Linguistically Light Lexical Extensions for Ontologies

Brian Davis[†], Siegfried Handschuh[†], Alexander Troussov[‡], John Judge[‡], Mikhail Sogrin[‡]

[†]Digital Enterprise Research Institute,
National University of Ireland Galway,
IDA Business Park, Lower Dangan,
Galway, Ireland
{brian.davis, siegfried.handschuh}@deri.org

[‡]IBM LanguageWare,
Dublin Software Lab,
IBM Technology Campus,
Dublin 15, Ireland
{atrousso, johnjudge, sogrimik}@ie.ibm.com

Abstract

An increasing number of enterprises are beginning to include semantic web ontologies into their Information Extraction (IE) and Text Analytics (TA) applications. This can be challenging for a TA group wishing to avail of semantic web ontologies due to the manual effort of retargeting and tailoring language resources within the TA system to a new domain to meet customer needs. A lightweight lexical layer within an ontology offers a solution to this problem. Furthermore, the identification of class instances within unstructured text for either the purposes of ontology population or semantic annotation are usually limited to term mentions of proper noun, personal noun or fixed key phrases within Text Analytics or Ontology Based Information Extraction (OBIE) applications. These systems do not generalise to cope with compound nominal classes of multi word expressions. LEON, a set of Lexical Extensions for Ontologies offers a solution to this problem. We describe LEON, which encodes light linguistic features of lexical entries for concepts within an ontology, as well as a lightweight lexical analyzer which complies the LEON metadata into efficient an dictionary format to drive large scale identification and semantic annotation of concepts mentioned in text.

1. Introduction

An increasing number of enterprises are beginning to include semantic web ontologies into their Information Extraction and Text Analytics process regardless of whether this is to model the application domain or to model the internal data structures of text analytics system itself.

The Semantic Web community is also increasingly becoming aware of the need to encode linguistic knowledge concerning concepts directly into ontologies. The incorporation of linguistic data within an ontology is frequently for descriptive purposes only or to support ontology localisation. However, other approaches have utilised lexical extensions within an ontology to drive an Ontology Based Information Extraction or a Semantic Annotation platform, whereby, the linguistic layer replaces the internal conceptual data structures of the IE engine eliminating the persistent need to map IE language resources to ontologies.

Therefore, a major challenge for a TA group wishing to avail of semantic web ontologies is how to minimise the costly manual effort of retargeting and tailoring dictionaries and language resources within the TA system to a new domain to ensure meeting customer needs and timelines. A lightweight lexical layer within an ontology offers a solution to this problem. The other major challenge involves deciding on how much linguistic expressivity to encode into the ontology in order to ensure high precision/recall without impacting on scalability, maintainability and usability of the TA application.

In order to illustrate the potential role of lexical extensions to an ontology, we present LEON, a set of Lexical Extensions for Ontologies, which can help in the identification of instances. We describe a use case based on requirements gathered from a leading industry TA group. We outline our conceptualisation of a lexical layer within an ontology. This "lexical layer" as such is subsequently statically compiled into efficient Finite State Machine (FSM) dictionaries of equivalent format to drive a lightweight lexical analyser for semantic annotation. Consequently, we have developed a Linguistic Light Scanner (LLS) based on the lexical layer format, and tested it with the MeSH vocabulary and PubMed documents corpus.

Finally, in order to illustrate the benefits of adding lexical layers to ontologies, we focus on identifying concepts represented as Multiword Expressions (MWE) — specifically compound nominals, in text.

2. Related Work

2.1. Linguistic Support for Ontologies and Information Extraction

The inclusion of a linguistic or lexical layer is by no means a new phenomenon. For example, LingInfo, was developed as part of the SmartWeb[1] project (Buitelaar et al., 2006). The work conceptualised the idea of a linguistic layer for a Semantic Web Ontology or more specifically a "multilingual/multimedia lexicon model for ontologies" (Buitelaar et al., 2006). Linguistic representation in LingInfo can consist of: a language identifier, Part of Speech (POS) tag, morphological data, and syntactic compositional data as well a contextual data in the form of grammar rules of n-grams. Furthermore, content and knowledge are organised into four layers, where the ontology layer is located at the central layer and linguistic features and their subsequent associations to the central layer are located in the outer middle layers with the outer layer containing textual content. The LingInfo model is applied to the SmartWeb Integrated Ontology SWIntO, whereby the linguistic feature layer is compiled into language resources (gazetteers) within the SProuT IE engine based on a mapping between the SWIntO and SProUT's Type Description Lan-

[1] http://smartweb.dfki.de/

guage (TDL). This mapping is applied to both SWIntO concepts and properties. The work of (Buitelaar et al., 2006) is influenced strongly by LMF, Lexical Markup Framework (Francopoulo et al., 2006), which is part of the ISO TC37/SC4[2] working group on the management of Language Resources. LMF has its origins in language engineering standardisation initiatives such as EAGLES[3] and ISLE[4].

2.2. Compound Nominals as Multiword Expressions

Compound nominals such as "*car park*", "*attorney general*", "*part of speech*" are similar to non-decomposable idioms in that they do not alter syntactically and can inflect for number (Sag et al., 2002). This can be catered for by simply adding an "s" to the string, making, for example, "*car parks*". However as the authors highlight, this is not the case for left-headed compounds such as "*attorney general*", "*congressman at large*" and "*part of speech*". Taking such a simple approach towards handling inflection would result in abnormalities such as "**congressman at larges*" and "**part of speechs*". To resolve this, a system could simply list all the singular and plural forms of each compound nominal. However, this leads to lexical proliferation, which is undesirable as it makes the lexicon difficult to maintain and increases the likelihood of human error being introduced. The authors note that the listing approach is less "dramatic" when compared to that of non-decomposable idioms. Nevertheless the listing approach is still very inefficient. Furthermore, not all compound nominals are syntactically unalterable and are furthermore categorised as semi-fixed expressions. Initially, one could specify lexical entries as in the list below (often referred to as citation form):

Marie Curie

Alzheimer's disease

Industrialised countries

Debt consolidation

This is very easy for a user building a lexicon to achieve but in real texts compound nominals appear in varied forms according to syntax and inflection. Hence, the above examples could occur within real text as such:

Marie and Pierre Curie

Diseases like Parkinson's and Alzheimer's

Industrialised and developing countries

Consolidation of credit card debts

The "listing" approach for tackling compound nominals does not cater for these variations in syntax.

[2]http://www.tc37sc4.org/

[3]http://www.ilc.cnr.it/EAGLES96/browse.html

[4]http://www.mpi.nl/ISLE/

3. Use Case for Linguistically Aware Ontology in a Commercial IE Setting

This work is based on a research collaboration between the Digital Enterprise Research Institute (DERI), Galway and the IBM LanguageWare[5] team in Dublin, which arose as a result of joint work on the NEPOMUK Project[6]. The purpose of this research collaboration is to investigate the leveraging of semantic technologies into commercial Information Extraction or more specially Ontology Based Information Extraction. Consequently, due to the commercial nature of the LanguageWare product[7], our requirements for employing the use of ontologies in the IE process are constrained as follows:

1. The system must minimise the retargeting of Language Resources (LR) to a new domain ontology

2. There must be a low manual cost for extending LRs in case of shift or extension within the domain ontology

3. Performance of the system must be scalable in order to be accepted by industrial enterprises and their customers

4. The system must ensure high quality precision/recall accuracy

5. The system must be able to cope with legacy knowledge or non Semantic web based ontologies, e.g. MeSH[8] and existing internal IBM taxonomies.

4. Implementation

4.1. Lexical Extensions for Ontologies — LEON

We call our approach for adding lexical extensions to an ontology LEON. It focuses not on the linguistic description of vocabulary associated with a concept but on the linguistic features of a given concept in order to identify class instances in text. In addition, our approach is deliberately less complex in order to cater for users, in particular knowledge engineers, who lack a linguistic background, but may wish to develop an ontology with linguistic features included. Moreover, we particularly focus on compound nominals as opposed to simple personal or proper noun recognition, in order to illustrate the benefits of including linguistic information within an ontology. LEON corresponds to the definition and inclusion of a lexical description as a meta class within an ontology as well the internal format of the subsequently compiled FSM dictionaries.

For each ontology entity which specifies a lexical entry, a citation form is included as well as a set of *user adjustable* constraints in order to improve extraction precision. Thus, the LEON metadata is structured as such:

[5]http://www-306.ibm.com/software/globalization/topics/languageware/index.jsp

[6]http://nepomuk.semanticdesktop.org

[7]http://www.alphaworks.ibm.com/tech/lrw

[8]http://www.nlm.nih.gov/mesh/meshhome.html

1. Citation form
2. Constraints:
   (a) Fixed/variable word order
   (b) Number of intervening tokens (i.e. "diseases like Alzheimer's" = +1 Token in comparison to the citation form "Alzheimer's disease")
   (c) Boolean values for exact capitalisation (based on citation form)
   (d) Boolean values for exact string match (based on citation form)

4.2. Linguistic Light Scanner

The Linguistic Light Scanner (LLS) is a simple lexical analyzer or lexer which can detect term mentions of concepts within text. During text processing, the scanner scans the text, normalises words, and detects if all the constituents of some LEON lexical units are present in a suitable window of text. When a signature of any lexical expression is detected, compiled LEON constraints (Section 4.1.) are used to accept or reject the possibility of the signature being a valid occurrence of the lexical entry.

4.2.1. Examples of Processing Compound Nominals

Compiled LEON metadata allows the Linguistic Light Scanner to distinguish between MWEs with fixed and variable word order and to handle syntactic alterations in the surface forms encountered in the text.

For example, LLS will detect the signature of the proper name MWE "*United Nations*" in the passage "*. . . nation united by . . .*" if compiled LEON constraints for this entry (user tailored or default) specify that word order, capitalisation or specific inflections of constituents in the citation form must be respected, then the entry will be rejected in this case.

However the LLS would detect the signature of the syntactically alterable MWE "*debt consolidation*" in the passage "*. . . consolidation of credit card debts . . .*", only if constraints for this lexical entry do not prohibit free word order of constituents as well as inclusion of additional tokens between constituents.

4.2.2. The LLS Dictionary

In order to to drive the IE process, all LEON instances within an ontology are traversed and compiled into lexico-semantic resources based on layered dictionaries, whereby lexical information and semantic information are mapped to independent layers, while at the same time linking them together when term mentions are encountered in text by using the LLS. The "key" for this linking is the citation form which will be recognised in text and the semantic concept(s) can be found using this citation form Figure 1 shows an overview of this layered approach.

There may be one to many, many to one or many to many mappings between citation forms (the "key") and concepts within the semantic layer. This essentially constitutes a need for instance disambiguation, which is deferred to the semantic layer. The resolution of such ambiguities can be handled be applying graph mining techniques to the ontology graph contained within the semantic layer (Judge et al., 2007).

The LLS dictionary is represented as a finite state machine (FSM), in which transitions are made upon locating a constituent of a multi-word expression. Each state in the FSM may have a flag indicating whether words found constitute a multi-word expression, as well as some constraints indicating how the MWE can inflect or be split up by intervening words, and the original citation form or the information necessary to reconstruct it.

During dictionary compilation time citation forms are converted into an unordered set of normalised constituents, which we call the *signature* of a lexical entry. For example, the lexical entry for "*United Nations*" might be stored by LLS internally as two transitions in FSM {"*nation*", "*united*"}, starting from the entry state, and the state after the last transition would contain a terminating flag and a set of constraints. Each state may actually contain several sets of constraints when there are multiple different MWEs which constituents after normalisation and reordering become identical.

4.2.3. LLS Algorithm

The LLS processes document text within a sliding window of an appropriate size (to ensure that the longest possible expression fits inside). Paragraphs and sentences may also provide natural boundaries. Words in this window are normalised and sorted in the same way as signatures for dictionary entries, and then checked against the dictionary FSM to find all potential matches. To do it, a subsequence matching algorithm is used, which finds all MWEs, all constituents of which are present.

Then, all constituents of a potential match are mapped back to the original text and expression constraints are checked. And finally, if necessary, a citation form of the found MWE is reconstructed from MWE constituents.

A possible complication of the sliding window approach used is that sometimes the window boundary breaks apart a multi-word expression. This means that another, shorter, expression can be found. In this case, if one expression is found to contain all the constituents of another, then the shorter expression will be discarded and the longer expression kept.

4.2.4. Implementation details

The reference implementation of LLS is done using Java programming language. A number of open-source and publicly available libraries are used:

- *Apache UIMA* (Unstructured Information Management Architecture)[9] provides document analysis and annotation framework. And Linguistic Light Scanner is implemented as an UIMA annotator.
- *ICU* (International Components for Unicode)[10] is a set of libraries providing Unicode and Globalisation support. It is used to perform text tokenisation and to compare and sort words with or without regard to capitalisation or accented characters. ICU can also use

[9] http://incubator.apache.org/uima/
[10] http://icu-project.org/

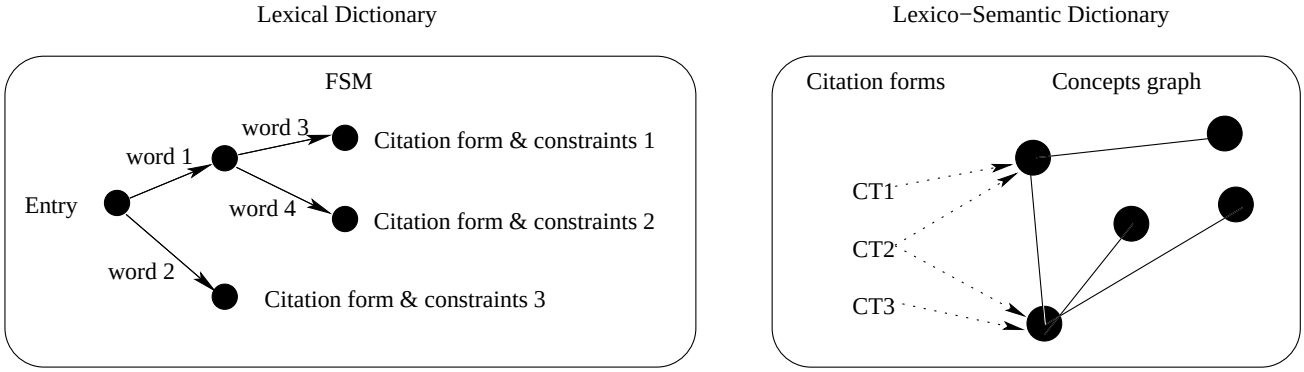

Figure 1: Lexical and Lexico-Semantic Dictionaries

conventions and standards of a particular language, region or country.

The LLS implementation allows customisation of morphological normalisers or lemmatisers. The reference implementation uses a simple S-suffix normaliser for English language, which strips plural or possessive case suffixes. A more advanced normaliser may be used, like the Porter stemmer (Porter, 1980), or normalisers for other languages.

5. Evaluation

Our evaluation of the lexical layer (LEON) and LLS was performed on documents from the medical domain. Such documents usually contain many multi-word expressions which are syntactically alterable. For example the MWE "*11 beta-hydroxysteroid dehydrogenase*" (citation form) can appear in text as "*hydroxysteroid 11-beta dehydrogenase*".

A LLS dictionary was built using MeSH (Medical Subject Headings) vocabulary, which contained 92,542 expressions. All expressions which were added were given default constraint values (variable word order, allow two intervening tokens, case-insensitive). Note that no human optimisation of the constraints for individual MWEs was performed. These default values were somewhat crudely applied, however, it represents a very basic set of LEON data to use as a baseline. In the real world the constraints would be set by the knowledge engineer encoding the lexical layer in the ontology.

The evaluation corpus used was PubMed[11], which is a collection of documents containing biomedical and life sciences journal articles. PubMed provides a specific collection of 63,430 open-access documents for data-mining purposes.

5.1. Results

Table 1 shows the results for analysis of the PubMed corpus. The Linguistic Light Scanner was used to detect MWEs in the texts using compiled LEON metadata.

After processing all PubMed documents, LLS found 17,666,135 expressions in the texts, of which 672,697 (3.81%) expressions contained some intervening tokens and 473,557 (2.68%) expressions had a different word order from the citation form, and 326,823 (1.85%) had both variable word order and intervening tokens. Therefore, in total, allowing variable word order and intervening tokens in expressions has helped the Linguistic Light Scanner to find and identify 4.86% more expressions.

Exprs with fixed word order & no intervening tokens found	17345322
Exprs with variable word order & intervening tokens found	17666135
Of which	
Variable order	473557
Intervening Tokens	672697
Both	326823

Table 1: Results using LEON and LLS on PubMed texts

If we compare the performance of the LLS with LEON data (Table 1) to that without the LEON data we see only 1.85% increase. But the difference in numbers is explained in that the latter system will often find shorter expressions in the same span of text. Imagine an example where a dictionary contains "*11*", "*beta*", "*hydroxysteroid*", "*dehydrogenase*" and also "*11 beta-hydroxysteroid dehydrogenase*", but a document contains the text "*hydroxysteroid 11-beta dehydrogenase*". The LLS can properly detect the mention of "*hydroxysteroid 11-beta dehydrogenase*", but if variable order is not allowed four other expressions will be found, "*11*", "*beta*", "*hydroxysteroid*", and "*dehydrogenase*". While both approaches give eligible answers, LLS detection is more correct because the whole compound term is returned.

6. Conclusions and Future Work

We have presented use case requirements for leveraging semantic web technologies into a real world commercial IE system. The lexical layer — LEON, for a given ontology can be statically compiled into efficient FSM dictionaries. Furthermore to illustrate the potential power of LEON combined with a lightweight linguistic scanner we have targeted compound nominals as an initial extraction problem. The evaluations show that LEON metadata can be used to improve the detection of lexical entries in text, particularly

[11] http://www.pubmedcentral.nih.gov/

those which are comprised of multiple words and which can inflect, vary syntactically and/or contain intervening tokens.

LEON differs from other approaches involving deep analysis which tend to suffer from idiomaticity and overgeneration problems while the shallower "words with spaces" approach frequently employed in Information Extraction and industrial text analytics systems lacks flexibility and is prone to lexical proliferation. Nor is LEON for the purpose of vocabulary description enhancement only. Our approach proposes to actually utilise the lexical information to identify more complex linguistic phenomena (compound nominals) during the IE process while simultaneously maintaining performance and scalability.

The experiments described here do not use a state-of-the-art Semantic web Ontology, instead favouring to use an existing resource (MeSH) which provides an extensive vocabulary of multiword lexical expressions which allows us to test the viability of the LEON and LLS approach to the problem. This however, is in line with our original use case definitions whereby the approach needs to be compatible with existing legacy data and ontologies.

Our future plans are to expand this work by using this method to augment existing Semantic Web resources and improve detection of concepts from complex lexical forms in Semantic Web applications.

Acknowledgments

The work presented in this paper was supported (in part) by the European project NEPOMUK No. FP6-027705 and (in part) by the Líon project supported by Science Foundation Ireland under Grant No. SFI/02/CE1/I131.

7. References

Paul Buitelaar, Thierry Declerck, Anette Frank, Stefania Racioppa, Malte Kiesel, Michael Sintek, Ralf Engel, Massimo Romanelli, Daniel Sonntag, Berenike Loos, Vanessa Micelli, Robert Porzel, and Philipp Cimiano. 2006. LingInfo: Design and Applications of a Model for the Integration of Linguistic Information in Ontologies. In *Proceedings of the 5^{th} International Conference on Language Resources and Evaluation (LREC)*.

Gill Francopoulo, Monte George, Nicoletta Calzolari, Monica Monachini, Nuria Bel, Mandy Pet, and Claudia Soria. 2006. Lexical Markup Framework (LMF). In *Proceedings of the 5^{th} International Conference on Language Resources and Evaluation (LREC)*.

John Judge, Mikhail Sogrin, and Alexander Troussov. 2007. Galaxy: IBM Ontological Network Miner. In Sören Auer, Christian Bizer, Claudia Müller, and Anna V. Zhdanova, editors, *CSSW*, volume 113 of *LNI*, pages 157–160. GI.

Martin F. Porter. 1980. An algorithm for suffix stripping. *Program*, 14(3):130–137, July.

Ivan A. Sag, Timothy Baldwin, Francis Bond, Ann A. Copestake, and Dan Flickinger. 2002. Multiword Expressions: A Pain in the Neck for NLP. In *CICLing*, pages 1–15.