ACSIJ Advances in Computer Science: an International Journal, Vol. 5, Issue 4, No.22 , July 2016
ISSN : 2322-5157
www.ACSIJ.org

ACSIJ
WWW.ACSIJ.ORG

Multilingual extraction and editing of concept strings for the legal domain

Andrea Varga[1], and Andrew N. Edmonds[2]

[1] The Content Group,
Godalming, GU7 1JX, United Kingdom
varga.andy@gmail.com

[2] The Content Group,
Godalming, GU7 1JX, United Kingdom
andy@docandys.com

Abstract

Identifying semantic expressions (so-called *concept strings* (CSs)) in multilingual corpora is an important NLP task, as it allows web search engines to define and perform semantic queries over large collection of documents. Existing web search engines in the legal domain are mainly limited to keyword search, in which the query word is matched against the textual content of the documents. This paper presents a novel framework named the Concept Strings Framework that makes use of CSs for representing the content of the documents, and for allowing semantic search over them. These CSs can consist of individual knowledge base (KB) concepts (e.g. WordNet concepts) or combination of them. In addition, this paper presents an interactive web-based toolkit, called the Template Editor that enables the creation, editing and evaluation of CSs. Experiments on two publicly available legislation websites show satisfactory results.

Keywords: *Semantic Search; Concept Strings; Knowledge Base; WordNet*

1. Introduction

To operate efficiently, financial institutions need to regularly create and update documents, comply with the laws concerning the management of the documents, and keep track of the changes made in the legislation. For instance, the regulations can specify requirements that the documents must fulfill, such as the time period which a document must be retained for (*retention* requirement), the format in which the document must be kept (*format* requirement), the time when the document must be submitted to an agency (*submission* requirement) or completely destroyed (*destruction* requirement). This task is currently done by domain experts, who employ various state-of-the-art keyword based search engines (e.g. legislation.gov.uk for the UK, and http://www.ecfr.gov for the USA) to find appropriate requirement laws, and then review these laws manually. The output returned by such tools however depends on the keywords used by the experts, introducing the risk of missing out some important information, due to the various expressions used to describe relevant information.

In order to avoid this, this paper presents a novel framework called the Concept Strings Framework that makes use of multilingual knowledge bases (KBs) to understand the content of the documents and to formulate semantic queries over them. The created semantic queries rely on CSs [1] that consists of KB concepts and arbitrary combination of them. Additionally this framework exploits the hierarchical structure of KBs to obtain synonyms of these concepts.

For the representation of CSs a standard language was proposed. This paper further presents a web-based toolkit called the Template Editor, which using the proposed language permits the creation, editing and evaluation of CSs. The website can be used by multiple users simultaneously, providing an efficient way for the visual exploration of CSs.

The main contributions of this paper are as follows: a) a framework for extracting CSs and performing semantic search b) an interactive web-based toolkit for editing, visualizing and evaluating multilingual CSs, and c) a new language for encoding the concepts defined in CSs.

2. Related Work

The semantic processing of legal documents has gained much attention in recent years, due to the organization of conferences (e.g. JURIX[1], ICAIL[2]), legal tracks at the TREC[3] conference, and biannual LREC SPLet workshop[4] focusing on this topic. The main NLP tasks explored are information extraction, co-reference resolution, keyword extraction, document classification, dependency parsing, summarization, and search.

The majority of web search engines developed is keyword based. Most countries provide a web search engine for their legislation, for instance legislation.gov.uk in the UK, boe.es in Spain, and legifrance.gouv.fr in France. In addition, there has been some work on building ontologies for the legal domain in the SALEM (Semantic Annotation for LEgal Management) [2], [3], and the LOIS (Lexical Ontologies for Legal Information Sharing) [4] projects. In the SALEM project a small ontology was built to cover eight legislative provision types, including three major categories such as obligations, definitions and modifications. The goal of the project was to assign each law paragraph a given provision type, and to annotate parts of paragraphs with semantic roles identifying legal entities (e.g. actors, actions and properties) referred to in the provision. In the LOIS project a multilingual ontology was created by localizing WordNets to Italian, English, German, Czech, Portuguese and Dutch languages. The main purpose of this project was to allow cross-lingual retrieval across different national collection of laws. [5] used LOIS for query expansion, focusing on the terminology for the same legal jurisdiction. In this approach one or two words provided in the query are searched in the KB, and a weighting applied: a weight of 1 is given for synonyms of a term, a weight of 0.5 is given for subterms, and a weight of 0.25 is given for all meaningful terms mentioned in a definition.

In contrast to these approaches, we present a semantic search system that allows to formulate queries using arbitrary combination of KB concepts, having more than two concepts, and makes use of WordNet hierarchies to build such queries. In addition we enrich WordNet with specialized glossaries from the legal domain and domain specific concepts from legislation websites.

3. Multilingual Wordnet

Princeton Wordnet (WN) [6] is the original open source WordNet project developed for English, which has over 150,000 concepts. As a resource, a WN is a huge net, consisting of nouns, verbs, adjectives and adverbs, that are grouped into sets of synonyms (called synsets), each expressing a distinct concept. Synsets are also interlinked through various relations such as antonymy, meronymy (is part of), holonymy (opposite of meronymy), hypernymy (is kind of) and hyponymy (opposite of hypernymy). Over the past few decades various projects have been developed to build WNs for different languages [7]. One example is the Open Multilingual WordNet [8], an open source multilingual resource, which contains over 2 million senses, distributed over 150 languages, all linked to Princeton WN. Out of the available languages, the CSs Framework makes use of the Arabic [9], Spanish [10], French [11], German [12], Portuguese [13], and English languages.

The selected WNs were further analyzed and converted into language models, fully connected object oriented models.
In addition, further adjustments have been made to the English language model to tailor it to the legal domain. Similar developments will be done for the other languages in the future. Firstly, domain experts were asked to review the synonym hypernymy trees of each concept defined in the requirements (e.g. for the *retention* requirement: all synonyms of document types, time expressions) and exclude senses that are irrelevant in the legal context. Secondly, a list of legislation sources were researched to identify domain specific glossaries and legitimate sources that define domain specific concepts (e.g. the FCA glossary, UK Companies Act, and the https://www.gov.uk/ website). These concepts were then added to the English language model.

*1	Any Multiple At Least One	*0	Any Or None	!1	Any Single At Least One	!0	Any Single Or None
*1v	Multiple At Least One Verb	*0v	Multiple Verb	*1n	Multiple At Least One Noun	*0n	Multiple Noun
*1d	Multiple At Least One Adverb	*0d	Multiple Adverb	*1a	Multiple At Least One Adjective	*0a	Multiple Adjective
*1p	Multiple At Least One Pronoun	*0p	Multiple Pronoun	*1r	Multiple At Least One Proper Noun	*0r	Multiple Proper Noun
*1m	Multiple At Least One Modal Verb	*0m	Multiple Modal Verb				
!1v	Single Verb	!0v	Single Verb Or None	!1n	Single Noun	!0n	Single Noun Or None
!1d	Single Adverb	!0d	Single Adverb Or None	!1a	Single Adjective	!0a	Single Adjective Or None
!1p	Single Pronoun	!0p	Single Pronoun Or None	!1r	Single Proper Noun	!0r	Single Proper Noun Or None
!1m	Single Modal Verb	!0m	Single Modal Verb Or None				

Fig. 1 Wildcards used in templates.

4. The Concept Strings Framework

In this section we describe our approach for extracting and matching CSs in legal corpora, called the Concept Strings Framework, written in C#. A CS contains an array of WN concepts, each annotated with an array of possible meanings and its inferred part-of-speech (POS). The elements of CS can be combined with *wildcards*, such expressions are called *templates*. Wildcards are special selections of symbols that indicate that a match can be made with a certain number of words of a particular type (e.g. noun, verb, adjective, adverb, modal verb). A list of

[1] http://jurix.nl/
[2] http://sites.sandiego.edu/icail/
[3] http://trec-legal.umiacs.umd.edu/
[4] https://sites.google.com/site/splet2014workshop/

possible wildcards are shown in Figure 1. To create a short, concise template set, *template variables* (introduced by $) can be used. These elements allow the definition of a group of repeated concepts that can then be referenced inside the templates. For example we can define the following template (displayed in Figure 2) *keep *0 $documenttypes *0 $timeexpressions*, where "keep" stands for the verb keep, meaning "retain possession of", "*0" denotes any or no concept, and $documenttypes $timeexpressions are two template variables. $documenttypes can be any of the following concepts = {document, information, content}, while $timeexpressions can be = {day, month, year}. The main goal of this template is to match sentences such as "*the firm must keep documents for three years*". The proposed XML language for the *retention* template set looks as follows:

```
<templates>
<language>en</language>
<variables>
 <variable>
        <name>$documenttypes</name>
        <variableconcept>
          <source>content</source>
          <pos>
          <postype>noun</postype>
          <word>content</word>
          <concept><ref>6611268</ref>
          <description>what a
                communication ...
          </description>
          </concept>
          </pos>

        </variableconcept> ...
 <variable>...
</variables>
<template>
  <source>keep *0 $documenttypes *0 $timeexpressions
</source>
  <pos><postype>mathsymbol</postype>
  <word>$documenttypes</word>
  <concept><ref>20000257</ref>
  <description>...</description>
  </concept></pos>
</template>...
</templates>
```

The pre-processing of sentences is done using standard NLP pipeline, including tokenization, stemming, and part-of-speech (POS) tagging[1]. Given a pre-processed text, the pattern matching engine will match sentences where the words in the text are textually the same with the words defined in the template, and the POS of the words in the text agrees with the POS of words defined in the template. Furthermore, a match can be made for each pair of concepts from each word pair in the matching sequence, if the concepts share some similarities based on the hypernymy trees in WN. This is done by searching through the neighborhood of trees to examine if a concept is the parent of the other or if they have a near mutual ancestor. For example, for the above template, we will also match sentences where the word keep is replaced with its synonyms: held, maintain, store, file or retain (including all the three forms of the verbs).

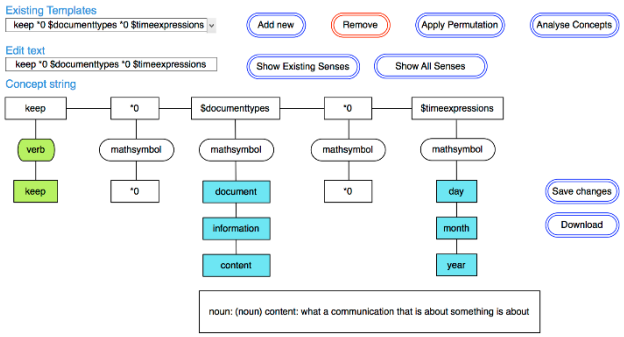

Fig. 2 Example retention template.

The CSs Framework has a data structure called a *Concept Tree* that efficiently holds sets of templates and permits them to be matched against text. Effectively the text is read in as a stream and the trees is passed over it. The Concept Tree matches the incoming text against directly and against the hypernymy tree, while considering multiple templates and the many virtual paths that wildcard handling creates.

5. The Template Editor

The Template Editor follows a simple yet powerful web-based architecture. The editor is written in C# using MVC design pattern and uses Microsoft SQL Server to store the data. It is tested on Internet Explorer, Firefox, and Chrome browsers.

5.1 Template Creation and Editing

Figure 2 shows a screen shoot of the user interface for template creation and editing for a given language (e.g. English). The interface is split into three main parts. The top left corner shows the existing templates defined in the current template set, and allows the editing of templates. The right part displays a list of available operations that can be done on a template set: e.g. creation of a new template (Add new button), removal of a template from a template set (Remove button), permutation of a template

[1] we use Stanford POS tagger

(Apply Permutation button), and analysis of concepts referenced in the template set (Analyse Concepts button - see section 6.2). When editing a template two options are available: Show All Senses, which generates a new set of concepts holding all possible meanings, and Show Existing Senses, which displays senses that were already used in previous templates, reducing the number of possible senses to be chosen for a given concept.

Generally several templates are created for a given requirement because one template specifies one sequence of concepts (one idea), and often an idea can be expressed using a different concept order. In order to help annotators create all possible combination of concept orders, the Apply Permutation button was developed, which automatically transforms an active voice template into its passive voice counterpart. For instance, for the template *keep *0 $documenttypes *0 $timeexpressions*, the *$documenttypes *0 keep *0 $timeexpressions*, (matching the sentence "documents must be kept for three years"), and the *$timeexpressions *0 $documenttypes *0 keep* (matching the sentence "for a period of three years documents must be kept") templates are created.

The center part visualizes the currently selected template using the concept string diagram. The diagram follows changes in the text of the template and interactively permits the user to determine the meaning of the concept. Hovering over a concept brings up a tooltip with the concept definition, and clicking on a concept will delete the concept that is not the one sought. After each edit, the templates can be saved by pressing the Save changes button, and the final template set downloaded, using the Download button.

5.2 Template Analysis and Reduction

One of the main benefits of the CSs Framework is that in order to find all the synonyms of a given concept in a text, it is enough to define only the most generic concept in the templates. This allows the template set to be short, and easily manageable. In the first corpus analysis phase, where annotators collect relevant concepts for a given template set, there is a need to analyze the collected concepts to examine which ones to keep. This functionality is provided in the Template Analysis tab, shown in Figure 3.

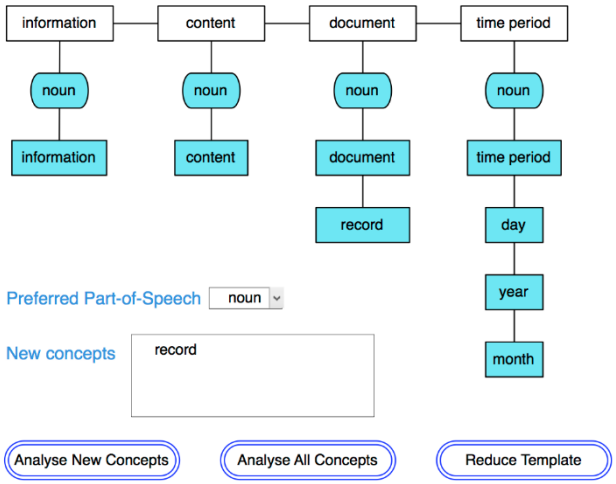

Fig. 3 Analyzing noun concepts used in the retention requirement (template set).

The interface allows the inspection of concept hierarchies based on POS. Two operations are available: analysis of concepts defined in a given template set (Analyse All Concepts button) and comparison of new concepts with the template concepts (New concepts textbox and Analyse New Concepts button). In both cases, the hierarchy of concepts is displayed for the selected concepts. Disconnected concepts, concepts that don't share any relationship with each other, are shown in a horizontal line one after another, while concepts that are children of another concept are displayed vertically just below the parent concept. For this reason it is enough to only keep in the template set concepts that are at level 1 on the diagram. In the example provided in Figure 3, we can see that record is a child of document, while day, year, and month are children of time period, and therefore only document and time period are kept in the template set.

In addition to performing the manual analysis, there is also the possibility to remove duplicate templates automatically by pressing the Reduce Template button. An algorithm was developed that deletes templates that have same sequence of concepts as an existing template, or a sequence where one or more of the concepts are children of the matching concepts in another template, and the rest are the same. For example, in the template set with sequences A, B; A1, B2; A2, B1; A3, B3, where A and B are two concepts with child concepts A1, A2, and A3, and B1, B2 and B3 respectively, the last 3 templates are deleted.

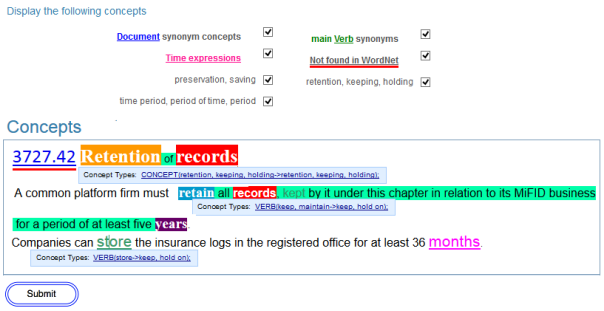

Fig. 4 Analyzing of template coverage for the retention requirement (template set).

5.3 Template Coverage

Having the template set created, the next step consists in evaluating it on real word text. For this purpose the Template Coverage editor (shown in Figure 4) was developed that provides an in-depth analysis of the results by highlighting the matched results, which we call *extract*, in colors (the whole matched phrase in green, while the matched concepts using various colors), and by inspecting the coverage of the template set. For the matched concepts a tooltip is also displayed, showing the parent concept for the concepts (e.g. store is a child of (->) keep). The main goal of the editor is to display all mentions of concepts of a given type: e.g. all document concepts (Document synonym concepts checkbox), all time expressions (Time expressions checkbox), and all verb concepts (main Verb synonyms checkbox - for the *retention* template the main verbs are keep and preserve). This allows to detect concepts that are not covered by the template set (not highlighted as a known concept type by the editor), and thus must be added to it (e.g. insurance log in the example provided in Figure 4). Furthermore, there is also the possibility to highlight words that are not found in WN (Not found in WordNet checkbox), that can be used to extend the language model for the language used by the template set.

6. Evaluation of Concept Strings

We used the Template Editor for an experimental analysis of CSs. The analysis had two major goals: to validate the effectiveness of CS extraction and to identify common error classes. In the evaluation we focused on finding all relevant information, favoring recall instead of precision.
We defined two qualitative categories for the evaluation of CSs: "Relevant" (REL) and "Irrelevant" (IRREL). We labelled a matched result as REL if it is semantically correct and applies to financial companies. A match is semantically correct if the matched concepts are semantically related (e.g. the document concept is the direct object of the verb keep; the time expression refers to the document concept to be retained). Correspondingly, we labelled a result as IRREL if it is semantically wrong or does not apply to financial companies.

Table 1: Legislation sources used in the experiments. #Res stands for number of extracts, #Sent for the average number of sentences per page.

Sources	#Res	#Sent	#REL	#IRREL
Comp. Act	35	123	14	21
FCA	106	289.01	66	40

Two different sources were used in the experiments: the UK Companies (Comp.) Act from 2006, where 35 results were found from 14 out of 1,695 pages, and the FCA Handbook with 106 results found from 66 out of 3,655 pages. The results were compared against manual annotations done by domain experts. In all cases we can see that the CSs Framework significantly reduces the number of pages and extracts to be reviewed, easing the tedious and costly task of reading thousands of pages. More importantly, the framework returned all relevant information previously found by the experts manually, resulting in 100% recall on both websites. In terms of precision, as shown in Table 1, we can observe that it performed better on handbooks, achieving 62.26% precision (76.74% F1), and it performed less well on the Companies Act, reaching 40% precision (57.14% F1).
The evaluation was performed by two domain experts, who identified four main error types: two of the cases relate to the correctness of concept synonyms (ErDoc, ErVerb), one relates to syntactical error (ErPOS), and another one encompasses semantic mistakes in the matched concept sequence (ErSem). The inter-annotator Kappa agreement between the annotators was 0.85. The average tagging speed was 8 minutes per extract.

Table 2: Distribution of error classes in the two sources analysed.

Errors	ErDoc	ErVerb	ErPos	ErSem
Comp. Act	11	11	5	18
FCA	19	34	7	39

The distribution of error classes is presented in Table 2. The most common error type found is ErSem, where the verb and the document concept are not semantically related. One typical example is when the results span across several sentences, such as in "*Explanatory **Notes** (1)Every public company must **hold** a general meeting as its annual general meeting in each period of 6 **months**, (2)* . . . ". In such cases an enumerator extractor needs to be employed that correctly identifies the sentence boundaries, and the CSs Framework must be constrained to only

consider concepts that are within a single sentence. Building an enumerator extractor is however a challenging task due to the various enumeration formats employed in legislation, and the irregular capitalization used inside the enumerations. Furthermore, to ensure that the document concept is the direct object of the verb (the notes are held), deep semantic analysis, a dependency parser will need to be applied. The second most common error type is ErVerb, where the meaning of the matched verb is not keep or preserve. For example, in the sentence "*The **report** must set out the steps the HRA has **taken** during the **year***", take is a wrong synonym of keep. This error can be corrected by applying a word sense disambiguation (WSD) system. Such system is aimed to be incorporated into CSs Framework in the future. Similarly, the ErDoc error occurs when the meaning of the matched noun concept is not document. For example, in the sentence "*In this **case**, the firm can **store** insurance logs for three **years***", case is a wrong synonym of document. This error can also be corrected by a WSD system. Common to both error types are the mistakes done by the POS tagger. For example in the sentence "*If, the firm has not been **trading** for three months in a business line, then it must use the **records** that are available to it and must also factor in reasonable forecasts, to make up a three **month** reference period.*", trading is a verb instead of noun (synonym of document), and records is a noun instead of verb (synonym of keep). In order to address this case, the POS tagger will need to be improved.

7. Conclusions

This paper presented the CSs Framework, a semantic search system in the legal domain that makes use of CSs to match sequences of text with the same meaning. The creation, editing and evaluation of CSs was enabled using an interactive web-based toolkit, the Template Editor. Experimental results demonstrated that our approach works well in finding relevant information, being able to return all examples previously found by domain experts by hand, reaching 100% recall. Future work will include the followings: a) implementation of an enumeration extractor that helps identify the sentence boundaries b) implementation of a WSD system that helps filtering out wrong results c) incorporation of a dependency parser into the CSs Framework for obtaining the direct object of verbs in a sentence, ensuring that the matched concepts are semantically related (e.g. the document concept is the direct object of the verb) , and d) extension of the evaluation to other requirement types and languages.

Acknowledgments

The authors wish to thank the domain experts for their help in evaluating the Concept Strings Framework.

References

[1] A. Edmonds. Using concept structures for efficient document comparison and location. In Proceedings of IEEE Symposium on Computational Intelligence and Data Mining, 2007.
[2] C. Soria, R. Bartolini, A. Lenci, S. Montemagni, and V. Pirrelli. Automatic extraction of semantics in law documents. In Proceedings of the V Legislative XML Workshop, 2007.
[3] R. Bartolini, A. Lenci, S. Montemagni, V. Pirrelli, and C. Soria. Automatic classification and analysis of provisions in Italian legal texts: a case study. In Proceedings of OTM Confederated International Conferences, 2004.
[4] L. Dini, W. Peters, D. Liebwald, E. Schweighofer, L. Mommers, and W. Voermans. Cross-lingual legal information retrieval using a WordNet architecture. In Proceedings of the 10th international conference on Artificial intelligence and law, 2005.
[5] E. Schweighofer, and A. Geist. Legal query expansion using ontologies and relevance feedback. In Proceedings of the 2nd Workshop on Legal Ontologies and Artificial Intelligence Techniques, 2007.
[6] G. A. Miller. Wordnet: A lexical database for english. Commun. ACM, 1995.
[7] F. Bond, and K. Paik. A survey of wordnets and their licenses. In Proceedings of the 6th Global WordNet Conference, 2012.
[8] F. Bond, and R. Foster. Linking and extending an open multilingual wordnet. In Proceedings of the ACL. Association for Computational Linguistics, 2013.
[9] W. Black, S. Elkateb, and P. Vossen. Introducing the arabic wordnet project. In Proceedings of the third International WordNet Conference, 2006.
[10] A. F. Montraveta, G. Vazquez, and C. Fellbaum. The spanish version of wordnet 3.0. In Text Resources and Lexical Knowledge, 2008.
[11] B. Sagot, and D. Fier. Building a free French wordnet from multilingual resources. In Ontolex, 2008.
[12] B. Hamp, and H. Feldweg. Germanet - a lexical-semantic net for german. In Proceedings of ACL workshop Automatic IE and Building of Lexical Semantic Resources for NLP Applications, 1997.
[13] V. dePaiva, and A. Rademaker. Revisiting a brazilian wordnet. In Proceedings of Global Wordnet Conference. Global Wordnet Association, 2012.

Andrea Varga received the BSc in computer science from the Babes-Bolyai University in 2007, the MSc degree in Intelligent Systems from the Babes-Bolyai University in 2008, and the PhD degree in text mining from the University Of Sheffield in 2015. She is currently a data scientist at The Content Group, United Kingdom, working on text mining. Her research focuses on natural language processing (text classification, topic classification, semantic search, social network analysis, and semantic web).
Andrew N. Edmonds received the PhD degree in artificial intelligence from the University of Bedfordshire in 1996. He is currently a data scientist at Dr Andy's IP Ltd. His research focuses on natural language processing (text classification, word sense disambiguation, and semantic search) and chaos theory.