KNEWS: Using Logical and Lexical Semantics to Extract Knowledge from Natural Language

Valerio Basile\textsuperscript{1} and Elena Cabrio\textsuperscript{2} and Claudia Schon\textsuperscript{3}

Abstract. We present KNEWS, a pipeline of NLP tools that accepts natural language text as input and outputs knowledge in a machine-readable format. The tool outputs frame-based knowledge as RDF triples or XML, including the word-level alignment with the surface form, as well as first-order logical formulae. KNEWS is freely available for download. Moreover, thanks to its versatility, KNEWS has already been employed for a number of different applications for information extraction and automatic reasoning.

1 Introduction

Machine Reading \cite{7} is the task of extracting formally encoded knowledge from natural language text. It subsumes Natural Language Understanding: after resolving the ambiguities in the text, the information extracted is grounded by linking it to a knowledge base. A complete machine reading tool is a step towards the construction of large repositories of general knowledge without having to rely on human-built resources. Such system can automatically learn, for instance, that the Knife is used as Instrument to Cut the Bread, where Knife, Instrument, Cut and Bread are all entries in some knowledge base on the Web. Moreover, a machine reading component can play an important role in other environments: in many automated reasoning applications, it is necessary to use large ontologies as background knowledge. Machine reading helps to disambiguate predicate names. This supports selection methods like SInE \cite{7} to select very focused background knowledge.

Several tools have been published to solve parts of this complex task, notably the extraction of entities \cite{15,16}, relations \cite{12}, semantic roles \cite{10}, and more. FRED \cite{14} is a notable example, proposing a complete machine reading pipeline that extracts RDF triples comprising every aspect of the semantics of the input text.

In this demo we present KNEWS, a complete and versatile text-to-knowledge pipeline for machine reading, that tries to overcome some of the limits of existing systems. In particular, in comparison with FRED, from which we drew the initial inspiration, KNEWS is configurable to use different external modules, provides different kind of meaning representations as output, and, last but not least, its source code is freely available and not bound to online APIs. KNEWS is available at \url{https://github.com/valeriobasile/learningbyreading} a demo is at \url{http://gingerbeard.alwaysdata.net/knews/}

2 NLP Pipeline for Knowledge Extraction

KNEWS is a pipeline system. The main components are a semantic parser and two modules for word sense disambiguation and entity linking. KNEWS works by running these components separately on a text, then it aligns the output of the the semantic parser to the output of the other two modules (Figure 1).

Semantic parsing. The semantic parsing module must process the input text and output a complete formal representation of its meaning. To this aim, KNEWS employs the C&C tools and Boxer. The C&C tools \cite{6} are a pipeline of statistical NLP tools including a tokenizer, a lemmatizer, named entity and part-of-speech tagger, and a parser that creates a Combinatorial Categorial Grammar representation of the natural language syntax. Boxer \cite{5} is a rule-based system that builds an abstract meaning representation on top of the CCG analysis. Such structures contain, among other information, predicates representing the roles of the entities with respect to the detected events, e.g., event(A), entity(B), agent(A,B) to represent B playing the role of the agent of the event A.

Word sense disambiguation and Entity Linking. In order for the semantic representations to be useful in contexts such as knowledge representation and automatic reasoning, the predicates need to be linked to a knowledge base. KNEWS uses WordNet \cite{11} to represent concepts and events, DBpedia \cite{11} for named entities, and FrameNet’s frames \cite{11} to represent events, integrating the mapping with the WordNet synsets provided by FrameBase \cite{7}. The inventory of thematic roles used by Boxer is taken from VerbNet \cite{18}, while KNEWS employs the mapping provided by SemLinks \cite{13} to link them (whenever possible) to FrameNet roles. By linking the discourse referents representing concepts in a DRS to WordNet synsets, entities to DBpedia and events to FrameNet frames, KNEWS is able to extract complete meaning representations from natural language text linked to Linked Open Data knowledge bases.

1 Université Côte d’Azur, Inria, CNRS, I3S, France, \{valerio.basile@inria.fr\}
2 Université Côte d’Azur, Inria, CNRS, I3S, France, \{elena.cabrio@inria.fr\}
3 Universität Koblenz-Landau, Germany \{schon@uni-koblenz.de\}

\url{https://dbpedia.org}
3 Output Modes

Frame-based Semantics The first output mode of KNEWS is frame instances, sets of RDF triples that contain a unique identifier, the type of the frame, the thematic roles involved in the instance, and the concepts or entities that fill the roles. The format follows the scheme of FrameBase [12], which offers the advantage of inter-operability with other resources in the Linked Open Data cloud. An example of frame instance, extracted from the sentence “A robot is driving the car” is given by the following triple [5]

\[
\text{fb:fi-lc59fa56 a fb:frame-operate_vehicle-drive-v} .
\text{fb:fi-lc59fa56 fb:fe-Driver dbr:Robot} .
\text{fb:fi-lc59fa56 fb:fe-Vehicle wn:02961779-n} .
\]

This output mode of KNEWS has been employed in [4] to create a repository of general knowledge about objects.

Word-aligned Semantics The second output mode of KNEWS is similar to the previous one (frame instances) with the difference that it contains as additional information the alignment with the text. We exploit the Discourse Representation Graph [2] output of Boxer to link the discourse referents to surface forms, i.e., span of the original input text, resulting in a word-aligned representation as follows:

\[
\text{<frameinstances>}
\text{<frameinstance id="9a3fa55e" type="Operate_vehicle-drive-v" internalvariable="el">}
\text{<framelexicalization>A robot is driving the car</framelexicalization>}
\text{<frameinstances>}
\text{<frameinstance id="fb:fi-x1b07969" role="Driverv" internalvariable="x1">}
\text{<concept>http://wordnet-rdf.princeton.edu/wn31/02961779-n</concept>}
\text{<rolelexicalization>x1 is driving the car</rolelexicalization>}
\text{<frameinstances>}
\text{<frameinstance id="fb:fi-x25408" role="Vehicle" internalvariable="x2">}
\text{<concept>http://wordnet-rdf.princeton.edu/wn31/02961779-n</concept>}
\text{<rolelexicalization>x2 is driving the car</rolelexicalization>}
\text{<frameinstances>}
\text{</frameinstances>}
\text{</frameinstance>}
\text{</frameinstances>}
\]

Such surface forms can be complete (e.g., “A robot”) or incomplete (e.g., “x1 is driving x2”) and can be composed to recreate the original, as well as lexicalizations for new knowledge [3].

First-order Logic In the third output mode, KNEWS is able to generate first-order logic formulae representing the natural language text given as input. For example the input “She won a spelling bee.” leads to the following first-order logic formula:

\[
\exists A, B, C (r1\text{Theme}(A, C) \land r1\text{Agent}(A, B) \land 201102550-v(A) \land 200940051-n(C) \land r1of(C, B) \land 107907011-n(B) \land ndfemale(B))
\]

 corresponding to a disambiguated version of the following one:

\[
\exists A, B, C (r1\text{Theme}(A, C) \land r1\text{Agent}(A, B) \land \text{win}(A) \land \text{bee}(C) \land r1of(C, B) \land r1spelling(B) \land ndfemale(B))
\]

Providing the predicates as Wordnet synsets has the advantage that this information can be exploited to select background knowledge in a much more focused manner, as proposed in [8]. In the example above, the use of KNEWS ensures that “spelling bee” is distinguished from the insect bee. This can be used to avoid selecting background knowledge on insects. The syntax of the formula returned by KNEWS is very similar to the well-known TPTP format [19] — in fact we plan to provide TPTP syntax directly as output of KNEWS.

REFERENCES


