Linguakit

Developed by ProLNat@GE Group (http://gramatica.usc.es/pln/), CiTIUS, University of Santiago de Compostela, Galiza.

LinguaKit is a Natural Language Processing tool containing several NLP modules (constantly updated and improved):

- Dependency parser (DepPattern)
- PoS tagger
- NER (named entity recognition)
- NEC (named entity classification)
- Coreference resolution of named entities
- Sentiment analysis
- Multiword extraction
- Keyword extraction
- Relation extraction
- Language recognition
- Tokenizer
- Sentence segmentation
- Lemmatization
- Keyword in context
- Entity linking and semantic annotation
- Summarizer
- Verb conjugator
- Language checker (spelling, lexicon, grammar)

Demo

A web interface to LinguaKit is available at LinguaKit.com

Description

The command **linguakit** is able to process 4 languages: Portuguese, English, Spanish and Galician. Since February 2018, a new language has been added: historical galician-portuguese (**histgz**), by Xavier Canosa, which is still a prototype that will be improved. The following tools are available. Scroll down for additional documentation and usage examples.

- **Dependency parser** (parameter **dep**): Runs parsers. The parsers are implemented in PERL and stored in the **parsers** file. The parsers were compiled from formal grammars (more information). There are several parameters to control output: basic triplets (**-a**), triplets with morphological information (**-fa**), the same output as the input (**-c**) for correction purpose, and CoNLL format (**-conll**). These parameters are further explained in the section *Dependency Parser* below.
- **PoS tagger** (parameter **tagger**): Provides the PoS tagger *CitiusTools*. It returns one PoS tag and one lemma per token. This is also known as PoS tagging disambiguation. The module is provided with two submodules: **NER** (**-ner**) and **NEC** (**-nec**). The NEC module returns semantic tags for named entities: **NP0SP00** (Person), **NP00G00** (Location), **NP00O00** (Organization), **NP00V00** (Miscelaneous).
- **COREF** (parameter **coref**) labels the different named entities of the text (identified by the **NER** and **NEC**) with a numeric id which represents the discourse entity they refer to (e.g., "Bob Marley NP00SP0 (1)", "Jimi Hendrix NP00SP0 (2)", "Marley NP00SP0 (1)", "Hendrix NP00SP0 (2)", etc.). **COREF** allows the **-crnec** option (experimental) which relabels some named entities based on the results of the coreference analysis. Please note that the **COREF** modules may slow down the execution of the system when analyzing large texts. Also, remember that **COREF** is performed document by document, so it is not recommended to run it in a large corpus containing several documents.
- **Multiword extraction** (parameter **mwe**): Extracts multiwords from PoS tagged text. There are several optional

parameters, each one being a specific lexical association measure for ranking the candidate terms: chi square (-chi, default), loglikelihood (-log), mutual information (-mi), symmetrical conditional probability (-scp), simple co-occurrences (-cooc).

- Keyword extraction (parameter key): Extracts keywords (lexemes and proper names) from PoS tagged text and ranked them using a reference corpus and chisquare.
- Sentiment analysis (parameter sent): Returns POSITIVE, NONE (neutral) or NEGATIVE, using a polarity lexicon and a classifier trained from annotated tweets. Given an input text, this module returns a polarity value for each paragraph, namely it returns three columns for each paragraph: the text in the first column, the polarity (pos, neg, none) in the second column, and the polarity score (from 0 to 1) in the third column. The last line of the output returns the overall score computed as the average of all paragraphs.
- Relation extraction (parameter rel): Returns triples SUBJECT - RELATION - OBJECT using methods based on Open Information Extraction.
- Language recognition (parameter recog): Returns the language of the input text:en, es, pt, gl, gz (agal galician variety), fr, eu, ca, bn (bengali), ur (urdu), hi (hindi), ta (tamil). This module is also used by other modules to recognize the language before processing (only for the supported languages: pt, en, es, gl).
- Tokenizer (parameter tok): Returns a tokenized text. Parameter-split splits word contractions and verb clitics. Parameter -sort ranks tokens by frequency.
- Sentence segmentation (parameter seg): Returns a sentence per line. Sentence segmentation is the problem of dividing a string of written language into its component sentences.
- Lemmatization (parameter lem): Returns all the lemmas of each token and all the morphological information (PoS tags) associated to each lemma. This is the process running before PoS tagging disambiguation.
- Keyword in context (parameter kwic): Returns a target word in context (window: 10 tokens). Option-tokens returns tokens as context. This module requires the keyword to be searched as an additional argument.
- Entity linking (parameter link): Returns a list of terms which represent Wikipedia entities. Besides, the input text is annotated with those terms and their links to Wikipedia. Requires Internet conection since it runs via Web API service. The output can be in two formats: json (default) and xml.
- Summarizer (parameter sum): Returns an abstract of the input text. You can choose the percentage of the text to be summarized by using as option a number from 1 to 100. The code was developed by Fernando Blanco Dosil when it was working in Cilenis Language Technology.
- Conjugator (parameter conj): Returns the verb inflection if you enter the infinitive form. Pay attention that the input is not a file but a string, the infinitive verb, and the module should be used like this: ./linguakit conj pt "fazer" -s -pb. The module is working for three languages: Galician, Spanish and Portuguese. In the case of portuguese verbs, you can choose among 4 language varieties: european portuguese after the spelling agreement (-pe), brasilian portuguese after the spelling agreement (-pb), european portuguese before the spelling agreement (-pen), brasilian portuguese before the spelling agreement (-pbn). The output is in json format. This module requires Internet conection since it runs using a Web API.
- Language checker (parameter aval): Returns the language errors found in the input sentence. Several types of errors are considered: spelling, lexical, and grammatical issues. Suggestions of correction are provided as well as a linguistic explanation for each specific type of error. Requires Internet conection since it runs via Web API service. The output can be in two formats: json (default) and xml. By now, it is only available for Galician language (gl).

Requirements

Depending on your GNU/Linux version or distribution, you may need to install some CPAN Perl modules: **Getopt::ArgParse* Perl module. * *LWP::UserAgent* Perl module. * *HTTP::Request::Common* Perl module. * *Storable* Perl module.

To install them, you may use the -MCPAN interface at the command line:

```
sudo perl -MCPAN -e 'install (Getopt::ArgParse)'
sudo perl -MCPAN -e 'install (LWP::UserAgent)'
sudo perl -MCPAN -e 'install (HTTP::Request::Common)'
sudo perl -MCPAN -e 'install (Storable)'
```

Installation

Using Git

```
git clone https://github.com/citiususc/Linguakit.git
```

ZIP Download

Download Linguakit-master.zip and then:

```
unzip Linguakit-master.zip
```

As all modules are been updated regularly, you'd better use`git` to install and update the system.

A New more efficient version of LinguaKit was released in 22 June 2017 by César Piñeiro (also for Windows`linguakit.bat` command).

Usage (Linux)

Run `./linguakit --help` to see the modules:

```
./linguakit <module> <lang> <input> [options]
cat <input> |./linguakit  <module> <lang>  [options]

    module = dep, tagger, mwe, recog, sent, rel, tok, seg, kwic, link, sum, conj
    language = gl, es, en, pt, histgz
    input = path of the input (by default a txt file or gz/zip)

    'dep'     dependency syntactic analysis
    'tagger'  part-of-speech tagging
    'mwe'     multiword extraction
    'key'     keyword extraction
    'recog'   language recognition
    'sent'    sentiment analysis
    'rel'     relation extraction
    'tok'     tokenizer
    'seg'     sentence segmentation
    'lem'     lemmatization
    'kwic'    keyword in context (concordances)
    'link'    entity linking and semantic annotation
    'sum'     text summarizer
    'conj'    verb conjugator (the input is just a verb)
    'coref'   named entity coreference solver
    'aval'    language checker: avalingua
```

Run `./linguakit <module> --help` to see the options of a module. These are the available command-line options:

```
-a       'dep' option: simple dependency analysis (by default syntactic output)
-fa      'dep' option: full dependency analysis
-c       'dep' option: tagged text with syntactic information (for correction rules)
-conll   'dep' option: CoNLL output style

-noner   'tagger' option: no NER or NEC is processed (by default PoS tagger output)
-ner     'tagger' option: PoS tagger with Named Entity Recognition - NER (only with 'tagger' module)
-nec     'tagger' option: PoS tagger with Named Entity Classification - NEC (only with 'tagger' module)

-crnec   'coref' option: NEC correction with NE Coreference Resolution

-chi     'mwe' option: chi-square co-occurrence measure (by default)
-log     'mwe' option: loglikelihood
-scp     'mwe' option: symmetrical conditional probability
-mi      'mwe' option: mutual information
-cooc    'mwe' option: co-occurrence counting

-split   'tok' option: tokenization with splitting
-sort    'tok' option: tokenization with tokens sorted by frequency

-tokens  'kwic' option: contexts are tokens
        *The kwic option is mandatory and also requires another argument: the keyword to be searched

-json    'link' option: json output format of entity linking (by default)
-xml     'link' option: xml output format of entity linking

-p 1-99 'sum' option: percentage of the input text that will be summarized (by default 10%)

-pe      'conj' option: the verb conjugator uses European Portuguese (by default)
-pb      'conj' option: the verb conjugator uses Brasilian Portuguese
-pen     'conj' option: the verb conjugator uses European Portuguese before the spell agreement
-pbn     'conj' option: the verb conjugator uses Brasilian Portuguese before the spell agreement
-json    'aval' option: json output format of language spelling (by default)
-xml     'aval' option: xml output format of language spelling

-s       'sent', 'recog' and 'conj' option: if <input> is a string and not a file
```

Usage in Windows

The same syntax with linguakit.bat command. You must install Perl and insert the path for the interperter.

Examples

Return a dependency-based analysis in CoNLL format:

```
./linguakit dep pt test/pt.txt -conll
```

Return the PoS tags with NEC information for named entities (in historical galician-portuguese - histgz):

```
./linguakit tagger histgz test/histgz.txt -nec
```

Return the PoS tags with NEC information for named entities and Coreference Resolution:

```
./linguakit coref en test/en.txt
```

Return a sentiment value:

```
./linguakit sent en "I don't like the film" -s
echo  "I don't like the film" | ./linguakit sent en
```

Make multiword extraction ranked with chi-square:

```
./linguakit mwe pt test/pt.txt -chi
```

Generate the context in tokens of the keyword ***presidente*** (concordances or keyword in context).

```
./linguakit kwic pt test/pt.txt -tokens "presidente"
```

Return triples (relations):

```
./linguakit  rel en test/en.txt
```

Return an abstract or summary of the input text (50%):

```
./linguakit sum en test/en.txt -p 50
```

Return the european portuguese inflection of the input verb:

```
./linguakit conj pt "fazer" -s -pe
echo "fazer" | ./linguakit pt conj -pe
```

Input file

The input must be in plain text format, and encoded in UTF8.

Lexicons

Lexicons (electronic dictionaries) are in `tagger/*/lexicon/dicc.src` files. If you modify them, then you should recompile them by running:

```
./lexicon_compiler.sh
```

Dependency parser

Output format

Parameter `-a` gives as output the basic dependency-based analysis. Each analysed sentence consists of two elements:

1. A line containing the POS tagged lemmas of the sentence. This line begins with the tag `SENT`. The set of tags used here are listed in file `TagSet.txt`. All lemmas are identified by means of a position number from 1 to N, where N is the size of the sentence.
2. All dependency triplets identified by the grammar. A triplet consists of: `(relation;head_lemma;dependent_lemma)`

For instance, the sentence "*I am a man.*" generates the following output using the parameter `-a`:

```
SENT::<I_PRO_0_<number:0|lemma:I|possessor:0|case:0|genre:0|person:0|politeness:0|type:P|token:I|>
am_VERB_1_<number:0|mode:0|lemma:be|genre:0|tense:0|person:0|type:S|token:am|>
a_DT_2_<number:0|lemma:a|possessor:0|genre:0|person:0|type:0|token:a|>
man_NOUN_3_<number:S|lemma:man|genre:0|person:3|type:C|token:man|> ._SENT>
(Lobj;be_VERBF_1;I_PN_0)
(Spec;man_NOM_3;a_DT_2)
(Robj;be_VERBF_1;man_NOM_3)
```

Parameter `-fa` gives rise to a full represention of the depedency-based analysis. Each triplet is associated with two pieces of information: morpho-syntactic features of both the head and the dependent.

Parameter -c generates as output with the same format as the input (a tagged text) but with some corrections proposed by the grammar. This option is useful to identify and correct regular errors of PoS taggers using grammatical rules.

Parameter -conll generates an output with the format defined by CoNLL-X, inspired by Lin (1998). This format was adopted by the evaluation tasks defined in CoNLL.

More information »

PoS tagger

The EAGLES convention is being followed.

More Information »

Coreference Resolution

More information »

Sentiment analysis

The input can be either a file (by default) or a string (option -e). The output is POSITIVE, NONE, OR NEGATIVE, and a score between 0 and 1. The classifier was trained with tweets, so the input should be just one sentence or a small paragraph.

More Information »

Multiword Extraction

More Information »

Language identification

More Information »

Entity linking

This module reads the input text and returns three types of terms and concepts:

- MAIN_TERM is a term occurring in the text that is linked to a Wikipedia concept.
- NEW_TERM is a Wikipedia concept which does not occur in the text but is semantically related to some of the main terms.
- CATEGORY is a Wikipedia category used to categorize and classify the text.

Also returns an annotated version of the input text, namely the text is annotated with tags identifying the main terms. Each MAIN_TERM is enriched with a link to its corresponding concept in Wikipedia.

For more information, you can look up for our paper:

> Gamallo, Pablo and Marcos Garcia (2016) "Entity Linking with Distributional Semantics", PROPOR 2016, LNAI 9727.

Additional Documentation and Bibliography

More information on the modules can be found in papers stored in the docs directory.

References

The toolkit

> Gamallo, Pablo, Marcos Garcia, César Piñeiro, Rodrigo Martínez-Castaño and Juan C. Pichel (2018). LinguaKit: a Big Data-based multilingual tool for linguistic analysis and information extraction. In Proceedings of The Second International Workshop on Advances in Natural Language Processing (ANLP 2018) co-located at SNAMS-2018, pp. 239-244. http://dx.doi.org/10.1109%2FSNAMS.2018.8554689.
>
> Gamallo P. , Garcia M. (2017) LinguaKit: uma ferramenta multilingue para a análise linguística e a extração de informação, Linguamática , 9(1).

Dependency analysis

Gamallo P. , González I. (2011) A Grammatical Formalism Based on Patterns of Part-of-Speech Tags , International Journal of Corpus Linguistics , 16(1), 45-71. ISNN:1384-6655

Gamallo, P. 2015. Dependency Parsing with Compression Rules, The 14th International Conference on Parsing Technologies (IWPT-2015) p. 107-117, Bilbao. ISBN 978-1-941643-98-3

Gamallo, P., González, I. 2012. DepPattern: A Multilingual Dependency Parser, Demo Session of the International Conference on Computational Processing of the Portuguese Language (PROPOR 2012) , April 17-20, Coimbra, Portugal.

PoS tagging, NEC and Coreference Resolution

Garcia, M. and Gamallo, P. 2015. Yet another suite of multilingual NLP tools, Symposium on Languages, Applications and Technologies (SLATE 2015) p. 81-90. ISBN 978-84-606-8762-7.

Garcia, M., 2016. Incorporating Lexico-semantic Heuristics into Coreference Resolution Sieves for Named Entity Recognition at Document-level. In Proceedings of the 10th edition of the Language Resources and Evaluation Conference (LREC 2016), Portorož: 3357-3361.

Abuín, José Manuel, Juan Carlos Pichel, Tomás Fernández Pena, Pablo Gamallo e Marcos Garcia (2014). Perldoop: Efficient Execution of Perl Scripts on Hadoop Clusters, IEEE International Conference on Big Data (IEEE Big Data 2014).

Garcia, M. and Gamallo, P. 2014. An Entity-Centric Coreference Resolution System for Person Entities with Rich Linguistic Information. In Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, Dublin: 741-752.

Gamallo P., Garcia, M. (2011) A Resource-Based Method for Named Entity Extraction and Classification , Lecture Notes in Computer Science, vol. 7026 , Springer-Verlag, 610-623. ISNN: 0302-9743

Sentiment analysis

Gamallo, P. and Garcia, M. 2014. Citius: A Naive-Bayes Strategy for Sentiment Analysis on English Tweets, In Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014), Dublin: 171-175.

Gamallo, P., Garcia, M. and Fernández-Lanza, S. (2013). TASS: A Naive-Bayes strategy for sentiment analysis on Spanish tweets, Proceedings of XXIX Congreso de la Sociedad Española de Procesamiento de lenguaje natural. Workshop on Sentiment Analysis at SEPLN (TASS2013), Madrid. pp. 126-132. ISBN: 978-84-695-8349-4. (FIRST system in the task of polarity detection at the entity level)

Multiword extraction

Barcala M., E. Domínguez-Noya, P. Gamallo, M.López, E. Moscoso, G. Rojo, P. Santalla, S. Sotelo. (2007) A Corpus and Lexical Resources for Multi-word Terminology Extraction in the Field of Economy, 3rd Language & Technology Conference(LeTC'2007), Poznan, Poland (355-359).

Relation extraction

Gamallo, P. and Marcos Garcia (2015). Multilingual Open Information Extraction, Lecture Notes in Computer Science, 9273, Berlin: Springer-Verlag: 711-722. ISNN: 0302-9743.

Gamallo, P. 2014. An Overview of Open Information Extraction, In Proceedings of the Third Symposium on Languages, Applications and Technologies (SLATE-2014), Bragança, Portugal: 13-16.

Gamallo, P., Garcia, M. Fernández-Lanza, S. 2012. Dependency-Based Open Information Extraction, In Joint Workshop on Unsupervised and Semi-Supervised Learning in NLP (ROBUS-UNSUP 2012), at the 13th Conference of the European Chapter of the Association for Computational Linguistics (EACL 2012). Avignon.

Entity linking

Gamallo, Pablo and Marcos Garcia (2016). "Entity Linking with Distributional Semantics", PROPOR 2016, Lecture

Notes in Computer Science, 9727. Berlin: Springer-Verlag, ISNN: 0302-9743.

Language checker (Avalingua)

Gamallo, P., Garcia, M., del Río, I., González 2015. Avalingua: Natural Language Processing for Automatic Error Detection, In: Marcus Callies, Sandra Götz (Eds.), Learner Corpora in Language Testing and Assessment, John Benjamins Publishing Company, pp. 35-58. ISBN: 978-90-272-0378-6.

INFORMACIÓN

Investigadores
Pablo Gamallo Otero

Licenza

DESCARGAR

- Repositorio Gitlab
- Descargar de Gitlab
- Repositorio Github

PUBLICACIÓNS

Multilingual Open Information Extraction
17th Portuguese Conference on Artificial Intelligence, EPIA 2015, Coimbra, Portugal, September 8-11, 2015. Proceedings, 2015

PROXECTOS DE INVESTIGACIÓN

TELEPARES: Tecnologías de la lengua para análisis de opiniones en redes sociales

DEMOSTRADORES

Linguakit