Designing a Model of Contexts for Word-Sense Disambiguation in a Semantic Network

Egor Dubenetskiy[0000−0002−4681−5518], Evgenij Tsopa[0000−0002−7473−3368], Sergey Klimenkov[0000−0001−5496−6765], and Alexander Pokid[0000−0003−1733−4042]

Computer Science Department, ITMO University, Saint Petersburg, Russia
https://en.itmo.ru/en/
{egor.dubenetskiy,evgenij.tsopa,serge.klimenkov,alexander.pokid}@cs.ifmo.ru

Abstract. This paper is dedicated to the problem of word-sense disambiguation in natural language processing. In this study, relations between a word's sense and its surrounding context are examined and a probabilistic model of a context is designed. An approach for testing such a model and developing a disambiguation algorithm is described as well.

Keywords: Word-sense disambiguation · Resolving homonymy · Semantic analysis · Semantic network · Natural language processing

1 Introduction

One of the crucial problems of artificial intelligence is natural language processing. To get closer to the solution of this problem, semantic networks are created that store semantic information regarding one or several domains, and algorithms over these networks are developed.[1]

One of the difficulties met in semantic analysis is word-sense disambiguation. The WSD problem is to determine what sense corresponds to an ambiguous word depending on the context. It is caused by such phenomena in languages like homonymy and polysemy.[2]. Nowadays, two types of approaches[3] are continuously improved and developed: supervised WSD[4,5], where manually sense-annotated corpora are used for models training, and knowledge-based[6,7], when developed algorithms rely on artificially constructed structures or content.

Semantic networks are a popular source of information for the knowledge-based approach. WordNet[8] was the first open ontology that helped scientists to use synset structure information in application to WSD methods. The main WordNet's disadvantage is that it was manually created and cannot cover all real-world changes. Popular successors of WordNet, such as BabelNet[9], are created semi-automatically, and contain million of sense nodes and lexemes for many languages of the world, as well as information from free online dictionaries.

This article will examine the application of the context mechanism to a general-purpose semantic network developed with an automatic resolving sense reference approach in [10] on the basis of the open thesaurus *Wiktionary* [11].

The purpose of this study is to develop a model of the context that can be further applied to the problem of lexical disambiguation, and to elaborate an approach for using that model in the existing semantic network.

2 Background

This study is based on the semantic network described earlier in the paper [10]. The following categories of objects are contained in the semantic network:

1. *Entities* — an object or category of objects that can act as the meaning of a word. There are two kinds of entities:
   (a) *Senses* – abstract concepts of a given domain. Usually a sense is denoted by a common name (e.g. "`ANIMAL`") and has a short lexical definition, or *a gloss*. A sense corresponds to a single article of a thesaurus. Examples: "`CAR` – a motor vehicle", "`ADDRESS` – a conventional codification of where something is located in space".
   (b) *Instances* – specific objects that represent distinguishable units of a sense. An instance is often (but not always) denoted by a proper name. The boundary between senses and instances is defined by the domain of a semantic network application. Examples of instances: "BMW 315" (an instance of sense `CAR`), "Elbrus 2000" (an instance of sense `MICROPROCESSOR`), "wolf" (an instance of sense `ANIMAL`).
   An instance of a sense may consist of several instances of other lower-level senses; such an instance is called *a relation* (the concept of relation is borrowed from relational algebra). For example, an instance representing a mailing address may be presented as a tuple (house, street, city, country), e.g. "49, Kronverkskiy prospekt, Saint Petersburg, Russia".
2. *Wordforms*, that are grouped into sets representing distinct linguistic paradigms, or *lexemes*. For example, Russian wordforms *автобус*, *автобусом*, *автобуса* correspond to the sense `АВТОБУС` (`BUS`).
3. *Properties*. A property is a link between two senses, declaring that one of them is an attribute of the other. E.g. `ADDRESS` is a property of `ORGANIZATION`.
4. *References*. A reference is a connection between two entities (either senses or instances) that defines a semantic link between them. The following types of semantic links are supported in the network:
   – synonymy and antonymy,
   – meronymy and holonymy,
   – hyponymy and hypernymy.
5. *Lexes*. A lex is a symbolic representation of a wordform. We do not use the term "lexeme" for lex here because it can contain wordforms for instances, that can not be always interpreted as lexemes. Lexes are stored in a string pool and are necessary for re-use in homonymic wordforms to reduce memory usage.

Due to the structure of the semantic network and the field of its application, numerous limitations and requirements arise. They will be described below.

3 Problem statement

The task is to transform a stream of tokens into a stream of entities (senses or instances) that can be later used for syntactic and semantic analysis. In this paper, a model of contexts will be elaborated to help solving this problem.

3.1 Requirements and limitations

The following points originate from the essence of the existing semantic network and will be taken into account when designing a context model:

1. The structure of a semantic network is well-defined (see its description above) and cannot be modified, but can be extended.
2. The input stream of tokens is finite and can be fully stored into memory.
3. There is a fixed number of contexts and senses against which homonymy is being resolved.
4. For any word in a given text, all its meanings can be found in the semantic network as references to senses/instances.
5. The developed mechanism of contexts should allow disambiguation in texts including several contexts that overlap and smoothly flow into each other.

4 Modeling a Context

What is proposed in this paper is to develop a mechanism of contexts for resolution of homonymy.

According to an English language dictionary [12], a *context* is the part of the text which surrounds a particular word and determines its meaning.

Therefore, in order to resolve homonymy with contexts, we need to determine in which context a given word is located. This consideration allows us to create a functional model of a context.

4.1 Functional model of a context

A context c can be considered as a function mapping each word of input $w \in W$ to its corresponding entity $\epsilon \in (S \cup I)$ (a sense or an instance):

$$c : W \to (S \cup I), \tag{1}$$

This model is based on the assumption that, given a context, same words are resolved into same entities. Hence in order for this model to function correctly, the context must be well defined. As long as within a document the same words may have different meanings, the context needs to be updated as the resolution algorithm moves along the document.

4.2 Probabilistic model of a fuzzy context

Because of the limited computational resources, a context cannot always be defined statically or determined dynamically in such a way that all homonymical word forms are resolved to corresponding entities. Moreover, the word-sense disambiguation problem is AI-complete [13]. This is why a model of a fuzzy context can be useful.

Let us imagine the context as the function that maps words to multiple entities, i.e. for each pair word-entity in a context what is defined is a probability to meet the word in the given context in the given sense.[1] This definition of a context c can be expressed formally:

$$c : W \times E \to [0, 1]; \tag{2}$$

$$\forall w : \sum_{\epsilon \in E} c(w, \epsilon) = 1, \tag{3}$$

where $E = S \cup I$ – set of entities (senses and instances).

As it is shown below, this definition leads us to an disambiguation algorithm.

5 Developing a Mechanism for Disambiguation

5.1 Extending the semantic network's structure with contexts

In order to breathe life into the designed model and provide the semantic network with the capability of homonymy resolution, the authors suggest that the following actions should be taken.

The structure of the semantic network imposes certain restrictions on how a context mechanism can be designed within it.

Three modifications are proposed to be applied to the existing semantic network in order to provide a framework for word-sense disambiguation:

1. Create a table containing information about contexts.
2. Extend the existing data structure holding information about senses with an optional attribute "topic" referring to a context in which this sense is commonly met.
3. Add an optional attribute "context" to links between wordforms and senses. If the field value is undefined, the homonymy is considered unresolved.

5.2 Elaborating a disambiguation algorithm

An algorithm for ambiguity resolution is suggested below. It is based on the developed context model and consists of the two phases:

[1] The numerical simulation of this probability is based on a (hypothetical) text corpus which contains text fragments belonging to the given context and is large enough to be an unbiased sample of language material.

1. *preparation phase*, when only monosemous words are annotated in the input as defining a context,
2. *feeding phase*, when remaining words are read by the algorithm left to right and assigned contexts to which they belong.

Both phases are described below in details.

Preparation phase. This phase will require to have some entities in the semantic network preliminarily annotated with links to contexts in which the sense or instance is most commonly used. One of the possible source of such information can be found in Wiktionary [11]. Topics can be extracted from tags in the bottom of each Wiktionary page. It is possible to define what tags correspond to what contexts statically due to the requirement on finite number of contexts.

At the preparation phase, for each word in the input text, corresponding entities are found in the semantic network. If there is exactly one relevant sense or instance, i.e. the word is unambiguous, the token t_i is marked as defining the context c_k. Such a token is called *a context anchor* and denoted with a_j. Set of anchors for context c_k is given the symbol $A[c_k]$.

After the preparation phase, what we get is the input text annotated with possible contexts. Let us call them *initial contexts*.

Feeding phase. At the feeding phase, the algorithm is given the tokens of input from left to right. For each token t_i at position i (i between 1 and N, where N is the length of the document), weights of initial context anchors a_j (where j is the position of the anchor in the input token) are computed with the formula:

$$w_{a_j}(t_i) = \begin{cases} 1/\ln(D+1) & \text{if } D \geq 0, \\ e^{-x} & \text{if } D < 0, \end{cases} \tag{4}$$

where $D = j - i$ – distance from token t_i to the context anchor a_j. The formulae are heuristic.

Then for each anchored context c_k its weight is calculated as:

$$w_{c_k}(t_i) = \sum_{a_j \in A[c_k]} w_{a_j}(t_i), \tag{5}$$

where $A[c_k]$ is the set of anchors for context c_k. After that, the context with the highest weight is selected. The token t_i is marked as an anchor for this context, and a link between this token and the corresponding contextual entity is created. The newly created anchor will be taken into account in further iterations to increase the algorithm's precision.

After consuming the whole text, what we get is the input text with links to corresponding senses or instances in the semantic network. This data can be used later for semantic analysis.

6 Methods

To test the above mechanism of contexts, the following methodology is proposed. The following tasks should be addressed regarding verification:

1. assembling a semantic concordance,
2. implementing the disambiguation algorithm in software,
3. executing the developed programs,
4. measuring and evaluating the results.

6.1 Building a Semantic Concordance for Testing

To test the developed algorithm, a semantic concordance[2] was required. Such an annotated textual corpus in which all words are guaranteed to reference the corresponding entities of the semantic network would be useful to measure the precision of the algorithm, as described below in section 6.4.

For the purposes of this study, the lexical database *WordNet 3.1* [15] was chosen as a source of data for the semantic network. A custom parser has been written to import database files of WordNet into the existing structure of semantic database management system (see section 2). Senses and semantic pointers of WordNet were mapped to senses and references of the test database.

The characteristics of the semantic network can be found in the table 1.

No. of senses and instances	40215
No. of sense-to-sense references	43684
No. of word-to-sense references	103027

Table 1. Characteristics of the imported semantic network.

What is used in the present paper as a textual corpus is the SENSEVAL-3 set of lexical samples [14]. Word entries in the corpus are linked to the exact wordings of glosses in the WordNet thesaurus [15], so an additional step of mapping glosses to appropriate senses was involved.

The characteristics of the corpus are given in the table 2.

6.2 Developing Software

A proof-of-concept implementation of the algorithm described above was created using the Ruby programming language. The program scans through the text and keeps track of the current context. After consuming each word it re-calculates the weights of contexts (as described above, in section 5.2) and annotates the last word with the sense from a context with the highest weight.

[2] As defined in [16], a *semantic concordance* is a textual corpus in which every substantive word is linked to its appropriate sense in a semantic network.

Number of documents	7860
Number of annotated homonyms	8529
Total number of words	954042

Table 2. Characteristics of the constructed textual corpus.

6.3 Conducting an Experiment

To gather results, the developed program will be passed the textual corpus as an input and will mark all the wordforms in it with references to some entities (senses or instances). Some of these references are expected to be misplaced. Therefore, in terms of binary classification, successfully resolved references are considered true positives, unresolved non-existing links are true negatives, resolved non-existing references are false positives, and unresolved existing links are false negatives. Based on the ratio of "hits" and "misses", some measures will be calculated as described below.

6.4 Measuring the Results

A *precision* measure will be introduced to evaluate results:

$$\text{precision} = \frac{\text{true positives}}{\text{true positives} + \text{false positives}}, \tag{6}$$

and a *recall* measure will be computed:

$$\text{recall} = \frac{\text{true positives}}{\text{true positives} + \text{false negatives}}. \tag{7}$$

After that, the F_1 score will be calculated as the harmonic mean of precision and recall:

$$F_1 = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}. \tag{8}$$

To review the results, a reference value of the F_1-measure will be calculated by running a dummy algorithm that randomly resolves all cases of ambiguity, and then this reference score will be compared with the real value of F_1-measure produced by the described mechanism of contexts.

7 Results

The random algorithm renders the result of $F_1 = 0.366398$. With precision $= 0.647877$ and recall $= 0.608246$ our algorithms gives the final F_1 score value of 0.627436. See table 3 for comparison.

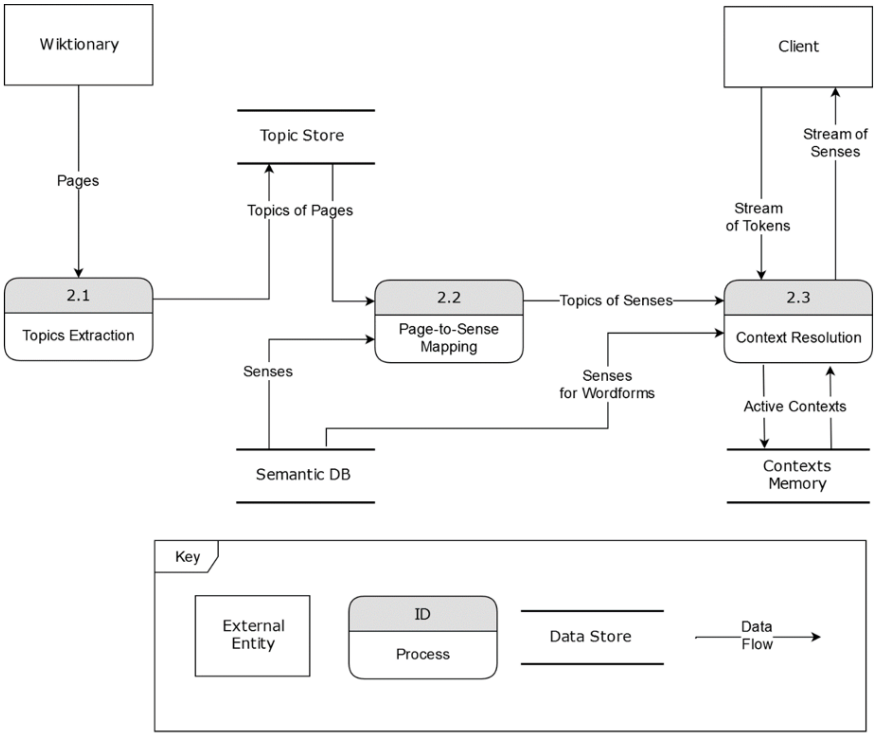

Fig. 1. Data flow diagram for the developed disambiguation mechanism

8 Conclusion

The developed model of a context requires certain improvements. For example, the heuristic formulae from section 5.2 may be adjusted to find out the values of coefficients giving the highest accuracy of the algorithm. It is a subject of further research.

References

1. Sussna, M. (1993, December). Word sense disambiguation for free-text indexing using a massive semantic network. In *Proceedings of the second international conference on Information and knowledge management (pp. 67-74)*. ACM.
2. Navigli, R. (2009). Word sense disambiguation: A survey. *ACM computing surveys (CSUR)*, 41(2), 10.
3. Raganato, A., Camacho-Collados, J., & Navigli, R. (2017, April). Word sense disambiguation: A unified evaluation framework and empirical comparison. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers* (pp. 99-110).

	Proof of concept	Random algorithm
Precision	0.647877	0.267186
Recall	0.608246	0.582800
F_1 score	0.627436	0.366398

Table 3. Results of the experiment

4. Raganato, A., Bovi, C. D., & Navigli, R. (2017, September). Neural sequence learning models for word sense disambiguation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing* (pp. 1156-1167).
5. Liu, F., Lu, H., & Neubig, G. (2018). Handling homographs in neural machine translation. In *Proceedings of NAACL, New Orleans, LA, USA.*
6. Tripodi, R., & Pelillo, M. (2017). A game-theoretic approach to word sense disambiguation. *Computational Linguistics*, 43(1), 31-70.
7. Chaplot, D. S., & Salakhutdinov, R. (2018, April). Knowledge-based word sense disambiguation using topic models. In *Thirty-Second AAAI Conference on Artificial Intelligence.*
8. Miller, G. A. (1995). WordNet: a lexical database for English. *Communications of the ACM*, 38(11), 39-41.
9. Navigli, R., & Ponzetto, S. P. (2010, July). BabelNet: Building a very large multilingual semantic network. In *Proceedings of the 48th annual meeting of the association for computational linguistics* (pp. 216-225). Association for Computational Linguistics.
10. Klimenkov, S., Tsopa, E., Pismak, A., & Yarkeev, A. (2016, November). Reconstruction of Implied Semantic Relations in Russian Wiktionary. In *KEOD* (pp. 74-80).
11. Wiktionary: Main Page. (2019, October 15). Wiktionary, The Free Dictionary. Retrieved 12:06, October 15, 2019 from https://en.wiktionary.org/.
12. context. (n.d.) American Heritage® Dictionary of the English Language, Fifth Edition. (2011). Retrieved November 2 2019 from https://www.thefreedictionary.com/context.
13. Navigli, R., & Velardi, P. (2005). Structural semantic interconnections: a knowledge-based approach to word sense disambiguation. *IEEE transactions on pattern analysis and machine intelligence*, 27(7), 1075-1086.
14. Mihalcea, R., Chklovski, T., and Kilgarriff, A. (2004, July). The Senseval-3 English lexical sample task. In *Proceedings of SENSEVAL-3, the third international workshop on the evaluation of systems for the semantic analysis of text* (pp. 25-28).
15. Princeton University "About WordNet." WordNet. Princeton University. 2010. Retrieved 14:30, December 12, 2019 from https://wordnet.princeton.edu/.
16. George A. Miller, Claudia Leacock, Randee Tengi, and Ross T. Bunker. 1993. A semantic concordance. In *Proceedings of the workshop on Human Language Technology (HLT '93)*. Association for Computational Linguistics, USA, 303–308. DOI:https://doi.org/10.3115/1075671.1075742