ReadME generation from an OWL ontology describing NLP tools

Driss Sadoun, Satenik Mkhitaryan, Damien Nouvel, Mathieu Valette

ReadME Generation from an OWL Ontology Describing NLP Tools

Driss Sadoun, Satenik Mkhitaryan, Damien Nouvel, Mathieu Valette
ERTIM, INALCO, Paris, France
firstname.lastname@inalco.fr

Abstract

The paper deals with the generation of *ReadME* files from an ontology-based description of NLP tool. *ReadME* files are structured and organised according to properties defined in the ontology. One of the problem is being able to deal with multilingual generation of texts. To do so, we propose to map the ontology elements to multilingual knowledge defined in a SKOS ontology.

1 Introduction

A *ReadMe* file is a simple and short written document that is commonly distributed along with a computer software, forming part of its documentation. It is generally written by the developer and is supposed to contain basic and crucial information that the user reads before installing and running the software.

Existing NLP software may range from unstable prototypes to industrial applications. Many of them are developed by researchers, in the framework of temporary projects (training, PhD theses, funded projects). As their use is often restricted to their developers, they do not always meet *Information technology* (IT) requirements in terms of documentation and reusability. This is especially the case for under-resourced languages, which are often developed by researchers and released without standard documentation, or written fully or partly in the developer's native language.

Providing a clear *ReadMe* file is essential for effective software distribution and use: a confusing one could prevent the user from using the software. However, there is no well established guidelines or good practices for writing a *ReadMe*.

In this paper we propose an ontology-based approach for the generation of ordered and structured *ReadMe* files for NLP tools. The ontology defines a meta-data model built based on a joint study of NLP tool documentation practices and existing meta-data model for language resources (cf. section 2). Translation functions (TFs) for different languages (currently eight) are associated to ontology properties characterising NLP tools. These *TFs* are defined within the *Simple Knowledge Organization System* (*SKOS*) (cf. section 2.2). The ontology is filled via an on-line platform by NLP experts speaking different languages. Each expert describes the NLP tools processing the languages he speaks (cf. section 3). A *ReadMe* file is then generated in different languages for each tool described within the ontology (cf. section 3). Figure 1 depicts the whole process of multilingual *ReadMe* generation.

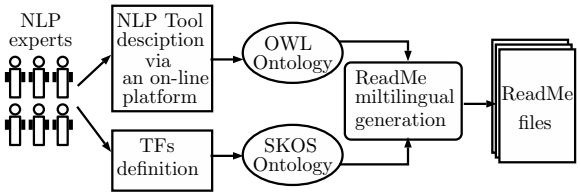

Figure 1: ReadMe generation process

2 NLP tools ontology

This work takes place in the framework of the project *MultiTal* which aims at making NLP tool descriptions available through an on-line platform, containing factual information and verbose descriptions that should ease installation and use of considered NLP tools. This project involves numerous NLP experts

Proceedings of the 2nd International Workshop on Natural Language Generation and the Semantic Web (WebNLG), pages 46–49,
Edinburgh, Scotland, September 6th, 2016.

in diverse languages, currently *Arabic*, *English*, *French*, *Hindi*, *Japanese*, *Mandarin Chinese*, *Russian*, *Ukrainian* and *Tibetan*. Our objective is to take advantage of the NLP experts knowledge both to retrieve NLP tools in their languages and to generate multilingual *ReadMe* files for the retrieved NLP tools. A first step to reach this goal is to propose a conceptual model whose elements are as much independent as possible from the language. Then, associate to each conceptual element, a lexicalisation for each targeted language.

2.1 Ontology conceptualisation

In order to conceptualise an ontology that structures and standardises the description of NLP tools we proceeded to a joint study of:

- Documentation for various NLP tools processing aforementioned languages that have been installed and closely tested;
- A large collection (around ten thousands) of structured *ReadMe* in the *Markdown* format, crawled from *GitHub* repositories;
- Meta-data models for Language Resources (LR) as the CMDI (Broeder et al., 2012) or META-SHARE meta-data model ontology (McCrae et al., 2015).

This study gave us guidelines to define bundles of properties sharing a similar semantic. For example, properties referring to the affiliation of the tool (as *hasAuthor*, *hasLaboratory* or *hasProjet*), to its installation or its usage.

We distinguish two levels of meta-data: 1) a *mandatory level* providing basic elements that constitute a *ReadMe* file and 2) a *non-mandatory level* that contains additional information as relations to other tools, fields or methods. These latter serve tools' indexation within the on-line platform.

Figure 2 details the major bundles of properties that we conceptualized to describe an NLP tool. The processed languages are defined within the bundle *Task*. Indeed, an NLP tool may have different tasks which may apply to different languages.

As our ambition is to propose pragmatic descriptions detailing the possible installation and execution procedures, we particularly focused on the decomposition of these procedures into atomic actions.

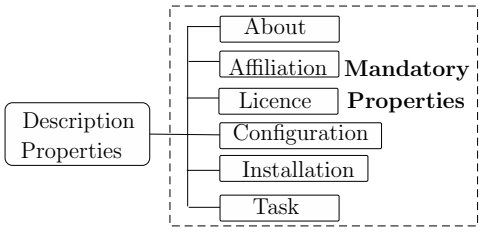

Figure 2: Bundles of properties representing ReadMe sections

2.2 Multilingual translation functions

Within the ontology, NLP tools are characterised by their properties. Values allocated to these properties are as much as possible independent of the language (date of creation and last update, developer or license names, operating system information, ...). Hence, what needs to be lexicalised is the semantic of each defined property. Each NLP expert associate to each property a translation functions (*TFs*) that formalise the lexical formulation of the property in the language he speaks. *TFs* are defined once for each language. The amount of work have not exceeded half a day per language to associate *TFs* to the around eighty properties of the ontology. In order to ensure a clean separation between the conceptual and the lexical layer, *TFs* are defined within a *SKOS* ontology. The *SKOS* ontology structure is automatically created from the OWL ontology. Thus, adding a new language essentially consists in adding within *SKOS TFs* in that particular language to each OWL property. Translation functions are of two kinds:

1. $P(V_1) \rightsquigarrow$ * V_1 **@lang*
2. $P(V_1,V_2) \rightsquigarrow$ * V_1 * V_2 * or * V_2 * V_1 * *@lang*

with P a property, * a set of words that can be empty, V_1, V_2 values of the property P and *@lang* an OWL language tag that determines the language in which the property is lexicalised. Below, two examples of *tranlation functions* for Japanese that have been associated to the properties *authorFirstName* and *download*.

- *authorFirstName*(V_1) $\rightsquigarrow$ 作成者名: V_1 @jp
- *download*(V_1,V_2) $\rightsquigarrow$ V_2 から V_1 をダウンロードする。@jp

3 Natural language generation of multilingual *ReadMe* files

In our framework, each NLP expert finds, installs and uses available NLP tools processing the language he speaks. Then, he describes every tool that runs correctly via an on-line platform connected to the ontology (cf. Figure 1). Elements of description do not only come from an existing *ReadMe* as if they exist, they are rarely exhaustive. Hence, experts also gather tool information from the web and during installing and testing each tool.

At this step, the *OWL* ontology is filled and the translated functions of each property are defined within the *SKOS* ontology. Our aim is to generate ordered and structured *ReadMe* files in different languages. To do so, we use *Natural language generation* (*NLG*) techniques adapted to the Semantic Web (also named *Ontology verbalisation*) (Staykova, 2014; Bouayad-Agha et al., 2014; Cojocaru and Trãuşan Matu, 2015; Keet and Khumalo, 2016). *NLG* can be divided in several tasks (Reiter and Dale, 2000; Staykova, 2014). Our approach currently includes: *content selection*, *document structuring*, *knowledge aggregation*, and *lexicalisation*. The use of more advanced tasks as *referring expression aggregation*, *linguistic realisation* and *structure realisation* is in our perspectives.

3.1 Ontology content selection and structuring

Unlike the majority of *ontology verbalisation* approaches, we do not intend to verbalise the whole content of the ontology. We simply verbalize properties and their values that characterise a pertinent information that have to appear in a *ReadMe* file. The concerned properties are those which belong to the *mandatory level* (cf. section 2.1).

The structure of *ReadMe* files is formalized within the ontology. First, *ReadMe* files are organised in sections based on bundles of properties defined in the ontology (cf. Figure 2). Within each section, the order of property is predefined. Both installation and execution procedures are decomposed to their atomic actions. These actions are automatically numbered according to their order of execution (cf. Figure 3). Different installation and execution procedures may exist according the operating system (Linux, Windows, ...), architecture (32bits, 64bites, 86bits, ...), language platform (JAVA 8, Python 3, ...) and so on. As well, execution procedures depend on tasks the NLP tool performs and the languages it processes. Thus, each procedure is distinguished and its information grouped under its heading. Moreover, execution procedures are also ordered as an NLP tool may have to perform tasks in a particular ordered sequence. This structuring is part of the ontology conceptualisation. It consists in defining property and sub-property relations and in associating a sequence number to each property that has to be lexicalised.

3.2 Ontology content aggregation and lexicalisation

Following the heuristics proposed in (Androutsopoulos et al., 2014) and (Cojocaru and Trãuşan Matu, 2015) to obtain concise text, OWL property values are aggregated when they characterise the same object. For example, if an *execution procedure* (ep_i) has two values for *operating system* (ex : Linux and Mac) then the two values are merged as the following below:

$hasOS(ep_i,\text{Linux}) \wedge hasOS(ep_i,\text{Mac})$
$\Rightarrow hasOS(ep_i,\text{Linux and Mac})$

The last step consists in *property lexicalisation*. While a number of approaches rely on ontology elements' names and labels (often in English) to infer a lexicalisation (Bontcheva, 2005; SUN and MELLISH, 2006; Williams et al., 2011), in our approach, the lexicalisation of properties depend only on their translation functions. During the ontology verbalisation, each targeted language is processed one after the other. The *TF* of encountered properties for the current language is retrieved and used to lexicalise the property. Property values are considered as variables of the *TFs*. They are not translated as we ensure that they are as much as possible independent of the language. Figure 3 gives an example of two installation procedures for the NLP tool *Jieba* that processes Chinese. In this example, actions are lexicalised in English. Furthermore, the lexicalised command lines appear in between brackets.

As a result of this generation, all *ReadMe* files have the same structure, organisation and, as much as possible, level of detail, especially regarding installation and execution procedures

which represent the key information for a tool usage. The resulted texts are simple which suits a *ReadMe*. However, it could be valuable to use more advanced NLG techniques as *referring expression aggregation*, *linguistic realisation* and *structure realisation* to produce more less simplified natural language texts.

Procedure name: *wget - ubuntu*

1- *download* jieba-0.38.zip *via wget* (wget https://pypi.python.org/packages/f6/86 /9e721cc52075a07b7d07eb12bcb5dde771d35332a 3dae1e14ae4290a197a/jieba-0.38.zip)
2- *unzip* jieba-0.38.zip (unzip jieba-0.38.zip)
3- *go to the directory* jieba-0.38 (cd jieba-0.38/)
4- *type the command:* python setup.py install

Procedure name: *pip - ubuntu*

1 - *type the command:* sudo pip install jieba

Figure 3: Two installation procedures of the NLP tool *Jieba* lexicalised in English.

4 Conclusion

We proposed an ontology-based approach for generating simple, structured and organised ReadMe files in different languages. *Readme* structuring and lexicalisation is guided by the ontology properties and their associated *translation functions* for the targeted languages. The generated *ReadMe*s are intended to be accessible via an on-line platform. This platform documents in several languages NLP tools processing different languages. In a near future, we plan to evaluate the complexity for end-users of different level of expertise to install and execute NLP tools using our generated *ReadMe* files. We also hope that, as a side-product, the proposed conceptualisation may provide a starting point to establish guidelines and best practices that NLP tool documentation often lacks, especially for under-resourced languages.

References

Ion Androutsopoulos, Gerasimos Lampouras, and Dimitrios Galanis. 2014. Generating natural language descriptions from OWL ontologies: the naturalowl system. *CoRR*, abs/1405.6164.

Kalina Bontcheva, 2005. *The Semantic Web: Research and Applications: Second European Semantic Web Conference, ESWC*, chapter Generating Tailored Textual Summaries from Ontologies, pages 531–545.

Nadjet Bouayad-Agha, Gerard Casamayor, and Leo Wanner. 2014. Natural language generation in the context of the semantic web. *Semantic Web*, 5(6):493–513.

Daan Broeder, Dieter Van Uytvanck, Maria Gavrilidou, Thorsten Trippel, and Menzo Windhouwer. 2012. Standardizing a component metadata infrastructure. In *LREC*, pages 1387–1390.

Dragoş Alexandru Cojocaru and Ştefan Trãuşan Matu. 2015. Text generation starting from an ontology. In *Proceedings of the Romanian National Human-Computer Interaction Conference - RoCHI*, pages 55–60.

C. Maria Keet and Langa Khumalo. 2016. Toward a knowledge-to-text controlled natural language of isizulu. *Language Resources and Evaluation*, pages 1–27.

John P. McCrae, Penny Labropoulou, Jorge Gracia, Marta Villegas, Víctor Rodríguez-Doncel, and Philipp Cimiano, 2015. *ESWC (Satellite Events)*, chapter One Ontology to Bind Them All: The META-SHARE OWL Ontology for the Interoperability of Linguistic Datasets on the Web, pages 271–282.

Ehud Reiter and Robert Dale. 2000. *Building Natural Language Generation Systems*. Cambridge University Press.

Kamenka Staykova. 2014. Natural language generation and semantic technologies. *Cybernetics and Information Technologies*, 14(2):3–23.

Xiantang SUN and Chris MELLISH. 2006. Domain independent sentence generation from rdf representations for the semantic web. In *Combined Workshop on Language-Enabled Educational Technology and Development and Evaluation of Robust Spoken Dialogue Systems, European Conference on AI*.

Sandra Williams, Allan Third, and Richard Power. 2011. Levels of organisation in ontology verbalisation. In *13th European Workshop on Natural Language Generation*, pages 158–163. Proceedings of the 13th ENLG.