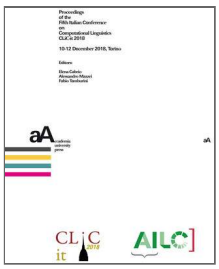
View metadata, citation and similar papers at core.ac.uk

brought to you by CORE

provided by OpenEdition



Elena Cabrio, Alessandro Mazzei and Fabio Tamburini (dir.)

Proceedings of the Fifth Italian Conference on Computational Linguistics CLiC-it 2018
10-12 December 2018, Torino

Accademia University Press

CoreNLP-it: A UD pipeline for Italian based on Stanford CoreNLP

Alessandro Bondielli, Lucia C. Passaro and Alessandro Lenci

DOI: 10.4000/books.aaccademia.3094
Publisher: Accademia University Press
Place of publication: Torino
Year of publication: 2018
Published on OpenEdition Books: 8 April 2019
Serie: Collana dell'Associazione Italiana di Linguistica Computazionale
Electronic ISBN: 9788831978682

http://books.openedition.org

Electronic reference
BONDIELLI, Alessandro ; PASSARO, Lucia C. ; and LENCI, Alessandro. *CoreNLP-it: A UD pipeline for Italian based on Stanford CoreNLP* In: *Proceedings of the Fifth Italian Conference on Computational Linguistics CLiC-it 2018: 10-12 December 2018, Torino* [online]. Torino: Accademia University Press, 2018 (generated 19 avril 2019). Available on the Internet: <http://books.openedition.org/aaccademia/3094>. ISBN: 9788831978682. DOI: 10.4000/books.aaccademia.3094.

CoreNLP-it: A UD pipeline for Italian based on Stanford CoreNLP

Alessandro Bondielli[1], **Lucia C. Passaro**[2] and **Alessandro Lenci**[2]
[1] Dipartimento di Ingegneria dell'Informazione (DINFO), Università degli studi di Firenze
[2] CoLing Lab, Dipartimento di Filologia, Letteratura e Linguistica (FiLeLi), Università di Pisa
alessandro.bondielli@unifi.it
lucia.passaro@fileli.unipi.it
alessandro.lenci@unipi.it

Abstract

English. This paper describes a collection of modules for Italian language processing based on CoreNLP and Universal Dependencies (UD). The software will be freely available for download under the GNU General Public License (GNU GPL). Given the flexibility of the framework, it is easily adaptable to new languages provided with an UD Treebank.

Italiano. *Questo lavoro descrive un insieme di strumenti di analisi linguistica per l'Italiano basati su CoreNLP e Universal Dependencies (UD). Il software sarà liberamente scaricabile sotto licenza GNU General Public License (GNU GPL). Data la sua flessibilità, il framework è facilmente adattabile ad altre lingue con una Treebank UD.*

1 Introduction

The fast-growing research field of Text Mining and Natural Language Processing (NLP) has shown important advancements in recent years. NLP tools that provide basic linguistic annotation of raw texts are a crucial building block for further research and applications. Most of these tools, like NLTK (Bird et al., 2009) and Stanford CoreNLP (Manning et al., 2014), have been developed for English, and, most importantly, are freely available. For Italian, several tools have been developed during the years such as TextPro (Pianta et al., 2008) and the Tanl Pipeline (Attardi et al., 2010) but unfortunately they are either outdated or not open source. An exception is represented by Tint (Aprosio and Moretti, 2016), a standalone freely available and customizable software based on Stanford CoreNLP. The main drawback of this solution is that it is a resource highly tailored for Italian in which some of the modules have been completely re-implemented on new classes and data structures compared to the CoreNLP ones. In addition, like for the other existing resources, it does not provide an output that is fully compatible with the Universal Dependency (UD) framework,[1] which is becoming the *de facto* standard especially for morpho-syntactic annotation, as well as for text annotation in general.

In this paper, we present CoreNLP-it, a set of customizable classes for CoreNLP designed for Italian. Our system, despite being simpler than any of the above mentioned toolkits, both in scope and number of features, has the advantage of being easily integrated with the CoreNLP suite, since its development has been grounded on the principle that all data structures be natively supported by CoreNLP.

The key properties of CoreNLP-it are:

- **UD based and compliant**: The toolkit and models are based on UD and follow its guidelines for token and parsing representation. It can provide all annotation required in the UD framework, and produces a CoNLL-U formatted output at any level of annotation, as well as any other type of annotation provided in CoreNLP.
- **Multi-word token representation**: Multi-word tokens (e.g., enclitic constructions) are handled by providing separate tokens. Moreover, the CoNLL-U output can represent such information following the UD guidelines.
- **Hybrid tokenization**: A fast and accurate hybrid tokenization and sentence splitting module replaces the original rule-based annotators for this task.
- **Integration with CoreNLP**: Given the way it is built (including the exclusive usage of

[1] http://universaldependencies.org/

CoreNLP classifiers and data structures), the add-on can be seamlessly integrated with the latest available version (3.9.1) of CoreNLP, and is expected to work with upcoming versions as well.

- **Support for other languages**: It provides out-of-the-box new capabilities of supporting basic annotations for other languages provided with a UD Treebank.

This paper is organized as follows: in Section 2, we present the architecture of the toolkit, whereas its core components (annotators) are described in Section 3. The results on Italian are discussed in Section 3.5. Section 4 shows preliminary experiments for the adaptation of the software to two additional languages provided with a UD treebank, namely Spanish and French.

2 Architecture

CoreNLP-it has been built as an *add-on* to the Stanford CoreNLP toolkit (Manning et al., 2014). CoreNLP offers a set of linguistic tools to perform core linguistic analyses of texts in English and other languages, and produces an annotated output in various formats such as CoNLL (Nivre et al., 2007), XML, Json, etc.

2.1 Stanford CoreNLP

The main architecture of CoreNLP consists of an *annotation* object as well as a sequence of *annotators* aimed at annotating texts at different levels of analysis. Starting from a raw text, each module adds a new annotation layer such as tokenizaton, PoS tagging, parsing etc. The behavior of the single annotators can be controlled via standard Java properties. Annotators can analyze text with both *rule-based* or *statistical-based* models. While rule-based models are typically language dependent, statistical based ones can be trained directly within the CoreNLP toolkit in order to improve the performance of the default models or to deal with different languages and domains.

2.2 CoreNLP-it

The main goal we pursued in developing CoreNLP-it was to keep the original CoreNLP structure and usage intact, while enabling it to deal with Italian texts in order to produce a UD-compliant and UD-complete output. More specifically, we aimed at building a system capable of providing all textual annotations required by the UD guidelines. Moreover, our system is also compatible with standard CoreNLP functions (e.g., Named Entity Recognition (NER) and Sentiment annotation). For these reasons,we implemented a series of *custom annotators* and *statistical models* for Italian. The custom annotators replace the corresponding CoreNLP annotators leaving intact the annotation structure and output of the annotators they are replacing.

For simplicity, we used only one of the UD treebanks available for Italian, namely the UD adaptation of the ISDT Italian Treebank (Bosco et al., 2013). The resource was used to build most of the new models, as well as for training standard statistical models (e.g., PoS tagging and Dependency Parsing) available in CoreNLP. More specifically, to obtain a UD-compliant output, we trained the Italian models on the *training*, *dev*, and *test* sets provided within the treebank.

The current version of CoreNLP-it can be easily integrated and configured into CoreNLP by adding the custom annotator classes and their respective models into the pipeline. Such classes and their properties can be added in a configuration file or called via the API interface. This procedure follows the standard CoreNLP documentation and guidelines for custom annotator classes. In addition, we provide a new class (resembling a CoreNLP one) for the training of the hybrid tokenization and sentence splitting. The configuration of the classifier and the required dictionaries (cf. Section 3.1) can be specified in a separate property file.

3 Modules

The annotators described in the following sections are aimed at producing a UD compliant and complete output. The following information is extracted from text: Sentences, Tokens, Universal PoS Tags, language specific PoS Tags, Lemmas, Morphological Features, and Dependency Parse Tree for each sentence.

In this section, we briefly describe each module of our linguistic pipeline, focusing on the annotators and models it implements.

3.1 Sentence Splitting and Tokenization

Sentence Splitting and Tokenization are handled by a single classifier, namely the annotator *it_tok_sent*. The process splits raw text into sen-

tences, and each sentence into tokens. Crucially, the tokenization process can deal with both single and multi-word tokens as specified by the CoNLL-U format.

Multi word tokens such as verbs with clitic pronouns (e.g., *portar-vi* "carry to you") and articulated prepositions (prep + determiner) (e.g., *della*, *di+la* "of the"), are split into their respective components. The information about the original word and its position in the sentence is however retained within each token by exploiting the *token span* and *original word* annotations.

Tokenization is usually solved with rule-based systems able to identify word and sentence boundaries, for example by identifying white spaces and full stops. However, in order to avoid encoding such set of rules, we implemented a model inspired by Evang et al. (2013). At its core, the process is driven by a hybrid model. First, it uses a character-based statistical model to recognize sentences, tokens, and clitic prepositions. Then, a rule based dictionary is used to optimize the multi-word tokens detection and splitting.

The classifier tags each character with respect to one of the following classes: i. `S`: start of a new sentence; ii. `T`: start of a new token; iii. `I`: inside of a token; iv. `O`: outside of a token; v. `C`: start of a clitic preposition inside a token (e.g. *mandar**vi***).

The classifier is a simple implementation of the maximum entropy *Column Data Classifier* available in the Stanford CoreNLP. To train the model, we used the following feature set: i. window: a window of n characters before and after the target character; ii. the case of the character; iii. the class of the previous character.

In order to deal with multi-tokens, the system allows for a full rule-based tagging of a parametric list of multi-tokens typically belonging to a strictly language dependent closed class words. In the Italian implementation, such words are *articulated prepositions* (prep + determiner). The word list to be ignored is fed to the classifier during training.

Moreover, an additional set of rules can be applied after the classification step in order to deal with possibly misclassified items. In particular, the system simply checks each token against a dictionary of multi-words and split them accordingly. In the case of Italian, we built a dictionary of *clitic verbs* (which are instead an open class) by bootstrapping the verbs in the treebank with all possible combinations of clitic pronouns. A final tagging phase was used to merge the rule-based and statistical predictions.

3.2 Part-of-Speech Tagging

The Maximum Entropy implementation of the Part-of-Speech Tagger (Toutanova et al., 2003) provided in the Stanford CoreNLP toolkit has been used to predict language dependant PoS Tags (xPoS).

In order to annotate Universal PoS (uPoS) tags, a separate annotator class, namely *upos*, has been implemented.

For what concerns the xPoS Tagger, the Maximum Entropy model was trained on the UD-ISDT Treebank. uPoS tags are instead approached with a rule based strategy. In particular, we built a mapping between xPoS and uPoS based on the UD-ISTD Treebank. The mapping is used within the annotator to assign the uPoS tag based on the predicted xPoS tag.

3.3 Lemmatization and Morphological Annotation

In order to annotate each token with its corresponding lemma and morphological features, we developed a rule-based custom annotator. The annotator exploits a parametric dictionary, to assign lemmas based on the word *form* and PoS. In particular, the dictionary contains the lemma and *UD morphological features* for n $(form, PoS)$ pairs. The form is used as the main access key to the dictionary, while PoS is used to solve ambiguity, e.g., between *amo* as "I love" or as "fishing hook". Finally, in cases of PoS ambiguity, corpus frequency is used to select the target lemma.

The dictionary can be manually built or extracted from a UD treebank. In the latter case, the provided *Vocabulary* class has methods to extract and build a serialized model of the vocabulary.

3.4 Dependency Parsing

The Neural Network Dependency Parser implemented in Stanford CoreNLP (Chen and Manning, 2014) allows models to be trained for different languages.

As for Italian, we used FastText (Joulin et al., 2016) Italian 300dim-pretrained embeddings described in Bojanowski et al. (2017). The dependency parser was trained with the default configuration provided in Stanford CoreNLP.

3.5 CoreNLP-it performances

Table 1 reports the global performances of the currently trained models. In particular, all our models were evaluated against the UD-ISDT Treebank test set.

With respect to the Tokenization, we measured the accuracy by considering the whole output of the tokenization process (i.e., the combination of the statistical classifier and rule based multi-word tokens detection). As for Lemmatization, we tested the system by predicting the lemmas for tokens in the UD-ISDT Italian test set. PoS Tagging and Dependency Parsing were tested with the system provided in CoreNLP.

Task	Tokens/sec	Results
Tok., S.Split.	17277.4	Accuracy: 99%
xPoS Tag	7575.4	F1: 0.97
Lemma	5553.1	Accuracy: 92%
Dep. Parsing	1717.8	LAS: 86.15
		UAS: 88.57

Table 1: Evaluation of CoreNLP-it modules on the UD-ISDT Treebank test set.

We must point out that one of the main shortcomings of implementing a more statistically oriented model for tokenization with respect to a rule based one is that it may underperform in the case of badly formatted or error-filled texts, which we cannot find in most Treebanks. However, we believe that such an approach could be nonetheless very useful in that it can be automatically scaled to different linguistic registers and text genres. Moreover, most typical errors could be avoided by means of data augmentation strategies and the use of more heterogeneous data for training, such as for example the PoSTWITA-UD Treebank (Sanguinetti et al., 2018).

It is important to stress that the main focus of this work was to build a framework allowing for a fast and easy implementation of UD models based on Stanford CoreNLP from a software engineering point of view. The basic pre-trained models are intended as a proof of concept, and will require further parameter tuning to increase their performance.

4 Flexibility Towards Other Languages

One of the key goals that has driven the development of CoreNLP-it is keeping the core code implementation as language independent as possible. To obtain the required linguistic knowledge, the framework exploits statistical models or external resources. On the one hand, the use of big linguistic resources to perform some of the tasks can affect the computational performances, but the system enables the construction of basic resources from the treebank used for training. On the other hand, this framework is very flexible, especially by considering tasks like tokenization and lemmatization. In particular, the system is able to produce a full UD-compliant Stanford Pipeline for languages for which an UD Treebank is available.

In order to validate this claim, we focused on two languages closely related to Italian, namely Spanish and French. We trained the respective models on the UD-adapted corpora ES-ANCORA (Taulé et al., 2008) and FR-GSD (Hernandez and Boudin, 2013). In these cases, to detect multiword tokens we exploited the information available in these corpora. It is clear that such models are intended as an interesting UD baseline, because the linguistic information they employ is not yet as optimized as the one used by the Italian models.

Since the core of the adaptation of the Stanford Pipeline to Universal Dependencies relies on the Tokenization phase, we report here the results obtained for this task. It is clear that the rest of the models (i.e., PoS tags and Parsing) can be trained simply by following the Stanford CoreNLP guidelines. Results obtained for the tokenization modules for French and Spanish are shown in Table 2.

Task	Language	Accuracy (%)
Tok., S.Split.	Spanish	99,9
	French	99,7
Lemma	Spanish	66
	French	69

Table 2: Evaluation of CoreNLP-it modules on Spanish and French.

All statistical models have similar performances with respect to Italian ones. The main differences, as expected, concern the tasks most dependent on external resources (e.g., Lemmatization). For example, we noticed a much lower recall for multiword token identification, given the exclusive use of the examples found in the training set. The approach shows very promising results especially for tokenization and sentence splitting modules which are central for all the subsequent levels of analysis

based on UD. It is clear that for PoS Tagging and Parsing further developments based on Stanford CoreNLP and language-specific resources are required to account for the specific features of each language.

5 Conclusion and Ongoing Work

In this paper, we presented CoreNLP-it, a set of add-on modules for the Stanford CoreNLP language toolkit. Our system provides basic language annotations such as sentence splitting, tokenization, PoS tagging, lemmatization and dependency parsing, and can provide a UD-compliant output. Our rule based and statistical models achieve good performances for all tasks. In addition, since the framework has been implemented as an add-on to Stanford CoreNLP, it offers the possibility of adding other new annotators, including for example the Stanford NER (Finkel et al., 2005). Moreover, first experiments on other languages have shown very good adaptation capability with very little effort.

In the near future, we plan to refine the core code by performing extensive tests to better deal with additional UD-supported languages and optimize their performances. We also plan to release the tool as well as the basic trained models for Italian. Moreover, we intend to perform data augmentation strategies to refine our models and make them able to work properly also with ill-formed or substandard text input.

References

Alessio Palmero Aprosio and Giovanni Moretti. 2016. Italy goes to Stanford: a collection of CoreNLP modules for Italian. *CoRR*.

Giuseppe Attardi, Stefano Dei Rossi, and Maria Simi. 2010. The tanl pipeline. In *LREC Workshop on WSPP*, pages 15–21, Valletta, Malta.

Steven Bird, Ewan Klein, and Edward Loper. 2009. *Natural Language Processing with Python*.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146.

Cristina Bosco, Simonetta Montemagni, and Maria Simi. 2013. Converting italian treebanks: Towards an italian stanford dependency treebank. In *Proceedings of the 7th Linguistic Annotation Workshop and Interoperability with Discourse*, pages 61–69, Sofia, Bulgaria.

Danqi Chen and Christopher Manning. 2014. A fast and accurate dependency parser using neural networks. In *Proceedings of EMNLP 2014*, pages 740–750, Doha, Qatar.

Kilian Evang, Valerio Basile, Grzegorz Chrupała, and Johan Bos. 2013. Elephant: Sequence labeling for word and sentence segmentation. In *Proceedings of EMNLP 2013*, pages 1422–1426, Seattle, Washington, USA. ACL.

Jenny Rose Finkel, Trond Grenager, and Christopher Manning. 2005. Incorporating non-local information into information extraction systems by Gibbs sampling. In *Proceedings of ACL 2005*, ACL '05, pages 363–370, Stroudsburg, PA, USA. ACL.

Nicolas Hernandez and Florian Boudin. 2013. Construction automatique d'un large corpus libre annoté morpho-syntaxiquement en français. In *Actes de la conférence TALN-RECITAL 2013*, pages 160–173, Sables d'Olonne, France.

Armand Joulin, Edouard Grave, Piotr Bojanowski, Matthijs Douze, Hervé Jégou, and Tomas Mikolov. 2016. Fasttext.zip: Compressing text classification models. *CoRR*.

Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J. Bethard, and David McClosky. 2014. The Stanford CoreNLP natural language processing toolkit. In *Association for Computational Linguistics (ACL) System Demonstrations*, pages 55–60.

Joakim Nivre, Johan Hall, Sandra Kübler, Ryan McDonald, Jens Nilsson, Sebastian Riedel, and Deniz Yuret. 2007. The CoNLL 2007 shared task on dependency parsing. In *Proceedings of The CoNLL Shared Task Session of EMNLP-CoNLL 2007*, pages 915–932, Prague, Czech Republic. ACL.

Emanuele Pianta, Christian Girardi, and Roberto Zanoli. 2008. The TextPro tool suite. In *Proceedings of LREC 2008*, pages 2603–2607, Marrakech, Morocco. European Language Resources Association (ELRA).

Manuela Sanguinetti, Cristina Bosco, Alberto Lavelli, Alessandro Mazzei, Oronzo Antonelli, and Fabio Tamburini. 2018. Postwita-ud: an italian twitter treebank in universal dependencies. In *Proceedings of LREC 2018*.

Mariona Taulé, Maria Antònia Martí, and Marta Recasens. 2008. AnCora: Multilevel annotated corpora for catalan and spanish. In *Proceedings of LREC 2008*, pages 96–101, Marrakech, Morocco.

Kristina Toutanova, Dan Klein, Christopher D. Manning, and Yoram Singer. 2003. Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proceedings of NAACL 2003*, NAACL '03, pages 173–180, Stroudsburg, PA, USA. ACL.