# ULISSE:
# an Unsupervised Algorithm for Detecting Reliable Dependency Parses

**Felice Dell'Orletta, Giulia Venturi and Simonetta Montemagni**
Istituto di Linguistica Computazionale "Antonio Zampolli" (ILC–CNR)
via G. Moruzzi, 1 – Pisa (Italy)
{felice.dellorletta,giulia.venturi,simonetta.montemagni}@ilc.cnr.it

## Abstract

In this paper we present ULISSE, an unsupervised linguistically–driven algorithm to select reliable parses from the output of a dependency parser. Different experiments were devised to show that the algorithm is robust enough to deal with the output of different parsers and with different languages, as well as to be used across different domains. In all cases, ULISSE appears to outperform the baseline algorithms.

## 1 Introduction

While the accuracy of state–of–the–art parsers is increasing more and more, this is still not enough for their output to be used in practical NLP–based applications. In fact, when applied to real–world texts (e.g. the web or domain–specific corpora such as bio–medical literature, legal texts, etc.) their accuracy decreases significantly. This is a real problem since it is broadly acknowledged that applications such as Information Extraction, Question Answering, Machine Translation, and so on can benefit significantly from exploiting the output of a syntactic parser. To overcome this problem, over the last few years a growing interest has been shown in assessing the reliability of automatically produced parses: the selection of high quality parses represents nowadays a key and challenging issue. The number of studies devoted to detecting reliable parses from the output of a syntactic parser is spreading. They mainly differ with respect to the kind of selection algorithm they exploit. Depending on whether training data, machine learning classifiers or external parsers are exploited, existing algorithms can be classified into *i)* supervised–based, *ii)* ensemble–based and *iii)* unsupervised–based methods.

The first is the case of the construction of a machine learning classifier to predict the reliability of parses on the basis of different feature types. Yates et al. (2006) exploited semantic features derived from the web to create a statistical model to detect unreliable parses produced by a constituency parser. Kawahara and Uchimoto (2008) relied on features derived from the output of a supervised dependency parser (e.g. dependency lengths, number of unknown words, number of coordinated conjunctions, etc.), whereas Ravi et al. (2008) exploited an external constituency parser to extract text–based features (e.g. sentence length, unknown words, etc.) as well as syntactic features to develop a supervised predictor of the target parser accuracy. The approaches proposed by Reichart and Rappoport (2007a) and Sagae and Tsujii (2007) can be classified as ensemble–based methods. Both select high quality parses by computing the level of agreement among different parser outputs: wheras the former uses several versions of a constituency parser, each trained on a different sample from the training data, the latter uses the parses produced by different dependency parsing algorithms trained on the same data. However, a widely acknowledged problem of both supervised–based and ensemble–based methods is that they are dramatically influenced by a) the selection of the training data and b) the accuracy and the typology of errors of the used parser.

To our knowledge, Reichart and Rappoport (2009a) are the first to address the task of high qual-



*Proceedings of the Fifteenth Conference on Computational Natural Language Learning*, pages 115–124,
Portland, Oregon, USA, 23–24 June 2011. 

ity parse selection by resorting to an unsupervised–based method. The underlying idea is that syntactic structures that are frequently created by a parser are more likely to be correct than structures produced less frequently. For this purpose, their PUPA (*POS–based Unsupervised Parse Assessment Algorithm*) uses statistics about POS tag sequences of parsed sentences produced by an unsupervised constituency parser.

In this paper, we address this unsupervised scenario with two main novelties: unlike Reichart and Rappoport (2009a), a) we address the reliable parses selection task using an unsupervised method in a supervised parsing scenario, and b) we operate on dependency–based representations. Similarly to Reichart and Rappoport (2009a) we exploit text internal statistics: but whereas they rely on features that are closely related to constituency representations, we use linguistic features which are dependency–motivated. The proposed algorithm has been evaluated for selecting reliable parses from English and Italian corpora; to our knowledge, this is the first time that such a task has been applied to a less resourced language such as Italian. The paper is organised as follows: in Section 2 we illustrate the ULISSE algorithm; sections 3 and 4 are devoted to the used parsers and baselines. Section 5 describes the experiments and discusses achieved results.

## 2 The ULISSE Algorithm

The ULISSE (*Unsupervised LInguiStically–driven Selection of dEpendency parses*) algorithm takes as input a set of parsed sentences and it assigns to each dependency tree a score quantifying its reliability. It operates in two different steps: 1) it collects statistics about a set of linguistically–motivated features extracted from a corpus of parsed sentences; 2) it calculates a quality (or reliability) score for each analyzed sentence using the feature statistics extracted from the whole corpus.

### 2.1 Selection of features

The features exploited by ULISSE are all linguistically motivated and rely on the dependency tree structure. Different criteria guided their selection. First, as pointed out in Roark et al. (2007), we needed features which could be reliably identified within the automatic output of a parser. Second, we focused on dependency structures that are widely agreed in the literature a) to reflect sentences' syntactic and thus parsing complexity and b) to impose a high cognitive load on the parsing of a complete sentence.

Here follows the list of features used in the experiments reported in this paper, which turned out to be the most effective ones for the task at hand.

**Parse tree depth**: this feature is a reliable indicator of sentence complexity due to the fact that, with sentences of approximately the same length, parse tree depth can be indicative of increased sentence complexity (Yngve, 1960; Frazier, 1985; Gibson, 1998; Nenkova, 2010).

**Depth of embedded complement 'chains'**: this feature is a subtype of the previous one, focusing on the depth of chains of embedded complements, either prepositional complements or nominal and adjectival modifiers. Long chains of embedded complements make the syntactic structure more complex and their analysis much more difficult.

**Arity of verbal predicates**: this feature refers to the number of dependency links sharing the same verbal head. Here, there is no obvious relation between the number of dependents and sentence complexity: both a small number and a high number of dependents can make the sentence processing quite complex, although for different reasons (elliptical constructions in the former case, a high number of modifiers in the latter).

**Verbal roots**: this feature counts the number of verbal roots with respect to number of all sentence roots in the target corpus.

**Subordinate vs main clauses**: subordination is generally considered to be an index of structural complexity in language. Two distinct features are considered for monitoring this aspect: one measuring the ratio between main and subordinate clauses and the other one focusing on the relative ordering of subordinate clauses with respect to the main clause. It is a widely acknowledged fact that highly complex sentences contain deeply embedded subordinate clauses; however, subordinate clauses are easier to process if they occur in post–verbal rather than in pre–verbal position (Miller, 1998).

**Length of dependency links**: McDonald and Nivre (2007) report that statistical parsers have a drop in

accuracy when analysing long distance dependencies. This is in line with Lin (1996) and Gibson (1998) who claim that the syntactic complexity of sentences can be predicted with measures based on the length of dependency links, given the memory overhead of very long distance dependencies. Here, the dependency length is measured in terms of the words occurring between the syntactic head and the dependent.

**Dependency link plausibility** (henceforth, ArcPOSFeat): this feature is used to calculate the plausibility of a dependency link given the part–of–speech of the dependent and the head, by also considering the PoS of the head father and the dependency linking the two.

### 2.2 Computation Score

The quality score (henceforth, $QS$) of parsed sentences results from a combination of the weights associated with the monitored features. ULISSE is modular and can use several weights combination strategies, which may be customised with respect to the specific task exploiting the output of ULISSE.

For this study, $QS$ is computed as a simple product of the individual feature weights. This follows from the necessity to recognize high quality parses within the input set of parsed sentences: the product combination strategy is able to discard low quality parse trees even in presence of just one low weight feature. Therefore, $QS$ for each sentence $i$ in the set of input parsed sentences $I$ is $QS(S_i) = \prod_{y=1}^{n} Weight(S_i, f_y)$, where $S_i$ is the i–th sentence of $I$, $n$ is the total number of selected features and $Weight(S_i, f_y)$ is the computed weight for the y–th feature.

Selected features can be divided into two classes, depending on whether they are computed with respect to each sentence and averaged over all sentences in the target corpus (global features), or they are computed with respect to individual dependency links and averaged over all of them (local features). The latter is the case of the ArcPOSFeat feature, whereas the all other ones represent global features.

For the global features, the $Weight(S_i, f_y)$ is defined as:

$$Weight(S_i, f_y) = \frac{F(V(f_y), range(L(S_i), r))}{|range(L(S_i), r)|}, \quad (1)$$

where $V(f_y)$ is the value of the y–th feature (extracted from $S_i$), $L(S_i)$ is the length of the sentence $S_i$, $range(L(S_i), r)$ defines a range covering values from $L(S_i) - r$ and $L(S_i) + r$, $F(V(f_y), range(L(S_i), r))$ is the frequency of $V(f_y)$ in all sentences in $I$ that has a value of length in $range(L(S_i), r^1)$ and $|range(L(S_i), r)|$ is the total number of sentences in $I$ with length in $range(L(S_i), r)$. For what concerns the local feature ArcPOSFeat, ULISSE assigns a weight for each arc in $S_i$: in principle different strategies can be used to compute a unique weight for this feature for $S_i$. Here, the sentence weight for the feature *ArcPOSFeat* is computed as the minimum weight among the weights of all arcs of $S_i$. Therefore, $Weight(S_i, ArcPOSFeat) = min\{weight((Pd, Ph, t)), \forall (Pd, Ph, t) \in S_i\}$, where the triple $(Pd, Ph, t)$ is an arc in $S_i$ in which $Pd$ is the POS of the dependent, $Ph$ is the POS of the syntactic head and $t$ is the type of the dependency relation and $weight((Pd, Ph, t))$ is the weight of the specific arc $(Pd, Ph, t)$. The individual arc weight is computed as follows:

$$\begin{aligned} weight((Pd, Ph, t)) = & \frac{F((Pd, Ph, t))}{F((Pd, X, t))} \cdot \\ & \cdot \frac{F((Pd, Ph, t))}{F((X, Ph, t))} \cdot \\ & \cdot \frac{F(((Pd, Ph, t)(Ph, Ph2, t2)))}{F((Pd, Ph, t))} \cdot \\ & \cdot \frac{F(((Pd, Ph, t)(Ph, Ph2, t2)))}{F((Ph, Ph2, t2))} \cdot \\ & \cdot \frac{F(((Pd, Ph, t)(Ph, Ph2, t2)))}{F((((Pd, X, t))(X, Ph2, t2)))}, \end{aligned}$$

where $F(x)$ is the frequency of $x$ in $I$, $X$ is a variable and $(arc1\ arc2)$ represent two consecutive arcs in the tree.

## 3 The Parsers

ULISSE was tested against the output of two really different data–driven parsers: the first–order Maximum Spanning Tree (MST) parser (McDonald et al., 2006) and the DeSR parser (Attardi, 2006) using Support Vector Machine as learning algorithm. The

[1] We set r=0 in the in–domain experiments and r=2 in the out–of–domain experiment reported in Sec 5.3.

former is a graph–based parser (following the so–called "all–pairs" approach Buchholz et al. (2006)) where every possible arc is considered in the construction of the optimal parse tree and where dependency parsing is represented as the search for a maximum spanning tree in a directed graph. The latter is a Shift–Reduce parser (following a "stepwise" approach, Buchholz et al. (2006)), where the parser is trained and learns the sequence of parsing actions required to build the parse tree.

Although both parser models show a similar accuracy, McDonald and Nivre (2007) demonstrate that the two types of models exhibit different behaviors. Their analysis exemplifies how different the two parsers behave when their accuracies are compared with regard to some linguistic features of the analyzed sentences. To mention only a few, the Shift–Reduce parser tends to perform better on shorter sentences, while the MST parser guarantees a higher accuracy in identifying long distance dependencies. As regards the identification of dependency types, the MST parser shows a better ability to identify the dependents of the sentences' roots whereas the Shift–Reduce tends to better recognize specific relations (e.g. Subject and Object).

McDonald and Nivre (2007) describe how the systems' behavioral differences are due to the different parsing algorithms implemented by the Shift–Reduce and the MST parsing models. The Shift Reduce parser constructs a dependency tree by performing a sequence of parser actions or transitions through a greedy parsing strategy. As a result of this parsing procedure, a Shift Reduce parser creates shorter arcs before longer arcs. The latter could be the reason for the lower accuracy in identifying longer arcs when compared to the MST parser. This also influences a lower level of accuracy in the analysis of longer sentences that usually contain longer arcs than shorter sentences. The MST parser's ability to analyze both short and long arcs is invariant as it employs a graph-based parsing method where every possible arc is considered in the construction of the dependency tree.

## 4 The Baselines

Three different increasingly complex baseline models were used to evaluate the performance of ULISSE.

The first baseline is constituted by a *Random Selection* (*RS*) of sentences from the test sets. This baseline is calculated in terms of the scores of the parser systems on the test set.

The second baseline is represented by the *Sentence Length* (*SL*), starting from the assumption, demonstrated by McDonald and Nivre (2007), that long sentences are harder to analyse using statistical dependency parsers than short ones. This is a strong unsupervised baseline based on raw text features, ranking the parser results from the shortest sentence to the longest one.

The third and most advanced baseline, exploiting parse features, is the PUPA algorithm (Reichart and Rappoport, 2007a). PUPA uses a set of parsed sentences to compute the statistics on which its scores are based. The PUPA algorithm operates on a constituency based representation and collects statistics about the POS tags of the words in the yield of the constituent and of the words in the yields of neighboring constituents. The sequences of POS tags that are more frequent in target corpus receive higher scores after proper regularization is applied to prevent potential biases. Therefore, the final score assigned to a constituency tree results from a combination of the scores of its extracted sequences of POSs.

In order to use PUPA as a baseline, we implemented a dependency–based version, hencefoth referred to as *dPUPA*. dPUPA uses the same score computation of PUPA and collects statistics about sequences of POS tags: the difference lies in the fact that in this case the POS sequences are not extracted from constituency trees but rather from dependency trees. To be more concrete, rather than representing a sentence as a collection of constituency–based sequences of POSs, dPUPA represents each sentence as a collection of sequences of POSs covering all identified dependency subtrees. In particular, each dependency tree is represented as the set of all subtrees rooted by non–terminal nodes. Each subtree is then represented as the sequence of POS tags of the words in the subtree (reflecting the word order of the original sentence) integrated with the POS of the leftmost and rightmost in the sentence (*NULL* when there are no neighbors). Figure 1 shows the example of the dependency tree for the sentence *I will give you the ball.*

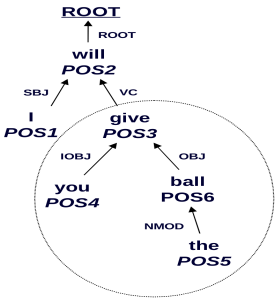



Figure 1: Example of dependency tree.

If we consider the subtree rooted by *give* (in the dotted circle), the resulting POS sequence is as follows: *POS2_POS3_POS4_POS5_POS6_NULL*, where *POS3_POS4_POS5_POS6* is the sequence of POS tags in the subtree, *POS2* is the left neighbor POS tag and *NULL* marks the absence of a right neighbor.

## 5 Experiments and Results

The experiments were organised as follows: a target corpus was automatically POS tagged (Dell'Orletta, 2009) and dependency–parsed; the ULISSE and baseline algorithms of reliable parse selection were run on the POS–tagged and dependency–parsed target corpus in order to identify high quality parses; results achieved by the selection algorithms were evaluated with respect to a subset of the target corpus of about 5,000 word–tokens (henceforth referred to as "test set") for which gold-standard annotation was available. Different sets of experiments were devised to test the robustness of our algorithm. They were performed with respect to i) the output of the parsers described in Section 3, ii) two different languages, iii) different domains.

For what concerns the languages, we chose Italian and English for two main reasons. First of all, they pose different challenges to a parser since they are characterised by quite different syntactic features. For instance, Italian, as opposed to English, is characterised by a relatively free word order (especially for what concerns subject and object relations with respect to the verb) and by the possible absence of an overt subject. Secondly, as it is shown in Section 5.1, Italian is a less resourced language with respect to English. This is a key issue, since as demonstrated by Reichart and Rappoport (2007b) and McClosky et al. (2008), small and big treebanks pose different problems in the reliable parses selection.

Last but not least, we aimed at demonstrating that ULISSE can be successfully used not only with texts belonging to the same domain as the parser training corpus. For this purpose, ULISSE was tested on a target corpus of Italian legislative texts, whose automatic linguistic analysis poses domain–specific challenges (Venturi, 2010). Out–of–domain experiments are being carried out also for English.

### 5.1 The Corpora

**The Italian corpora** Both parsers were trained on ISST–TANL[2], a dependency annotated corpus used in Evalita'09[3], an evaluation campaign carried out for Italian (Bosco et al., 2009). ISST–TANL includes 3,109 sentences (71,285 tokens) and consists of articles from newspapers and periodicals.

Two different target corpora were used for the in–domain and out–of–domain experiments. For the former, we used a corpus of 1,104,237 sentences (22,830,739 word–tokens) of newspapers texts which was extracted from the CLIC-ILC Corpus (Marinelli et al., 2003); for the legal domain, we used a collection of Italian legal texts (2,697,262 word–tokens; 97,564 sentences) regulating a variety of domains, ranging from environment, human rights, disability rights, freedom of expression to privacy, age disclaimer, etc. In the two experiments, the test sets were represented respectively by: a) the test set used in the Evalita'09 evaluation campaign, constituted by 260 sentences and 5,011 tokens from newpapers text; b) a set of 102 sentences (corresponding to 5,691 tokens) from legal texts.

**The English corpora** For the training of parsers we used the dependency–based version of Sections 2–11 of the Wall Street Journal partition of the Penn Treebank (Marcus et al., 2003), which was developed for the CoNLL 2007 Shared Task on Dependency Parsing (Nivre et al., 2007): it includes 447,000 word tokens and about 18,600 sentences.

As target data we took a corpus of news, specifically the whole Wall Street Journal Section of the

[2] http://medialab.di.unipi.it/wiki/SemaWiki
[3] http://evalita.fbk.eu/index.html

Penn Treebank[4], from which the portion of text corresponding to the training corpus was removed; the English target corpus thus includes 39,285,425 tokens (1,625,606 sentences). For testing we used the test set of the CoNLL 2007 Shared Task, corresponding to a subset of Section 23 of the Wall Street Journal partition of the Penn Treebank (5,003 tokens, 214 sentences).

### 5.2 Evaluation Methodology

Performances of the ULISSE algorithm have been evaluated i) with respect to the accuracy of ranked parses and ii) in terms of Precision and Recall. First, for each experiment we evaluated how the ULISSE algorithm and the baselines classify the sentences in the test set with respect to the "Labelled Attachment Score" (LAS) obtained by the parsers, i.e. the percentage of tokens for which it has predicted the correct head and dependency relation. In particular, we computed the LAS score of increasingly wider top lists of *k* tokens, where *k* ranges from 500 word tokens to the whole size of the test set (with a step size of 500 word tokens, i.e. k=500, k=1000, k=1500, etc.).

As regards ii), we focused on the set of ranked sentences showing a LAS $\geq \alpha$. Since imposing a 100% LAS was too restrictive, for each experiment we defined a different $\alpha$ threshold taking into account the performance of each parser across the different languages and domains. In particular, we took the top 25% and 50% of the list of ranked sentences and calculated Precision and Recall for each of them. To this specific end, a parse tree showing a LAS $\geq \alpha$ is considered as a *trustworthy analysis*. Precision has been computed as the ratio of the number of trustworthy analyses over the total number of sentences in each top list. Recall has been computed as the ratio of the number of trustworthy analyses which have been retrieved over the total number of trustworthy analyses in the whole test set.

In order to test how the ULISSE algorithm is able to select reliable parses by relying on *parse* features rather than on *raw text* features, we computed the accuracy score (LAS) of a subset of the top list of sentences parsed by both parsers and ranked by ULISSE: in particular, we focused on those sentences which were not shared by the MST and DeSR top lists.

[4]This corpus represents to the unlabelled data set distributed for the CoNLL 2007 Shared Task on Dependency Parsing, domain adaptation track.

### 5.3 Results

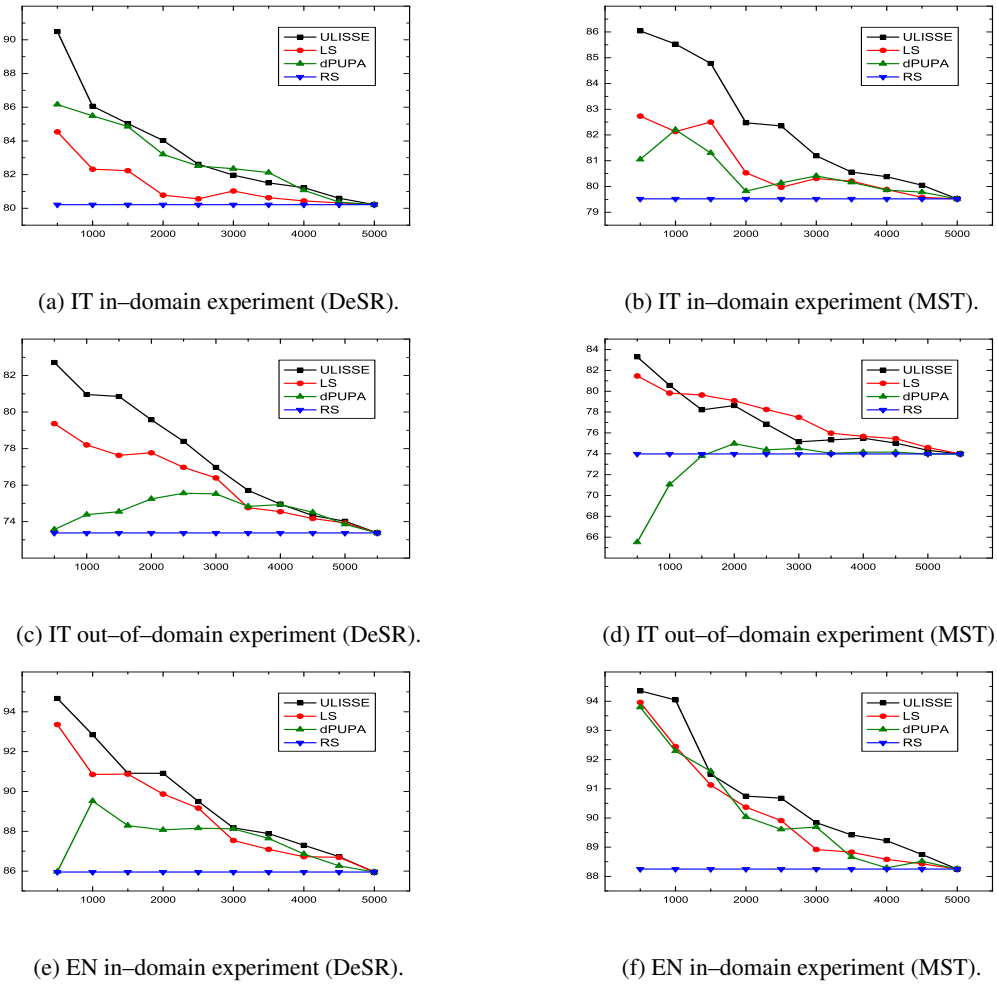
We will refer to the performed experiments as follows: "IT in–domain" and "IT out–of–domain" for the Italian experiments using respectively the ISST–TANL test set (henceforth ISST_TS) and the Legal-Corpus test set (henceforth Legal_TS); "EN in–domain" for the English experiment using the PTB test set (PTB_TS).

As a starting point let us consider the accuracy of DeSR and MST parsers on the whole test sets, reported in Table 1. The accuracy has been computed in terms of LAS and of Unlabelled Attachment Score (UAS), i.e. the percentage of tokens with a correctly identified syntactic head. It can be noticed that the performance of the two parsers is quite similar for Italian (i.e. wrt ISST_TS and Legal_TS), whereas there is a 2.3% difference between the MST and DeSR accuracy as far as English is concerned.

| | ISST_TS | | Legal_TS | | PTB_TS | |
|---|---|---|---|---|---|---|
| Parser | LAS | UAS | LAS | UAS | LAS | UAS |
| DeSR | 80.22 | 84.96 | 73.40 | 76.12 | 85.95 | 87.25 |
| MST | 79.52 | 85.43 | 73.99 | 78.72 | 88.25 | 89.55 |

Table 1: Overall accuracy of DeSR and MST parsers.

The plots in Figure 2 show the LAS of parses ranked by ULISSE and the baselines across the different experiments. Each plot reports the results of a single experiment: plots in the same row report the LAS of DeSR and MST parsers with respect to the same test set. In all experiments, ULISSE turned out to be the best ranking algorithm since it appears to select top lists characterised by higher LAS scores than the baselines. As Figure 2 shows, all ranking algorithms perform better than Random Selection (RS), i.e. all top lists (for each *k* value) show a LAS higher than the accuracy of DeSR and MST parsers on the whole test sets. In the EN in–domain experiment, the difference between the results of ULISSE and the other ranking algorithms is smaller than in the corresponding Italian experiment, a fact resulting from the higher accuracy of DeSR and MST parsers (i.e. LAS 85.95% and 88.25% respectively) on the PTB_TS. It follows that, for example, the first top list (with *k*=500) of the SL baseline has a




(a) IT in–domain experiment (DeSR).

(b) IT in–domain experiment (MST).

(c) IT out–of–domain experiment (DeSR).

(d) IT out–of–domain experiment (MST).

(e) EN in–domain experiment (DeSR).

(f) EN in–domain experiment (MST).

Figure 2: LAS of parses ranked by ULISSE algorithm and by the three baselines.

LAS accuracy of 93.36% and 93.96% respectively for DeSR and MST: even in this case, ULISSE outperforms all baselines. This is also the case in the IT out–of–domain experiment. As reported in Table 1, parsing legal texts is a quite challenging task due to a number of domain–specific peculiarities at the level of syntax: this is testified by the average sentence length which in the Legal_TS is 56 word tokens. Nevertheless, ULISSE is able also in this case to highly rank long sentences showing a high LAS. For example, while in the first top list of 500 word tokens the sentences parsed by DeSR and ordered by SL have an average sentence length of 24 words and a LAS of 79.37%, ULISSE includes in the same top list longer sentences (with average sentence length = 29) with a higher LAS (82.72%). Also dPUPA ranks in the same top list quite long sentences (with 27 average sentence length), but compared to ULISSE it shows a lower LAS (i.e. 73.56%).

| | IT in–domain | | IT out–of–domain | | EN in–domain | |
|---|---|---|---|---|---|---|
| | DeSR | MST | DeSR | MST | DeSR | MST |
| MST top–list | 80.93 | **80.27** | 68.84 | 74.58 | 83.37 | **90.39** |
| DeSR top–list | **82.46** | 77.82 | **75.47** | **74.88** | **86.50** | 86.74 |

Table 3: LAS of not–shared sentences in the DeSR and MST top–lists.

Results in Table 2 show that in the top 25% of the ranked sentences with a LAS $\geq \alpha$ ULISSE has the highest Precision and Recall in all experiments. We believe that the low performance of dPUPA with respect to all other ranking algorithms can be due to

| | DeSR | | | | | | | | MST | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 25% | | | | 50% | | | | 25% | | | | 50% | | | |
| | Prec | Rec | LAS | AvgSL | Prec | Rec | LAS | AvgSL | Prec | Rec | LAS | AvgSL | Prec | Rec | LAS | AvgSL |
| **IT in–domain: LAS $\geq$ 85%** (DeSR: 120 sentences; MST: 112 sentences) | | | | | | | | | | | | | | | | |
| ULISSE | **66.15** | **35.83** | **88.25** | 5.25 | **59.23** | **64.17** | 84.30 | 14.60 | **60** | **34.82** | **86.16** | 5.68 | **55.38** | **64.29** | **83.39** | 15.27 |
| LS | 63.08 | 34.17 | 84.54 | 4.15 | 53.08 | 57.50 | 82.07 | 11.90 | 58.46 | 33.93 | 82.73 | 4.45 | 53.08 | 61.61 | 82.14 | 12.75 |
| dPUPA | 61.54 | 33.33 | 86.89 | 6.68 | **59.23** | **64.17** | **84.36** | 14.82 | 53.85 | 31.25 | 82.26 | 8.61 | 50.00 | 58.04 | 79.94 | 17.04 |
| **IT out–of–domain: LAS $\geq$ 75%** (DeSR: 51 sentences; MST: 57 sentences) | | | | | | | | | | | | | | | | |
| ULISSE | **73.08** | **37.25** | **80.75** | 16.71 | **69.23** | **70.59** | **79.17** | 41.80 | **69.23** | **31.58** | **81.47** | 13.63 | 67.31 | 61.40 | 78.36 | 36 |
| LS | 53.85 | 27.45 | 76.71 | 12.63 | 67.31 | 68.63 | 78.34 | 34.14 | 61.54 | 28.07 | 78.42 | 11.30 | **69.23** | **63.16** | **79.78** | 30.54 |
| dPUPA | 57.69 | 29.41 | 73.97 | 15.67 | 61.54 | 62.74 | 75.24 | 40.39 | 46.15 | 21.05 | 72.08 | 22.56 | 57.69 | 52.63 | 74.86 | 42.91 |
| **EN in–domain: LAS $\geq$ 90%** (DeSR: 118 sentences; MST: 120 sentences) | | | | | | | | | | | | | | | | |
| ULISSE | **81.48** | **37.29** | **94.50** | 6.31 | **69.44** | **63.56** | 90.93 | 16.36 | **77.78** | **35** | **93.74** | 5.82 | **69.44** | **62.5** | **91.20** | 16.48 |
| LS | 77.78 | 35.59 | 93.39 | 4.87 | 65.74 | 60.17 | **91.01** | 13.67 | 75.92 | 34.17 | 93.55 | 4.79 | 68.52 | 61.67 | 90.84 | 13.44 |
| dPUPA | 74.07 | 33.90 | 89.76 | 7.95 | 65.74 | 60.17 | 88.37 | 18.14 | **77.78** | **35** | 93.43 | 5.08 | 68.52 | 61.67 | 91.03 | 14.49 |

Table 2: **In all Tables**: the number of sentences with a LAS $\geq \alpha$ parsed by DeSr and MST parsers (first row); Precision (Prec), Recall (Rec), the corresponding parser accuracy (LAS) of the top 25% and 50% of the list of sentences and ranked by the ULISSE algorithm, Length of Sentence (LS) and dependency PUPA (dPUPA) and the corresponding average length in tokens of ranked sentence (AvgSL).

the fact that PUPA is based on constituency–specific features that once translated in terms of dependency structures may be not so effective.

In order to show that the ranking of sentences does not follow from raw text features but rather from parse features, we evaluated the accuracy of parsed sentences that are not–shared by MST and DeSR top–lists selected by ULISSE. For each test set we selected a different top list: a set of 100 sentences in the IT and EN in–domain experiments and of 50 sentences in the IT out–of–domain experiment. For each of them we have a different number of not–shared sentences: 24, 15 and 16 in the IT in–domain, IT out–of–domain and EN in–domain experiments respectively. Table 3 reports the LAS of DeSR and MST for these sentences: it can be observed that the LAS of not–shared sentences in the DeSR top list is always higher than the LAS assigned by the same parser to the not–shared sentences in the MST top list, and viceversa. For instance, in the English experiment the LAS achieved by DeSR on the not–shared top list is higher (86.50) than the LAS of DeSR on the not–shared MST top list (83.37); viceversa, the LAS of MST on the not–shared DeSR top list is higher (86.74) than the LAS of MST on the not–shared MST top list (90.39). The unique exception is MST in the IT out–of–domain experiment, but the difference in terms of LAS between the parses is not statistically relevant (p–value $< 0.05$). These results demonstrate that ULISSE is able to select parsed sentences on the basis of the reliability of the analysis produced by each parser.

## 6 Conclusion

ULISSE is an unsupervised linguistically–driven method to select reliable parses from the output of dependency parsers. To our knowledge, it represents the first unsupervised ranking algorithm operating on dependency representations which are more and more gaining in popularity and are arguably more useful for some applications than constituency parsers. ULISSE shows a promising performance against the output of two supervised parsers selected for their behavioral differences. In all experiments, ULISSE outperforms all baselines, including dPUPA and Sentence Length (SL), the latter representing a very strong baseline selection method in a supervised scenario, where parsers have a very high performance with short sentences. The fact of carrying out the task of reliable parse selection in a supervised scenario represents an important novelty: however, the unsupervised nature of ULISSE could also be used in an unsupervised scenario (Reichart and Rappoport, 2010). Current direction of research include a careful study of a) the quality score function, in particular for what concerns the combination of individual feature weights, and b) the role and effectivess of the set of linguistic features. This study is being carried out with a specific view to NLP tasks which might benefit from the ULISSE algorithm. This is the case, for instance, of the domain adaptation task in a self–training scenario (McClosky et al., 2006), of the treebank construction process by minimizing the human annotators' efforts (Reichart and Rappoport, 2009b), of *n–best* ranking methods for machine translation (Zhang, 2006).

## References


Giuseppe Attardi. 2006. *Experiments with a multilanguage non-projective dependency parser*. In Proceedings of the Tenth Conference on Computational Natural Language Learning (CoNLL-X '06), New York City, New York, pp. 166–170.

Cristina Bosco, Simonetta Montemagni, Alessandro Mazzei, Vincenzo Lombardo, Felice Dell'Orletta and Alessandro Lenci. 2009. Parsing Task: comparing dependency parsers and treebanks. In Proceedings of Evalita'09, Reggio Emilia.

Sabine Buchholz and Erwin Marsi. 2006. *CoNLL-X shared task on multilingual dependency parsing*. In Proceedings of CoNLL.

Felice Dell'Orletta. 2009. *Ensemble system for Part-of-Speech tagging*. In Proceedings of Evalita'09, Evaluation of NLP and Speech Tools for Italian, Reggio Emilia, December.

Lyn Frazier. 1985. *Syntactic complexity*. In D.R. Dowty, L. Karttunen and A.M. Zwicky (eds.), *Natural Language Parsing*, Cambridge University Press, Cambridge, UK.

Edward Gibson. 1998. *Linguistic complexity: Locality of syntactic dependencies*. In *Cognition*, 68(1), pp. 1-76.

Daisuke Kawahara and Kiyotaka Uchimoto. 2008. *Learning Reliability of Parses for Domain Adaptation of Dependency Parsing*. In Proceedings of IJCNLP 2008, pp. 709–714.

Dekan Lin. 1996. *On the structural complexity of natural language sentences*. In Proceedings of COLING 1996, pp. 729–733.

Ryan McDonald, Kevin Lerman, and Fernando Pereira. 2006. *Multilingual dependency analysis with a two-stage discriminative parser*. In Proceedings of CoNLL 2006.

Ryan McDonald and Joakim Nivre. 2007. *Characterizing the Errors of Data-Driven Dependency Parsing Models*. In Proceedings of EMNLP-CoNLL, 2007, pp. 122-131.

Mitchell P. Marcus, Mary Ann Marcinkiewicz and Beatrice Santorini. 1993. *Building a large annotated corpus of English: the penn treebank*. In Comput. Linguist.,vol. 19, issue 2, MIT Press, pp. 313–330.

Rita Marinelli, et al. 2003. *The Italian PAROLE corpus: an overview*. In A. Zampolli et al. (eds.), *Computational Linguistics in Pisa*, XVI–XVII, Pisa–Roma, IEPI., I, 401–421.

David McClosky, Eugene Charniak and Mark Johnson. 2006. *Reranking and self–training for parser adaptation*. In Proceedings of ICCL–ACL 2006, pp. 337–344.

David McClosky, Eugene Charniak and Mark Johnson. 2008. *When is Self–Trainig Effective for parsing?*. In Proceedings of COLING 2008, pp. 561–568.

Jim Miller and Regina Weinert. 1998. *Spontaneous spoken language. Syntax and discourse*. Oxford, Clarendon Press.

Ani Nenkova, Jieun Chae, Annie Louis, and Emily Pitler. 2010. *Structural Features for Predicting the Linguistic Quality of Text Applications to Machine Translation, Automatic Summarization and Human–Authored Text*. In E. Krahmer, M. Theune (eds.), *Empirical Methods in NLG*, LNAI 5790, Springer-Verlag Berlin Heidelberg, pp. 222241.

Joakim Nivre, Johan Hall, Sandra Kübler, Ryan McDonald, Jens Nilsson, Sebastian Riedel, Deniz Yuret. 2007. *The CoNLL 2007 Shared Task on Dependency Parsing*. In Proceedings of the EMNLP-CoNLL, pp. 915–932.

Sujith Ravi, Kevin Knight and Radu Soricut. 2008. *Automatic Prediction of Parser Accuracy*. In Proceedings of the EMNLP 2008, pp. 887–896.

Roi Reichart and Ari Rappoport. 2007a. *An ensemble method for selection of high quality parses*. In Proceedings of ACL 2007, pp. 408–415.

Roi Reichart and Ari Rappoport. 2007b. *Self–Training for Enhancement and Domain Adaptation of Statistical Parsers Trained on Small Datasets*. In Proceedings of ACL 2007, pp. 616–623.

Roi Reichart and Ari Rappoport. 2009a. *Automatic Selection of High Quality Parses Created By a Fully Unsupervised Parser*. In Proceedings of CoNLL 2009, pp. 156–164.

Roi Reichart and Ari Rappoport. 2009b. *Sample Selection for Statistical Parsers: Cognitively Driven Algorithms and Evaluation Measures*. In Proceedings of CoNLL 2009, pp. 3–11.

Roi Reichart and Ari Rappoport. 2010. *Improved Fully Unsupervised Parsing with Zoomed Learning*. In Proceedings of EMNLP 2010.

Brian Roark, Margaret Mitchell and Kristy Hollingshead. 2007. *Syntactic complexity measures for detecting Mild Cognitive Impairment*. In Proceedings of ACL Workshop on Biological, Translational, and Clinical Language Processing (BioNLP'07), pp. 1–8.

Kenji Sagae and Junichi Tsujii. 2007. *Dependency Parsing and Domain Adaptation with LR Models and Parser Ensemble*. In Proceedings of the EMNLP–CoNLL 2007, pp. 1044–1050.

Giulia Venturi. 2010. *Legal Language and Legal Knowledge Management Applications*. In E. Francesconi, S. Montemagni, W. Peters and D. Tiscornia (eds.), *Semantic Processing of Legal Texts*, Lecture Notes in Computer Science, Springer Berlin / Heidelberg, vol. 6036, pp. 3-26.

Alexander Yates, Stefan Schoenmackers and Oren Etzioni. 2006. *Detecting Parser Errors Using Web–based Semantic Filters*. In Proceedings of the EMNLP 2006, pp. 27–34.

Victor H.A. Yngve. 1960. *A model and an hypothesis for language structure*. In Proceedings of the American Philosophical Society, pp. 444-466.

Ying Zhang, Almut Hildebrand and Stephan Vogel. 2006. *Distributed language modeling for N-best list re-ranking*. In Proceedings of the EMNLP 2006, pp. 216–223.