LiMoSINe pipeline: Multilingual UIMA-based NLP platform

Olga Uryupina[1], Barbara Plank[2], Gianni Barlacchi[1,3],
Francisco Valverde Albacete[4], Manos Tsagkias[5], Antonio Uva[1], and Alessandro Moschitti[6,1]
[1]Department of Information Engineering and Computer Science, University of Trento, Italy
[2]University of Groningen, The Netherlands
[3]SKIL - Telecom Italia, Trento, Italy
[4]Dept. Teoria de Señal y Comunicaciones, Universidad Carlos III de Madrid, Spain
[5]904Labs, Amsterdam, The Netherlands
[6]Qatar Computing Research Institute
uryupina@gmail.com, b.plank@rug.nl, gianni.barlacchi@unitn.it,
fva@tsc.uc3m.es, manos@904labs.com,
antonio.uva@unitn.it, amoschitti@gmail.com

Abstract

We present a robust and efficient parallelizable multilingual UIMA-based platform for automatically annotating textual inputs with different layers of linguistic description, ranging from surface level phenomena all the way down to deep discourse-level information. In particular, given an input text, the pipeline extracts: sentences and tokens; entity mentions; syntactic information; opinionated expressions; relations between entity mentions; co-reference chains and wikified entities. The system is available in two versions: a standalone distribution enables design and optimization of user-specific sub-modules, whereas a server-client distribution allows for straightforward high-performance NLP processing, reducing the engineering cost for higher-level tasks.

1 Introduction

With the growing amount of textual information available on an everyday basis, Natural Language Processing gets more and more large-scale. Moreover, a lot of effort has been invested in the recent years into the development of multi- and cross-lingual resources. To efficiently use large amounts of data for high-level tasks, e.g., for Information Extraction, we need robust parallelizable multilingual preprocessing pipelines to automatically annotate textual inputs with a variety of linguistic structures. To address the issue, we present the LiMoSINe Pipeline—a platform developed by the FP7 EU project LiMoSINE: Linguistically Motivated Semantic aggregation engiNes.

Several platforms and toolkits for NLP preprocessing have been made available to the research community in the past decades. The most commonly used ones are OpenNLP[1], FreeLing (Padró and Stanilovsky, 2012) and GATE (Cunningham et al., 2011). In addition, many research groups publicly release their preprocessing modules. These approaches, however, pose several problems:

- most of these tools require a considerable effort for installation, configuration and getting familiar with the software,
- parallelization might be an issue,
- for languages other than English, many modules are missing, while the existing ones often have only a moderate performance level.

In the LiMoSINe project, we focus on high-performance NLP processing for four European languages: English, Italian, Spanish and Dutch. We combine state-of-the-art solutions with specifically designed in-house modules to ensure reliable performance. Using the UIMA framework, we opt for a fully parallelizable approach, making it feasible to process large amounts of data. Finally, we release the system in two versions: a client application connects to the pipeline installed on the LiMoSINe server to provide the users with all the annotation they require. This does not require any advanced installation or configuration of the software, thus reducing the engineering cost for the potential stake holders. A local installation of the pipeline, on the contrary, requires some effort to get familiar with the system, but it also gives users a possibility to integrate their own modules, thus allowing for a greater flexibility. The pipeline is available at http://ikernels-portal.disi.unitn.it/projects/limosine/.

2 LiMoSINe pipeline: overall structure

Our platform supports various levels of linguistic description, representing a document from different angles. It should therefore combine outputs of numerous linguistic preprocessors to provide a uniform and deep representation of a document's semantics. The overall structure of our pipeline is shown on Figure 1. This complex structure raises an issue of the compatibility between preprocessors: with many NLP modules around—publicly available, implemented by the LiMoSINe partners or designed by potential stakeholders—it becomes virtually impossible to ensure that

[1]http://opennlp.apache.org

Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics—System Demonstrations, pages 157–162,
Berlin, Germany, August 7-12, 2016.

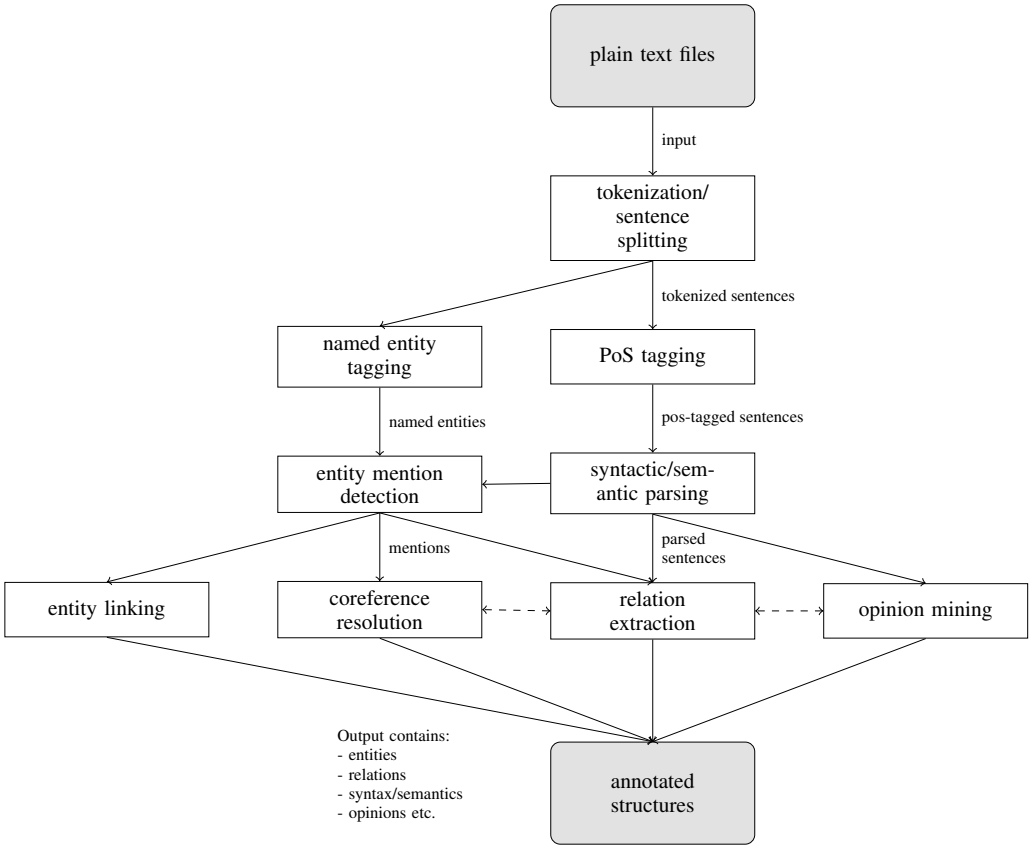

Figure 1: LiMoSINe pipeline architecture

any two modules have the same input/output format and thus can be run as a pipeline.

We have focused on creating a platform that allows for straightforward incorporation of various tools, coordinating their inputs and outputs in a uniform way. Our LiMoSINe Pipeline is based on Apache UIMA—a framework for Unstructured Information Management.[2] UIMA has been successfully used for a number of NLP projects, e.g., for the IBM Watson system (Ferrucci et al., 2010).

One of the main features of UIMA is its modularity: the individual annotators only incrementally update the document representation ("CAS"), but do not interact with each other. This allows for a straightforward deployment of new components: to add a new module to a UIMA system, one only has to create a wrapper converting its input and output objects into CAS structures. Moreover, UIMA allows for full parallelization of the processing flow, which is especially crucial when we aim at annotating large amounts of data.

UIMA-based systems can be deployed both locally or remotely. To run a UIMA application on a local machine, the user should follow the instructions on the UIMA web site to download and install UIMA. The LiMoSINe Pipeline should then be downloaded and run. While this requires some engineering effort, such an approach would allow the user to implement and integrate their own modules into the existing pipeline, as well as to re-optimize (e.g., retraining a parser to cover a specific domain).

A client-server version of the pipeline has been installed on the LiMoSINe server. The client application can be downloaded from the pipeline website. The users do not need to install any UIMA-related software to use this service. While this approach does not provide the flexibility of a local installation, it allows the users to obtain state-of-the-art NLP annotations for their textual inputs at no engineering cost at all. This might provide a valuable support for projects focusing on higher-level tasks, for example, on Question Answering, especially for languages other than English, considerably reducing the effort required for implementing and integrating all the preprocessing components needed.

3 Integrated modules

The LiMoSINe project has focused on four European languages: English, Italian, Spanish and Dutch. For all these languages, we have created a platform that provides robust parallelizable NLP processing up to the

[2] http://uima.apache.org/

syntactic parsing level. This already allows to create complex structural representations of sentences, to be used for higher-level tasks, such as Opinion Mining or Question Answering (cf. Section 4 below). In addition, where possible, we have integrated deeper semantic and discourse-level processing, such as relation extraction, coreference, opinion mining and entity linking. Table 1 provides an overview of all the currently supported modules.

The feasibility of our approach depends crucially on the performance of linguistic processors for a specific language and on the availability of the manually annotated data. Despite a growing interest in the multilingual processing in the NLP community, for a number of tasks no robust processors are available for languages other than English and for some others even a generic model cannot be retrained due to the lack of data. While we tried to rely as much as possible on the state-of-the-art technology, we had to implement or re-optimize a number of preprocessors.

3.1 English

Stanford tools. To provide basic preprocessing, required by our high-level components, we created UIMA wrappers for several Stanford NLP tools (Manning et al., 2014): the tokenizer, the parser and the named entity analyzer.

Entity Mention Detector. Both coreference resolver and relation extractor require information on mentions—textual units that correspond to real-world objects. Even though some studies focus on specific subtypes of mentions (for example, on pronominal coreference or on relations between named entities), we believe that a reliable pipeline should provide information on all the possible mentions.

An entity mention detector (EMD), covering a wide variety of mentions, has been developed at the University of Trento as a part of BART (see below). A more recent version has been proposed for the CoNLL-2011/2012 Shared Tasks (Uryupina et al., 2011; Uryupina et al., 2012). It is a rule-based system that combines the outputs of a parser and an NE-tagger to extract mention boundaries (both full and minimal nominal spans) and assign mention types (name, nominal or pronoun) and semantic classes (inferred from WordNet for common nouns, from NER labels for proper nouns). We are currently planning to integrate learning-based EMD (Uryupina and Moschitti, 2013) to cover additional languages, in particular, Arabic.

Opinion Mining. The opinion expression annotator is a system developed at the University of Trento by Johansson and Moschitti (2011). It extracts fine-grained opinion expressions together with their polarity. To extract opinion expressions, it uses a standard sequence labeler for subjective expression markup similar to the approach by (Breck et al., 2007). The system has been developed on the MPQA corpus that contains news articles. It internally uses the syntactic/semantic LTH dependency parser of (Johansson and Nugues, 2008). The opinion mining tool thus requires CoNLL-2008-formatted data as input, as output by the parser, and as such needs pre-tokenized and tagged input.

Relation Extraction. The relation extractor (RE) is a tree-kernel based system developed at the University of Trento (Moschitti, 2006; Plank and Moschitti, 2013). Tree kernel-based methods have been shown to outperform feature-based RE approach (Nguyen et al., 2015). The system takes as input the entity mentions detected by the EMD module (which provides information on the entity types, i.e. PERSON, LOCATION, ORGANIZATION or ENTITY).

The first version of the relation extractor was trained on the ACE 2004 data. It provides the following binary relations as output: Physical, Personal/Social, Employment/Membership, PER/ORG Affiliation and GPE Affiliation.

An extended version of the Relation Extractor includes an additional model trained on the CoNLL 2004 data (Roth and Yih, 2004) following the setup of Giuliano et al. (2007). The model uses a composite kernel consisting of a constituency-based path-enclosed tree kernel and a linear feature vector encoding local and global contexts (Giuliano et al., 2007). The CoNLL 2004 model contains the following relations: LiveIn, LocatedIn, WorkFor, OrgBasedIn, Kill.

Both models exhibit state-of-the art performance. For the ACE 2004 data, experiments are reported in (Plank and Moschitti, 2013). For the CoNLL 2004 data, our model achieves results comparable to or advancing the state-of-the-art (Giuliano et al., 2007; Ghosh and Muresan, 2012).

Coreference Resolution. Our coreference resolution Analysis Engine is a wrapper around BART—a toolkit for Coreference Resolution developed at the University of Trento (Versley et al., 2008; Uryupina et al., 2012). It is a modular anaphora resolution system that supports state-of-the-art statistical approaches to the task and enables efficient feature engineering. BART implements several models of anaphora resolution (mention-pair and entity-mention; best-first vs. ranking), has interfaces to different machine learners (MaxEnt, SVM, decision trees) and provides a large set of linguistically motivated features, along with the possibility to design new ones.

Entity Linking. The Entity Linking Analysis Engine ("Semanticizer") makes use of the Entity Linking Web Service developed by the University of Amsterdam

Annotator	English	Italian	Spanish	Dutch
tokenizer	Stanford	TextPro	IXA	xTas/Frog
POS-tagger	Stanford	TextPro	IXA	xTas/Frog
NER	Stanford	TextPro	IXA	xTas/Frog
Parsing	Stanford, LTH	FBK-Berkeley	IXA	xTas/Alpino
Entity Mention Detection	BART	BART-Ita	-	-
Opinion Mining	Johansson&Moschitti (2001)	-	-	-
Relation Extraction	RE-UNITN	RE-UNITN unlex	-	-
Coreference	BART	Bart-Ita	-	-
Entity Linking	Semanticizer	Semanticizer	Semanticizer	Semanticizer

Table 1: Supported modules for different languages

(Meij et al., 2012). The web service supports automatic linking of an input text to Wikipedia articles: the output of the web service API is a list of IDs of recognized articles, together with confidence scores as well as the part of the input text that was matched. This entity linking module can be considered as cross-lingual and cross-document co-reference resolution, since entity mentions in documents in different languages are disambiguated and linked to Wikipedia articles. Each annotation unit corresponds to a span in the document and is labeled with two attributes: the corresponding Wikipedia ID and the system's confidence.

3.2 Italian

For Italian, we have been able to integrate language-specific processors for tokenization, sentence splitting, named entity recognition, parsing, mention detection and coreference. For relation extraction, we have followed a domain adaptation approach, transferring an unlexicalized model learned on the English data. A detailed description of our annotators for Italian is provided below.

TextPro wrapper. To provide basic levels of linguistic processing, we rely on TextPro—a suite of Natural Language Processing tools for analysis of Italian (and English) texts (Pianta et al., 2008). The suite has been designed to integrate various NLP components developed by researchers at Fondazione Bruno Kessler (FBK). The TextPro suite has shown exceptional performance for several NLP tasks at multiple EvalIta competitions. Moreover, the toolkit is being constantly updated and developed further by FBK. We can therefore be sure that TextPro provides state-of-the-art processing for Italian.

TextPro combines rule-based and statistical methods. It also allows for a straightforward integration of task-specific user-defined pre- and post-processing techniques. For example, one can customize TextPro to provide better segmentation for web data.

TextPro is not a part of the LiMoSINe pipeline, it can be obtained from FBK and installed on any platform in a straightforward way. No TextPro installation is needed for the client version of the semantic model.

Parsing. A model has been trained for Italian on the Torino Treebank data[3] using the Berkeley parser by the Fondazione Bruno Kessler. The treebank being relatively small, a better performance can be achieved by enforcing TextPro part-of-speech tags when training and running the parser. Both the Torino Treebank itself and the parsing model use specific tagsets that do not correspond to the Penn TreeBank tags of the English parser. To facilitate cross-lingual processing and enable unlexicalized cross-lingual modeling for deep semantic tasks, we have mapped these tagsets to each other.

Entity Mention Detection. We have adjusted our Entity Mention Detection analysis engine to cover the Italian data. Similarly to the English module, we use BART to heuristically extract mention boundaries from parse trees. However, due to the specifics of the Torino Treebank annotation guidelines, we had to change the extraction rules substantially.

Relation Extraction. Since no relation extraction datasets are available for Italian, we have opted for a domain adaptation solution, learning an unlexicalized model on the English RE data. This model aims at capturing structural patterns characteristic for specific relations through tree kernel-based SVMs. This solution requires some experiments on making English and Italian parse trees more uniform, for example, on translating the tagsets. We extract tree-based patterns for CoNLL-2004 relations (see above) from the unlexicalized variant of the English corpus and then run it on modified Italian parse trees. Clearly, this model cannot provide robust and accurate annotation. It can, however, be used as a benchmark for supervised RE in Italian. To improve the model's precision, we have restricted its coverage to named entities in contrast to all the nominal mentions used by the English RE models.

Coreference Resolution. A coreference model for BART has been trained on the Italian portion of the SemEval-2010 Task 1 dataset (Uryupina and Moschitti, 2014). Apart from retraining the model, we have incorporated some language-specific features to account,

[3] http://www.di.unito.it/~tutreeb/

for example, for abbreviation and aliasing patterns in Italian. The Italian version of BART, therefore, is a high-performance language-specific system. It has shown reliable performance at the recent shared tasks for Italian, in particular, at the SemEval-2010 Task 1 (Broscheit et al., 2010) and at the EvalIta 2009 (Biggio et al., 2009).

Both our English and Italian coreference modules are based on BART. Their configurations (parameter settings and features) have been optimized separately to enhance the performance level on a specific language. Since BART is a highly modular toolkit itself and its language-specific functionality can be controlled via a Language Plugin, no extra BART installation is required to run the Italian coreference resolver.

3.3 Spanish

We have tested two publicly available toolkits supporting language processing in Spanish: OpenNLP and IXA (Agerri et al., 2014). The latter has shown a better performance level and has therefore been integrated for the final release of the LiMoSINe pipeline.

For tokenization, we rely on the `ixa-pipe-tok` library (version 1.5.0) from the IXA pipes project. Since it uses FSA technology for the tokenization and a rule-based segmenter, it is fast (tokenizing around 250K words/s) and expected to be valid accross several dialects of Spanish (Agerri et al., 2014).

The POS tags are assigned by using the IXA model for Maximum Entropy POS tagging, and reported to provide 98.88% accuracy (Agerri et al., 2014). Lemmatization uses the morfologik-stemming toolkit, based on FSA for a lower memory footprint (up to 10% the size of a full-fledged dictionary).

Named entities (PERSON, LOCATION, ORGANIZATION and MISC) are annotated using the Maximum Entropy model of IXA trained on the CONLL 2002 dataset and tags.

Finally, the IXA pipeline provides a module for constituency parsing trained on the (Iberian) Spanish section of the AnCora corpus.

3.4 Dutch

For Dutch, we have been able to integrate language-specific processors for tokenization, sentence splitting, lemmatization, named entity recognition, dependency tree, and part-of-speech tagging.

To provide basic levels of linguistic processing, we rely on xTas—a text analysis suite for English and Dutch (de Rooij et al., 2012). The suite has been designed to integrate various NLP components developed by researchers at University of Amsterdam and is extendable to work with components from other parties. xTas is designed to leverage distributed environments for speeding up computationally demanding NLP tasks and is available as a REST web service. xTas and instructions on how to install it and set it up can be found at `http://xtas.net`.

Most of the Dutch processors at xTas come from Frog, a third-party module. Frog, formerly known as Tadpole, is an integration of memory-based NLP modules developed for Dutch (van den Bosch et al., 2007). All NLP modules are based on Timbl, the Tilburg memory-based learning software package. Most modules were created in the 1990s at the ILK Research Group (Tilburg University, the Netherlands) and the CLiPS Research Centre (University of Antwerp, Belgium). Over the years they have been integrated into a single text processing tool. More recently, a dependency parser, a base phrase chunker, and a named-entity recognizer module were added.

For dependency parsing, xTas uses Alpino, a third-party module.[4] Annotation typically starts with parsing a sentence with the Alpino parser, a wide coverage parser of Dutch text. The number of parses that is generated is reduced through interactive lexical analysis and constituent marking. The selection of the best parse is done efficiently with the parse selection tool.

4 Conclusion and Future/Ongoing work

In this paper, we have presented the LiMoSINe pipeline—a platform supporting state-of-the-art NLP technology for English, Italian, Spanish and Dutch. Based on UIMA, it allows for efficient parallel processing of large volumes of text. The pipeline is distributed in two versions: the client application is oriented to potential users that need high-performance standard tools at a zero engineering cost. The local version, on the contrary, requires some installation and configuration effort, but in return it offers a great flexibility in implementing and integrating user-specific modules.

Since the beginning of the LiMoSINe project, the platform has been used for providing robust preprocessing for a variety of high-level tasks. Thus, we have recently shown how structural representations, extracted with our pipeline, improve multilingual opinion mining on YouTube (Severyn et al., 2015) or crossword puzzle resolution (Barlacchi et al., 2014).

The pipeline has been adopted by other parties, most importantly by the joint QCRI and MIT project IYAS (Interactive sYstem for Answer Selection). IYAS focuses on Question Answering, showing that representations, based on linguistic preprocessing, significantly outperform more shallow methods (Tymoshenko and Moschitti, 2015; Tymoshenko et al., 2014).

As part of the LiMoSINe project, we have created the LiMoSINe Common Corpus: a large collection of documents downloaded from different web resources

[4] `http://www.let.rug.nl/vannoord/alp/Alpino/`

in any of the four addressed languages. These data were annotated automatically. We illustrate the processing capabilities of our pipeline on the Spanish part of the corpus (EsLCC). To this end, we developed a UIMA Collection Processing Engine (CPE). Once the EsLCC was downloaded it was first tidied up with Apache Tika. The pipeline was then applied to clean text. It was capable of processing the approximately 103K EsLCC documents in a little bit more than 24 hours on an Ubuntu 14.04 with 16GB of RAM, on an Intel i7@3.50GHz $\times$ 8 core box.

Currently, the QCRI team is working on extending the pipeline, integrating various preprocessing modules for Arabic.

5 Acknowledgements

This work has been supported by the EU Projects FP7 LiMoSINe and H2020 5G-CogNet.

References

R. Agerri, J. Bermudez, and G. Rigau. 2014. IXA pipeline: Efficient and ready to use multilingual NLP tools. In LREC.

G. Barlacchi, M. Nicosia, and A. Moschitti. 2014. Learning to rank answer candidates for automatic resolution of crossword puzzles. In CoNLL-2014.

S. M. Bernaola Biggio, C. Giuliano, M. Poesio, Y. Versley, O. Uryupina, and R. Zanoli. 2009. Local entity detection and recognition task. In EvalIta-2009.

E. Breck, Y. Choi, and C. Cardie. 2007. Identifying expressions of opinion in context. In IJCAI.

S. Broscheit, M. Poesio, S.P. Ponzetto, K.J. Rodriguez, L. Romano, O. Uryupina, and Y. Versley. 2010. BART: A multilingual anaphora resolution system. In SemEval.

H. Cunningham, D. Maynard, K. Bontcheva, V. Tablan, N. Aswani, I. Roberts, G. Gorrell, A. Funk, A. Roberts, D. Damljanovic, T. Heitz, M.A. Greenwood, H. Saggion, J. Petrak, Y. Li, and W. Peters. 2011. Text Processing with GATE (Version 6).

O. de Rooij, J. van Gorp, and Maarten de Rijke. 2012. xtas: Text analysis in a timely manner. In DIR 2012: 12th Dutch-Belgian Information Retrieval Workshop.

D.A. Ferrucci, E.W. Brown, J. Chu-Carroll, J. Fan, D. Gondek, A. Kalyanpur, A. Lally, J.W. Murdock, E. Nyberg, J.M. Prager, N. Schlaefer, and Ch.A. Welty. 2010. Building Watson: An overview of the DeepQA project. AI Magazine, pages 59–79.

D. Ghosh and S. Muresan. 2012. Relation classification using entity sequence kernels. In COLING 2012, pages 391–400.

C. Giuliano, A. Lavelli, and L. Romano. 2007. Relation extraction and the influence of automatic named-entity recognition. ACM Trans. Speech Lang. Process., 5(1).

R. Johansson and A. Moschitti. 2011. Extracting opinion expressions and their polarities – exploration of pipelines and joint models. In ACL.

R. Johansson and P. Nugues. 2008. Dependency-based semantic role labeling of PropBank. In EMNLP.

C.D. Manning, M. Surdeanu, J. Bauer, J. Finkel, S.J. Bethard, and D. McClosky. 2014. The Stanford CoreNLP natural language processing toolkit. In ACL System Demonstrations.

E. Meij, W. Weerkamp, and M. de Rijke. 2012. Adding semantics to microblog posts. In WSDM.

A. Moschitti. 2006. Efficient convolution kernels for dependency and constituent syntactic trees. Machine Learning: ECML 2006.

T.H. Nguyen, B. Plank, and R. Grishman. 2015. Semantic representations for domain adaptation: A case study on the tree kernel-based method for relation extraction. In ACL.

L. Padró and E. Stanilovsky. 2012. Freeling 3.0: Towards wider multilinguality. In LREC, Istanbul, Turkey, May. ELRA.

E. Pianta, Ch. Girardi, and R. Zanoli. 2008. The TextPro tool suite. In LREC.

B. Plank and A. Moschitti. 2013. Embedding semantic similarity in tree kernels for domain adaptation of relation extraction. In ACL.

D. Roth and W. Yih. 2004. A linear programming formulation for global inference in natural language tasks. In CoNLL.

Aliaksei Severyn, Alessandro Moschitti, Olga Uryupina, and Barbara Plank. 2015. Multilingual opinion mining on YouTube. Information Processing and Management.

K. Tymoshenko and A. Moschitti. 2015. Assessing the impact of syntactic and semantic structures for answer passages reranking. In ACM CIKM.

K. Tymoshenko, A. Moschitti, and A. Severyn. 2014. Encoding semantic resources in syntactic structures for passage reranking. In EACL.

O. Uryupina and A. Moschitti. 2013. Multilingual mention detection for coreference resolution. In IJCNLP.

O. Uryupina and A. Moschitti. 2014. Coreference resolution for Italian: Assessing the impact of linguistic components. In CLIC-it.

O. Uryupina, S. Saha, A. Ekbal, and M. Poesio. 2011. Multi-metric optimization for coreference: The UniTN / IITP / Essex submission to the 2011 CONLL shared task. In CoNLL.

O. Uryupina, A. Moschitti, and M. Poesio. 2012. BART goes multilingual: The UniTN / Essex submission to the CoNLL-2012 Shared Task. In CoNLL.

A. van den Bosch, B. Busser, S. Canisius, and W. Daelemans. 2007. An efficient memory-based morphosyntactic tagger and parser for Dutch. In CLIN. Leuven, Belgium.

Y. Versley, S.P. Ponzetto, M. Poesio, V. Eidelman, A. Jern, J. Smith, X. Yang, and A. Moschitti. 2008. BART: a modular toolkit for coreference resolution. In ACL.