Visualizing Deep-Syntactic Parser Output

Juan Soler-Company[1] **Miguel Ballesteros**[1] **Bernd Bohnet**[2]
Simon Mille[1] **Leo Wanner**[1,3]
[1]Natural Language Processing Group, Pompeu Fabra University, Barcelona, Spain
[2]Google Inc.
[3]Catalan Institute for Research and Advanced Studies (ICREA)
[1,3]{name.lastname}@upf.edu [2]bohnetbd@google.com

Abstract

"Deep-syntactic" dependency structures bridge the gap between the surface-syntactic structures as produced by state-of-the-art dependency parsers and semantic logical forms in that they abstract away from surface-syntactic idiosyncrasies, but still keep the linguistic structure of a sentence. They have thus a great potential for such downstream applications as machine translation and summarization. In this demo paper, we propose an online version of a deep-syntactic parser that outputs deep-syntactic structures from plain sentences and visualizes them using the Brat tool. Along with the deep-syntactic structures, the user can also inspect the visual presentation of the surface-syntactic structures that serve as input to the deep-syntactic parser and that are produced by the joint tagger and syntactic transition-based parser ran in the pipeline before deep-syntactic parsing takes place.

1 Introduction

"Deep-syntactic" dependency structures bridge the gap between surface-syntactic structures as produced by state-of-the-art dependency parsers and semantic logical forms in that they abstract away from surface-syntactic idiosyncrasies, but still keep the linguistic structure of a sentence. More precisely, a deep-syntactic structure (DSyntS) is a dependency tree that captures the argumentative, attributive and coordinative relations between full (i.e., meaningful) words of a sentence. For illustration, Figure 1 shows a surface-syntactic structure (above) and deep-syntactic structure (below) for the sentence: *almost 1.2 million jobs have been created by the state in that time.*

DSyntSs have a great potential for such downstream applications as deep machine translation, summarization or information extraction. In deep machine translation as discussed, e.g., by Jones et al. (2012), DSyntSs simplify the alignment between the source and target language structures considerably. In extractive summarization, sentence fusion (Filippova and Strube, 2008) becomes much more straightforward at the level of DSyntSs. A stochastic sentence realizer that takes as input DSyntSs can then be used to generate surface sentences (Ballesteros et al., 2015). In information extraction (Attardi and Simi, 2014) the procedures for the distillation of the information to fill the slots of the corresponding patterns are also simpler at the DSyntS level.

However, it is only recently that deep-syntactic parsing has been introduced as a new parsing paradigm; see, e.g., (Ballesteros et al., 2014).[1] No visualization interfaces are available as yet to control the output of deep-syntactic parsers. In this paper, we propose such a visualization interface. The interface can be used for both a pipeline consisting of a syntactic parser and a deep parser and a joint syntactic+deep parser. In the first configuration, it facilitates the visualization of the output of the syntactic parser and of the output of the deep parser. In the second configuration, it visualizes directly the output of the joint parser.

In what follows, we present its use for the first configuration applied to English. As surface-syntactic parser, we use Bohnet and Nivre (2012)'s joint tagger+lemmatizer+parser. As deep parser, we use Ballesteros et al. (2014)'s implementation. Both have been trained on the dependency Penn Treebank (Johansson and Nugues, 2007), which has been extended by the DSyntS-annotation. The interface can be inspected online; cf. `http://dparse.`

[1]The source code of Ballesteros et al.'s deep parser and a short manual on how to use it can be downloaded from `https://github.com/talnsoftware/deepsyntacticparsing/wiki`.

Proceedings of NAACL-HLT 2015, pages 56–60,
Denver, Colorado, May 31 – June 5, 2015.

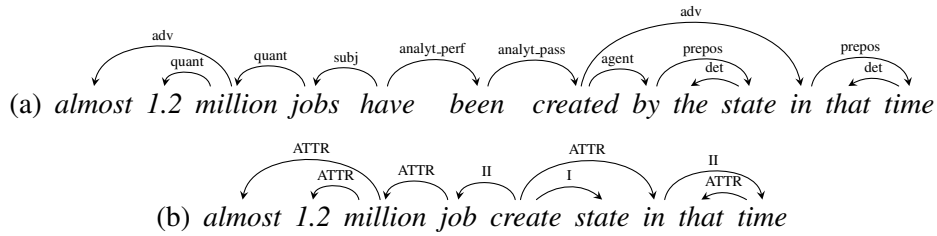

Figure 1: Sample equivalent (a) SSynt- and (b) DSynt-structures. A SSyntS contains all tokens of the sentence, while in the corresponding DSyntS the grammatical tokens that are void of lexical meaning are omitted.

`multisensor.taln.upf.edu/main`. It accepts as input an English sentence and delivers as output the surface- and deep-syntactic structures of this sentence.

Section 2 shows how the online model of the deep parser is trained and displays its performance for the English model. Section 3 describes the visualization interface and its use online. In Section 4, a number of other existing visualizers of the output of dependency parsers are briefly listed. Section 5, finally, concludes and makes some suggestions for future work.

2 Deep-Syntactic Parser

As already mentioned above, we use the joint PoS-tagger+lemmatizer+parser of Bohnet and Nivre (2012)[2] as surface parser, setting up a pipeline with the deep-syntactic parser of Ballesteros et al. (2014).[3] The output of the first serves as input to the latter.

The online versions of the joint PoS-tagger+lemmatizer+parser and the deep-syntactic parser have been trained on the dependency Penn Treebank (Johansson and Nugues, 2007) in CoNLL09 format. To have an English training dataset for the deep-syntactic parser, we derived DSyntSs from the syntactic structures of the dependency Penn Treebank, extending thus the Penn Treebank by a new layer of annotation, as described in Section 2.1. The performance figures obtained using this dataset are shown in Section 2.2.

[2]The joint PoS-tagger+Lemmatizer+parser is available for downloading at https://code.google.com/p/mate-tools/.

[3]The deep-syntactic parser is availabe for download at https://code.google.com/p/deepsyntacticparsing/.

2.1 Training Dataset for the Deep-Syntactic Parser

The English deep-syntactic dataset has been obtained using a rule-based graph transducer that converts the syntactic annotation of the dependency Penn Treebank into a DSyntS annotation in the CoNLL09 format. The conversion removes definite and indefinite determiners, auxiliaries, THAT complementizers, TO infinitive markers, and all functional (or *lexically-bound*) prepositions which we were able to recover in PropBank and NomBank. In these two resources, 11,781 disambiguated predicates are described and their semantic roles are listed. We use two fields of their XML files to gather prepositions: the last word of the field "descr" in "roles", and the first word of the field of the corresponding role in "example". In this way, we retrieve, for instance, for the lexical unit *beg.01* the preposition *from* for the second semantic role (as in *beg from someone*), and the preposition *for* for the third role (as in *beg someone for something*). The correspondence between prepositions and semantic roles is also used for the mapping of dependency relations (Mille and Wanner, 2015).

For each surface dependency relation, a default mapping that is conditioned by the encountered syntactic structure and dictionary entries is defined. Thus, a subject is by default mapped to a first argument *I* unless it is the subject of a passive verb. In this case, the subject is mapped to the second argument *II*. Along similar lines, a dictionary entry may specify in the subcategorization pattern of a headword the association of a given preposition to a different argument slot than indicated by the default mapping. For instance, in the sentence *Sony announced its plans to hire Mr. Guber*, *to* is a depen-

	POS	LEMMA	LAS	UAS
English	98.50	99.46	89.70	92.21

Table 1: Performance of Bohnet and Nivre's joint PoS-tagger+dependency parser trained on the PTB Treebank for English.

Hypernode Detection (English)

Measure	SSyntS–DSyntS Transducer
p_h	98.42 (41967/42461)
r_h	98.82 (41967/42467)
$F1_h$	98.62

Attachment and labeling (English)

Measure	SSynS–DSyntS Transducer
LAP	81.80 (34882/42461)
UAP	85.82 (36598/42461)
LA-P	89.11 (37998/42641)
LAR	82.14 (34882/42467)
UAR	86.18 (36598/42467)
LA-R	89.48 (37998/42467)

Table 2: Performance of the Ballesteros et al. deep-syntactic parser trained on the adapted version of the PTB Treebank for English.

dent of *plan* with the surface dependency *NMOD*. *NMOD* is by default mapped to the deep relation *ATTR*, but in the dictionary entry of *plan* it is stated that a dependent introduced by *to* is mapped to *II*, such that in the case of *plan*, the default will be overwritten in that *NMOD* will be mapped to *II*.

2.2 Parser Results

Our models offer state-of-the-art performance for part-of-speech tagging, lemmatization, syntactic dependency parsing and deep-syntactic parsing.[4] Tables 1[5] and 2[6] show the results of both parsers.

[4]This is the first attempt to build English deep-syntactic structures; Ballesteros et al. (2014) report results for Spanish only.

[5]'POS' stands for part-of-speech accuracy, 'LEMMA' for lemma accuracy, 'LAS' for labeled attachment score, and 'UAS' for unlabeled attachment score

[6]'p_h' stands for hypernode detection precision, 'r_h' for hypernode detection recall, '$F1_h$' for hypernode detection F1 measure, 'LAP' for labeled attachment precision, 'UAP' for unlabeled attachment precision, 'LA-P' for label accuracy precision, 'LAR' for labeled attachment recall, 'UAR' for unlabeled attachment recall, and 'LA-R' for label accuracy recall.

3 Tree Visualization

Our visualization interface is built with a Java HTTPServer, which is bound to an IP address and port number that listens to incoming connections from users. The HTTPServer Java class connects with both the joint tagger+lemmatizer+parser and the deep-syntactic parser and provides the output of plain text input sentences in real time. To ensure real time performance, a model of both parsers is already loaded, and the interface waits for new input given by the users.

The main page (see `http://dparse.multisensor.taln.upf.edu/main`) lets the user introduce a text and select what kind of parsing she wants to see in the output, the surface-syntactic, deep-syntactic or both at the same time. Depending on the choice of the user, after parsing the CoNLL outputs (surface- and/or deep-syntactic) are shown. If desired, they can be also downloaded. A click on the corresponding link takes the user to the graphic representation of the parse tree.

The visualization of the output is performed by the annotation tool Brat (Stenetorp et al., 2012). Brat takes an annotation file, which is produced by transforming the CoNLL files that the parsers output into Brat's native format, and generates the graphical interface for the dependency trees.

Figure 2 shows three sample surface syntactic structures in Brat. In Figure 3, their equivalent deep-syntactic structures are displayed. As already Figure 1, the figures illustrate the difference of both types of structures with respect to the abstraction of linguistic phenomena. The DSyntSs are clearly much closer to semantics. As a matter of fact, they are equivalent to PropBank structures (Palmer et al., 2005). However, this does not mean that they must *per se* be "simpler" than their corresponding surface-syntactic structures—compare, for instance, the structures (3a) and (3b) in Figures 2 and 3, where both SSyntS and DSyntS contain the same number of nodes, i.e., are isomorphic.

The structures (2a) and (2b) illustrate the capacity of the deep parser to correctly identify the arguments of a lexical item without that explicit hints are available in the surface structure.

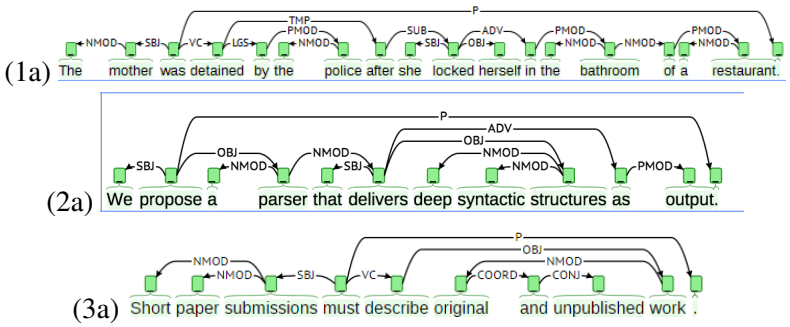

Figure 2: Visualization of surface syntactic structures with Brat

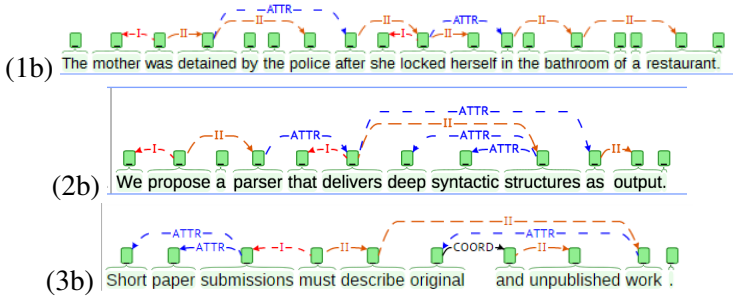

Figure 3: Visualization of deep-syntactic structures with Brat

4 Related Work

Visualization interfaces normally offer a universal and simple way to access the output of NLP tools, among them parsers. This leads to better comprehension of their outputs and a better usability for downstream applications. Therefore, it is not surprising that visualization interfaces have been a relevant topic during the last years in the NLP community; see, e.g., (Collins et al., 2008; Collins et al., 2009; Feng and Lapata, 2010). In the parsing area, tools such as MaltEval (Nilsson and Nivre, 2008), the Mate Tools (Bohnet and Wanner, 2010), XLDD (Culy et al., 2011), TreeExplorer (Thiele et al., 2013), ViZPar (Ortiz et al., 2014), MaltDiver (Ballesteros and Carlini, 2013), or XLike Services (Carreras et al., 2014) have been proposed for the visualization of parse trees and their subsequent evaluation. The interface described in this paper serves a similar purpose. To the best of our knowledge, it is the first interface that uses the flexible off-the-shelf tool Brat and that serves for the visualization of deep-syntactic structures.

5 Conclusions and Future Work

We have presented an operational interface for the visualization of the output of a deep-syntactic parser and of surface-syntactic structures that serve it as input. The interface is flexible in that it allows for the display of any additional structural information provided by an extended parsing pipeline. For instance, if the obtained deep-syntactic structure is projected onto a frame-like structure (Chen et al., 2010) with semantic roles as arc labels, this frame structure can be displayed as well. We are currently working on such an extension. Furthermore, we aim to expand our visualization interface to facilitate active exploration of linguistic structures with Brat and thus add to the static display of structures the dimension of *Visual Analytics* (Keim et al., 2008).

Acknowledgments

This work has been partially funded by the European Union's Seventh Framework and Horizon 2020 Research and Innovation Programmes under the Grant Agreement numbers FP7-ICT-610411, FP7-SME-606163, and H2020-RIA-645012.

References

G. Attardi and M. Simi. 2014. Dependency parsing techniques for information extraction. In *Proceedings of Evalita 2014*.

M. Ballesteros and R. Carlini. 2013. Maltdiver: A transition-based parser visualizer. In *Demonstrations of the Sixth International Joint Conference on Natural Language Processing*, page 25. IJCNLP.

M. Ballesteros, B. Bohnet, S. Mille, and L. Wanner. 2014. Deep-syntactic parsing. In *Proceedings of the 25th International Conference on Computational Linguistics (COLING)*.

M. Ballesteros, B. Bohnet, S. Mille, and L. Wanner. 2015. Data-driven sentence generation with non-isomorphic trees. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics – Human Language Technologies (NAACL HLT 2015)*.

B. Bohnet and J. Nivre. 2012. A transition-based system for joint part-of-speech tagging and labeled non-projective dependency parsing. In *EMNLP-CoNLL*.

B. Bohnet and L. Wanner. 2010. Open Source Graph Transducer Interpreter and Grammar Development Environment. In *Proceedings of the International Conference on Linguistic Resources and Evaluation (LREC)*.

X. Carreras, L. Padró, L. Zhang, Z. Rettinger, A.and Li, E. Garcıa-Cuesta, Z. Agic, B. Bekavec, B. Fortuna, and T. Štajner. 2014. Xlike project language analysis services. *Proceedings of the Demonstrations Session at EACL*, pages 9–12.

D. Chen, N. Schneider, D. Das, and N.A. Smith. 2010. Semafor: Frame argument resolution with log-linear models. In *Proceedings of the 5th international workshop on semantic evaluation*, pages 264–267. Association for Computational Linguistics.

C. Collins, G. Penn, and S. Carpendale. 2008. Interactive visualization for computational linguistics. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics on Human Language Technologies*, Stroudsburg, PA, USA. Association for Computational Linguistics.

C. Collins, S. Carpendale, and G. Penn. 2009. DocuBurst: Visualizing Document Content Using Language Structure. In *Proceedings of the Eurographics/IEEE-VGTC Symposium on Visualization (EuroVis '09)*, pages 1039–1046. Eurographics Association.

C. Culy, V. Lyding, and H. Dittmann. 2011. xLDD: Extended Linguistic Dependency Diagrams. In *Proceedings of the 2011 15th International Conference on Information Visualisation*, IV '11, pages 164–169, Washington, DC, USA. IEEE Computer Society.

Y. Feng and M. Lapata. 2010. Visual Information in Semantic Representation. In *Proceedings of the 2010 Conference of the North American Chapter of the Association for Computational Linguistics – Human Language Technologies (NAACL HLT 2010)*, pages 91–99, Stroudsburg, PA, USA. Association for Computational Linguistics.

K. Filippova and M. Strube. 2008. Sentence fusion via dependency graph compression. In *Proceedings of the International Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

R. Johansson and P. Nugues. 2007. Extended constituent-to-dependency conversion for English. In *Proceedings of the 16th Nordic Conference of Computational Linguistics (NODALIDA)*, pages 105–112, Tartu, Estonia, May 25-26.

B. Jones, J. Andreas, D. Bauer, K.-M. Hermann, and K. Knight. 2012. Semantics-based machine translation with hyperedge replacement grammars. In *Proceedings of the International Conference on Computational Linguistics (COLING)*.

D.A. Keim, F. Mansmann, J. Schneidewind, J. Thomas, and H. Ziegler. 2008. Visual Analytics: Scope and Challenges. In S. Simoff, editor, *Visual Data Mining, LNCS 4404*, pages 76–90. Springer Verlag, Berlin.

S. Mille and L. Wanner. 2015. Towards large-coverage detailed lexical resources for data-to-text generation. In *Proceedings of the First International Workshop on Data-to-Text Generation*, Edinburgh, Scotland.

Jens Nilsson and Joakim Nivre. 2008. Malteval: an evaluation and visualization tool for dependency parsing. In *Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC'08)*, Marrakech, Morocco, may. European Language Resources Association (ELRA).

I. Ortiz, M. Ballesteros, and Y. Zhang. 2014. ViZPar: A GUI for ZPar with Manual Feature Selection. *Procesamiento del lenguaje natural*, 53.

Martha Palmer, Paul Kingsbury, and Daniel Gildea. 2005. The Proposition Bank: An annotated corpus of semantic roles. *Computational Linguistics*, 31.

P. Stenetorp, S. Pyysalo, G. Topić, T. Ohta, S. Ananiadou, and J. Tsujii. 2012. BRAT: A Web-based Tool for NLP-Assisted Text Annotation. In *13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 102–107. Association for Computational Linguistics.

G. Thiele, M. Gärtner, W. Seeker, A. Björkelund, and J. Kuhn. 2013. Treeexplorer – An extensible Graphical Search Tool for Dependency Treebanks. In *Proceedings of the Demonstrations of the 51st Annual Meeting of the Association for Computational Linguistics (ACL 2013)*.