# Intelligent Approaches for the Automated Domain Ontology Extraction

Alexander Katyshev[1], Anton Anikin[1,3](✉), Mikhail Denisov[1], and Tatyana Petrova[2]

[1] Volgograd State Technical University, Volgograd, Russia
Anton@Anikin.name
[2] Volgograd State Socio-Pedagogical University, Volgograd, Russia
[3] Software Engineering School, Volgograd, Russia

**Abstract.** The chapter presents the review of modern approaches for the domain terminology extraction, concept discovery, concept hierarchy derivation, and learning of non-taxonomic relations steps in the ontology learning task. The chapter presents the review of not only various approaches to solving these NLP tasks but also ready-made tools that implement these approaches.

**Keywords:** Semantic link network · Ontology · Ontology learning · Natural language processing · Named entity recognition · Relations extraction · Information extraction

## 1 Introduction

Now in the world over the past 30 years, the volume of various kinds of information tends to constantly increase. To improve the efficiency of solving the problems of accumulation, processing, search, and analysis of information arrays, modern advances in the field of information technology are used. In particular, one of the components of modern promising approaches is the use of ontologies. Ontologies are developed and can be used to solve various problems, including for joint use by people or software agents, for the possibility of accumulation and reuse of knowledge in the specific domain, for the creation of models and programs that operate ontologies, rather than hard-coded data structures, for analysis of knowledge in the specific domain.

The ontology development process usually begins with what is composed of terminological concepts that are used in studies of the properties and characteristics of the terminology presented in it. Then, in a natural language, a list of exact definitions of terms presented in the glossary is created. This method can be based on several concepts (class hierarchies). Of the concepts not involved in the compilation of classification trees, class attributes and their possible meanings are distinguished. It is these concepts that establish the basic connections between classes.

The article consists of a description of ontology learning, natural language processing (specifically the description of information extraction task and relation extraction task), and also modern approaches to information and relation extraction from specific domain texts.

## 2 Ontology Learning

### 2.1 Ontology Tasks

The development of a domain ontology consists of the following steps [5]:

- domain terminology extraction;
- concept discovery and hierarchy derivation;
- relations extraction;
- ontology population;

and others.

Concepts, or classes, are considered in a broad sense as the conceptualization of all representatives of an entity or phenomenon. They may contain other concepts, individual instances, or combinations thereof. In ontologies and ontological-type resources, concepts are usually organized into a taxonomy—a hierarchical structure with generic relations. Relations are a type of interaction between concepts. The most common type of relation used in all ontologies is generic relations, also called inclusion, categorization, IS-A, class–subclass, general–private, taxonomic, and hyperhyponymic relations. Relations, like concepts, can themselves form a hierarchy [1].

### 2.2 Ontology Application

In modern search engines, texts are automatically indexed by the set of words that make up these texts.

Such a presentation of texts as a simple set of words ("bag of words")[10] has a large number of obvious drawbacks that make it difficult to find relevant texts, for example:

- redundancy: the word-by-word index uses synonyms that express the same concepts;
- text words are considered independent of each other, which does not correspond to the properties of a connected text;
- polysemy of words: since polysemous words can be considered as a disjunction of two or more concepts expressing different meanings of a polysemantic word, it is unlikely that all the elements of this disjunction are of interest to the user.

These shortcomings are deprived of the so-called conceptual indexing, that is, such indexing when the text is indexed not according to words, but according to the concepts that are discussed in this text. With this technology:

- all synonyms are reduced to the same concept;
- ambiguous words are assigned to different concepts;
- links between concepts and corresponding words are described and can be used in the analysis of the text.

In order to try to implement a scheme of automatic conceptual indexing and conceptual search, you must have a resource that describes the system of concepts of a given specific domain, that is, an ontology in a given specific domain.

## 3 Natural Language Processing

Natural language processing (NLP) is the general direction of artificial intelligence and mathematical linguistics. It studies the problems of computer analysis and synthesis of natural languages. In relation to artificial intelligence, analysis means understanding the language, and synthesis means generating literate text. Solving these problems will mean creating a more convenient form of interaction between a computer and a person. One of the main tasks of NLP is the analysis of texts in natural language. This common task includes several key sub-tasks: text categorization, information extraction, information retrieval, etc.

In this article, we analyze in detail the NLP task of extracting information from the text in a natural language and modern approaches to its solution for automating the construction of domain ontologies.

### 3.1 Information Extraction

The extraction of information is primarily associated with the search for entities and relations. This is one of the key stages of text preprocessing necessary for the implementation of more complex models and programs. Knowledge bases are used to remove homonymy, in word processing, semantic search, question–answer systems, and automatic understanding of the text without a teacher (machine reading). Entities should be categorized. A special place in the extraction of entities is occupied by the problems of identifying named entities and coreference (resolving anaphoric relations). Most of the problems that arise when automatically building a knowledge base on the web are related to the amount and heterogeneity of the data. Today you can find millions of entities, hundreds of thousands of classes, hundreds of types of relations, and hundreds of thousands of facts. The greatest difficulties arise when extracting knowledge from open areas, as well as when processing "temporary" knowledge.

### 3.2 Relation Extraction

There are two key sources for highlighting semantic relations between concepts in a language. First, you can use text boxes—this method was used, in particular, by the authors of Princeton WordNet [7]. Relations are extracted using both rules and machine learning methods on the case with pairs already marked up.

The rules have the form of regular expressions of the type "such (1) as (2), (3), (4)," where a hyperonym falls into slot 1, and hyponyms fall into slots 2–4. Similar heuristics can distinguish relations of other types [3,17].

The second principal source is machine-readable explanatory and encyclopedic dictionaries. Vocabulary definitions are a partially structured text and, as a rule, are constructed according to uniform templates, which facilitates their processing and the creation of general algorithms for identifying relations. The lexical information in the dictionary, in contrast to the plain text, is presented in a concentrated form and provided with labels that can be used as indicators of a relation between the word being defined and part of the definition [12].

## 4 Modern Approaches to Information and Relations Extraction

### 4.1 Using Context-Free Grammar

Grammar features, such as parts of speech, allow you to encode additional information about the language. One of the most effective ways to improve the quality of a model is to introduce grammars and parsers to create lightweight syntactic structures that directly affect dynamic collections of text, which can be of great importance.

To get information about the language in which the sentence is written, we need a set of grammatical rules that determine the components of correctly formed sentences in this language—this is what the grammar gives. Grammar is a set of rules that describe how syntactic units (sentences, phrases, etc.) in a language should be divided into their constituent elements. Using grammars, you can define a variety of rules for assembling phrases or fragments from parts of speech. A context-free grammar (CFG) is a set of rules for combining syntax components into meaningful strings [4].

The task of extracting named entities is currently well studied, and there are many commercial and open solutions for English such as Spacy, Stanford NER, OpenNLP, NLTK [9], MITIE, Google Natural Language API, ParallelDots [16], Aylien, Rosette, and TextRazor.

For example, Stanford NER markup named entities in raw text. The parser marks each token in the sentence, and based on the context of the sentence, the parser can define named entities among the three classes (PERSON, ORGANIZATION, LOCATION) [8].

There are good solutions for Russian too, but they are mostly closed: for example, PullEnti is an SDK for developers of information systems dealing with unstructured data—texts in natural language; functionality: selection of named entities (named entity recognition), morphology, semantics, and various processing procedures; types of entities: persons, organizations, dates, countries, and decrees. All algorithms are rule-based and languages: Russian and Ukrainian.

There are very few open-source solutions for the Russian language. For example, there is a Tomita parser from Yandex. It is designed to extract structured data from natural language text. Facts are extracted using CFG and keyword dictionaries. The parser allows to write custom grammar and add dictionaries for the desired language.

There is one more tool for the Russian language: This is Natasha. Natasha is an analog of Tomita parser for Python and a set of ready-made rules for retrieving names, addresses, dates, amounts of money, and other entities.

Table 1 provides a comparative analysis of several modern tools for extracting facts from the text.

Quality assessment was determined based on the F-measure.

$$F = 2 * \frac{P * R}{P + R}$$

where $P$ is a precision; $R$ is a recall.

**Table 1.** Analysis of modern tools for extracting facts from the text

| Tool | Speed | Settings | Flexibility | Quality |
|---|---|---|---|---|
| PullEnti | Middle | Easy.<br>Input data: raw text and work type | Low.<br>Supports a small set of types of named entities | 0.8672 |
| Tomita parser | Fast | Very hard.<br>User must create some required files including his own CFG and 2–3 setting file | High.<br>Named entities depend on user's CFG | 0.8584 |
| Natasha | Slow | Easy.<br>Python package with ready CFG.<br>Input data: raw text | Low.<br>Supports a small set of types of named entities | 0.6598 |

## 4.2 Neural Networks

Since 2015, most NLP tasks have been solved using vector representations. Each word in the raw text is converted to its corresponding vector. Vectors are built on the basis of the context of use and the mutual occurrence of words in sentences. With these vectors, you can then carry out various mathematical operations. So, for example, the cosine distance between vectors shows how two and several words are similar to each other within the context of this text.

This neural network approach is to solve various NLP tasks such as machine translation, question answering, and others.

With the advent of powerful pre-trained representations, trained using some flavor of a language modeling objective such as ELMO [13], OpenAI [14] GPT, and BERT [6], the de facto technique for NLP has become to take some sort of off-the-shelf model pre-trained on gargantuan amounts of data and fine-tune to your task with some smaller in domain corpus. Indeed, this strategy has successfully achieved tremendous SOTA results on existing NLP benchmarks.

Let us consider the BERT tool in detail. BERT is a neural network from Google, which showed by a large margin state-of-the-art results on a number of tasks [6]. Using BERT, you can create AI programs for processing a natural language: answer questions asked in any form, create chatbots, automatic translators, analyze text, and so on. To submit text to the input of a neural network, you need to present it in the form of numbers somehow. In practice, each word is assigned not one number, but several. For example, it is a vector of 32 numbers. And the distances are measured as the distances between the points that these vectors point to in the space of the corresponding dimension (for a vector 32 digits long, this is a space with 32 dimensions, or with 32 axes). This allows you to compare one word at once with several words that are close in meaning (depending on which axis to count). Moreover, arithmetic operations can be performed with vectors.

This approach is called embeddings [11]. Many packages, such as Python packages, allow the first layer of the neural network to put a special layer of embedding layer, which does this automatically. That is, at the input of the neural network we submit the usual word number in the dictionary, and embedding layer, self-learning, translates each word into a vector of the specified length, say, 32 numbers.

It is much more profitable to pre-train such a vector representation of words on some huge corpus of texts, for example, on the whole Wikipedia, and to use ready-made word vectors in specific neural networks, rather than re-train them every time [2].

In the summer of 2018, OpenAI noticed [15] that if you pre-train a neural network on the transformer [18] architecture on large volumes of text, then it unexpectedly and by a large margin shows excellent results on a variety of different natural language processing tasks. In fact, such a neural network at its output creates vector representations for words and even whole phrases. And by hanging on top of such a language model a small block of a couple of additional layers of neurons, you can train this neural network for any tasks, for example, extracting facts and relations from a text in a natural language.

In addition to BERT, there is another XLNet model [19]. XLNet is an autoregressive language modeling (AR LM). It is trying to predict the next token from the sequence of the previous ones. In classic autoregressive models, this contextual sequence is taken independently from two directions of the original string.

XLNet generalizes this method and forms a context from different places in the source sequence. How it does it? It takes all (in theory) possible permutations of the original sequence and predicts each token in the sequence from the previous ones.

If we draw analogies with the BERT, it turns out that we do not mask the tokens in advance, but rather use different sets of hidden tokens for different permutations. At the same time, the second problem of BERT disappears—the lack of hidden tokens when using the pre-trained model. In the case of XLNet, the entire sequence, without masks, is already input.

Comparison with SOTA results on the test set of RACE, a reading comprehension task shows on the Table 2. * indicates using ensembles. “Middle” and “High” in RACE are two subsets representing middle and high school difficulty levels.

**Table 2.** Comparison with SOTA results on the test set of RACE

| RACE | Accuracy | Middle | High |
|---|---|---|---|
| GPT | 59.0 | 62.9 | 57.4 |
| BERT | 72.0 | 76.6 | 70.1 |
| BERT+OCN* | 73.5 | 78.4 | 71.5 |
| BERT+DCMN* | 74.1 | 79.5 | 71.8 |
| XLNet | 81.7 | 85.4 | 80.2 |

Currently, such models are created for some separate domains. Such models cannot be extended to other domains for solving the problems of extracting concepts and domain relations to automate the construction of domain ontologies, since to solve these problems it is necessary to create and train a separate model for this domain.

## 5 Conclusion

This article has described in detail modern approaches to solving problems such as extracting facts and relationships in a natural language text. In addition, a detailed review and comparison of modern tools and libraries that implement these approaches were carried out. Each of the above tools has its own advantages and disadvantages. But a universal (as well as a multilingual solution) has not been found so far.

**Acknowledgements.** This chapter presents the results of research carried out under the RFBR grants 18-07-00032 “Intelligent support of decision making of knowledge management for learning and scientific research based on the collaborative creation and reuse of the domain information space and ontology knowledge representation model” and 20-07-00764 “Conceptual modeling of the knowledge domain on the comprehension level for intelligent decision-making systems in the learning”.

## References

1. Alatrish, E., Tošić, D., Milenkovic, N.: Building ontologies for different natural languages. Comput. Sci. Info. Syst. **11**, 623–644 (2014)
2. Anikin, A., Katyshev, A., Denisov, M., Smirnov, V., Litovkin, D.: Using online update of distributional semantics models for decision-making support for concepts extraction in the domain ontology learning task. In: IOP Conference Series: Materials Science and Engineering, vol. 483, 012073 (2019)
3. Anikin, A., Sychev, O., Gurtovoy, V.: Multi-level modeling of structural elements of natural language texts and its applications. In: Biologically Inspired Cognitive Architectures, pp. 1–8. Springer International Publishing (2018)
4. Bengfort, B., Bilbro, R., Ojeda, T.: Applied Text Analysis with Python. O'Reilly Media, Inc. (Jun 2018)
5. Cimiano, P., Völker, J., Studer, R.: Ontologies on demand? A description of the state-of-the-art, applications, challenges and trends for ontology learning from text, pp. 315–320 (2006)
6. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: Bert: Pre-training of deep bidirectional transformers for language understanding. In: NAACL-HLT (2019)
7. Fellbaum, C. (ed.): WordNet: an electronic lexical database. In: Language, Speech, and Communication, MIT Press, Cambridge, MA (1998)
8. Finkel, J.R., Grenager, T., Manning, C.D.: Incorporating non-local information into information extraction systems by Gibbs sampling. In: ACL (2005)
9. Loper, E., Bird, S.: Nltk: the natural language toolkit. CoRR cs.CL/0205028 (2002)
10. Ma, S., Sun, X., Wang, Y., Lin, J.: Bag-of-words as target for neural machine translation (2018)
11. Mikolov, T., Chen, K., Corrado, G.S., Dean, J.: Efficient estimation of word representations in vector space. CoRR abs/1301.3781 (2013)
12. Pantel, P., Pennacchiotti, M.: Automatically harvesting and ontologizing semantic relations (2008)
13. Peters, M.E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., Zettlemoyer, L.S.: Deep contextualized word representations. arXiv:1802.05365 (2018)
14. Radford, A.: Improving language understanding by generative pre-training (2018)
15. Radford, A., Narasimhan, K., Salimans, T., Sutskever, I.: Improving Language Understanding with Unsupervised Learning. Tech. rep, Technical report, OpenAI (2018)
16. Singh, P., Varadarajan, S., Singh, A., Srivastava, M.: Multidomain document layout understanding using few shot object detection (2018)
17. Snow, R., Jurafsky, D., Ng, A.: Learning syntactic patterns for automatic hypernym discovery, vol. 17 (2004)
18. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., Polosukhin, I.: Attention is all you need. In: NIPS (2017)
19. Yang, Z., Dai, Z., Yang, Y., Carbonell, J.G., Salakhutdinov, R., Le, Q.V.: Xlnet: generalized autoregressive pretraining for language understanding (2019). arXiv:1906.08237