# Ontology-Based Information Extraction: A machine learning approach

Slavko Žitnik and Marko Bajec

Faculty of Computer and Information Science, University of Ljubljana,
Tržaška cesta 25, 1000 Ljubljana, Slovenia
{slavko.zitnik,marko.bajec}@fri.uni-lj.si

**Abstract.** Information extraction (IE) task is to retrieve certain types of information from natural text. Traditional systems are using manually defined extraction rules. Later, some statistical and machine learning algorithms were adapted to solve the task. Recently, ontology-based information extraction (OBIE) emerged as a subfield of IE. Ontologies provide formal and explicit specifications of conceptualizations which plays a crucial role in the IE process. We are proposing the Intelligent OBIE system and present it by using an arbitrary ontology, taking it as possible input. We use multiple stages iteratively, combining entity and relation extraction, co-reference resolution along pre- and post-processing steps. The proposed system may improve the IE task over known systems.

**Keywords:** ontology, information extraction, text analysis

## 1 Introduction

Machine understanding of textual documents, which is needed for IE task, has been challenging for scientists since early computer-era. Russell and Norvig state that IE lies between information retrieval systems [8], which find documents related to user's requirements, and text understanding systems that attempt to analyze text and extract their semantic contents. IE methods were at first naive and manual rule based. By using machine learning techniques and induction methods they made progress over half-, automatic wrapper based and seed expansion approaches. Because the construction of wrappers is slow, consumes lots of space and learning examples, we will try new approaches and rules using machine learning with ontologies for semantic elevation, similar to newer approaches.

This paper presents the Intelligent Ontology-Based Information Extraction (IOBIE) system. It's aim is to automatically extract information from text using an ontology and state-of-the-art methods from the IE field. The system consists of pre-processing, our iterative method, post-processing and visualization. At pre-processing we identify the topic of the text to select the most appropriate ontology and parse sentences with full parser [5]. The core of IOBIE is iterative-based using our own methods for entity, relation extraction and co-reference resolution. As also Bengtson and Roth [1] have found out, the key to performance

increase in co-reference resolution is obtaining and selecting right features. We believe this stands for all IE tasks and that is why IOBIE system is focusing on remodeling of known concepts and using appropriate features. The post-processing step implements entity resolution [2] and redundancy elimination [10]. The output is then prepared for visualization as a graph using ontology. We will use simple graph as visualization technique as this area is not focus of our research.

The rest of paper is organized as follows. Section 2 presents prior work on some areas (Parsing, Coreference and Entity resolution, Entity extraction), which we will use for our research. We also explain why each area is important for our work. Section 3 describes the IOBIE system and then section 4 concludes the paper.

## 2 Related Work

### 2.1 Natural Language Parser

Parsing is the process of analyzing a text as a sequence of tokens to determine its grammatical structure with respect to a given grammar.

Shallow parsing is to highlight words (determined by verbs, nouns) without the structure of sentences. An example is the Illinois Chunker [1]. It's task is simpler than full parsing where parser is trying to label words and uncover the dependencies. For example, which groups of words go together and which words are the subject or the object of a verb. In the output we get semantically partitioned, part-of-speech (POS) tagged (i.e. nouns, verbs, adjectives, ...) words and dependencies between parts of the input sentence.

Underneath we see Stanford full parser's [5] output of the sentence "The PICACSA Conference will be held in Ljubljana." in a Penn Treebank notation.

```
(ROOT
  (S
    (NP (DT The) (NN PICACSA) (NN Conference))
    (VP (MD will)
      (VP (VB be)
        (VP (VBN held)
          (PP (IN in)
            (NP (NNP Ljubljana))))))
    (. .)))
```

The parser tagged starting *The* as determiner **DT** for the noun phrase **NP** *The PICACSA Conference*. The verb *be* has modal verb **MD** *will* and past participle **VBN** *held* next to it. Proper noun in singular **NNP** *Ljubljana* with preposition **PP** *in* is dependent to the verb.

[1] Illinois Chunker - (`http://cogcomp.cs.illinois.edu/page/software_view/13`)

### 2.2 Coreference resolution

Coreference resolution is the task of grouping all the mentions of entities into equivalent classes. It is somewhat similar to entity resolution which we present in 2.4. Mentions are all references to a selected entity. In text mentions for a person can be represented for example as names, nicknames and pronouns. Let us see an example:

> *The PICACSA Conference*$_1$ *will be held in* *Ljubljana*$_2$. *Its*$_1$ *purpose is to bring together a panel of* *postgraduate students*$_4$. *They*$_4$ *will present contributed papers*$_5$ *and have* *round table discussions*$_6$ *about* *them*$_5$. *The technical part*$_7$ *of* *the conference*$_1$ *will be accompanied by* *cultural and social events*$_8$.

In the example we found a total of 11 mentions about 8 entities. The coreference resolution method should find those 8 classes of entities.

Recently a simple but state-of-the-art pairwise model was proposed [1] and improved by hypergraph partitioning [3]. They solve problem as a graph problem. Each mention represents a node and each coreference an edge. Let $B_m$ be the set of mentions appearing before mention $m$ in document $d$. Let $a = \arg\max_{b \in B_m}(pc(b, m))$. If scoring function $pc(a, m)$ is above selected threshold, an edge $am$ is added to coreference graph. The resulting graph contains connected components, each representing one equivalence class. They learn $pc$ function using an averaged perceptron learning algorithm with selected features. As they showed, right features are extremely important at this task. They use different types of features: string relations (e.g. head, extent match, alias), semantic (e.g. gender, number match, synonyms, hypernyms), relative location (e.g. apposition, distance), learned (e.g. anaphoricity), memorization (e.g. last words) and predicted types (e.g. entity type).

### 2.3 Structure and entity extraction

This is the main topic we will try to improve. Like we said before, early research was about uncovering the structure of documents. Rules were manually created for specific extractions. The technique of set expansion [12] was popular if at least half-structured documents were to be processed. This was also a way of enriching their gazeteer lists. These lists often contain all or majority of the instances of some classes of the ontology. Initial lists must be errorless and easily obtained (e.g. Slovene post-offices). Later, wrapper induction approaches appeared and more also recently machine learning techniques. Enrycher [11] is one of the prototypes of extracting entities and their relations. More general approach is used by OntoSyphon [6]. It accepts ontology as an input and learns itself to extract entities. It was evaluated against food, animal and actors domain.

Different classification techniques have been used in IE, especially sequence tagging techniques as Hidden Markov Models and Conditional Random Fields. These models can be in general classified into five groups:

**1)** Lexicons- We check gazeteer lists for each of the entity type.
**2)** Pre-segmented classification: At preprocessing, we tokenize text and then classify each token separately.
**3)** Sliding window: We chose a fixed size $l$ of the window. Then we "slide" through the text and try to classify $l$ tokens at once.
**4)** Boundary models: We need to predict start, end position and length and then try to classify an instance.
**5)** Finite State Machines: At first we need to choose state values (e.g. words within sentence or letters within word). Then we try to determine most likely state sequence.

We will use Hidden Markov Models (HMM). They have been widely used for POS tagging, dialog act modelling, speech and text recognition. Rabiner has written one of the first tutorial [7] how to use them. In the Figure 1 we see an example of HMM for IE of research paper headers. A Markov chain (named

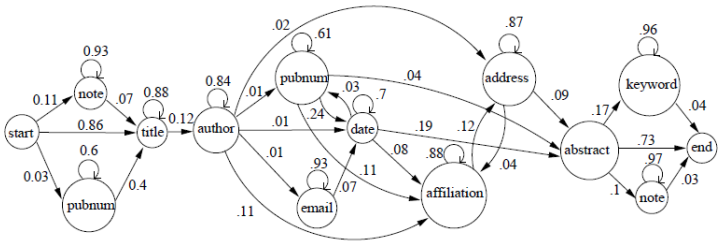


**Fig. 1.** Learned HMM example for research paper headers [9]

by Andrey Markov) is a mathematical system that models transitions from one state to another, where next state depends only on the current and not past. Model is called hidden because the final transitions between states are learned and hidden to the user. At the classification we start at a state, determined by the initial state distribution $\pi$ and then emit an observable symbol at each transition. The input which is hard to determine is the number of states $N$ which must be known before HMM construction. For $M$ observable symbols, the HMM is represented as a tuple $\lambda = (A, B, \pi)$, where $A$ is $N \times N$ transition probability matrix between states and $B$ $M \times N$ matrix of observation symbol probability distribution for each state.

### 2.4 Entity resolution

The aim of entity resolution is matching references to the same instance. This is important for us, while references to the same entities can be mentioned in different parts of a document or among different documents. One of more successful algorithms for entity resolution is collective entity resolution by Lise Getoor et. al. [2].

They first sort all the references into smaller buckets according to simple rule or metric to reduce the number of pairwise checks. After these checks they get possible cluster pairs containing one or more references that may be matched. These pairs are filled into a priority queue and iteratively checked for matching. At this part relational metrics are also used with linear combination with attribute ones. Iterating stops when no more cluster pairs similarities are above threshold and the priority queue is empty. The result of the algorithm are clusters containing references to the same entity.

## 3 IOBIE - Intelligent Ontology-Based Information Extraction

Figure 2 shows the overall design of our system. Our main goal will be to identify entities, relations and co-references. Our methods are marked as the inner rectangle within IOBIE system. Here we give the description:

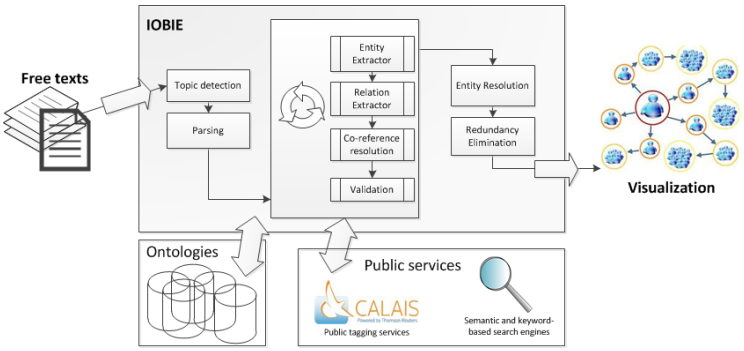


**Fig. 2.** IOBIE architecture

IOBIE system will contain a number of general and domain ontologies. As an input it will receive one or more texts where the detected entities and relations between them will be from. Annotated text output will be returned and could be visualized as a graph. Each vertex in a graph will represent an entity or attribute and each edge a certain relations.

At preprocessing we plan to detect topics of document and parse text using full parser. Next step will be an iterative method of extracting entities, relations, coreference resolution and validation. Iterations will repeat until the result of an iteration is unchanged or the maximum number of iterations is reached. Once all the entities and relations will be identified, we will start with postprocessing. At postprocessing we will do entity resolution and then redundancy elimination for cleaning. At the time of writing this paper we do not know whether it is more wisely to combine the entity resolution with our iterative method or not.

We will test both approaches and use more appropriate. The systems's result is annotated text that can be visualized as a graph and easily browsed by a man.

We will evaluate our methods on Wikipedia and check results with DBPedia, which is duplicated as an ontology with instances. We will also test our methods against others on specific datasets. Evaluating IOBIE on Wikipedia will show what are the optimal paramaters for the methods which might be different than using them standalone.

**Preprocessing.** For each text, topic will be discovered by the use of clustering. This technique will use bag-of-words and most common words. Based on the outcome of this step, an appropriate ontology will be selected. Next step is text parsing. We will use the full Stanford parser. Results of parsing will serve as additional attributes to our entity and relation extraciton method, to determine which ontology to use during the processing and to fine-tune methods' parameters.

Ontologies will help us with the formal model of the domain. At first we will not deal with a population of them, integration of multiple ontologies or building our own ontology. As a general ontology we are going to use FOAF [2] to model people and their relations. Because our system will be designed to accept or choose an ontology as one of the inputs, we are going to use ontologies such as GENIA or MeSH [3], made specifically for certain evaluation datasets. Software integration and ontology handling will be done by Jena Toolkit [4].

**Entity Extraction** using Hidden Markov Models (HMM). Methods require considerable training data, so we will provide training sets for each ontology to specifically learn models. HMMs will be used because they successfully tag sequences. For our problem, sequences are words within sentenced and tags are entity types from selected ontology. Because we want our system to improve over time, it will consist of multiple models for each ontology as an ensemble. When ensemble will contain sufficient number of models, we will remove the one with the worst scores while creating a new one. New data will be received by users of the system. They will be able to repair results or just mark errors. Features that will be used are parser's results, keyword-based and semantic search engine results and possible other annotation services. Additional attributes will be type of words left and right of the current word, coding style of current word (capital letters, special characters), endings and well known extraction rules. We will also assist ourselves with gazeteer lists that will be kept separately for each type of entity and relation in the ontology.

We will use ergodic (i.e. fully-connected) type of HMM. At first number of states will be defined by the number of possible entity types according to an ontology. In the further test we will try to build also models with more states.

[2] Friend of a Friend - `http://xmlns.com/foaf/spec/`
[3] Medical Subject Headings - `http://www.nlm.nih.gov/mesh/`
[4] Jena Toolkit - `http://jena.sourceforge.net/`

We will choose numbers, words and characters delimited by special characters (i.e. .,/,-,?,#,$,...) as observable symbols and each checked sequence will be one sentence.

**Relation Extraction.** The extraction of relations is a similar problem to extraction of entities. In each sentence we will examine the dependencies between the different parts. Relations are largely hidden in the verbs, which will also be stored and updated in ontologies gazeteer lists for each relation. We will use the same methods as with the entity extractor by mostly using dependency parse trees as features as presented in [4].

**Co-reference resolution.** We will present co-reference resolution as a graph problem, where nodes represent entities. First we will solve the problem within a sentence, then we will look at the entire text of a window of three sentences. Algorithm will work on the whole graph, similar to Bengtson and Roth [1]. We will also use additional relational features like the contradiction of two-entity neighbours and relations between neighbouring entities according to the ontology (e.g. matching two references when one is friend and another enemy of third person is not likely to happen). Other attributes are indicators - pronoun, name, occurrence in a subsidiary part, appearance of a pronoun left or right, the number of other entities between. To help solve this problem we will start with the naive Bayes classifier for creating edges in the result graph.

**Validation.** At the validation we will review found solution and, if necessary, repeat the iteration. Iteration will be repeated if the current solution was changed in this iteration and the iteration counter will be less than the maximum set value which will be determined empirically during learning. In the further research we will also support active learning for IOBIE in this phase.

**Postprocessing.** In the last phase we will clean the data and refactor it to a format suitable for visualization. The visualization methods will not be part of our research, that is why we will use visualization libraries.

In the Entity resolution module we have already implemented a proposal of collective ER [2]. We will improve their algorithm by adding additional attribute of ontology class membership to support our ontologies and using relational metrics on the mapped ontology. Entity resolution is similar to coreference resolution and is also solved as a graph problem. The main difference is that entity resolution is more powerful by matching references to same entities using attribute and relational metrics than coreference resolution where is the only task to group the same mentions together.

The Redundancy elimination module will do the final data purification. Up to this step, each entity could be presented as a set of references and at this step they will be combined into only one reference. Representative reference will be selected by a pairwise function and will in the first system naively return entity which had highest score at Entity extraction and is not pronoun.

## 4 Conclusion

In this paper we presented a new approach in building OBIE systems. As far as we know, there does not exist an IE system that would combine so many intelligent methods in a way as IOBIE, that is why we believe IOBIE will perform well. Expected IOBIE's results can be widely used in information retrieval, bioinformatics, semantic technologies, text databases analysis within organizations and more.

Currently we have implemented post-processing methods and got appropriate pre-processing methods. We have also overviewed tools and algorithms for information extraction tasks. Further we are going to develop our iterative method as presented in section 3. As we have already stated, we currently do not know exactly whether it is more wisely to combine the entity resolution with our iterative method or not. We will be clear about it when we build IOBIE.

## References

1. E. Bengtson and D. Roth. Understanding the value of features for coreference resolution. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, page 294–303, 2008.
2. I. Bhattacharya and L. Getoor. Collective entity resolution in relational data. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 1(1):5–es, 2007.
3. J. Cai and M. Strube. End-to-end coreference resolution via hypergraph partitioning. In *Proceedings of the 23rd International Conference on Computational Linguistics*, page 143–151, 2010.
4. K. Fundel, R. Küffner, and R. Zimmer. RelExRelation extraction using dependency parse trees. *Bioinformatics*, 23(3):365, 2007.
5. D. Klein and C.D. Manning. Accurate unlexicalized parsing. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics-Volume 1*, pages 423–430. Association for Computational Linguistics, 2003.
6. L. McDowell and M. Cafarella. Ontology-driven information extraction with ontosyphon. *The Semantic Web-ISWC 2006*, page 428–444, 2006.
7. L. R Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.
8. S.J. Russell, P. Norvig, J.F. Candy, J.M. Malik, and D.D. Edwards. *Artificial intelligence: a modern approach*. Prentice hall, 2010.
9. K. Seymore, A. McCallum, and R. Rosenfeld. Learning hidden Markov model structure for information extraction. In *AAAI-99 Workshop on Machine Learning for Information Extraction*, pages 37–42, 1999.
10. L. ubelj, D. Jelenc, E. Zupani, D. Lavbi, D. Trek, M. Krisper, and M. Bajec. Merging data sources based on semantics, contexts and trust. *IPSI Trans. Inter. Res.*, 7(1), 2011.
11. T. Štajner, D. Rusu, L. Dali, B. Fortuna, D. Mladenić, and M. Grobelnik. Enrycher: service oriented text enrichment. *Proceedings of SiKDD*, 2009.
12. R. C Wang and W. W Cohen. Language-independent set expansion of named entities using the web. In *icdm*, page 342–350, 2007.