Using OpenNLP for Realizing Natural Language Pre-Processing Pipeline as an Interface towards Semantic Web

[1]Aashima Saxena, [2]Sanjay Kumar Malik
[1]M.Tech (IT), [2]Associate Professor
USIC&T
Guru Gobind Singh Indraprsatha University, New Delhi, India

Abstract: The goal of web semantics is to convert unstructured data to meaningful representation, which may be largely benefited by exploiting many techniques like natural language processing (NLP). NLP plays a significant role in understanding, analysing and generating techniques like information extraction, parsing and semantic analysis. NLP models and Semantic Web support each other and have complementary roles in data management.
In this paper, first NLP and its techniques have been revisited and its role has been discussed. Second, NLP pre-processing pipeline has been explored and a stack for the same has been presented. Third, a few significant NLP toolkits and the challenges faced during implementation have been explored. Fourth, OpenNLP has been used to realize a few components of the NLP pre-processing pipeline like tokenizer, parsing and sentence splitter. Finally, the last section concludes the paper and future work has been stated.

IndexTerms **- Natural Language Processing, semantic web, openNLP, parsing, NLP pipeline.**

I. INTRODUCTION

In the present era of information search which is keyword based, a lot of irrelevant results explode on our screens. There is a need of making this search more concise, effective and efficient which needs the web to be smart and intelligent. Therefore, there is need of a Semantic Web. Semantic Web is based on the idea that the content on web may be expressed not only in natural language, but even in some language which may be understood and interpreted unambiguously. This enables software agents to locate, integrate, share, and reuse information more efficiently.

Semantic web is not just a tool to conduct human tasks, if properly designed, it can be used to revolutionize the human knowledge base as a whole [9].

As we know, computers require us to communicate in an unambiguous, well-constructed and a formal but straight forward language in the machine understandable format. Therefore, the key for moving to the next level is to extract the information efficiently, in a structured form and designate them in a way that it beholds its semantics thereby allowing open data linking at a global scale.

Information extraction consists of three main tasks [2], which are:

- Linguistic pre-processing
- Event extraction
- Named entity recognition

These tasks can be achieved by using the basic mechanisms, as simple as the *pre-processing pipeline* of Natural Language Processing, whose primary goal is to extract data in a structured format from the text-based documents. The data thus obtained, is linked to pre-existing data (residing in various databases) via technologies of semantic web thereby bridging the gap between unstructured and structured data. [8]. Also, this will aid in enhancing the human factor in our vision of a Semantic Web. We will then be able to visualise a web having software agents roaming from one webpage to another, having the ability to communicate with humans in natural language to deliver results by performing an efficient semantic search.

II. NLP AND ITS STACK

Natural language processing i.e. NLP is a branch of artificial intelligence which deals with understanding analysing and then generating the natural languages. It automatically processes the text written in natural language. With the aid of NLP, the unstructured data in the web can be segregated into a structured format of subject-predicate-object by defining rules in its pre-processing pipeline stages thereby enhancing the data on the web. The NLP techniques aids in enhancing the data on the web with semantics. One can also view NLP as a medium of communication between humans and the software agents roaming from one webpage to another. There is a desperate need of a high-level interface language which permits humans to annotate in a familiar notation which is unambiguous, offering precision equivalent to a formal language [6], where NLP pipeline acts a low-level interface.

The NLP pre-processing pipeline constitutes of various steps, namely: Tokenization, Sentence Splitting, P-O-S (Parts-of-Speech) Tagging, Morphological Analysis, Parsing and Chunking. It automatically adds information about entities and a URI is assigned to each.

Parsing, Information extraction, knowledge representation and semantic analysis are a few operations that are applied to the natural language query for extracting and retrieving keywords, noun-adjective-verb tokens and are thus stemmed to their appropriate roots [11]. A protocol stack of the above discussed pre-processing pipeline has been given [2].

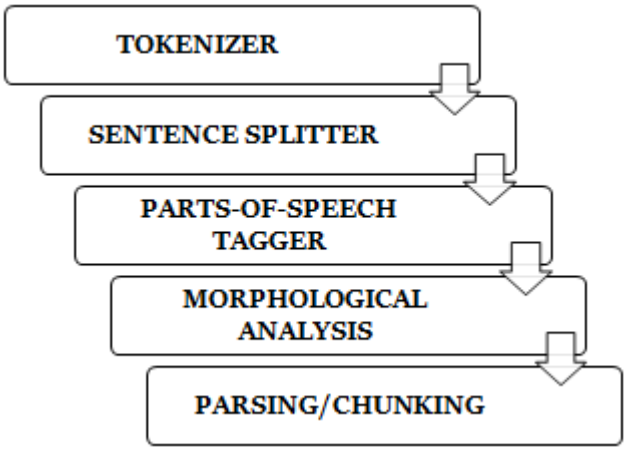

Fig.1. NLP pre-processing stack [2]

Fig. 1 describes the flow for processing a document to extract its semantics. Tokenizer is an essential component for it serves as an input to other components. It breaks the text up into tokens, where tokens are referred to words, punctuation marks and other special characters.

After tokenizing, Sentence splitter is used to find boundaries of a sentence by predicting for the ending of the sentence for each full stop. For tagging the tokens, to train which word is what and how it is or can be used we use the parts-of-speech tagger. Morphological analysis being an optional field, depends on the functionality of the project and deals with the internal structure of the word, starting from the root source to the end-use of the word. By parsing or chunking, we are able to analyse the overall grammatical structure of the natural language sentences. The output of this pipeline stack can be used for more complex applications to semantically augment information for further retrieval and analysis.

III. NLP TOOLS AND TOOLKITS

There are several variants of toolkits available which offer similar yet different set of tools. These differ in areas such as performance, processing speed, accuracy, modelling and many other factors. A specific tool/toolkit must be critically chosen to perform tasks. A brief description of some toolkits has been stated in Table 1 below.

Table 1. Brief Description of a few NLP Tools [2]

Tool	Description
OpenNLP	• Open source machine learning toolkit for language processing. • Runs on command line via JAVA API • Sentence splitter can be run independently here.
NLTK	• Open source toolkit. • Python based. • Has a command line interface. • Provides different variants, both rule-based and learning-based.
Stanford CoreNLP	• Open source annotation pipeline framework. • Uses JAVA API. • Used via command line; no other complex framework required. • High quality results. • Used where simple linguistic information is required.
GATE-ANNIE	• Knowledge-based; easily adaptable. • Java based. • Easy integration; platform independent.

	• Contains ready-made pipeline for information extraction. • Provides a few open source linguistic pre-processing components.

3.1 Challenges faced while using NLP with Generic Web Directories

To combine NLP techniques for semantic web, it is first required to solve the related challenges while converting or processing the information, namely: locating named entity, tagging parts-of-speech, parsing, and more. These arise primarily because of the distinct formats of the current web directories and the rules used by the NLP tools to implement the discussed pipeline. The general NLP Tools developed cannot be directly implemented in a web framework due to various reasons [3] as stated below in Table 2, which explains the difference in context of the rules used in both of them.

Table 2. Contextual Rule difference between Web Directory and NLP Tools [3]

Web Directory	**NLP Tools**
• Labels are short phrases.	• A take on full-fledged sentences.
• Very rare use of pronouns and verbs, most of the words are nouns and articles, adjectives, conjunctions and prepositions	• In full-fledged sentences occurrence of verbs and pronouns is very common.
• Knowledge-based, named entities occur densely.	• Named entities are sparse here.
• Every word begins with a capital letter leaving conjunctions and prepositions.	• In case of full-fledged sentences, every first word of the sentence and all proper names commences with capital letters.

IV. OPENNLP ILLUSTRATIONS

OpenNLP is a toolkit which consists of java libraries used for realizing NLP techniques. The most common tasks supported by openNLP are sentence segmentation, tokenisation, named entity recognition, parts-of-speech tagging, co-reference resolution and chunking. A few of these tasks have been explored and implemented.

4.1 Sentence Splitter

In openNLP, the sentence splitter can be implemented independent of the tokenizer. The 'SentenceDetectorME' class contains methods to split text into sentences by evaluating the 'end-of-sentence' characters to determine if they signal the end of a sentence.

In the below mentioned program, the end-of-characters are defined as a full stop (.), a question mark (?), and an exclamation mark (!); as soon as these are encountered the sentence is split into two or more sentences.

Program 1: Sentence Splitter

```
public class SentenceDetection_ME {
public static void main(String args[]){
String sentence = " Hii. How are you?
Lets try implementing NLP processing pipeline ." + "web semantics; techniques ;various NLP technologies." +
"hii?ii" ;
String simple = "[.?!;]";
String[] splitString =sentence.split(simple));
for (String string : splitString)
System.out.println(string);       }  }
```

Output:

```
Hii
How are you
Let's try implementing NLP processing pipeline
web semantics
techniques
various NLP technologies
```

hii
ii

4.2 Tokenizer

OpenNLP provides us with variants of classes to perform tokenization on our texts/documents, one of them being the 'WhitespaceTokenizer'. This class tokenizes our raw text by using whitespace as the parameter.

In the given program, the method 'tokenize ()' will convert the raw text into tokens as soon as it encounters a whitespace in between.

Program 2: Tokenizer

```
public class WhitespaceTokenizer{
public static void main(String args[]) {
String sentence = "Hi. NLP processing pipeline ." + "web semantics; techniques ;various NLP technologies." +
"hi?ii" ;
WhitespaceTokenizer tokenizer= WhitespaceTokenizer.INSTANCE;
String tokens[] = (tokenizer.tokenize(sentence));
for(String token : tokens)
System.out.println (token);  } }
```

Output:

```
Hi.
NLP
processing
pipeline
.
web
semantics;
techniques
;various
NLP
technologies.
hi?ii
```

4.3 Parts-of-Speech Tagger

For given sentence or multiple sentences, Parts-of-Speech Tagger analyses and tags tokens present into corresponding part of speech e.g. nouns, pronouns, verbs etc. It requires a base knowledge repository containing mapping of words to corresponding part of speech. OpenNLP library appends supported tag identifier at token separated by an underscore '_'. Table 3 tabulates common tag identifiers supported by OpenNLP library.

Table 3: OpenNLP library Tag Identifiers[1]

Part of Speech	Tag Identifier
Noun (Singular)	NN
Noun (Plural)	NNS
Proper Noun (Plural)	NNPS
Proper Noun (Singular)	NNP
Verb	VB
Pronoun	PRP
Adjective	JJ
Adverb	RB
Conjunction	CC
Interjection	UH

An illustration of original sentence and tagged sentence is shown below:

Sentence: I love books and flowers

TaggedSentence:
I**_PRP**love**_VBP**books**_NNS**and**_CC**flowers**_NNS**

V. CONCLUSION AND FUTURE SCOPE

This paper presents a pre-processing pipeline stack of NLP and showcases how its components can be implemented using the openNLP tool to extract the structured content. A brief discussion over various NLP toolkits available and the challenges faced during their implementation has also been presented. At present it is still being learnt how the challenges faced during the combination of combining the NLP techniques with the Semantic web vision at the early stages can be overcome. In the times to come, all the components of the discussed NLP pre-processing pipeline will be implemented to realize the merging of NLP and semantic web successfully in order to explore the role of NLP towards semantic web.

REFERENCES

[1] E. Kaufmann and A. Bernstein, How Useful Are Natural Language Interfaces to the Semantic Web for Casual End-Users?
[2] Isabelle Augenstein, 2014, Natural Language Processing for the Semantic Web.
[3] I. Zaihrayeu et al, From Web Directories to Ontologies: Natural Language Processing Challenges.
[4] Khan et al, 2016, Sentiment analysis and the complex natural language
[5] M. Henderson et al, 2017, Efficient Natural Language Response Suggestion for Smart Reply.
[6] N.E. Fuchs and R. Schwitter, Web-Annotations for Humans and Machines.
[7] Reagan et al, 2017, Sentiment analysis methods for understanding large-scale texts: a case for using continuum-scored words and word shift graphs.
[8] Sukhdev Singh et al, 2017, Natural Language Processing: State of The Art, Current Trends and Challenges.
[9] Tim Berners-Lee, 2001, Scientific American: Feature Article: The Semantic Web.
[10] Vietnam J, 2014, Automatic question generation for supporting argumentation.
[11] Vijayarajan et al, 2016, A generic framework for ontology-based information retrieval and image retrieval in web data.

Footnotes:
[1]https://web.archive.org/web/20130517134339/http://bulba.sdsu.edu/jeanette/thesis/PennTags.html