28.6	74.9	45.6
Term enrichment	Voting	18.2	61.9	34.1
	ML	13.8	92.0	33.0

As we can see, although implemented approach corresponds to state-of-the-art of automatic terminology recognition (and the same system shows much better results for other datasets, e.g. GENIA [11]), current results are too low to be applicable in practice. Nevertheless, based on high recall, we suppose semi-automatic methods to be promising for this task. In the simplest form, we can provide the expert with recognized terms along with contexts of occurrences and ask him/her to mark each one as domain-specific or not; tracking of user decisions could further improve productivity.

4.3. Concepts formation

To evaluate the approach for homonymy detection, we manually marked up 75 specific terms of the board games domain: 43 terms with existing concepts and 32 with new ones. We performed 4-fold cross-validation.

The results are presented in Table 3. Baseline means the method based on WSD confidence [10].

Table 3. Results for homonymy detection

	Precision	Recall	F-measure
Baseline	63%	67%	65%
Our approach	74%	83%	78%

As we can see, our approach significantly outperforms the baseline method.

4.4. Relation extraction

Unlike the other steps, extraction of semantic relatedness cannot be evaluated by direct comparison with manually prepared gold standard. However, as gold standard we can use domain-specific concepts that are already contained in Wikipedia. Normally we filter out terms of such concepts during the first step; for this evaluation scenario, we keep those automatically recognized terms, each of which has only one meaning in Wikipedia and this meaning is domain-specific.

Since we use extracted neighbors only for semantic relatedness computation, we compare this function for these two sets: neighbors that we extracted for newly

formed concepts (OE set) and neighbors that Wikipedia stores for the corresponding concepts. Moreover, we are interested in relative values of semantic relatedness; in particular, WSD uses this function to compare concepts by their relatedness to context. Therefore we evaluate the ranking of semantic relatedness function based on two neighbor sets. To be exact, we use mean average precision metric (MAP): having set of concepts, we choose one and rank others by their semantic relatedness to the chosen one; then we compute average precision for two lists obtained by both semantic relatedness; finally, we repeat the whole procedure for each concept in the input set and average these values.

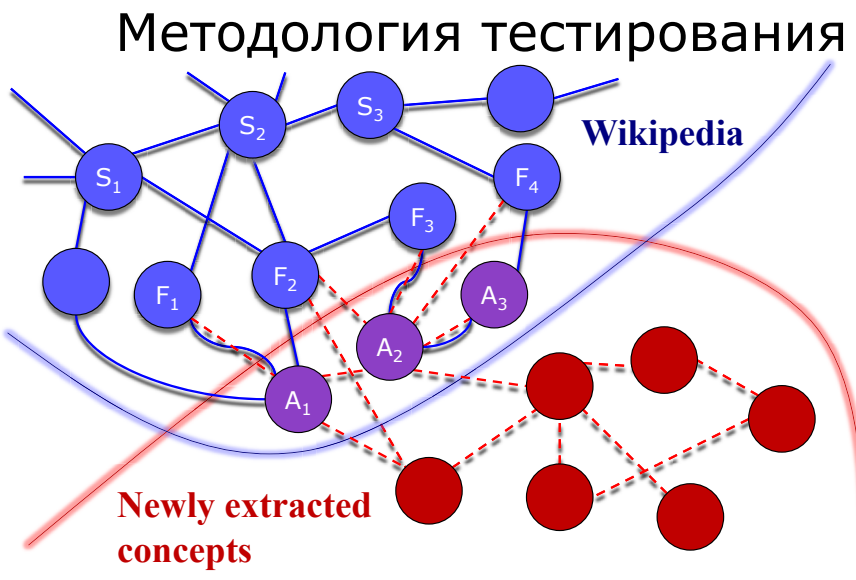

Figure 3. Sets of neighbors for evaluation of relation extraction algorithm. Blue solid lines mean links in Wikipedia; red dashed lines mean neighbors extracted by our system.

Since Dice measure is based on common neighbors, we compute MAP for 3 sets (see Figure 3):

- A—extracted concepts that are in Wikipedia
- F—first neighbors of A in OE set
- S—second neighbors of A in OE set (we use subset of 2000 concepts due to performance issue)

Results are shown in Figures 4–6.

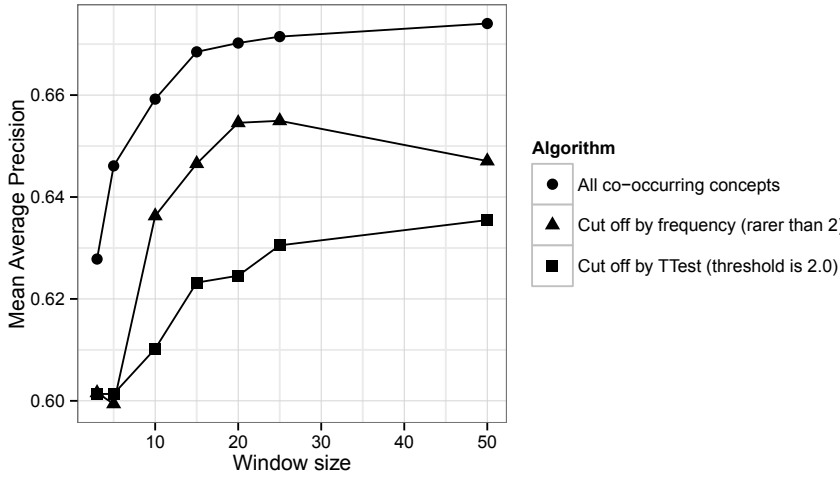

Figure 4. Mean Average Precision over A (extracted concepts that are in Wikipedia)

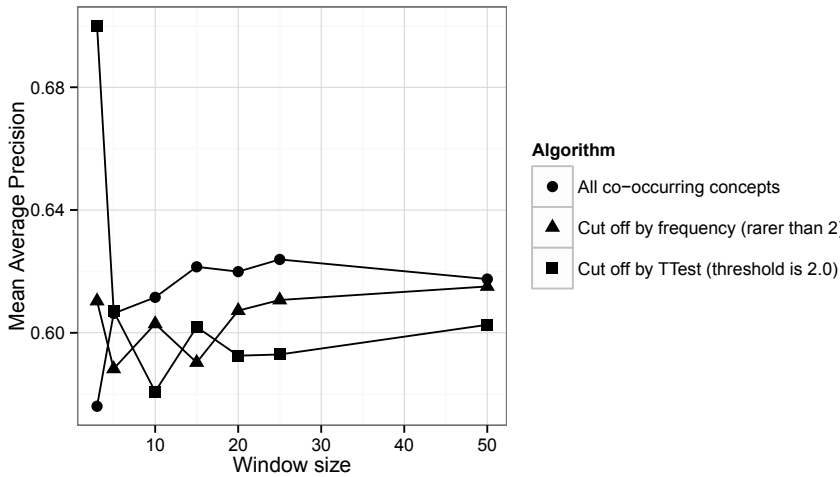

Figure 5. Mean Average Precision over F (first neighbors of A)

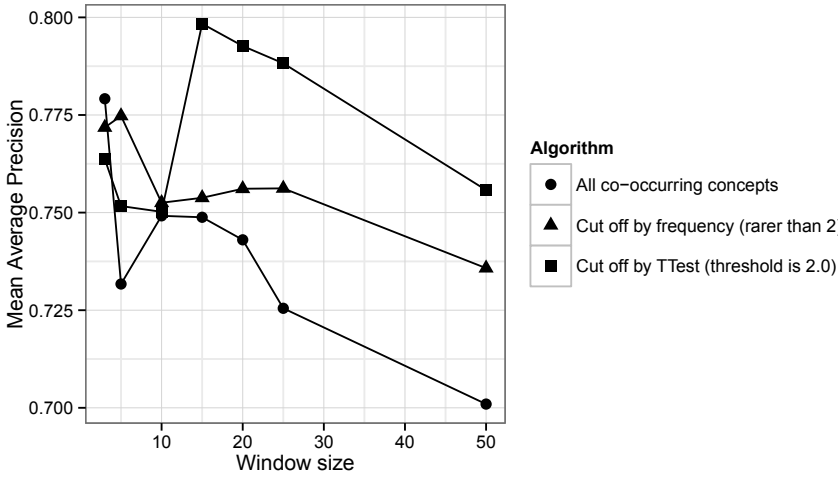

Figure 6. Mean Average Precision over S (second neighbors of A)

Note that we compute average precision between two lists of the same concepts (ordered by different algorithms), so in worst case—our semantic relatedness has nothing in common with Wikipedia's one—it would be 0.5. Our results are not so higher than 0.5, we believe it happens because there are few occurrences for most concepts, so there are few co-occurring concepts, so we cover less context than Wikipedia. It also explains why taking a huge amount of concepts as neighbors does not degrade results and why results for A improve with increasing window size: extracted neighbors serve mostly to connect to Wikipedia and to enable its set to participate in semantic relatedness computing.

5. Conclusions and Future work

This work addresses the problem of ontology incompleteness, in particular for wikification systems. We formulated the task as ontology construction from domain-specific texts and broke it into the steps according to Ontology learning cake. We used state-of-the-art approach for terminology extraction; our own method—for concepts formation; classic methods from the related field—for relations extraction. Another contribution is the methodology for dataset preparation and its usage for each step evaluation.

The main drawback of this work is low precision of terminology extraction. We plan to experiment with semi-automatic approaches, since recall is rather acceptable. In addition, if we ask the user to firstly provide domain-specific concepts that are already presented in Wikipedia, the modified task would be similar to *entity set expansion*, which has a lot of promising approaches [18][8]. Besides, currently we do not use existing ontology, although it contains necessary background knowledge; we are going to implement bootstrapping approach: to correct term recognition mistakes on the basis of extracted relations.

Also we plan to extend current dataset and to test on other domains, including cross-domain evaluation.

References

1. *Astrakhantsev, N. A., Turdakov, D. Y.* (2013). Automatic construction and enrichment of informal ontologies: A survey. Programming and Computer Software, 39(1), 34–42.
2. *Biemann, C.* (2005). Ontology Learning from Text: A Survey of Methods. In LDV forum (Vol. 20, No. 2, pp. 75–93).
3. *Buitelaar, P., Cimiano, P.* (Eds.). (2008). Ontology learning and population: bridging the gap between text and knowledge (Vol. 167). Ios Press.
4. *Cheng, X., Roth, D.* (2013). Relational Inference for Wikification. Urbana, 51, 61801.
5. *Chifu, E. T., Le Ia, I. A.* (2008). Text-based ontology enrichment using hierarchical self-organizing maps.
6. *Cimiano, P., Hotho, A., Staab, S.* (2005). Learning Concept Hierarchies from Text Corpora using Formal Concept Analysis. J. Artif. Intell. Res. (JAIR), 24, 305–339.
7. *Daille, B., Habert, B., Jacquemin, C., Royauté, J.* (1996). Empirical observation of term variations and principles for their description. Terminology, 3(2), 197–257.
8. *Dalvi, B., Callan, J., & Cohen, W.* (2011). Entity list completion using set expansion techniques. Carnegie-Mellon Univ Pittsburgh Pa Language Technologies Inst.
9. *Drumond, L., Girardi, R.* (2008). A Survey of Ontology Learning Procedures. WONTO, 427.
10. *Erk, K.* (2006). Unknown word sense detection as outlier detection. In Proceedings of the main conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics (pp. 128–135). Association for Computational Linguistics.
11. *Fedorenko, D. G., Astrakhantsev, N. A., Turdakov, D. Y.* (2013). Automatic recognition of domain-specific terms: an experimental evaluation. SYRCoDIS (pp. 15–23).
12. *Fedorenko, D. G., Astrakhantsev, N. A.* (2013). Automatic Extraction of New Concepts from Domain-Specific Terms [Izvlechenie novykh kontseptov predmetnospetsifichnykh terminov]. Proceedings of the Institute for System Programming of RAS, volume 25 (pp. 167–178).
13. *Frantzi, K. T., Ananiadou, S.* (1996). Extracting nested collocations. In Proceedings of the 16th conference on Computational linguistics-Volume 1 (pp. 41–46). Association for Computational Linguistics.
14. *Grineva, M., Grinev, M., Lizorkin, D., Boldakov, A., Turdakov, D., Sysoev, A., Kiyko, A.* (2011, March). Blognoon: exploring a topic in the blogosphere. In Proceedings of the 20th international conference companion on World wide web (pp. 213–216). ACM.
15. *Harris, Z. S.* (1954). Distributional structure. Word.
16. *Jimeno-Yepes, A., Berlanga-Llavori, R., Rebholz-Schuhmann, D.* (2010). Ontology refinement for improved information retrieval. Information Processing & Management, 46(4), pp. 426–435.
17. *Jurafsky, D., James, H.* (2000). Speech and language processing an introduction to natural language processing, computational linguistics, and speech.

18. *Li, X. L., Zhang, L., Liu, B., & Ng, S. K.* (2010, July). Distributional similarity vs. PU learning for entity set expansion. In Proceedings of the ACL 2010 Conference Short Papers (pp. 359–364). Association for Computational Linguistics.
19. *Lin, T., Etzioni, O.* (2012). No noun phrase left behind: detecting and typing unlinkable entities. In Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (pp. 893–903). Association for Computational Linguistics.
20. *Manning, C. D.* (1999). Foundations of statistical natural language processing. H. Schütze (Ed.). MIT press.
21. *Milne, D., Witten, I. H.* (2008). Learning to link with wikipedia. In Proceedings of the 17th ACM conference on Information and knowledge management (pp. 509–518). ACM
22. *Nakagawa, H., Mori, T.* (2002). A simple but powerful automatic term extraction method. In COLING-02 on COMPUTERM 2002: second international workshop on computational terminology-Volume 14 (pp. 1–7). Association for Computational Linguistics.
23. *Navigli, R., Velardi, P.* (2002). Semantic interpretation of terminological strings. In Proc. 6th Int'l Conf. Terminology and Knowledge Eng (pp. 95–100).
24. *Nenadié, G., Ananiadou, S., McNaught, J.* (2004). Enhancing automatic term recognition through recognition of variation. In Proceedings of the 20th international conference on Computational Linguistics (p. 604). Association for Computational Linguistics.
25. *Neshati, M., Alijamaat, A., Abolhassani, H., Rahimi, A., Hoseini, M.* (2007, November). Taxonomy learning using compound similarity measure. In Web Intelligence, IEEE/WIC/ACM International Conference on (pp. 487–490). IEEE.
26. *Noy, N. F., Klein, M.* (2004). Ontology evolution: Not the same as schema evolution. Knowledge and information systems, 6(4), 428–440.
27. *Park, Y., Byrd, R. J., Boguraev, B. K.* (2002). Automatic glossary extraction: beyond terminology identification. In Proceedings of the 19th international conference on Computational linguistics-Volume 1 (pp. 1–7). Association for Computational Linguistics.
28. *Patry, A., Langlais, P.* (2005). Corpus-based terminology extraction. In Proceedings of the 7th International Conference on Terminology and Knowledge Engineering (pp. 313–321).
29. *Pazienza, M. T., Pennacchiotti, M., Zanzotto, F. M.* (2005). Terminology extraction: an analysis of linguistic and statistical approaches. In Knowledge Mining (pp. 255–279). Springer Berlin Heidelberg.
30. *Peñas, A., Verdejo, F., Gonzalo, J.* (2001). Corpus-based terminology extraction applied to information access. In Proceedings of Corpus Linguistics (Vol. 2001).
31. *Slozhenikina, J. V.* The Term: Real as Life (Why Term Can and Should Have Variants), [Termin: zhivoi kak zhizn' (pochemu termin mozhet i dolzhen imet' varianty)] "Znanie. Ponimanie. Umenie", 2010, vol. 5
32. *Schutz, A., Buitelaar, P.* (2005). Relext: A tool for relation extraction from text in ontology extension. In The Semantic Web–ISWC 2005 (pp. 593–606). Springer Berlin Heidelberg.

33. *Strube, M., Ponzetto, S. P.* (2006). WikiRelate! Computing semantic relatedness using Wikipedia. In AAAI (Vol. 6, pp. 1419–1424).
34. *Turdakov, D., Velikhov, P.* (2008). Semantic relatedness metric for wikipedia concepts based on link analysis and its application to word sense disambiguation. SYRCoDIS, In CEUR Workshop Proceedings, vol. 355.
35. *Turdakov, D., Astrakhantsev, N., Nedumov, Y., Sysoev, A., Andrianov, I., Mayorov, V., Fedorenko, D., Korshunov, A., Kuznetsov, S.* (2014) Texterra: A Framework for Text Analysis [Texterra: infrastruktura dlya analiza tekstov]. Proceedings of the Institute for System Programming of RAS, volume 26, Issue 1, pp. 421–438
36. *Unger, C., Cimiano, P.* (2011). Pythia: Compositional meaning construction for ontology-based question answering on the Semantic Web. In Natural Language Processing and Information Systems (pp. 153–160). Springer Berlin Heidelberg.
37. *Wong, W., Liu, W., Bennamoun, M.* (2012). Ontology learning from text: A look back and into the future. ACM Computing Surveys (CSUR), 44(4), 20.