# Information Extraction Based on Extraction Ontologies: Design, Deployment and Evaluation

Martin Labský, Vojtěch Svátek, and Marek Nekvasil

University of Economics, Prague, Dept. Information and Knowledge Engineering,
Winston Churchill Sq. 4, 130 67 Praha 3, Prague, Czech Republic
labsky@vse.cz, svatek@vse.cz, nekvasim@vse.cz

**Abstract.** Most IE methods do not provide easy means for integrating complex prior knowledge that can be provided by human experts. Such knowledge is especially valuable when there are no or little training data. In the paper we elaborate on the extraction ontology paradigm; the distinctive features of our system called *Ex* are 1) probabilistic reasoning over extractable attribute and instance candidates and 2) combination of the extraction ontology approach with the inductive and (to some degree) wrapper approach. We also discuss the issues related to the deployment and evaluation of applications based on extraction ontologies.

## 1 Introduction

In the last decade, *web information extraction* (WIE) was dominated by two styles. One—*wrapper*-based—is quite reliable but strictly depends on the formatting regularity of pages. The other—*inductive*—paradigm assumes the presence of annotated training data, which is rarely fulfilled in real-world settings, and manual labelling of training data is often unfeasible. In addition, both approaches usually deliver extracted information as weakly semantically structured; if WIE is to be used to fuel *semantic web* repositories, secondary mapping to *ontologies* is typically needed, which makes the process complicated and may introduce additional errors [4].

There were recently proposals for pushing ontologies towards the actual extraction process as immediate prior knowledge. *Extraction ontologies* [3] define the concepts the instances of which are to be extracted in the sense of various attributes, their allowed values as well as higher-level Extraction ontologies are assumed to be hand-crafted based on observation of a sample of resources; they allow for rapid start of the actual extraction process, as even a very simple extraction ontology (designed by a competent person) is likely to cover a sensible part of target data and generate meaningful feedback for its own redesign. The clean and rich conceptual structure (allowing partial intra-domain reuse and providing immediate semantics to extracted data) makes extraction ontologies superior to ad-hoc hand-crafted patterns used in early times of WIE. However, many aspects of their usage still need to be explored.

Section 2 of the paper briefly reviews the features of our WIE tool named *Ex* (see [5] for a more thorough description). Section 3 drafts a larger context of the ontology-based extraction task, namely, the design of extraction ontologies (incl. their relationship to usual domain ontologies), practical aspects of their usage, and their evaluation. Finally, Section 4 summarises the contributions of the paper.

## 2 Brief Overview of the Ex system

### 2.1 Main Characteristics of *Ex*

Our approach to WIE was originally inspired by that developed by Embley and colleagues at BYU [3]. The main distinctive features of *Ex* are:

1. The possibility to provide extraction evidence with *probability estimates* plus other quantitative info such as value distributions, allowing to calculate the likelihood for every attribute and instance candidate using pseudo-probabilistic inference.
2. The effort to combine hand-crafted extraction ontologies with other sources of information: HTML formatting and/or training data. *HTML formatting* is exploited via formatting pattern induction, cf. Section 2.4. *Training data* can be exploited via incorporating external *inductive learning tools*, currently those from Weka.[1]

*Ex* currently has two major real-world applications (details on both are in [5]):

- In the EU (DG SANCO) MedIEQ project[2] *Ex* acts as one of the major IE engines assisting medical website labelling authorities in assigning quality labels to websites based on several dozens of *medical website quality criteria*. Several criteria have been operationalised wrt. automatically-extractable information; most extensive experiments so far concerned the presence and richness of *contact information*, so far in three languages (English, Spanish and Czech).
- In cooperation with a large Czech *web portal* we extract information about *products* sold or described online, such as TV sets, computer monitors and bicycles.

In addition, for experimental purposes, we also systematically address other domains such as weather forecasts [8] or seminar announcements (see subsection 2.3).

### 2.2 Structure of *Ex*(traction) Ontologies

Extraction ontologies in *Ex* are designed so as to extract occurrences of *attributes* (such as 'speaker' or 'location'), i.e. standalone named entities or values, and occurrences of whole *instances* of *classes* (such as 'seminar'), as groups of attributes that 'belong together', from HTML pages or texts in a domain of interest.

*Attributes* are identified by their name, equipped with a data type (string, long text, integer or float) and accompanied by various forms of *extraction evidence* relating to the attribute value or to the context it appears in. Attribute *value* evidence includes (1) textual value patterns; (2) for integer and float types: min/max values, a numeric value distribution and possibly units of measure; (3) value length in tokens: min/max length constraints or a length distribution; (4) axioms expressing more complex constraints on the value and (5) coreference resolution rules. Attribute *context* evidence includes (1) textual context patterns and (2) formatting constraints.

*Patterns* in *Ex* (for both the value and the context of an attribute or class) are nested regular patterns defined at the level of tokens (words), characters, formatting tags

[1] `http://www.cs.waikato.ac.nz/ml/weka`
[2] `http://www.medieq.org`

(HTML) and labels provided by external tools. Patterns may be inlined in the extraction ontology or sourced from (possibly large) external files, and may include e.g. fixed lexical tokens, token wildcards, character-level regexps, formatting tags, labels representing the output of external NLP tools or references to other patterns or attribute candidates. For *numeric* types, default value patterns for integer/float numbers are provided.

For both attribute and class definitions, *axioms* can be specified that impose constraints on attribute value(s). For a single attribute, the axiom checks the to-be-extracted value and is either satisfied or not (which may boost or suppress the attribute candidate's score). For a class, each axiom may refer to all attribute values present in the partially or fully parsed instance. For example, a start time of a seminar must be before the end time. Arbitrarily complex axioms can be authored using JavaScript. Further attribute-level evidence includes *formatting constraints* (such as not allowing the attribute value to cross an HTML element) and *coreference resolution scripts*.

Each *class definition* enumerates the attributes which may belong to it, and for each attribute it defines a *cardinality* range. Extraction knowledge may address both the content and the context of a class. *Class content patterns* are analogous to the attribute value patterns, however, they may match *parts* of an instance and must contain at least one *reference* to a member attribute. Class content patterns may be used e.g. to describe common wordings used between attributes or just to specify attribute ordering. For each attribute, the *engagedness* parameter may be specified to estimate the apriori probability of the attribute joining a class instance (as opposed to standalone occurrence). Regarding class context, analogous *class context patterns* and similar *formatting constraints* as for attributes are in effect.

In addition, constraints can be specified that hold over the whole sequence of extracted objects. Currently supported are minimal and maximal instance counts to be extracted from a document for each class.

All types of extraction knowledge mentioned above are pieces of evidence indicating the presence (or absence) of a certain attribute or class instance. Every piece of evidence may be equipped with two probability estimates: precision and recall. The *precision* of evidence states how probable it is for the predicted attribute or class instance to occur given the evidence holds, disregarding the truth values of other evidence. The *recall* of evidence states how abundant the evidence is among the predicted objects, disregarding whether other evidence holds.

### 2.3 Example

In order to illustrate most of the above features, we present and explain an example from the *seminar announcement extraction* task[3], in which the speaker name, location and start and end times (*stime*, *etime*) are to be extracted. Fig. 1 shows the structure of an extraction ontology for this task. Fig. 2 displays a part of the corresponding code in the XML-based ontology definition language, dealing with the name of the speaker and start time. Note that the extraction ontology defines some extra attributes like *date*, *host* and *submitter*; these are 'helper' attributes extracted in order for the system not to confuse them with the remaining attributes.

[3] Compiled by A. McCallum, `http://www.cs.umass.edu/~mccallum/code-data.html`.

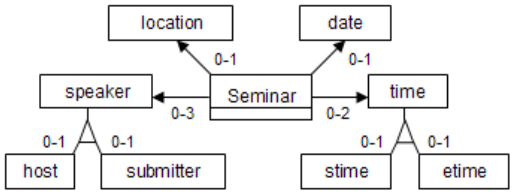


**Fig. 1.** General scheme of seminar extraction ontology

```
<class id="Seminar" counts="1">
<axiom cover="0.8" cond="all">
 diff = timeDiff($etime, $stime); diff > 29 && diff < 4*60;
</axiom>
<attribute id="speaker" card="0-1" eng="0.8">
 <pattern id="init"> <tok>^[A-Z]$</tok> .? </pattern>
 <pattern id="preamble"> speaker | presenter | talk by | who </pattern>
 <value>
  <pattern cover="0.5" p="0.8" case="CA|UC" ignore="lemma">
   <pattern src="first.txt" />
   <pattern ref="init" />?
   (<pattern src="last.txt" /> | <tok type="alpha"/>)
  </pattern>
  <length> <distribution min="1" max="6" /> </length>
  <refers> nameRefersTo($, $other) </refers>
 </value>
 <context>
  <pattern cover="0.1" p="0.6"> <pattern ref="preamble"/> :? $ </pattern>
 </context>
</attribute>
<attribute id="time" card="0-2" eng="0.8">
 <pattern id="hr"> <tok>^(0?[1-9]|1[0-2])$</tok> </pattern>
 <pattern id="mi"> <tok>^[0-5][05]$</tok> </pattern>
 <value>
  <pattern cover="0.3" p="0.8"> <pattern ref="hr"/> (:|.)? <pattern ref="mi"/> </pattern>
  <axiom> checkTime($) </axiom>
  <refers> sameTime($, $other) </refers>
 </value>
</atribute>
<attribute id="stime" card="0-1" extends="time" eng="1">
 <context>
  <pattern cover="0.02" p="0.4" ignore="case,lemma">
   (start|begin) at? :? | start? time: | from :?) $
  </pattern>
 </context>
</attribute>
```

**Fig. 2.** Fragment of code of seminar extraction ontology

In the global scope of the model, extraction knowledge affecting more than one attribute is defined: an axiom states that in 80% of cases, the duration of a seminar is between 30 minutes and 4 hours. The axiom is conditioned so that it only applies when both *stime* and *etime* are specified.

The *speaker* attribute shows the usage of nested regular patterns defined at the level of both words and characters. The 'value' section contains a sample pattern that is assumed to be exhibited by 50% of valid speaker names and its expected precision is 80%: the pattern partially relies on frequent first-name and surname lists. This value-related evidence is combined with contextual evidence stating that at least 10% of speaker names are preceded by indicative word sequences that, when present, identify a subsequent speaker name with 60% precision. A user-defined person-name co-reference resolution script is used to uncover multiple mentions of the same speaker.

Next, a generic *time* attribute follows which extracts time references from the input text. It contains an axiom that checks the time validity and also a time co-reference rule that identifies when two time entries are the same (like "noon" and "12pm"). Then two specializations of *time* are defined: the start and end times (only the start time is shown).

The system specializes an attribute value when it finds some evidence that indicates the specialization (a context pattern in this sample). All properties of the general attribute are inherited to the child.

### 2.4 Extraction Process

The inputs to the extraction process are the extraction ontology and a set of documents. The process consists of five phases with feed-back looping; further details are in [5]:

- *Document pre-processing*, including DOM parsing, tokenisation, lemmatisation, sentence boundary detection and optionally execution of a POS tagger or external named entity recognisers.
- *Generation of attribute candidates* (ACs) based on value and context patterns; an AC lattice is created.
- *Generation of instance candidates* (ICs) for target classes in a bottom-up fashion, via gluing the ACs together; high-level ontological constraints are employed in this phase. The ICs are eventually merged into the AC lattice.
- *Formatting pattern induction* allowing to exploit local mark-up regularities. For example, having a table with the first column listing staff names, if e.g. 90 person names are identified in such column and the table has 100 rows, patterns are induced at runtime that make the remaining 10 entries more likely to get extracted as well.
- *Attribute and instance parsing*, consisting in searching the merged lattice using dynamic programming. The most probable sequence of instances and standalone attributes through the analysed document is returned.

## 3 Ontology Design, Deployment and Evaluation

### 3.1 Design and Deployment of Extraction Ontologies

Clearly, the critical aspect of the WIE approach relying on extraction ontologies is the design of such ontologies. So far, in the projects mentioned in section 2, all ontologies were designed manually by experienced knowledge engineers; the time required for the initial design was in the order of several person-weeks. For some attributes it may prove difficult to enter all needed extraction knowledge manually and still achieve acceptable error rates. This can be due to large heterogeneity of the extracted values and due to the complexity or large amounts of the required extraction knowledge, or simply because of lack of the designer's knowledge (e.g. extraction from different languages). We are working in different directions to alleviate this problem:

- *Inductive models* can be trained to classify selected attributes for which training data are available. The classifier's decisions are then used within the extraction ontology patterns and can be augmented with further expert knowledge.
- When no training data are available, the designer can perform *mining* over the current extraction results in order to find frequent phrases that occur in different positions wrt. so-far extracted attribute values. The positions include left and right context, prefix, content and suffix, and different types of string overlap. The mined phrases can guide the designer in creating indicative context and content patterns.

– An alternative to building complex extraction models is to utilize evidence related to the *larger context* of data. For example, in the MedIEQ project, the extraction ontologies initially extract generic 'contact information' which is then specialized (e.g. to 'person responsible for medical content' or to 'administrative staff') using *post-processing rules* relying on page category determined by other tools.

We also investigate possibilities for reducing the amount of work in building the *conceptual structure* of the extraction ontology. Our hypothesis, partially confirmed by experiments described in [8], is that existing domain ontologies and possibly other models can be used as starting point for semi-automatically designing the structure of extraction ontologies via a set of *transformation rules*. As extraction ontologies are pre-dominantly tree-structured (they reflect the presentation structure of web/text documents), the transformation mostly has the character of *serialisation*, including steps such as converting a terminal subclass partition to an attribute of the superclass. Moreover, if even an authoritative domain ontology (DO) does not exist, state-of-the-art ontology engineering technology may allow to build it on the fly. From within large repositories of ontologies relevant ontologies can be retrieved via ontology *search* tools;[4] they can be *selected* based on their *quality evaluation* and partially *merged*.

The high-level workflow can be initiated either by adopting (or building) a DO or by directly writing an EO. In the latter case, we however lack a *target* ontological structure to be populated by extracted data. We thus assume that a DO could be *re-engineered* from an EO by following the transformation rules backwards (though such 'deserialisation' would require more human investment). Even though populating the DO using transformation rules will be a non-trivial process, it is likely to be more transparent compared to IE approaches that do not exploit ontological structures.

Finally, we assume that the EO could also be purely syntactically transformed to a semantic web (i.e. OWL) ontology, let us call it *Ex2OWL ontology*, that would serve as a DO (at the cost of being skewed towards document-oriented view).

Figure 3 depicts the scheme of prospective high-level workflow around EO-based IE. Solid edges correspond to fully-supported processes (now only the actual Ex-based IE), dashed edges to processes currently subject to intense research (the flow from ontology repository through the target DO to the EO), and dotted[5] edges to processes weakly elaborated to date (some of them amounting to mere syntactic transformations).

### 3.2 Evaluation of Ontology-Based Extraction

Common approaches to IE evaluation, have they been developed at the level of formal models [2] or e.g. pragmatically applied in the ACE programme,[6] solely focus on metrics for *result quality*. Even the presence of ontologies in IE is only reflected in scoring formulae modified so as to handle taxonomic similarity instead of exact in/correctness of results [7]. In reality, however, the result quality (typically measured by extraction accuracy) is only one factor of the overall cost; another one is the cost of *procurement of extraction knowledge*. An exception is the extraction of notorious types of generic

[4] We so far mainly experimented with *OntoSelect*, `http://olp.dfki.de/ontoselect`.

[5] Undirected edges do not refer to processes but merely to the 'population' relationship.

[6] `http://www.nist.gov/speech/tests/ace/ace07/doc/ace07-evalplan.v1.3a.pdf`

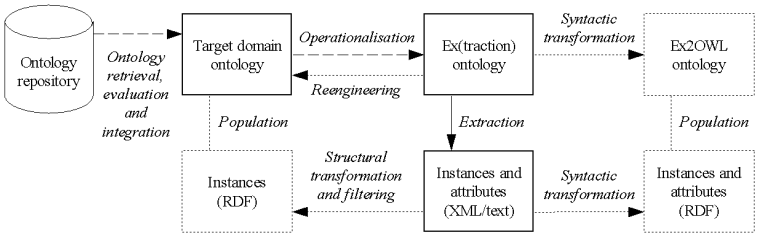


**Fig. 3.** High-level schema of (*Ex*)traction-ontology-based IE

named entities (such as peoples' names or locations in English) for which reasonably performing, previously trained tools already exist. However, in most cases, the potential user has to deal with a specific task for which no extraction model exists yet. The extreme alternatives now are 1) to let humans manually label a decent sample of the corpus and train a model, or 2) to prepare the extraction patterns by hand, e.g. in the form of an extraction ontology. Various middle ways are of course possible.

Let us sketch a very simple evaluation model that would allow to compare dissimilar IE methods including the model-building context. Instead of directly comparing the accuracy of different methods, we can declare the minimal accuracy value required for the target application (target accuracy – TA). Then we will calculate the overall cost (in terms of the human power consumed) required by those different methods in order for the TA to be reached. For a purely inductively-trained model, the cost amounts to

$$C_I = c_{annot}.n_I \tag{1}$$

where $c_{annot}$ is the cost of annotating one elementary unit (such as ontological instance) and $n_I$ is the number of annotations needed to learn a model reaching the TA. Similarly, for an extraction ontology that only uses manual extraction evidence, the cost is

$$C_O = c_{inspect}.n_O + C_{ODesign} \tag{2}$$

where $c_{inspect}$ is the cost of merely inspecting (viewing) one elementary unit and $n_O$ is the number of units that had to be viewed by the extraction ontology designer to build a model reaching the TA; $C_{ODesign}$ then is the cost of designing the actual extraction ontology. It is important to realise that $c_{inspect} << c_{annot}$ (among other, $c_{inspect}$ does not have to deal with exact determination of entity boundaries, which is a well-known problem in creating the ground truth for IE) and most likely also $n_O < n_I$; what now matters is whether this lower cost in $C_O$ is/not outweighed by the relatively high cost of $C_{ODesign}$. The model can be arbitrarily extended: e.g. for hybrid approaches (such as that we use in *Ex*) we could also consider the cost of deciding which attributes are to be extracted using which method—inductive vs. manual.

Let us, eventually, briefly touch another problem, that of *cross-validation*, which is a standard approach in evaluating inductive IE methods. While in the inductive approach an annotated dataset can be repeatedly partitioned and presented to the learning tool, we cannot do the same with the human designer of the extraction ontology, as s/he is not

as 'oblivious' as a machine. The only way of simulating cross-validation in this context thus seems to be the inclusion of multiple designers, which is in most cases prohibitive.

As partial illustration of the mentioned concepts, let us tentatively compute the cost of IE over *seminar announcements*. The utilized dataset contained 485 annotated documents, of which 240 were made available to the extraction ontology designer (who finally only needed to see a subset of these) and the remaining 245 were used for testing. After about 8 person days of development, the extraction ontology attained, on the test set, precision/recall values roughly comparable to those reported in literature; with F-measure reaching 94% for both *stime* end *etime*, and 69% and 77% for *speaker* and *location*, respectively. The accuracy for the two latter fields did not reach the best results[7] achieved by inductive algorithms like LP2 [1]. However, we can hypothesise that the total cost $C_O = c_{inspect}.240 + 8PD$ was possibly lower than $C_I = c_{annot}.485$. The comparison is further skewed by the different train/test set partitioning: one-way cut in our approach in contrast to 10-fold cross-validation used for other systems.

## 4 Conclusions

Thanks to their short development cycle, extraction ontologies are an interesting alternative for WIE when there are no or little training data. State-of-the-art ontological engineering techniques can be employed to ease their development. Fair evaluation of their performance however needs to take into account a larger context of their creation.

*The research was supported by the EC, FP6-027026, Knowledge space of semantic inference for automatic annotation and retrieval of multimedia content—K-Space.*

## References

1. Ciravegna, F.: LP2 – an adaptive algorithm for information extraction from web-related texts. In: Proc IJCAI-2001.
2. Desitter, A., Calders, T., Daelemans, W.: A Formal Framework for Evaluation of Information Extraction. Online `http://www.cnts.ua.ac.be/Publications/2004/DCD04`.
3. Embley, D. W., Tao, C., Liddle, D. W.: Automatically extracting ontologically specified data from HTML tables of unknown structure. In: Proc. ER '02, pp. 322–337, London, UK, 2002.
4. Kiryakov, A., Popov, B., Terziev, I., Manov, D., Ognyanoff, D.: Semantic annotation, indexing, and retrieval. *J. Web Sem.*, volume 2, pp. 49–79, 2004.
5. Labský, M., Svátek, V., Nekvasil, M., Rak, D.: The *Ex* Project: Web Information Extraction using Extraction Ontologies. In: ECML/PKDD'07 Workshop on Prior Conceptual Knowledge in Machine Learning and Knowledge Discovery (PriCKL'07), Warsaw, 2007.
6. Lafferty, J., McCallum, A., Pereira, F.: Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In Proc. ICML'01, 2001.
7. Maynard, D., Peters, W., Li, Y.: Metrics for Evaluation of Ontology-based Information Extraction. In: Workshop EON'06 at WWW'06.
8. Nekvasil, M., Svátek, V., Labský, M.: Transforming Existing Knowledge Models to Information Extraction Ontologies. In: 11th International Conference on Business Information Systems (BIS'08), Springer-Verlag, LNBIP, Vol.7., pp. 106–117.

[7] `http://tcc.itc.it/research/textec/tools-resources/learningpinocchio/CMU/others.html`