Adapting the TANL tool suite to Universal Dependencies

Giuseppe Attardi, Maria Simi
Dipartimento di Informatica
Largo B. Pontecorvo 3, 56127 Pisa, Italy
E-mail: attardi@di.unipi.it, simi@di.unipi.it

Abstract

TANL is a suite of tools for text analytics based on the software architecture paradigm of data driven pipelines. The strategies for upgrading TANL to the use of Universal Dependencies range from a minimalistic approach consisting of introducing pre/post-processing steps into the native pipeline to revising the whole pipeline. We explore the issue in the context of the Italian Treebank, considering both the efforts involved, how to avoid losing linguistically relevant information and the loss of accuracy in the process. In particular we compare different strategies for parsing and discuss the implications of simplifying the pipeline when detailed part-of-speech and morphological annotations are not available, as it is the case for less resourceful languages. The experiments are relative to the Italian linguistic pipeline, but the use of different parsers in our evaluations and the avoidance of language specific tagging make the results general enough to be useful in helping the transition to UD for other languages.

Keywords: Linguistic pipeline, dependency parsing, Universal Dependencies

1. Introduction

TANL (Natural Language Text Analytics) is a suite of tools for text analytics based on the software architecture paradigm of data pipelines. TANL pipelines are data driven, i.e. each stage pulls data from the preceding stage and transforms them for use by the next stage. Since data is processed as soon as it becomes available, processing delay is minimized improving data throughput. The processing modules can be written in either C++, Python or PHP and can be combined using few lines of scripts to produce full NLP applications. TANL provides a set of modules, ranging from tokenization to POS tagging, from parsing to NE recognition. A TANL pipeline can be processed in parallel on a cluster of computers by means of a map/reduce streaming framework.

The architecture of the TANL pipeline, its modules and some sample applications where presented at LREC 2010 (Attardi et al. 2010).

Given that most of the tools are based on machine learning techniques, the pipeline can be used to process texts in several languages: English, Italian, Spanish.

For Italian in particular we have been contributing to the development of training resources for the various tools: an Italian lexicon, a resource for training the POS tagger, and, along the years, several treebanks of increasing size for training dependency parsers: ISST-CoNLL (released for CoNLL-2007), MIDT (the Merged Italian Dependency Treebank), ISDT (the Italian version of the Stanford Dependencies).

Our team joined since the beginning the Universal Dependencies (UD) project, a recent initiative to develop cross-linguistically consistent treebank annotations for several languages, that aims to facilitate multilingual parser development and cross-language parsing (Nivre, 2015). As part of this effort, we released in May 2015 the first version of the Italian treebank adhering to the UD guidelines: UD-it 1.1.

In this paper we analyze how to adapt the TANL linguistic pipeline in order to fully support the Universal Dependencies style of text annotation.

Given that specific guidelines are being developed for any level of analysis (data format, sentence splitting, tokenization, POS tagging, morphology, parsing), full support of the new standard has an impact on the whole chain of linguistic tools and training resources, as well as their use in a cascade.

In particular for parsing we need to carefully evaluate two alternative strategies: using native parsers and converters or using parsers directly trained on the UD resources. For this issue we will report the results of a comparative experiment for Italian.

The benefits in return from this endeavor are the ability to support language analytics applications for a wider range of languages: the number of available UD treebanks is indeed growing, counting to 33 at the time of this writing.

2. The Tanl Pipeline

The following modules are currently available as part of the TANL pipeline:

- Sentence Splitter: splits the text into sentences
- Word Tokenizer: deals with the segmentation of a sentence into tokens
- Word Aggregator: combines polyrematic expressions of common use into a single token (e.g. “a meno che” becomes “a_meno_che”)
- POS Tagger: enriches tokens within a sentence with attributes representing the POS and lemma
- Morph Splitter: splits the POS of each token into separate POS and morpho-features and also splits clitic forms into two or more tokens (e.g. the verb “avercelo” becomes “aver- ce- lo”)
- Parser: parses sentences producing dependency parse trees. The module uses DeSR, a state-of-the-art multilingual dependency parser based on the transition-based paradigm (Attardi, 2006; Attardi et

al., 2009a and 2009b).

- Sequence tagger: a generic tagger for sequences based on Conditional Markov Models, applicable to Named Entity tagging and SuperSense tagging.
- Time annotation tagger based on HeidelTime (Strötgen and Gertz, 2013).

The current pipeline for Italian is based on TANL POS tags, which are the same ones used in MIDT and ISDT and can therefore run smoothly with those treebanks.

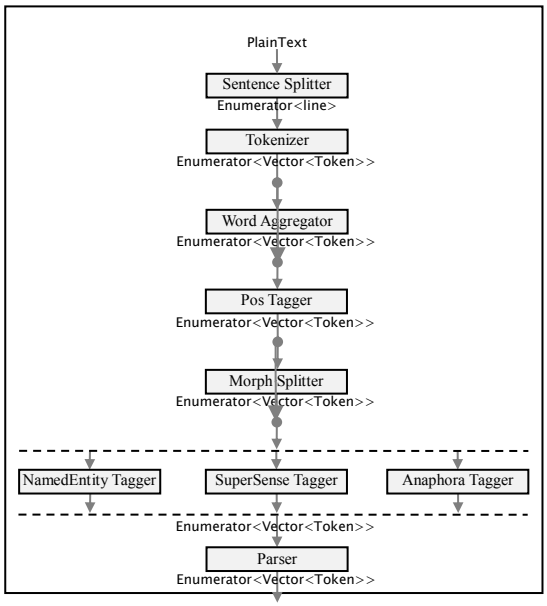

Figure 1. The TANL pipeline

3. Universal dependencies

Universal Dependencies are an attempt at standardization of syntactic annotations that provide detailed guidelines for all levels of linguistic analysis[1]. In particular:

- data encoding: UTF-8;
- data format: an extension of the tab separated CoNLL-X format;
- sentence splitting: only one root per sentence;
- tokenization: one syntactic word per token; multiword expressions span several tokens; no empty tokens or traces are allowed;
- PoS tags and morphological features are to be taken from a fixed inventory (Zeman, 2008); a column with language specific tags is also allowed.
- dependencies: categories from a fixed inventory along with careful guidelines for their use.

3.1 Italian UD

The core of the UD-it Italian treebank is the result of a conversion from the ISDT (Italian Stanford Dependency Treebank), released for the shared task on dependency parsing of Evalita-2014 (Bosco et al., 2013 and 2014). ISDT is a resource annotated according to the Stanford dependencies scheme (de Marneffe et al. 2008, 2013a, 2013b): it was obtained through a semi-automatic conversion process starting from an extended version of MIDT, the Merged Italian Dependency Treebank (Bosco, Montemagni, Simi, 2012).

MIDT in turn was obtained by merging two existing Italian treebanks, with different annotation schemes: TUT, the Turin University Treebank (Bosco et al. 2000), and ISST-TANL, first released as ISST-CoNLL for CoNLL-2007 (Montemagni and Simi, 2007). The ISDT corpus consists of 97,500 tokens derived from the TUT and 81,000 tokens derived from the ISST-TANL. Moreover a gold test dataset of 9,442 tokens was produced for the Evalita 2014 shared task.

UD-it is a larger resource including the previous texts, a new corpus of questions, and data obtained from ParTUT (the Multilingual Turin University Treebank) for a total of 316,660 tokens (12,880 sentences). For release 1.2, UD-it was randomly split into train, development and test data sets. Both development and test include 489 sentences each (~12,600 tokens).

3.2 Novelties introduced by UD

Universal Dependencies can be seen as an evolution of the Stanford Dependencies into a multi- language framework and introduce significant novelties. Affecting the conversion from MIDT to UD, is for example the decision that articulated preposition are to be split into their components (article and preposition), coherently to what was already done (in MIDT and ISDT) for clitics.

At the level of dependencies structure, UD introduces two major changes (deMarneffe et al., 2014), concerning: (i) the treatment of copulas and (ii) the treatment of prepositions with case marking.

Stanford Dependencies already recommended a treatment of the copula "to be" ("*essere*" in Italian) as dependent of a lexical predicate. In UD this becomes prescriptive and is motivated by the fact that many languages often lack an overt copula. This entails that the predicate complement is linked directly to its subject argument and the copula becomes a dependent of the predicate.

The second major change is the decision to fully adhere to the design principle of directly linking content words, and to abandon treating prepositions as a mediator between a modified word and its object: prepositions (but also other case-marking elements) are treated as dependents of the noun with specific *case* or *mark* labels.

The combined effect of these two decisions leads to parse trees with substantially different structure from those of MIDT and ISDT, as illustrated in figures 2 and 3.

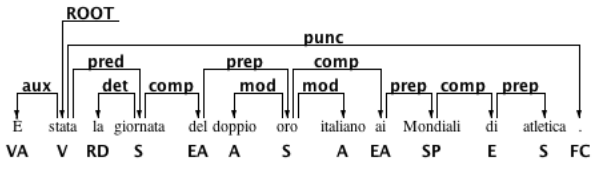

Figure 2. Example parse tree in MIDT

[1] https://universaldependencies.github.io/docs/

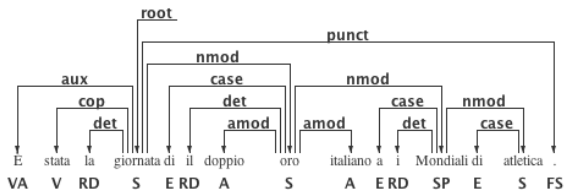

Figure 3. Example parse tree in UD

Moreover UD introduces an extension of the classical CoNLL-X tab separated format, called CoNLL-U. The main difference is the introduction of a notation for representing aggregated words (e.g. verbs with clitics or articulated prepositions), which are split into their constituents. The representation allows preserving the aggregated form by using a separate line, which is numbered with the range of the ID's of the individual tokenized constituents. The following is an example from the guidelines: "*vámonos al mar*" [let's go to the sea]:

```
1-2     vámonos     _
1       vamos       ir
2       nos         nosotros
3-4     al          _
3       a           a
4       el          el
5       mar         mar
```

4. Pipeline Enhancements for UD

Relevant research questions in adapting the TANL linguistic pipeline are:

(i) Which extensions to the pipeline are most appropriate for dealing with the new CoNLL-U format? Should we create a new pipeline accommodating the CoNLL-U format natively or just add pre-processing and post-processing steps for converting to/from CoNLL-U?

(ii) Given that UD-it is obtained by conversion from an enhanced version of MIDT, should we use a MIDT trained parser and convert to UD afterwards or use an UD trained parser directly?

(iii) Can we avoid any loss of linguistic information and loss of accuracy in the process?

(iv) Can rich POS tags, which are language specific, be dispensed by using word embedding features or word clusters?

4.1 POS tagging and morphology

The Universal tag set used in the UD (Zeman, 2008) consists of 17 categories and is not as rich as the Italian tag set, which consists of 37 categories.

Moreover, for the purpose of the pipeline, the TANL POS tags are combined with morphological features making up around 165 categories, which are dealt by a single efficient and accurate POS and morphology tagger. Fortunately, this level of detail can be retained in the language specific XPOSTAG column of the CoNLL-U format.

Writing a converter to produce the UPOSTAG column with universal tags it is a trivial task. Moreover the UD specifications introduce very detailed and potentially useful morphological features. Most of them can be obtained from the output of the TANL tagger using a different Morph Splitter; some however were not considered in the original design of the TANL tag set. Here are a few examples:

- clitic pronouns (when they do not play specific syntactic roles) are tagged as PC; UD specifications require a finer grained distinction in order to identify reflexive and impersonal pronouns and encode them as morphological features;
- Similarly, the UD specifications for morphology contemplate specific features for superlatives and comparative adjectives or adverbs.

The support for these additional lexical features would entail a revision of the lexicon and of the training resource for the tagger.

While POS tags and morphological features are linguistically relevant, they might not be essential for the purpose of parsing. We will explore later whether they could be replaced by features extracted from distributional word representations, i.e. word embeddings, which can be created by unsupervised learning methods and thus can be easily produced for any language.

4.2 Dealing with CoNLL-U

In the CoNLL-U numbering scheme, multiword tokens are included in separate lines indexed with integer ranges corresponding to component tokens. Lines with comments may also be introduced before sentences.

This extra information can be preserved in the TANL pipeline by means of the context field present in the representation of tokens, which can hold arbitrary attributes and can be nested and carried along the stages, even if not used.

5. Experiments

At LREC 2014 (Simi, Bosco, Montemagni, 2014) we compared the performance of a statistical parser (DeSR), trained on an augmented version of the Merged Italian Dependency Treebank (MIDT), with the results of the same parser directly trained on the Italian version of the Stanford Dependencies (ISDT). The experiments demonstrated that the accuracy of the DeSR parser trained on the reduced dependencies inventory of MIDT, followed by a conversion to ISDT, was slightly higher than the performance of the parser directly trained on ISDT. "*Less is more*" was our conclusion.

The starting point of our investigation is the observation that the order in which the different tools are used may have an impact in the accuracy of the final result. For this reason, we wanted to evaluate different strategies for parsing, before engaging in the revision of the linguistic pipeline. In the experiments, three different pipeline strategies were compared according to the accuracy of the final output. In the following we describe the experimental setup, the parsers used in the evaluation, the pipeline strategies and the results obtained.

5.1 Experimental setup

In the experiments we used as a benchmark the UD-it 1.2 version of the Italian Treebank and its official split in training, development and test.

An issue that we did not consider so far are revisions to the first stages of the pipeline, namely tokenization, sentence splitting, POS tagging and the generation of morphological features. For one thing, sentence splitting and tokenization are already compliant with the UD specifications; POS tagging could be revised to produce finer grained morphological features, but the impact on the final performance is expected to be minimal. This is confirmed by the experiments presented in section 6.

Given that the first stages (up to the morphological analysis) are fixed, we are still left with alternatives on how to perform dependency parsing. The strategies range from a minimalistic approach, consisting of introducing post-processing steps after the native pipeline, to revising and adapting the pipeline to the new standard before parsing.

In order to support automatic conversion into UD, the MIDT Treebank was extended with extra information (we call the enhanced version MIDT+). In particular: additional categories for *vocative*, *discourse*, and appositions (*appos*) had to be manually introduced[2]; similarly, open clausal complements had to be annotated manually in order to account for the subtle *xcomp*/*ccomp* distinction.

The tools we developed or used for the experiments are the following:

1. *Morph Splitter*: from a rich TANL POS tag which includes morphological features, it produces the CPOSTAG, POSTAG and MFEATS columns of the CoNLL-X format. This is a native tool in the TANL pipeline.
2. MIDT+toUD: a converter from MIDT+ to UD in CoNLL-X format; it takes care of converting POS, morphology and dependencies and producing the CPOSTAG, POSTAG, MFEATS and DEPREL fields according to the UD specs.
3. toCoNLL-U: a converter from CoNLL-X format to CoNLL-U format; it takes care of splitting articulated prepositions and of producing lines with index ranges for verbs with clitics and articulated prepositions.
4. *ConvertFeats*: converts only POS and morphological features into the UD standard.

Moreover we used three different state-of-the-art parsers in order to make our results more general.

5.2 Dependency parsers

In addition to DeSR, the fast transition based parser that we use in our pipeline, we repeated the experiments with two other state-of-the-art parsers, both graph-based: MATE (Bohnet, 2010; Bohnet and Kuhn 2012) and Turbo Parser (Martins et al., 2013).

[2] These extensions to MIDT are different from the ones proposed for the conversion into ISDT.

DeSR was chosen as a representative of transition-based parsers for two main reasons, besides our own interest in developing this technology: (i) it allowed us to experiment with different feature sets, by exploiting its declarative configuration mechanism; (ii) other parsers in this category, in particular MaltParser (Nivre et al.), were consistently reported to achieve inferior results in previous Evalita evaluation campaigns for Italian.

In our experiments we avoided resorting to parser combinations, which typically provide some small improvements in accuracy, since we are interested in a relative comparison of parser effectiveness and, besides, parser combinations may not be a viable solution for a pipeline where efficiency is important. We provide below a short description of the state-of-the-art parsers chosen for our experiments.

DeSR MLP is a transition-based parser that uses a Multi-Layer Perceptron algorithm (Attardi 2006, Attardi et al., 2009a and 2009b). We trained it on 300 hidden variables, with a learning rate of 0.01, and early stopping when validation accuracy reaches 99.5%. The feature model used in the experiments is detailed in (Attardi, Saletti, Simi 2015).

TurboParser is a graph-based parser that uses third-order feature models and a specialized accelerated dual decomposition algorithm for making non-projective parsing computationally feasible (Martins et al., 2013). TurboParser was used in configuration "full", enabling all third-order features.

Mate is a graph-based parser that uses passive aggressive perceptron and exploits reach features (Bohnet, 2010; Bohnet and Kuhn 2012). The only configurable parameter is the number of iterations (set to 25).

5.3 Pipeline strategies

The three pipelines we have experimented with share the following preliminary stages[3]:

1. Sentence splitting
2. Tokenization
3. POS Tagging

Moreover we have assumed gold data as a result of the first stages for the three pipelines. Therefore the final results are discounted of errors coming from sentence splitting, tokenization and POS tagging. As a consequence, the results can only be used for the sake of comparison but they do not mean to account for the performance of the whole analysis process. The experiments intend to measure instead different strategies according to the order in which the different tools are applied at the parsing level.

The tables show the LAS and UAS of the three parsers on the resulting parse tree with respect to the official gold test. The evaluation scores do not include punctuation. In the final results the development dataset has been included in the train dataset.

[3] Word aggregation is not performed.

5.3.1 Pipeline 1: parsers trained on MIDT+

In the first pipeline the conversion to UD was performed as post-processing step of the native MIDT pipeline. The parsers where trained on MIDT+, the test data were parsed with MIDT+ parsers and evaluated against a MIDT+ annotated gold standard. After conversion to UD in CoNLL-X format, the resulting test set was evaluated again. A final step takes care of producing the CoNLL-U format and evaluating the output against the gold CoNLL-U data test. The steps involved in the experiment (following steps 1-3) are:

4. Parsing with parsers trained on MIDT+
5. MIDT+toUD conversion, in CoNLL-X format.
6. Conversion to CoNLL-U format.

Table 1 shows the results of the output of the corresponding three steps of this process in terms of Labeled Attachment Score (LAS) and Unlabeled Attachment Score (UAS).

	MIDT+ parser		**MIDT+toUD**		**toCoNLL-U**	
	LAS	**UAS**	**LAS**	**UAS**	**LAS**	**UAS**
DeSR-MLP[4]	87.55	90.72	87.90	90.38	89.03	91.32
Mate	**89.99**	**93.33**	**90.52**	**92.93**	**91.22**	**93.45**
TurboParser	88.78	92.27	89.07	91.76	89.85	92.34

Table 1. Results of Pipeline 1.

5.3.2 Pipeline 2: parsers trained on UD-it in CoNLL-X format

In the next experiment we trained the parsers on the same resources converted to the UD annotation scheme in CoNLL-X format, and converted the results to CoNLL-U afterwards, taking care of splitting articulated prepositions after parsing. The steps involved in the experiment are:

4. Parsing with parsers trained on UD-it in CoNLL-X format (before splitting articulated prepositions)
5. Conversion to CoNNL-U.

Table 2 shows the evaluation of the output of UD parsers and the test data obtained after the final conversion step.
The configuration of DeSR for UD was different from the case of the MIDT parser. The feature model is described in detail in (Attardi, Saletti, Simi 2015). For the experiments on the UD corpus, the base feature model was used with 28 additional 3rd order features. MATE and Turbo Parser where instead used in the same configuration.

	UD-it parser		**toCoNLL-U**	
	LAS	**UAS**	**LAS**	**UAS**
DeSR-MLP[5]	87.06	89.54	88.02	90.31
Mate	**89.86**	**92.32**	**90.60**	**92.88**
TurboParser	88.77	91.54	89.58	92.15

Table 2. Results of Pipeline 2.

[4] With configuration 160-m

[5] With configuration 3-g

5.3.3 Pipeline 3: parser trained on UD-it in CoNLL-U format

As a final experiment we trained the parsers on the final resources in CoNLL-U format, where articulated prepositions are split as well. The only step involved in the experiment is parsing with parsers trained on UD-it in CoNLL-U format, after splitting articulated prepositions.

	UD-it parser	
	LAS	**UAS**
DeSR-MLP-3g	87.74	90.11
Mate	**90.46**	**92.60**
TurboParser	89.40	92.07

Table 3. Results of pipeline 3

5.4 Discussion

These experiments seem to confirm our initial intuition that there is an advantage for dependency parsers in dealing with a simpler annotation scheme, such as MIDT, and convert to UD as a final step. The best results in fact, with all the parsers we tested, where achieved through the process described as Pipeline 1.
Looking in more detail at the data provided in Table 1, we observe that the LAS scores for the parsers trained on MIDT are not the highest, but the parsers seem to be able to predict attachments fairly well (as shown by the UAS scores); eventually, since the conversion step takes care of converting and normalizing labels, this leads to the most accurate parse trees, also in terms of LAS.
In comparison the results of parsers trained directly on UD-it resources in CoNLL-X format are worse both in terms of LAS and UAS (ref. Table 2). Not surprisingly the simple mechanical step of splitting articulated prepositions (carried out by toCoNLL-U) improves the result in both Pipeline 1 and 2.
Pipeline 3 performs better on average than Pipeline 2: the difference in the resource for training and testing is only the presence of additional dependencies that are easy to learn for a statistical parser, those connecting articles and prepositions to a noun (with dependency *det* and *case* respectively).

6. Dispensing with language specific tags

In the previous experiments we took the first stages of pipeline for granted, discounting the errors that may be produced in sentence splitting, tokenization and POS tagging. Instead of measuring the performance of the corresponding tools, we decided, in a different set of experiments, to evaluate whether POS and morphology could be avoided altogether. Since this linguistic information may not be available for some language, this can be seen as a way to generalize the results obtained for Italian.

6.1.1 Dropping POS tags and morphology

We started by measuring the contribution of Italian specific POS tags and morphology on parser accuracy by

getting rid of the fine-grained POS and morphology fields and retaining only the universal tags. We obtained the following results:

	UD-it parser	
	LAS	**UAS**
DeSR-MLP-3g	86.92	89.51
Mate	**89.53**	**91.57**
TurboParser	88.45	91.40

Table 4. Results without XPOSTAG and no morphology

Table 4 shows that parsing performance decreases for all the parsers but the effect is not as dramatic as could be expected (less than 1% decrease in LAS wrt results reported in Table 3). This is encouraging.

6.1.2 Using word clusters

We further explored whether one could replace the language specific POS with information obtained from word embeddings.

We trained word embeddings using *word2vec* (Mikolov et al., 2013) on the text extracted from the Italian Wikipedia using WikiExtractor[6]. We tested vectors of size 50, 100 and 200.

We then produced clusters from these vectors using the *dbscan* algorithm[7] with distance sizes of 0.8, 1.2 and 1.6. This produces sets of clusters of sizes 59, 575 and 975 respectively. We trained the parsers using the cluster number as a feature instead of the language specific POS, obtaining the following results:

		UD-it parser	
	clusters	**LAS**	**UAS**
DeSR-MLP	59	87.53	90.40
DeSR-MLP	575	87.40	90.06
DeSR-MLP	975	85.93	89.15
TurboParser	59	**89.50**	**92.22**
TurboParser	575	88.89	91.57
TurboParser	975	89.18	92.01

Table 5. Results using clusters instead of POS.

We are not reporting results with the Mate parser, since it cannot be configured for dealing with alternative features to POS tags and morphology, and using clusters instead of POS causes a major drop in accuracy.

By comparing the best results in Table 5 (obtained with fewer clusters) with results in Table 3, we can notice that they are comparable and the loss in accuracy involved in giving up language specific POS and morphology, is nearly recovered by information gathered by an unsupervised process of computing word clusters.

[6] https://github.com/attardi/wikiextractor

[7] http://scikit-learn.org/stable/modules/generated/sklearn.cluster.dbscan.html

7. Conclusions

Moving from the native annotation style of the treebank of one language to the common style of Universal Dependencies impacts the whole pipeline used to perform text analysis. The strategies for upgrading to the use of UD range from a minimalistic approach consisting of introducing pre/post-processing steps into the native pipeline or revising the whole pipeline.

We explored the issue in the context of the Italian Treebank, considering both the efforts involved, how to avoid losing linguistically relevant information and the loss of accuracy in the process.

The experiments were aimed at finding the best trade-off between the efforts involved in revising the TANL pipeline and accuracy of the resulting output. Efficiency was also considered as a side but important issue.

This evaluation may influence the decision of whether to maintain the Italian Treebank in the intermediate MIDT+ annotation style or to abandon it in favour of UD. In the first case, extending the treebank with new data should be done by producing and revising the MIDT+ annotations. On the one hand this scheme is simpler for human annotators and makes the more performant Pipeline 1 viable, but on the other hand it forces contributors to learn an additional annotation language and corresponding guidelines. MIDT served well its purpose of bridging two pre-existing Italian resources, but its design choices were mostly dictated by practical considerations rather than linguistically motivated. In the long run, as soon as new gold data becomes available for Italian annotated directly in UD, we will have to consider abandoning the MIDT resource and consider restructuring strategies as a pre-processing step of the UD parser, or find other ways to improve on parser technology.

Considering the three parsers we tested, we observe that graph based parsers consistently achieve higher accuracy. While for certain applications or for producing new gold resources one needs to use the most accurate parser that he can get, a transition-based parser still has an advantage in raw parsing speed and is competitive for large scale or online applications[8].

The experience reported here might be useful in helping the transition to UD for other languages, since similar issues will have to be dealt.

8. Acknowledgements

Simonetta Montemagni and Alessandro Lenci contributed to the development of TANL Pipeline. With Cristina Bosco, they are part of the team developing UD guidelines for Italian and the UD-it treebank.

9. References

Giuseppe Attardi. 2006. Experiments with a Multilanguage Non-Projective Dependency Parser, Proc. of the Tenth Conference on Natural Language

[8] Disregarding speed-ups due to multithreading, the parsing time to analyze the test set was around 16 seconds for DeSR, 47 seconds for Turbo Parser and nearly 3 minutes for Mate.

Learning, New York, (NY).

Giuseppe Attardi, Felice Dell'Orletta. 2009. Reverse Revision and Linear Tree Combination for Dependency Parsing. In: *Proc. of Human Language Technologies: The 2009 Annual Conference of the NAACL, Companion Volume: Short Papers*, 261–264. ACL, Stroudsburg, PA, USA.

Giuseppe Attardi, Felice Dell'Orletta, Maria Simi, Joseph Turian. 2009. Accurate Dependency Parsing with a Stacked Multilayer Perceptron. In: *Proc. of Workshop Evalita 2009*, ISBN 978-88-903581-1-1.

G. Attardi, S. Dei Rossi, M. Simi. The Tanl Pipeline. *Proc. of LREC Workshop on WSPP*, Malta, 2010.

G. Attardi, S. Saletti, M. Simi. Evolution of Italian Treebank and Dependency Parsing towards Universal Dependencies, Proceedings of CLIC-it 2015, Trento, Dec 2015.

Bernd Bohnet. 2010. Top accuracy and fast dependency parsing is not a contradiction. In Proc. of Coling 2010, pp. 89–97, Beijing, China. Coling 2010 Organizing Committee.

Bernd Bohnet and Jonas Kuhn. 2012. The Best of Both Worlds -- A Graph-based Completion Model for Transition-based Parsers. Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics (EACL), pages 77–87.

Cristina Bosco, Vincenzo Lombardo, Leonardo Lesmo, Daniela Vassallo. 2000. Building a treebank for Italian: a data-driven annotation schema. In Proceedings of LREC 2000, Athens, Greece.

Cristina Bosco, Simonetta Montemagni, Maria Simi. 2012. Harmonization and Merging of two Italian Dependency Treebanks, Workshop on Merging of Language Resources, in Proceedings of LREC 2012, Workshop on Language Resource Merging, Istanbul, May 2012, ELRA, pp. 23–30.

Cristina Bosco, Simonetta Montemagni, Maria Simi. 2013. Converting Italian Treebanks: Towards an Italian Stanford Dependency Treebank. In: *ACL Linguistic Annotation Workshop & Interoperability with Discourse*, Sofia, Bulgaria.

Andre Martins, Miguel Almeida, and Noah A. Smith. 2013. Turning on the turbo: Fast third-order non-projective turbo parsers. In: Proc. of the 51st Annual Meeting of the ACL (Volume 2: Short Papers), 617–622, Sofia, Bulgaria. ACL.

Cristina Bosco, Felice Dell'Orletta, Simonetta Montemagni, Manuela Sanguinetti, Maria Simi. 2014. The Evalita 2014 Dependency Parsing task, CLiC-it 2014 and EVALITA 2014 Proceedings, Pisa University Press, ISBN/EAN: 978-886741-472-7, 1–8.

Marie-Catherine de Marneffe and Christopher D. Manning. 2008. The Stanford typed dependencies representation. In COLING Workshop on Cross-framework and Cross-domain Parser Evaluation.

Marie-Catherine de Marneffe, Miriam Connor, Natalia Silveira, Bowman S. R., Timothy Dozat, Christopher D. Manning. 2013. More constructions, more genres: Extending Stanford Dependencies, Proc. of the Second International Conference on Dependency Linguistics (DepLing 2013), Prague, August 27–30, Charles University in Prague, Matfyzpress, Prague, 187–196.

Marie-Catherine de Marneffe and Christopher D. Manning. 2013. Stanford typed dependencies manual, September 2008, Revised for the Stanford Parser v. 3.3 in December 2013.

Marie-Catherine De Marneffe, Timothy Dozat, Natalia Silveira, Katri Haverinen, Filip Ginter, Joakim Nivre, Christopher D. Manning. 2014. Universal Stanford Dependencies: a Cross-Linguistic Typology. In: *Proc. LREC 2014*, Reykjavik, Iceland, ELRA.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. *Advances in Neural Information Processing Systems*.

Simonetta Montemagni, Maria Simi. 2007. The Italian dependency annotated corpus developed for the CoNLL–2007 shared task. Tech. Report, ILC–CNR.

Joakim Nivre. 2015. Towards a Universal Grammar for Natural Language Processing, CICLing (1) 2015: 3–16.

Joakim Nivre, Johan Hall, and Jens Nilsson. 2006. Maltparser: a data-driven parser-generator for dependency parsing. In Proceedings of LREC-2006, volume 2216–2219.

Maria Simi, Cristina Bosco, Simonetta Montemagni. 2008. Less is More? Towards a Reduced Inventory of Categories for Training a Parser for the Italian Stanford Dependencies. In: *Proc. LREC 2014*, 26–31, May, Reykjavik, Iceland, ELRA.

Jannik Strötgen and Michael Gertz. 2013. Multilingual and Cross-domain Temporal Tagging. Language Resources and Evaluation, number 1, 269–298. Springer.

Daniel Zeman. 2008. Reusable Tagset Conversion Using Tagset Drivers. In *Proceedings of LREC 2008*.

10. Language Resource References

ISST-CoNLL: http://medialab.di.unipi.it/isst/
MIDT: http://medialab.di.unipi.it/wiki/MIDT
ISDT:http://medialab.di.unipi.it/Evalita2014/resources.html
TUT: http://www.di.unito.it/~tutreeb/
UD: https://lindat.mff.cuni.cz/repository/xmlui/handle/11234/1-1548