Semantic Extraction with Use of Frames

Jakub Dutkiewicz, Maciej Falkowski, Maciej Nowak, and Czesław Jędrzejek

Institute of Control and Information Engineering, Poznań University of Technology, Poznań, Poland

Abstract. This work describes an information extraction methodology which uses shallow parsing. We present detailed information on the extraction process, data structures used within that process as well as the evaluation of the described method. The extraction is fully automatic. Instead of machine learning it uses predefined frame templates and vocabulary stored within a domain ontology with elements related to frame templates. The architecture of the information extractor is modular and the main extraction module is capable of processing various languages when lexicalization for these languages is provided.

Keywords: Information extraction, event extraction, semantics, frames.

1 Introduction

Most of the data stored in the Internet today is unstructured and contaminated. Methods of information extraction (IE) are often designed for a pure language. We present a methodology for the extraction from contaminated data. There are basically two methods of information extraction. The first method, open extraction systems based on statistical classifiers and machine learning are scalable, but not very accurate. Rule based, domain specific systems that use linguistic patterns are more accurate. The most important element of extraction is an event, represented by a set of relations. Such an occurrent plays the central role in describing a situation. An extraction of an event encompasses three layers: syntactic, semantic and pragmatic. In our previous paper we have shown the basic idea of how to use the shallow parsing method to extract events from natural language resources [1]. The first two layers of processing are based on the process of recognizing the meanings of the extracted phrases within the sentence. This paper extends our work with presentation of the detailed architecture and methodology used in the extraction tool, including handling language ambiguity. The method is novel for the Polish language. Comparison of our method with other approaches for the English language will also be shown.

2 System Architecture

CAT IE Extractor is a system which has been developed to perform information extraction from natural language texts taken from the Internet. This type

A. Przepiórkowski and M. Ogrodniczuk (Eds.): PolTAL 2014, LNAI 8686, pp. 208–215, 2014.
© Springer International Publishing Switzerland 2014

of texts often lacks proper punctuation and may be grammatically and orthographically incorrect. Also, the language used within these texts often contains many abbreviations and phrases (absent in official and literary language), which hinders their analysis. Methodology described in this section takes such realistic language properties into account.

The Extractor processes data in several steps. The extraction itself is the final step and is performed on the structures representing a parsed sentence. Figure 1 describes consecutive stages of the analysis (inner boxes) and the names of the tools that perform those stages for the Polish language (outer boxes). The IE system itself is capable of extracting information from texts of any language, providing set of preliminary processing tools is available. For the inital stages of processing the English language we use the Sundance [6] system. (The result for the English language will be published elsewhere.)

The architecture described here is quite typical for systems of this kind. For the Polish language we use the following tools:

- TaKIPI [2] – morphosyntactic tagger,
- Liner2 [3] – named entity recognizer,
- Spejd [4] – shallow parser.

Because errors are common in analyzed texts, the first step is to normalize input, mainly its punctuation. We use our own tool for this purpose (Text Segmenter), which attempts to fix proper markings at the end of sentences. It also adds the commonly skipped diacritic signs, replaces the phrases with common ortographical errors and replaces commonly used abbreviations with their non-abbreviated equivalents. Before we proceed to the details of the IE system, it is crucial to introduce the key concepts and structures we use in the information extraction process, namely frames, frame instances and extraction patterns.

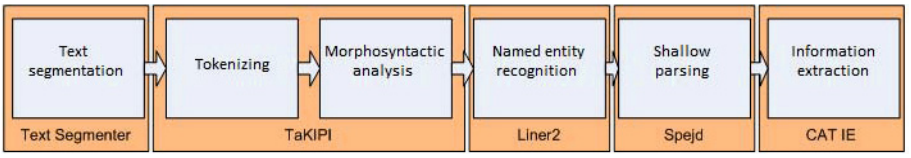

Fig. 1. Pipeline of architecture components

2.1 Frames

Data extracted as events from texts are expressed with artificial intelligence structures called frames. Frames can be considered as a domain data model. A frame is defined as a pair (R, S), where R is the semantic type of frame, and S is the set of its slots. Every slot in S is defined as a pair (R, T), where R is the relation, and T is a list of allowed semantic types of its object. An example of a frame of a type #Purchase, which describes the event of purchasing is

presented in Table 1. The semantic types of frames and slots are expressed as ontology classes. This ensures richer semantics in the extracted data and also allows binding the vocabulary used to express slots with the ontology. This construction makes our system flexible and it is a step towards connecting existing lexical resources with our ontology (e.g. using a system similar to Lemon [5]).

Table 1. Data structures for the extraction of the *Purchase* event

Frame		**Template**		**Instance**	
Type: Purchase		Anchor: Active verb		bought	
Slots:		Slots:		Slots:	
Relation (attribute)	Allowed type	Case	prepositions	Value	Type
Buyer	#Person, #Organization	nominative	–	Jan Kowalski	#Man
Seller	#Person, #Organization	dative	–	–	–
Bought object	#Thing	accusative	–	new Fiat	#Car
Price	#Money	accusative	“za" (“for")	for a song	#Money
Place	#Place	locative	“w" (“in")	–	–
Date	#Date	locative	“w" (“in")	last year	#Time

2.2 Frame Extraction Template

A frame extraction template specifies a set of rules, because of which frames can be extracted from the text. A template consists of an anchor and a set of slots (attributes of an event). The extraction process starts with finding an anchor. An anchor is mostly a verb or a verb phrase, when it appears in a derivation (i.e. a syntactically parsed sentence), it triggers the process of recognition of an event. Once the anchor is found, the process attempts to assign slots. Matched phrases are called anchor fillers and slot fillers, respectively. The matching conditions are expressed by the restrictions of two types – first grammatical (making a phrase a candidate for an anchor or a slot) and lexical. A template defines the interface between texts and a data model. A sample template is illustrated in Table 1.

Grammatical conditions consist of the allowed grammatical cases of the phrase and the allowed prepositions at the beginning of the phrase. Note, that in Polish, each case of a noun has its own morphological form. If there are no prepositions defined in the template, prepositional phrases will not be taken into consideration. If there is at least one preposition in the template, phrases with no preposition will not be taken into consideration.

Once the grammatical conditions are fulfilled, the system attempts to find a semantic type of template element which corresponds to the textual value of the head of the phrase. The system uses dictionaries stored within the ontology to perform this task. The matching process has two stages. First, the system verifies the matching of at least one of the allowed semantic types of an anchor.

If the anchor matching is successful, the system next attempts to match slots. There are three possible outcomes of this process:

- Slot matched – The textual value of a phrase matches any allowed semantic type defined in the ontology.
- Slot not matched – The textual value of a phrase matches no allowed semantic type defined in the ontology.
- No match – The textual value of does not appear in the ontology; the phrase is either marked as a slot value candidate (aggressive mode), or it is rejected (conservative mode).

In this way determined template elements become an input in the evaluation process. Usually the conservative mode generates the correct extraction, but at the same time it rejects many correct slot matches, thereby making the extraction error. This is caused mainly by the quality of vocabulary stored in the ontology. On the other hand, the aggressive mode generates many more extractions than the conservative one, though, at the expense of their quality; in this mode incorrectly filled slots and incorrect frames are more common.

2.3 Frame Instance

A unit of information that is created as an effect of the extraction process is a frame instance (FI). A frame instance is created by filling slots of an appropriate frame with textual values including the semantic type of the slot. This semantic type can be obtained either from the Named Entity Recognition (NER) subsystem (if the phrase was recognized as a named entity) or from the ontology subsystem, which maps the phrases to ontological semantic types with its vocabulary. As an illustration in Table 1 of the process we used the following sentence: "Apparently Jan Kowalski bought a new Fiat for a song last year" *"Podobno Jan Kowalski kupił w zeszłym roku nowego Fiata za bezcen"*. The type of the extracted frame is #Purchase – it contains information describing the event of purchasing. Only a part of the possible slots are filled with values, since there were no remarks in the text about a place of purchase nor information about a seller.

2.4 Sentence Disambiguation

Let us define the sentence as a set of phrases.

$$\{p_{\mathrm{i}} : p_{\mathrm{i}} \in S\} \tag{1}$$

According to this definition, sentence contains a number of phrases. Due to the ambiguity of natural language, the result of shallow parsing does not give exactly one correct interpretation. Instead, it returns all available interpretations of phrases in the sentence as it is defined in (2). We assume that each phrase has the denoted, most probable phrase interpretation, as it is described in (3).

$$\{p_{\mathrm{i,j}} : p_{\mathrm{i,j}} \in p_{\mathrm{i}}\} \tag{2}$$

$$\forall p_{\mathrm{i}} \in S \; \exists p_{\mathrm{i,j}} \in p_{\mathrm{i}}, \; p'_{\mathrm{i}} = p_{\mathrm{i,j}} \tag{3}$$

The entire pool $I(S)$ of available sentence S interpretations could be interpreted as following:

$$I(S) = \{s_{\mathrm{i}} : \forall p_{\mathrm{j}} \in S \; \exists p_{\mathrm{j,k}} \in p_{\mathrm{j}}, \; p_{\mathrm{j,k}} \in s_{\mathrm{i}}\} \tag{4}$$

Unless the size of a set $I(S)$ exceeds a large number, the extraction process is performed on every single sentence interpretation. If that number exceeds predefined large number, extraction is performed on a sentence that consists solely of the most probable phrases. Out of all the extraction results one is chosen using the disambiguation process. The disambiguation process uses two arguments. The first one is the normalized number of changes between the most propable phrase interpretations and actual phrase interpretations in the sentence. That normalized number of changes $c(s_{\mathrm{i}}, S)$ with the interpretation s_{i} of sentence S is defined in (5)

$$c(s_{\mathrm{i}}, S) = \frac{|p'_{\mathrm{i}} : p'_{\mathrm{i}} \in s_{\mathrm{i}}|}{|p_{\mathrm{i}} : p_{\mathrm{i}} \in S|} \tag{5}$$

The second argument of the disambiguation process is the actual interpretation of results. Currently we are using the aggressive approach – the more slots filled, the better. The $c(s_{\mathrm{i}}, S)$ argument in the disambiguation process is secondary to the amount of extraction. If we use the quality function to reduce the number of interpretations to one, we remove the ambiguity of the sentences at this point. To denote that function let us define $|E(s_{\mathrm{i}})|$ as the a number of extracted slots from the interpretation s_{i}. The disambiguation process is described in (6) and (7).

$$s_{\mathrm{i}} \in I(S') \iff \forall (s_{\mathrm{j}} \in I(S), s_{\mathrm{j}} \neq s_{\mathrm{i}}) \; E(s_{\mathrm{i}}) \geq E(s_{\mathrm{j}}) \tag{6}$$

$$s_{\mathrm{disamb}} = s_{\mathrm{i}} \iff s_{\mathrm{i}} \in I(S'), \forall (s_{\mathrm{j}} \in I(S'), s_{\mathrm{i}} \neq s_{\mathrm{j}}) \; c(s_{\mathrm{i}}, S') \geq c(s_{\mathrm{j}}, S') \tag{7}$$

If the disambiguation process returns more than one sentence, the disambiguated sentence is chosen randomly out of all returned interpretations.

3 Evaluation

The evaluation process uses frame instances (not individual elements) with filled candidate anchors and slots. To be evaluated positively a template has to have the correctly assigned Agent and Patient (if it appears) thematic roles. Let us give examples for the #Killing event. A result depends on the ontology or named entity recognition along with annotation rules. Annotator decisions (as determined by “gold standard") are positive or negative. The system outcome can also be positive or negative. Examples are presented in Table 2. Case 1 – Alice was recognized by NER, *książka* is in the ontology. Case 2 – *Alice* was not recognized by NER, *książka* is not in the ontology. To evaluate the accuracy of described method we have downloaded data from the National Corpus of Polish (NKJP) [7]. We have randomly chosen 1000 sentences with the word “kill" and 1000 sentences with the word “purchase". We have manually annotated each of the 2000 sentences. If the event was present in the sentence, the annotator was

Table 2. Example evaluation results

Case	Polish clause	English clause	Assignement (aggressive mode)
Case 1/2	*Jan zabił Alę*	John killed Alice	True positive
Case 1	*Jan zabił książkę*	John killed a book	True negative
Case 2	*Jan zabił książkę*	John killed a book	False positive

Table 3. Precision and recall measures for the extraction process

	Recognizing events		Recognizing slots	
Event type	Precision	Recall	Precision	Recall
#Kill	33%	99%	48%	46%
#Purchase	52%	98%	48%	46%

meant to define all apparent slots for the event. Another restriction given to annotators was that, they annotate only #Kill and #Purchase events. Within the 1000 sentences with “kill" based words, annotators chose 268 with the event and have marked 424 slots. For the sentences with “purchase" based words, annotators chose 319 sentences and marked 599 slots within those sentences.

Exactly the same set of sentences was used to perform the extraction with the IE system. To succesfully run the experiment in the conservative mode, we would need to implement a large resource vocabulary to our ontology. The vocabulary we have implemented thus far lets us run the process in the aggressive mode. The system chose 603 sentences with the #Purchase event and 788 sentences with #Kill event. The set of 603 sentences contained all annotated sentences but 3. The set of 788 sentences contained all annotated sentences but 6. The extractor chose 277 slots correctly and 292 slots incorrectly for the #Purchase event: it chose 262 slots correctly and 184 slots incorrectly for #Kill event. The measures evaluated are presented in Table 3.

Table 4. Sample extraction

Sentence:	At the online auction, Heinz bought a cube for a pile of money.
Sentence (literally):	*Za grube pieniądze Heinz kupił kostkę na aukcji internetowej.*
Bought object:	a cube
Price:	for a pile of money
Place:	at the online auction

An example of the extraction is presented in Table 4. An error in the presented extraction highlights one of the major problems with the methodology. There is a high possibility, that a word of foreign origin will be marked with incorrect grammatical tags. This, low level error propagates trough the entire process. Propagation of errors is an issue, as a mistaken slot filler makes the extraction partially incorrect. Evaluation of the partially incorrect frame instances is summarised in Table 5. The $Precision_{\mathrm{er}}$ measure is the precision of event recognition,

Table 5. Distribution of results regarding the number of errors within frames

Number of errors	Event type	Precision	Recall	$\frac{Precision}{Precision_{er}}$
0	#Kill	17%	50%	51%
<2	#Kill	28%	84%	85%
<3	#Kill	33%	91%	100%
0	#Purchase	17%	31%	33%
<2	#Purchase	36%	69%	69%
<3	#Purchase	47%	90%	90%

as it is presented in Table 3. The $\frac{Precision}{Precision_{er}}$ is the precision of extraction if we take only the frames which were correctly recognized as events.

4 Conclusions

In this work we improved our previous methodology [4] to enhance domain extraction. We created a mechanism that uses constraints described in the structualized data to perform extractions. We used the pattern based approach (with aggressive mode of matching). The next step would be to create a probabilistic model to find missing thematic roles outside of sentences with anchoring keywords. The methods described in this paper could be enhanced in several ways. The first, similar to the methods used in Boxer, would be based on the construction of a large ontology for most of verbs used in Polish. Such an ontology would look very similar to Verbnet. Valence dictionaries [4] would be helpful in this task. With a large enough ontology we could be able to extract and represent entire messages in a discourse representation compatible with ontological structures. This approach is scalable to a point: not only ontologies have to be enhanced but annotated corpora would likewise require enhancement. This way of improving our methodology would also require methods of representing the relations between events. On a deeper level the improvement of evaluation measures would also require enhancmenet of the shallow grammar parser used and neccesitate solving pronoun and coreference problems. Our method is similar to the Boxer [9] semantic parser, applied mostly to the English language. For selected events, used in this paper our thematic roles are more specific then for Verbnet derived eqiuvalent roles. For example, we use Perpetrator instead of Agent and Victim instead of Patient. There is a difference in the process of template matching for the Polish and English languages. Creating rule based matching templates for Polish is easier than for English, because of the greater expressive power of Polish with regards to its morphology. We plan to demonstrate this in our future work . That is why the recent work for template matching for English currently uses statistical methods [8].

Acknowledgements. This work was supported by the Polish National Centre for Research and Development (NCBR) No O ROB 0025 01 and 04/45/DSPB/0105 grants.

References

1. Dutkiewicz, J., Jędrzejek, C.: Ontology-based event extraction for the Polish language. In: LTC 2013 (2013)
2. Piasecki, M.: Polish Tagger TaKIPI: Rule Based Construction and Optimisation (2007)
3. Marcińczuk, M., Kocoń, J., Janicki, M.: Liner2 – A Customizable Framework for Proper Names Recognition for Polish. In: Bembenik, R., Skonieczny, Ł., Rybiński, H., Kryszkiewicz, M., Niezgódka, M. (eds.) Intell. Tools for Building a Scientific Information. SCI, vol. 467, pp. 231–254. Springer, Heidelberg (2013)
4. Przepiórkowski, A.: Powierzchniowe przetwarzanie języka polskiego, Akademicka Oficyna Wydawnicza EXIT, Warszawa (2008)
5. McCrae, J., Spohr, D., Cimiano, P.: Linking lexical resources and ontologies on the semantic web with lemon. In: Antoniou, G., Grobelnik, M., Simperl, E., Parsia, B., Plexousakis, D., De Leenheer, P., Pan, J. (eds.) ESWC 2011, Part I. LNCS, vol. 6643, pp. 245–259. Springer, Heidelberg (2011)
6. Riloff, E., Phillips, E.: An introduction to the sundance and autoslog systems (2004)
7. Lewandowska-Tomaszczyk, B., Bańko, M., Górski, R.L., Łazinski, M., Pęzik, P., Przepiórkowski, A.: Narodowy Korpus Języka Polskiego: geneza i dzień dzisiejszy (2010)
8. Huang, R., Riloff, E.: Modeling Textual Cohesion for Event Extraction. In: Proceedings of the 26th Conference on Artificial Intelligence, AAAI 2012 (2012)
9. Curran, J.R., Clark, S., Bos, J.: Linguistically Motivated Large-Scale NLP with C&C and Boxer. In: Proceedings of the ACL 2007 Demonstrations Session (ACL-07 demo), pp. 33–36 (2007)