Method of Semantic Refinement for Enterprise Search

Alexey Pismak[a], Serge Klimenkov[b], Eugeny Tsopa[c], Alexandr Yarkeev[d], Vladimir Nikolaev[e] and Anton Gavrilov[f]
ITMO University, Kronverksky pr 49, Saint-Petersburg, Russia

Keywords: Semantic Networks, Translingual Data, Apache Lucene, Semantic Queries, Semantic Search, Pertinence of Search Results, Ontologies.

Abstract: In this paper, we propose an approach of using the semantic refinement of the input search query for the enterprise search systems. The problem of enterprise search is actual because of the amount of processed data. Even with a good organization of documents, the process of searching for specific documents or specific data in these documents is very laborious. But even more significant problem is that the required content may have the matching meaning, but expressed with different words in the different languages, which prevents it from appearing in the search result. The proposed approach uses semantic refinement of the search query. First, the concepts are extracted from the semantic network based on translingual lexemes of the user query string, allowing to perform the search based on the senses rather than word forms. In addition, several rules are applied to the query in order to include or exclude senses which can affect the relevance and the pertinence of the search result.

1 INTRODUCTION

Search systems are the mandatory component of any digital environment of a modern enterprise. Generally, the search in document databases is carried out by methods of grammatical full-text search. This variant of work with the database of documents has high relevance of the search results, but at the same time, the value of the pertinence is still quite low. In order to increase relevance, some authors propose full-stack linguistic analysis based on production rules (Ogarok, 2020). This approach shows positive results in a question-based search system, but it mainly uses the prepared subset of search queries.

This problem is important because of the amount of information required to be processed. It is especially actual in enterprise search tasks which can be characterized by the following set of features:

1. Domain homogeneity. In the most cases the individual data elements in the enterprise system data set are closely related to each other and they usually belong to a common domain.
2. Large number of documents. Typical enterprise system stores a large set (from thousands to millions) of different documents in various formats.

Even a relatively small enterprise has a set of accounts, various acts, price lists, tax documents, employee documents, and internal documentation of the company. Even with a good organization of all documents, the process of search of specific data in these documents is very resource consuming. However, the domain-specific search systems show their effectiveness, especially when the specific ontology is used (Formica et al., 2020).

The usage of semantic tags to guide the navigation during the search was proposed (Solskinnsbakk and Gulla, 2011), however, this method doesn't solve the problem of sense disambiguation. The problem is that the required content may have similar meanings, but at the same time may have different representations (expressed in other words or in another language). All

[a] https://orcid.org/0000-0001-7459-1622
[b] https://orcid.org/0000-0001-5496-6765
[c] https://orcid.org/0000-0002-7473-3368
[d] https://orcid.org/0000-0001-9682-7253
[e] https://orcid.org/0000-0003-1889-3137
[f] https://orcid.org/0000-0002-9917-6609

Pismak, A., Klimenkov, S., Tsopa, E., Yarkeev, A., Nikolaev, V. and Gavrilov, A.
Method of Semantic Refinement for Enterprise Search.
DOI: 10.5220/0010159703070312
In *Proceedings of the 12th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K 2020) - Volume 2: KEOD*, pages 307-312
ISBN: 978-989-758-474-9

questions concerning the semantic processing of texts in the natural language require the formulation of a narrow range of problems to be solved and further research on the possibility of their resolution. An example of such a range of problems is the semantic search (Rashid and Nisar, 2016).

Theoretically, the semantic approach to text processing is designed to solve the main problem of lexical search that is huge number of errors during the incorrect resolution of the polysemy of search query lexemes.

The possible way of eliminating such errors is the usage of the ontology-based semantic graph to keep the knowledge needed to improve the search quality (Modoni et al. 2014). In their article, the authors offer the general architecture that has several advantages regarding the quality of results and the usability to formulate the queries, but their main focus is on the data mining needed to collect and fill the knowledge base. Another way of resolving word-sense disambiguation is based on the usage of entity linking in queries following by choice between supervised and unsupervised alternatives (Hasibi et al, 2016).

As a part of this work, we propose a method for implementing enterprise search based on the semantic data retrieved from the ontological network. Using the semantics of the search query we can significantly increase the pertinence of the response, and therefore the proposed method is based on using the semantic relations of the ontological network, the lexical information of semantic values and the translingual data.

2 SEMANTIC NETWORKS AND LEXICAL INFORMATION

Semantic networks are graph structures with nodes that store semantic values (senses that represent concepts), and the edges between nodes indicate the relative semantic affiliation of one concept with respect to another. Examples of such relations can be synonymy, hyponymy, meronymy, and their reverse relations: antonymy, hypernymy, and holonymy (Stern D., 2015). These elementary semantic relations between senses can be used to construct more complex relations, such as cohyponyms, converses, and others.

It is important to note that the semantic network described in this paper doesn't conform to the LMF (Lexical Markup Framework) (Francopoulo G., 2013) or UBY-LMF (Eckle-Kohler J. et al, 2015) standards, because of some limitations imposed by the object-oriented model. Instead, we used the semantic representation based on a labeled oriented graph structure, where nodes correspond to senses of several types, and edges provide links between nodes (Klimenkov et al, 2020). In addition, each node corresponding to a sense is connected to all possible lexemes used to represent the sense in different languages. The ontology is formed from several semi-structural sources (Pismak et al, 2019) and the translingual lexemes are collected during the process of sense-to-sense relation reconstruction (Osika et al, 2017). Such a graph structure allows us to eliminate the needs for word-sense disambiguation due to usage of reverse sense-to-lexeme relations while providing the possibility for a quick search of sense nodes by lexemes (Pokid et al, 2017). This lexico-semantic structure contains the following types of nodes:

A semantic node is the type of node for storing data about semantic values. In its general form it is an abstract node that does not store specific information about the meaning of the sense, but only positions it with respect to other concepts in the semantic network;

A lexical node is the type of node for storing a certain lexeme. Lexical nodes are always associated with a sematic node representing a sense which can be expressed with a given lexeme. It is important that lexical nodes also contain information about the language. It is used for applying the translingual functions of the semantic search.

The types of relationships are determined by the set of permissible combinations of node types and can take the following values:

1. Sense-to-sense-synonymy;
2. Sense-to-sense-antonymy;
3. Sense-to-sense-hyponymy;
4. Sense-to-sense-hypernymy;
5. Sense-to-sense-holonymy;
6. Sense-to-sense-meronymy;
7. Sense-to-lexeme.

One of the advantages of such a lexical-semantic structure is the elimination of ambiguity resolving. While working with semantic nodes we use all word forms that express associated senses. Another benefit is that sense-to-sense relations can be taken into account, which makes it possible to refine the particular concept for a given semantic meaning. And the last but not the least advantage is the using of sense-to-lexeme relations to provide translingual search due to keeping word forms in different languages.

3 SEARCH ALGORITHM

3.1 General Description

The idea of the proposed approach is that a user formulates a search query with knowledge of required sense. Given this fact, we can force him generate the search query from semantic value identifiers instead of lexemes. Taking the semantic identifiers as the initial data, we can obtain corresponding senses from the semantic network, and then operate with lexical nodes that express required senses. The method of accepting the user feedback and providing the user the adjusted queries to choose from has been proposed by some authors (Bi et al, 2019), but in that case the search is performed in two stages, which is not always the preferred way of user interaction.

In the first stage of the algorithm, we make a selection of the necessary semantic values and associated lexical data. Then it is necessary to form a search query from existing lexical units and submit it as an input data to an existing search system, such as Apache Lucene (Apache, 2011-2020) or Sphinx (Sphinx, 2001-2020).

In the current approach, we propose to use several rules for the retrieval of semantic nodes and related lexical units. The rules are used to form the sets that encompass all user provided senses and to eliminate documents which can reduce the pertinence of the search result.

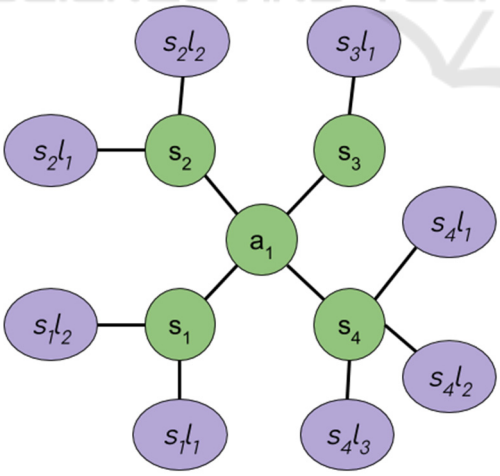

Figure 1: Fragment of the semantic network.

Let's look at the application of the rules to the fragment of the semantic network presented in Figure 1. The semantic nodes are green and the lexical nodes are purple.

Let's introduce several functions to operate on the value sets in the semantic network. To obtain a set of lexemes expressing the semantic meaning s, we introduce the function *lex(s)*. For example, according to Figure 1, the function *lex(s1)* will evaluate to the following result:

$$\text{lex}(s_1) = \{ s_1l_1, s_1l_2 \} \quad (1)$$

To obtain the set of lexemes that can express semantic values of the set S $\{s_1, s_2, \ldots s_n\}$, let's introduce the function *slex(S)*:

$$\text{slex}(S) = \text{lex}(s_1) \cup \text{lex}(s_2) \cup \ldots \cup \text{lex}(s_n) \quad (2)$$

The result of this function contains a set of lexemes that includes translingual data. It is a great advantage to use translingual data since the user can specify abstract semantic concepts in the search query, and the search process will use lexical units in all languages available in the semantic network.

Using these functions we introduce rules for the construction of a search query.

3.2 Hyponyms Rule

Sometimes the user performing the search specifies more general senses in the query assuming that all concrete senses will be included in the search query as well. For example, if the sense *car* is included in the search, the user expects that the occurrences of the specific car brands will also match the query. Traditional search queries require significant user efforts to achieve the result.

For the automatic selection and use of concrete senses in the enterprise search query let's introduce the function *hyp(s)*. This function returns the set of semantic hyponyms of argument s. Then having the subset of all concrete sense values of the argument, we can use the function slex to select all word forms of this subset:

$$Ls = \text{slex}(\text{hyp}(a)) \quad (3)$$

The result of this function (3) is the set of lexemes Ls used for the construction of the search query. Query result is passed to the search engine system. However, before we proceed to the phase of constructing such a query, we should introduce two new rules that will help us to refine the set of lexemes corresponding to the required semantic senses.

3.3 Synonyms Rule

Having the semantic node *a* as an input parameter, in the second step it is necessary to expand the set of appropriate senses by synonyms. Let semantic nodes s_x (Figure 2) be synonyms with respect to the node *a*. Then we introduce the function syn(s) that returns the synset for the semantic value *a*. For example, for a_1, this function returns the following value:

$$\text{syn}(a_1) = \{ s_1, s_2, s_3, s_4 \} \quad (4)$$

Given the synonymous senses, let's introduce the function that selects a set of lexemes for the required semantic node with all hyponyms and associated with them synonymous values. The resulted function will look like:

$$L_s = \text{slex}(\bigcup_{x \in hyp(S)} syn(x)) \quad (5)$$

As can be seen from the formulas, we select for each hyponym its synonyms and the resulting set of senses are passed as a parameter for the function slex. As a result, we get the set of lexemes that can be used to build the required search query.

3.4 Antonyms Rule

However, the search of all lexemes acquired on the previous step yields a large number of erroneous results that reduce the pertinence of the resulting set of found documents. To solve this problem authors propose to make an adjustment to the algorithm of lexemes selection. The main idea is that while selecting senses in the search query user does not expect to get as result documents that contain antonymous values to the specified argument. To implement it we propose to truncate required wordforms at the query level. The general form of the query Q can be defined as the set difference:

$$Q = L_s \setminus L_a \quad (6)$$

In this case, L_a is a set of lexemes that can be obtained with the function:

$$L_a = \text{slex}(\text{ant}(a)) \quad (7)$$

, ant(a) is a function that extracts all antonyms of the argument of sense a.

Thus, having the sets L_s (5) and L_a (7)), let's look at the principle of query construction using the example with the Apache Lucene search tool.

3.5 Building and Execution of Queries

In this paper, the authors propose the implementation of the described method for semantic search using the Apache Lucene platform.

The general architecture of the proposed implementation and the mapping of sets, extracted from the semantic network to Apache Lucene, are presented in Figure 2.

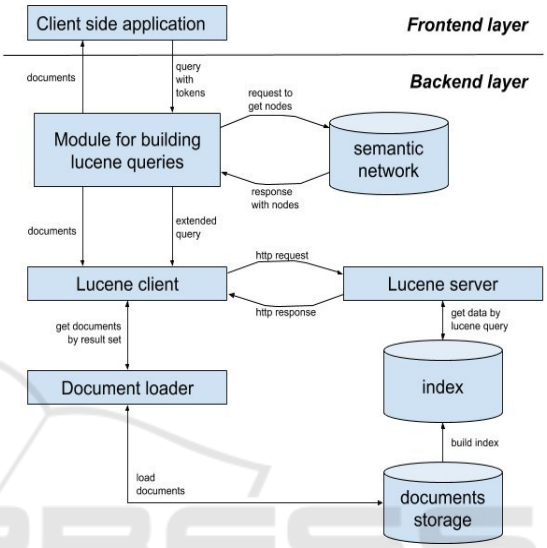

Figure 2: The general architecture for Lucene.

There are two layers of implementation:

Frontend - web interface that has the field for the input of required senses with the autocomplete function for possible senses;

Backend - applications integrated into a common infrastructure: application that implements the construction of queries in the Lucene language based on data of the semantic query; database with documents that are used for search; the semantic network in the form of a graph database.

The translation into the query for Apache Lucene is based on three simple rules:

- To intersect sets of lexemes use the AND operator
- To combine lexemes and their sets use the OR operator
- To calculate the difference of sets apply the NOT operator

4 RESULTS

Mainly in the existing search engines relevance and pertinence are used for the result evaluation (Omri,

2012). Many search systems in global networks use their own algorithms for the evaluation of results of the search, in particular the method based on user actions mentioned earlier.

The evaluation of the developed approach is based on the pertinence. This value is assumed to be within the range from zero to one. If all found documents correspond to the expectations of the user, we assume that the pertinence is equal to one. In case if all results were "useless" in terms of user expectations, we assume that the pertinence is equal to zero.

We conducted the experiment to evaluate the pertinence of the results for the following cases:

1. Grammatical (Lucene standard) search without the use of semantic network.
2. Semantic search without any rules.
3. Semantic search with synset rule applied.
4. Semantic search with synset and hyponym rules applied.
5. Semantic search with additional selection of translingual lexemes.

The experiment was done on a prepared document database including about 1000 files. The file set consisted of various documentation about the household and machine equipment, for example, price lists, user manuals, and other documents.

The value of the pertinence was calculated based on its average value for the set of queries to the system with different configurations. The configuration was used to change the algorithm in order to reveal the value of pertinence while using different features of the semantic network.

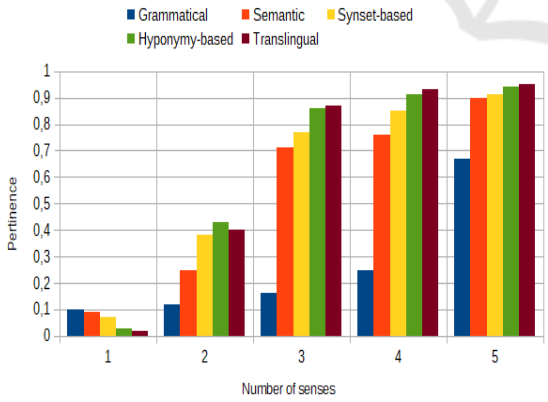

Figure 3: Experimental results.

For example, the maximum value of pertinence is reached while specifying five senses. At the same time, this category (five senses) shows maximum results while using all features of the semantic network, and without the usage of translingual information. However, this property relates to specific features of the database of documents, which contains data mainly on the same language. The results are shown in Figure 3.

5 CONCLUSIONS

The proposed approach is based on ideas that have many directions for development. In particular, one direction is to search texts for semantic constructions that could describe whole situations.

Also, each of the drawbacks of this approach specifies a vector for further development of this approach as a mechanism for natural language processing. Among the revealed disadvantages there are the following:

1. The user spends more time to prepare the query.
2. The performance is lower for semantic networks with high coherence.
3. For semantic networks with low coherence, the probability of search errors is higher.

REFERENCES

Apache Software Foundation, 2011-2020, Apache Lucene - Welcome to Apache Lucene, lucene.apache.org/

Bi, K., Ai, Q., Croft, W. B., 2019. Iterative relevance feedback for answer passage retrieval with passage-level semantic match. In ECIR'19. 558–572.

Eckle-Kohler, J., McCrae, J. P., Chiarcos, C., 2015, LemonUby–A large, interlinked, syntactically-rich lexical resource for ontologies. Semantic Web 6 (4), 371–378

Formica, A., Pourabbas, E., Taglino, F., 2020. Semantic Search Enhanced with Rating Scores. Future Internet. 12. 67.

Francopoulo, G. (ed.), 2013. LMF Lexical Markup Framework. ISTE Ltd and John Wiley & Sons, Inc., London, Hoboken.

Hasibi, F., Balog, K., Bratsberg, S., 2016. Exploiting entity linking in queries for entity retrieval. ACM SIGIR International Conference on the Theory of Information Retrieval (ICTIR '16). 209-

Klimenkov S. V., Nikolaev V. V., Kharitonova A. E., Gavrilov A. V., Pismak A. E., Pokid A. V., 2020. Using the semantic network for storing semi-structured data. Engineering Journal of Don 2(62), 27-47 (in Russian)

Modoni, G., Sacco, M., Terkaj, W., 2014. A semantic framework for graph-based enterprise search. Applied Computer Science. 10. 66–74

Ogarok, A., 2020. Method of semantic search and analysis of information. Informatization and communication 2020(1). 75-80 (in Russian).

Omri, M. N., 2012. Relevance Feedback for Goal's Extraction from Fuzzy Semantic Networks. Asian Journal of Information Technology. 3. 434-440.

Osika, V., Klimenkov, S., Tsopa, E., Pismak, A., Nikolaev, V., Yarkeev, A., 2017. Method of Reconstruction of Semantic Relations using Translingual Information. Proceedings of 9th International Conference on Knowledge Engineering and Ontology Development, 239-245.

Pismak, A. E., Klimenkov, S., Tsopa, E., Slobodkin, A. Yu., Nikolaev, V. V., 2019. Merging of semantic networks based on equivalence of topologies. Izvestiâ vysših učebnyh zavedenij. Priborostroenie. 62. 50-55. (In Russian)

Pokid A. V., Klimenkov S. V., Tsopa E. A., Zhmylev S. A., Tkeshelashvili N.M., 2017 Quick search method for nodes of a semantic network by exact word forms matching. Journal of Instrument Engineering. Vol. 60, N 10. P. 932—939 (in Russian).

Rashid, J., Nisar, M., 2016. A study on semantic searching, semantic search engines and technologies used for semantic search engines. International Journal of Information Technology and Computer Science (IJITCS). 10. 82-89

Solskinnsbakk, G., Gulla, J., 2011. Contextual search navigation using semantic tag signatures. ACM International Conference Proceeding Series. 34.

Sphinx Technologies Inc. “Open Source Search Engine.” Sphinx, 2001-2020, sphinxsearch.com/.

Stern, D., 2015. Making Search More Meaningful: Action Values, Linked Data, and Semantic Relationships. Online Searcher 39 (5), 55-58 (September/October 2015).