Natural Language Processing Using Natural Language ToolKit

Asmaa M. Hamandi Alia Karim AbdulHassan Hala Bahjat

Computer Science Dept., University of Technology, Baghdad, Iraq.

Abstract

Natural Language Processing (NLP) is an application field of Artificial Intelligence (AI). It refers to all processes that applied to text such as: tokenization, stemming, lemmatizing, and other operations. In this paper, NLP applied to English text in order to find Part Of Speech POS tagging that corresponding to the English sentence by using NLTK. The first part of any machine translation system is NLP. The source language text must be processed by applying NLP tools to produce good translation. In order to simplify the most widely used NLP operations, the built _ in toolkits has been produced. They provide modules and libraries that are easy to use and apply NLP operations. Using NLTK as an NLP tool recorded accuracy ratio of 70% , with the test cases have been selected randomly. The system has been implemented on PC with WINDOWS8.1 using Python programming language.

Keywords: NLP; NLTK; Machine Translation; NLP Toolkit; AI; POS tagging

معالجة اللغات الطبيعية باستخدام معدات اللغات الطبيعية

المستخلص

معالجة اللغات الطبيعية هو تطبيق من تطبيقات الذكاء الصناعي، وهو يشير الى جميع عمليات المعالجة التي يمكن تطبيقها على النصوص، مثل : تجزئة النص إلى مفردات، حذف الإضافات على الكلمات، إسترجاع مصدر الكلمة وعمليات أخرى. في هذا البحث يتم إجراء عملية معالجة النص واستخراج نوع الكلمة من حيث أجزاء الكلام. يعد الجزء الأول لأي نظام ترجمة آلية هو معالجة لغات طبيعية، حيث يتم تطبيق عمليات معالجة اللغات الطبيعية على اللغة المصدر للحصول على ترجمة صحيحة. لتبسيط عمليات معالجة اللغات الطبيعية ظهرت معدات معالجة اللغات الطبيعية لضم جميع عمليات المعالجة شائعة الاستخدام بحيث تكون مبنية داخلياً. هذه المعدات تضم مكتبات ونماذج توفر سهولة استخدام وتطبيق عمليات معالجة اللغات الطبيعية. إستخدام معدات اللغات الطبيعية سجل نسبة دقة 70% لحالات

اختبار تم اختيارها عشوائيا. تم تطبيق النظام باستخدام جهاز حاسوب شخصي يعمل بنظام تشغيل WINDOWS8.1 وباستخدام لغة البرمجة Python.

الكلمات المفتاحية: معالجة اللغات الطبيعية، معدات معالجة اللغات الطبيعية، الترجمة الآلية، الذكاء الصناعي، أجزاء الكلام

1. Introduction

NLP is a field of computer science, artificial intelligence, and computational linguistics concerned with the interactions between human languages and computers. As such, NLP is related to the area of human–computer interaction. The main challenge in NLP involve natural language understanding, that is, making computers to generate meaning from human or natural language input, and in the other hand involve natural language generation.

The term NLP contains a broad set of techniques for automated generation, analysis and manipulation of natural or human languages. Although most NLP techniques inherit largely from Linguistics and Artificial Intelligence, they are also influenced by relatively newer areas such as Machine Learning, Cognitive Science and Computational Statistics [1].

2. NLP Concepts and Terminology

General concepts and terminologies should be known when discussing NLP, they represent the basic vocabularies in text processing.

2.1. Sentences and Words

Words are the most fundamental building blocks of the language. Words consist of characters. Sentences are set of words ordered in an understandable and grammatical sequence.

2.2. Sentence Breaking

Sentences can be broken according to the sentence boundaries. Sentence boundaries are often marked by periods or other punctuation marks, but these same characters can serve other purposes (such as marking abbreviations) [2].

2.3. Word Segmentation

Separate a chunk of continuous text into separate words. For the English language, words are usually separated by spaces. However, for some other languages text segmentation is a significant task requiring knowledge of the morphology of words and vocabulary in the language.

2.4. Part-of-speech (POS) tag and POS tagging

A word can be classified into one or more of a set of lexical or part-of-speech categories such as Nouns, Verbs, Adverbs, Adjectives and other part of speech. A POS tag is a symbol representing such a lexical category - NN (Noun), JJ (Adjective), VB (Verb), and others. Table (1) contains some of these tags.

POS Tagging: is the task of automatically assign POS tags to each word in the sentences. It is a common language processing.

There are many methods applied to POS tagging. Most of the modern methods use some forms of machine learning. Such methods as: manually tagging, N-grams model, Transform-Based tagging, Decision Trees and Maximum Entropy [3].

The proposed system used the Penn Tree Bank POS tags that using the Maximum Entropy method.

2.5. Token and Tokenization

Token: it is a unit of a text, thus before applying real processing for the input text, it should be segmented into linguistic units such as words, numbers, and punctuation or alphanumeric. These units are known as tokens.

Tokenization: The process of segmentation of a text into basic units or tokens. For the English language, the existence of whitespace makes tokenization relatively easier. Tokenization based on whitespace is not enough for many applications because it merges punctuation together with words. Thus, it needs first to remove the punctuations.

2.6. Text Corpus:

A large and structured set of texts (usually electronically stored and processed). They are used to do hypothesis testing and statistical analysis, different word meanings what is called

ambiguity, frequency of words, checking occurrences or validating linguistic rules within a specific language type. Plural of corpus is corpora.

2.7. Normalization, Stemming and Lemmatization

As a preprocessing step, Normalization is a simple operation that applied on the input text. For the proposed system the normalization is to convert the letters of the input text into upper case letters.

Stemming: A further step of normalization is to strip off any affixes. Lemmatization: it is another further step of stemming is to make sure that the resulting form is a known word in a dictionary, this checking is additional process that makes the lemmatizes slower than the stemmers.

Table (1) Some of POS Tag Set

Tag	Description	Examples
CC	conjunction, coordinating	and but both either less for minus neither or nor therefore plus so yet times v. versus vs. whether
CD	numeral, cardinal	mid-1890 nine-forty fifty-one three-tenth million ten 0.5 forty-six 1987 twenty 79 zero two 75-degrees eighty-five IX '60s .025 fifteen 271,124 dozen quintillion DM2,000 ...
DT	determiner	another an all any both del either each half every la many neither much no some such that the them these those this
JJ	adjective or numeral, ordinal	Third pre-war ill-mannered regrettable oiled calamitous first separable battery-powered ectoplasmic participatory fourth still-to-be-named multi-disciplinary multilingual...
TO	"to" as infinitive marker or preposition	to
VB	verb, base form	ask assemble assess assign assume atone attention avoid bake balkanize bank begin behold believe bend benefit bevel beware bless boil bomb boost brace break bring broil brush build ...
VBD	verb, past tense	pleaded dipped swiped soaked regummed tidied convened registered halted cushioned exacted speculated snubbed figgered strode aimed adopted belied contemplated wore appreciated ...

3. Natural Language Processing Toolkits

There are many available toolkits for NLP, some of them are free to use, others are for commercial use and few of them are available only for researchers are provided by a center of researches. Table (2) contains a detailed list of the most widely used available toolkits.

3.1. Natural Language ToolKit NLTK

NLTK is a collection of modules and corpora, released under an open source license, that provides students and researchers in NLP with very useful tools. The most important advantage of using NLTK is that it is entirely self-contained. Not only providing wrappers and convenient functions that can be used as building blocks for common NLP tasks, it also provides raw and pre-processed versions of standard corpora used in NLP courses and literature [4].

NLTK is a leading platform for building programs in Python to work with human language data. It provides easy-to-use interfaces to over 50 corpora and lexical resources such as WordNet, along with a suite of text processing libraries for tokenization, classification, stemming, parsing, tagging and semantic reasoning, wrappers for industrial-strength NLP libraries, and an active discussion forum.

NLTK was originally created in 2001 as part of a computational linguistics course in the Department of Computer and Information Science at the University of Pennsylvania. After that it has been developed and expanded with the help of dozens of contributors. It has now been adopted in courses in several universities, and serves as the basis of many research projects [5].

NLTK is suitable for linguists, students, engineers, educators, researchers, and industry users alike. NLTK is available for Windows, , Linux and Mac OS X. Best of all, NLTK is a free, open source, community-driven project.

NLTK has been called a greatful tool for working in and teaching, computational linguistics using Python Programming Language , and an amazing library to play with natural language.

That was the reason for using NLTK as an NLP toolkit in the system in order to obtain the POS tags for the input text. In additional to that, Python programming language is very easy to use and easy to learn programming language. As one of the system contribution is to use NLTK as an NLP tool as a first stage of the converting English text into ASL system.

3.2. Open NLP

The Apache OpenNLP library is a toolkit based on machine learning for the processing of natural language text.It supports the most common NLP tasks, such as sentence segmentation, tokenization, named entity extraction, part-of-speech tagging, chunking, parsing, and coreference resolution. These tasks are usually required to provide more advanced text processing operations. OpenNLP also includes perceptron and maximum entropy based machine learning.

3.3. Stanford CoreNLP

Stanford CoreNLP provides a set of natural language analysis tools which can take raw text input and give the base forms of words, whether they are names of companies, people, or others, their parts of speech, normalize dates, times, and numeric quantities, and mark up the structure of sentences in terms of phrases and word dependencies, indicate which noun phrases refer to the same entities, indicate sentiment, or others. Stanford CoreNLP is an integrated framework. The goal of this kit is to make it very easy to apply a bunch of linguistic analysis tools to a piece of text.

Stanford CoreNLP is written in Java and licensed under the GNU General Public License (v3 or later; in general Stanford NLP code is GPL v2+, but CoreNLP uses several Apache-licensed libraries, and so the composite is v3+). Source is included. The full GPL, which allows many free uses, but not its use in proprietary software which is distributed to others. The download is 260 MB and requires Java 1.8.

Table (2): Most Widely Used NLP Toolkits

Name	Language	License	Creators	Website
DELPH-IN	C++, LISP	MIT, LGPL, ...	Deep Linguistic Processing with HPSG Initiative	http://www.delph-in.net/
General Architecture for Text Engineering (GATE)	Java	LGPL	GATE open source community	http://gate.ac.uk/
LinguaStream	Java	Free for research	University of Caen, France	http://www.linguastream.com/
MontyLingua	Java, Python	Free for research	MIT	http://web.media.mit.edu/~hugo/montylingua/
Natural Language Toolkit (NLTK)	Python	Apache 2.0		http://www.nltk.org
ClearNLP	Java	Apache License 2.0	Emory University	http://www.clearnlp.com

4. Applying NLTK as an NLP Toolkit

After initialization (reading the input English text) the first stage of the processing is the NLP stage, that takes the input English text as an input and produces the Part-Of-Speech (POS) tagging as an output for this stage according to several sub steps. Algorithm (1) represents the NLP stage main steps. In order to perform this task, an installation of the NLTK required, for getting a trust NLP tasks. The produced POS tags is according to the default tagger of NLTK that used Penn Treebank Tag Set that depends on max entropy approach. Table (1) illustrates the details of the tag set that is interdependent.

In Algorithm (1), the calling of the NLTK is the first step in order to be ready to use. Using the NLTK has been allowed in order to get the POS tagging using the tools that available in the kit. The input sentence must be processed by removing the punctuations (comma, semi-colon, colon, exclamation mark, question mark and other punctuations), then must be tokenized into words (according to the space place in the sentence), then the POS tagging must be obtained. All the details of the NLP operations that have been used for getting the POS tagging using the NLTK tools. Figure (1) shows the NLP main steps.

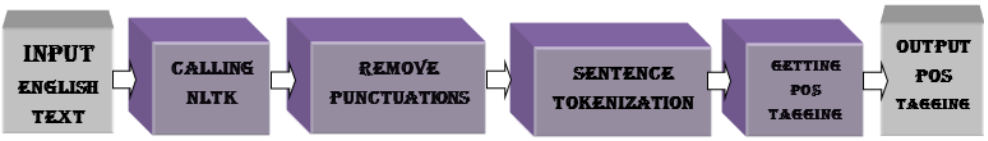

Figure (1): Block Diagram for the System

Algorithm (1): (NPL stage: getting the POS tagged from English text

Input: English text

Output: POS tagged

Calling NLTK toolkit;

Step 1: Get the input paragraph or sentence

Str= readfile (sentence);

Step 2: Remove the punctuations (str);

Step 3: Convert the input sentence into tokens

Words= tokenization(str1);

Step 4: Produce the POS tags

Tagged= POS (words);

Step 5: Return tagged

5. Experimental Example

Applying the algorithm (1) to an input English statement, and finding out the effect of each step in order to finally find POS tags for each word in the input sentence. The following steps describe the example:

a. input English Sentence: " The secret of life is not to do what you like, but to like what you do."

b. Calling NLTK toolkit, by importing it inside the program.

c. Applying the Steps

Step (1): Get the input paragraph or sentence

str = " The secret of life is not to do what you like, but to like what you do."

Step (2): Remove the punctuations from the input text

str1= " The secret of life is not to do what you like but to like what you do."

Step 3: Convert the input sentence into tokens

words= ['The', 'secret', 'of', life', 'is', 'not', 'to', 'do', 'what', 'you', 'like', 'but', 'to', 'like', 'what', 'you', 'do.']

Step 4: Produce the POS tags

tagged= [(The, DT), (secret, NN),(of, IN), (life, NN), (is, VBZ), (not, RB), (to, TO), (do, VB), (what, WP), (you, PRP), (like, VBP), (but, CC), (to, TO), (like, VB), (what, WP), (you, PRP), (do, VBP)]

Step 5: Return tagged

6. Results and Discussion

For evaluating the proposed system, a simple database has been built. This database contains more than one hundred English sentences with its corresponding POS tags, this database has been built for evaluation purpose only. For evaluating the accuracy of the system, a similarity measure has been needed. One of the most common get similarity measures is the Jaccard Similarity Index, which is based on a simple set operations union and intersection. The standard Jaccard Similarity equation is (Equ. 1), using this measure with additional processing steps to find the evaluation accuracy measure [6].

Jaccard Similarity Ratio = Intersection (set A, set B) / Union (set A, set B)
Equ(1)

Several processing steps have been applied to get the evaluation results. Algorithm (2) describes these steps. If the input text was a paragraph or a text file that contains several sentences, the algorithm has segmented it into individual sentences dealing with them one after another. The algorithm steps then would be applied to an individual sentence.

After reading the input sentence, a process of removing punctuations from the input sentence. After that, removing English stop words (words that are high frequency and low content like: but, is , or and if). Then, the normalization step by changing each letter in the sentence to its uppercase. Step (6) in the algorithm is to open the database to find out the most similar sentence in it to the input sentence, the most similar sentence is identified by computing the Jaccard similarity.

For each input sentence, Algorithm (2) have been applied to find the most similar English sentence saved in the database to the input English sentence. After processing steps and finding the POS tags for the input sentence, then comparing the output with the POS tags for the most similar English sentence. The comparison has been performed also using Jaccard similarity but in different way, without needing to Tokenization or deleting punctuations. The success case is to return the same index in the two times. That means the process has been performed correctly. It is worth mentioning to know that most of the failure cases belongs to that the test sentence does neither exists in the database nor similar to any of the existence sentence in the corpus. This problem could be challenged by enlarge the database over time.

The results of the system are differ according to number of words, characters, and the existence of the sentence itself or similar to it in the database or not. Selecting sentences for testing operation depends on whether the test sentences were the same sentences of database that has been build for evaluation purpose or nearing them, or the test sentences may be different of them. Different samples of test sentences has been selected giving different results. Table (3) and table (4) shows these different results.

Algorithm (2): Evaluation process

Processing the input sentence to find the index of the most similar sentence in the database.

Input: Input Sentence .

Output: The most similar sentence to the input one in the database.

Step (1): Get the input paragraph or sentence

str = readfile(sentence);

Step (2): Remove the punctuations from the input text

str1= remove-punctuations(str);

Step (3): Remove English Stop Words from the sentence

str2= remove-stop-words (str1);

Step (4): Convert the input sentence into tokens

Words= tokenization (str2);

Step (5): Normalization process by converting each token into Uppercase

Words-norm= Uppercase (words);

Step (6): Searching the database to compute the Jaccard similarity ratio and finding the most similar one to the input sentence.

Semilar-sent=get-sent(Jaccardmax(words-norm,database-sentences));

Step(6): Return Semilar-sent;

Table (3) contains different cases (English sentences) that are selected carefully, such that, most of the sentences are similar to (not exactly the same) the sentences that have been saved in the database. Different sentences have been used, long sentences and short ones with different number of words.

Table (3): Different Results for Different Selected Test Cases

No. of words	No. of characters	Max Jaccard for English Sentence	Max Jaccard for POS tags	Success flag
11	66	1.0	1.0	1
7	44	0.75	1.0	1
13	65	0.8571	1.0	1
11	53	0.6	1.0	1
10	54	1.0	1.0	1

Maximum Jaccard Similarity for English sentences sometimes couldn't reach to (1.0), while for the POS tags could always reach to the maximum value (1.0) that belongs to the generality of the

POS tags. The POS tags for a sentence is more general than the corresponding English. Such that, in English sentence a noun may be people's name, or place name or any different name; while in the POS tag for all these different names is 'NN'. The same thing is for different verbs.

The success flag is always (1) that refers to all the selected sample of sentences success and have a Jaccard similarity by finding its similar sentences (English and POS tags) in the database.

Table (4): Different Results for Different Random Test Cases

No. of words	No. of characters	Max Jaccard for English Sentence	Max Jaccard for POS tags	Success flag
12	62	0	1.0	0
8	39	0	0.8	0
7	33	0	0.6666	0
12	75	0.2857	0.75	1
9	41	0.25	1.0	1

Table (4) shows the results for five cases in which produces random samples of sentences from the web. Number of words is different from each case to another as well as number of characters. It is obvious that the results in table (4) is less efficient than results in table (4). That belongs to the randomly selected English sentences. Results in table (4) are still to be considered somewhat good because they referred to varietal the sentences that saved in the database. Such that there exists only more than one hundred sentences and however that, the success result (even if it was only one) is considered to be good mark that be given to both the system and the database.

According to results in both tables (3) and (4), each sentence is either computed as success of failure. The sentence computed as failure in one of two possibilities:

1. **Couldn't find the similar sentence in the database.**
2. **Found an English sentence that is similar to the failure sentence, but couldn't find the corresponding POS for it in parallel with the English one, or vice versa.**

According to the first possibility, the problem could be solved by enlarging the database to contain more different sentences in order to find similar sentences for all sentences as more as possible. The second possibility is referring to an error in finding the POS tags, or in the parallel English-POS database. It is worth mentioning, according to all cases in the two tables (3 and 4) all failure sentences are belong to the first possibility that all failure sentences couldn't be found in the database. That means, the system and the structure of the database have not recorded any error cases.

7. Conclusions and Future Works

The system is designed for produce NLP by finding the POS tags for each input sentence by using NLTK as an NLP toolkit.

7.1. Conclusion

According to the results of tables 3 and 4, conclude the following:

1. **Using the NLTK for NLP in order to obtain the POS tags of the sentence resulting into trusted system according to the NLTK toolkit.**
2. **From tables 3 and 4, the success flag is 1 in most cases that refers to the system and the structure of the database have not recorded any error cases.**
3. **The total using NLTK accuracy ratio for table 3 and 4 is 70%.**
4. **If an error has been found, that means couldn't find the similar sentence in the database.**
5. **The building of large database for evaluation purpose provides good evaluation result. Thus, the largest the database, the best results obtained.**

7.2. Future Works

As a future works suggestions

1. **Using this system as an important tool for Machine Translation system from English language into any other**

language, as the first stage of any translation system is to perform NLP for the source language.

2. **Developing the database by enlarging it with adding different sentences over time.**
3. **Using another toolkit such as CoreNLP or another one and compare the results.**

References

[1] Nitin Madnani, "Getting started on natural language processing with Python: an interview with Jennifer Lai", *Crossroads Crossroads,* vol. 13, no. 4, pp. 06-01, 2007.

[2] Daniel Jurafsky & James H. Martin.C., Speech and Language Processing: An introduction to natural language processing, computational linguistics, and speech recognition, 2007.

[3] Ahmed Hussein Aliwy January, *Arabic Morphosyntactic Raw Text Part of Speech Tagging System*, Warsaw University Faculty of Mathematics, Informatics and Mechanics, 2013.

[4] Edward Loper and Steven Bird. NLTK: The Natural Language Toolkit. In Proceedings of the ACL Workshop on Effective Tools and Methodologies for Teaching Natural Language Processing and Computational Linguistics, pages 62–69. Somerset, NJ: Association for Computational Linguistics, 2002.

[5] Steven Bird, Ewan Klein, and Edward Loper, Natural Language Processing with Python. USA: O'Reilly Media, 2009.

[6] William W. Cohen , Pradeep Ravikumar , Stephen E. Fienberg, "A comparison of string distance metrics for name-matching tasks ", *American Association for Artificial Intelligence*American Association for Artificial Intelligence, USA, 2003, pp. 73, 2003.

[7] Steven Bird, NLTK: The Natural Language Toolkit. Department of Computer Science and Software Engineering, University of Melbourne, Victoria3010. USA, 2006.